

IndraMotion MLC 14VRS

Robot-Control V2

Referenz
R911341586

Ausgabe 07



Titel	IndraMotion MLC 14VRS Robot-Control V2
Art der Dokumentation	Referenz
Dokumentations-Type	DOK-MLC***-ROCO****V14-RE07-DE-P
Interner Ablagevermerk	RS-527e93f498bd3aa00a6846a501f57a0f-7-de-DE-13

Änderungsverlauf

Ausgabe	Stand	Bemerkung
Ausgabe 01	2015-01	Erstausgabe für IndraMotion MLC 14VRS
Ausgabe 02	2015-11	Update, KinInterface/ML_KinTech/Visu, ML_Robot/ ML_RobotMovePoint, Trafo 22/23/39
Ausgabe 03	2016-06	Neu: KinInterface-HMI-Template Neu: RoCo_03_073_4-Achs-Scara mit Kurbel, Update: Ro- Co_03_018_Umschlingungskinematik ML_Robot: ML_RobotGetSafeZones
Ausgabe 04	2016-08	Robot-Control: Funktionsbeschreibung Neu: MoveID, Jump ML_Robot: ML_RobotMoveJumpAbsolute, ML_RobotGet- MoveID, ML_RobotSetMoveID
Ausgabe 05	2017-02	Update Oberflächenfunktionen, Funktionsbeschreibung HalfJump/Jump, Transformation 81: 3-Achs-Canfield
Ausgabe 06	2017-09	Integration Hexpod - Trafo, Neues Kapitel: "Anbindung von Robotik-Fremdsystemen", Beschreibung uniVAL drive
Ausgabe 07	2018-03	Neue Kapitel "Punkt", "Achstypen", Ergänzung "Armigkeits- wahl bei PTP-Bewegungen"

Schutzvermerk	© Bosch Rexroth AG 2018 Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.
Verbindlichkeit	Die angegebenen Daten dienen allein der Produktbeschreibung und sind nicht als zugesicherte Eigenschaften im Rechtssinne zu verstehen. Änderungen im Inhalt der Dokumentation und Liefermöglichkeiten der Produkte sind vorbehalten.
Redaktion	System IndraLogic & Basis Motion Logic, MaEI (TaDo)

Inhaltsverzeichnis

	Seite
1 Zu dieser Dokumentation.....	13
1.1 Gültigkeit der Dokumentation.....	13
1.2 Gliederung der Dokumentation.....	13
1.3 Gebrauch der Sicherheitshinweise.....	14
1.3.1 Aufbau der Sicherheitshinweise.....	14
1.3.2 Erläuterung der Signalwörter und der Signalgrafik.....	14
1.3.3 Verwendete Symbole.....	15
1.3.4 Erläuterung der Signalgrafik auf dem Gerät.....	15
1.4 Erforderliche und ergänzende Dokumentationen XLC/MLC.....	15
1.5 Bezeichnungen und Abkürzungen.....	21
1.6 Kundenfeedback.....	22
2 Funktionsbeschreibung.....	23
2.1 Systemübersicht.....	23
2.2 Koordinatensysteme und Transformationen.....	24
2.2.1 Koordinatensysteme.....	24
Koordinatensysteme, Übersicht.....	24
Koordinatensystemtypen.....	24
Achsen-Koordinatensystem (ACS).....	26
Basis-Koordinatensystem (BCS).....	26
Maschinen-Koordinatensystem (MCS).....	26
Produkt-Koordinatensystem (PCS).....	27
2.2.2 Transformationen.....	28
Transformationen, Übersicht.....	28
ACS-BCS-Transformation.....	28
BCS-MCS-Transformation.....	29
MCS-PCS-Transformation.....	29
2.3 Achstypen.....	30
2.3.1 Achstypen, Überblick.....	30
2.3.2 Positionierungsachsen.....	30
2.3.3 Orientierungsachsen.....	30
2.3.4 Hilfsachsen.....	31
2.3.5 Bandachsen.....	31
2.4 Punkt.....	31
2.4.1 Punkt, Übersicht.....	31
2.4.2 Konfiguration der Punktdarstellung.....	32
2.5 Zustandsdiagramm einer Kinematik.....	33
2.6 Robot-Control, Kommandos und Funktionen.....	35
2.6.1 Robot-Control, Kommandos und Funktionen, Überblick.....	35
2.6.2 Zustand eines Kommandos.....	36
2.6.3 Liste aller Kommandos und Funktionen.....	37
2.7 Bewegungsarten.....	39
2.7.1 Bewegungsarten, Überblick.....	39
2.7.2 Bewegungsart Linear.....	40

Inhaltsverzeichnis

	Seite
2.7.3 Bewegungsart Circular.....	41
2.7.4 Bewegungsart Punkt zu Punkt.....	42
2.7.5 Bewegungsart Jump.....	42
2.7.6 Bewegungsart HalfJump.....	45
2.7.7 Absolute Positionierung.....	48
2.7.8 Relative Positionierung.....	48
2.7.9 Kommandieren von modulo-gewichteten Achsen.....	48
2.8 Handbetrieb.....	49
2.8.1 Handbetrieb, Allgemein.....	49
2.8.2 Schrittbetrieb.....	50
2.8.3 Kontinuierlicher Betrieb.....	50
2.9 Teach-in.....	50
2.9.1 Teach-in, Allgemein.....	50
2.9.2 Aufbau der Punktedatei.....	51
2.9.3 Schreibe Punkt.....	52
2.9.4 Lese Punkt.....	52
2.9.5 Lese nächsten Punkt.....	52
2.9.6 Lösche Punkte.....	53
2.9.7 Speichere Punktedatei.....	53
2.9.8 Lade Punktedatei.....	54
2.9.9 Lösche Punktedatei.....	54
2.10 Überschleifen.....	54
2.10.1 Überschleifen, Allgemeines.....	54
2.10.2 Überschleifradius.....	55
2.10.3 Slope-Arten.....	55
2.10.4 Block_Slope.....	55
2.10.5 Cont_Slope.....	58
2.10.6 Überschleifmöglichkeiten bei verschiedenen Bewegungsarten.....	61
2.11 Robot Control, Move ID.....	62
2.11.1 Robot Control, Move ID, Setzen.....	62
2.11.2 Robot Control, Move ID, Überschleifen.....	63
2.12 Stoppen, Unterbrechen, Wiederaufnahme von Bewegungen.....	64
2.12.1 Stoppen von Bewegungen.....	64
2.12.2 Unterbrechen von Bewegungen.....	65
2.12.3 Wiederaufnahme von Bewegungen.....	66
2.12.4 Warten.....	67
2.13 Bandsynchrone Bewegung.....	67
2.13.1 Bandsynchrone Bewegung, Allgemeines.....	67
2.13.2 Konfiguration eines Bandes.....	68
2.13.3 Bandwerte und Punkte auf dem Band.....	69
2.13.4 Betrieb eines Bandes.....	72
2.13.5 Adaptive Motion.....	74
2.13.6 Überschleifen und Bandsynchron.....	75
2.13.7 Zusammenfassung.....	75
2.14 Schalten auf der Bahn.....	76
2.15 Verfahrbereichsgrenzen.....	79

Inhaltsverzeichnis

	Seite
2.16 Fehler und Fehlerreaktion.....	81
2.16.1 Fehler und Fehlerreaktion, Allgemeines.....	81
2.16.2 Fehlerreaktion der kommandierten Bewegung.....	82
2.16.3 Bandspezifische Fehlerreaktion.....	82
2.17 Zyklische Sollwertvorgabe.....	83
2.17.1 Zyklische Sollwertvorgabe, Allgemeines.....	83
2.17.2 Start und Betrieb der zyklischen Sollwertvorgabe.....	83
2.17.3 Die zyklische Sollwertvorgabe.....	84
2.17.4 Ende der zyklischen Sollwertvorgabe.....	85
2.18 Performancemessung.....	85
3 Oberflächenfunktionen.....	87
3.1 Oberflächenfunktionen, Allgemeines.....	87
3.2 Ordner Robot.....	87
3.2.1 Ordner Robot, Allgemeines.....	87
3.2.2 Hinzufügen einer neuen Kinematik.....	87
3.2.3 Kinematikinbetriebnahme.....	89
Übersicht.....	89
Sicherheitsfunktionen.....	91
Gruppieren einer Kinematik.....	92
Tippen.....	93
Positionieren.....	94
Bandkonfiguration.....	95
Produktkoordinaten.....	96
SPS-Konfiguration.....	97
3.2.4 PathViewer.....	98
Übersicht.....	98
Navigation und Bedingung.....	99
Kontextmenü des PathViewers.....	100
3.3 Ordner Kinematik.....	101
3.3.1 Ordner Kinematik, Allgemeines.....	101
3.3.2 Kinematik - Kontextmenü.....	102
Kinematik - Kontextmenü bzw. Menüpunkt Kinematik, Allgemeines.....	102
Erstinbetriebnahme.....	102
Diagnose.....	105
3.3.3 Kinematik - Dialoge.....	107
Dialoge im Überblick.....	107
Wichtung/Maßeinheiten.....	107
Achslängen und Kopplung.....	108
Bewegungsgrenzwerte.....	108
Koordinatenverschiebung.....	109
4 Transformationsschemen.....	111
4.1 Transformationsschemen, Überblick.....	111
4.2 Transformationsschemen.....	112

Inhaltsverzeichnis

	Seite
4.2.1	Kartesiser..... 112
4.2.2	2-Achs-Delta mit optional 1 Orientierungsachse..... 113
4.2.3	4-Achs Scara mit linearer Hubachse..... 115
4.2.4	3-Achs-Scara mit optional 1 Orientierungsachse..... 119
4.2.5	4-Achs-Scara mit optional 1 Orientierungsachse..... 123
4.2.6	2-Achs-Parallelogramm..... 127
4.2.7	2-Achs-Parallelogramm invers..... 129
4.2.8	2-Achs Presse..... 130
4.2.9	Umschlingungskinematik..... 132
4.2.10	2-Achs-Parallelogramm V1..... 133
4.2.11	3-Achs-Delta mit optional 1 Orientierungsachse..... 135
4.2.12	2-Achs-Parallelogramm V2..... 137
4.2.13	3-Achs-Delta Linear mit optional 1 Orientierungsachse..... 138
4.2.14	5-Achs-Delta..... 141
4.2.15	Adabot..... 145
4.2.16	2-Achs-Delta mit asymmetrischem TCP..... 148
4.2.17	3-Achs-Twisting Crane..... 149
4.2.18	3-Achs-Flex Stamper..... 151
4.2.19	3-Achs-Delta mit Linearachsen mit optional 1 Orientierungsachse..... 152
4.2.20	2-Achs-Delta Slide..... 156
4.2.21	3-Achs-Parallelogramm V3..... 157
4.2.22	6-Achs-Kartesiser..... 158
4.2.23	3-Achs-Delta mit 90 Grad Winkeln..... 160
4.2.24	4-Achs-Speedpicker..... 163
4.2.25	Stewart-Plattform 165
4.2.26	4-Achs-Scara mit Kurbel..... 169
4.2.27	3-Achs Canfield 173
4.3	Benutzerdefinierte Transformation..... 176
4.4	Erweiterte Benutzerdefinierte Transformation..... 178
4.5	Mechanische Informationen..... 183
5	ML_Robot.library..... 185
5.1	Einführung und Übersicht..... 185
5.2	Globale Variablen..... 186
5.2.1	Einführung und Übersicht..... 186
5.2.2	GVL_RobotConstant, Globale Konstanten..... 187
5.2.3	KinData[] - Kinematikdaten im Shared Memory 187
5.2.4	DV_Kin[] - Kinematikdaten über Funktionsvariablenzugriff 188
5.3	Funktionsbausteine..... 190
5.3.1	Funktionsbausteine, Übersicht..... 190
5.3.2	Standardschnittstellen an Funktionsbausteinen..... 193
	Motivation..... 193
	Bausteintypen..... 193
	Bausteintypen und ihre Schnittstellen..... 194
	Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung..... 196
	Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung..... 199

Inhaltsverzeichnis

	Seite
Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung.....	202
Ein- und Ausgänge flankengesteuerter Funktionsbausteine, permanente Bearbeitung.....	205
5.3.3 InBuffer-Ausgang an ML_Robot-Funktionsbausteinen.....	207
5.3.4 Kinematic/ Config.....	208
MC_GroupReset.....	208
MC_SetCartesianTransform.....	209
MC_SetKinTransform.....	211
MC_SetKinTransformExt.....	213
MC_UngroupAllAxes.....	214
ML_AddMechanicData.....	215
ML_GroupAllAxes.....	216
ML_GroupSetSafeZone.....	218
ML_SetBeltDesyncRelConfiguration.....	219
ML_SetBeltConfiguration.....	220
ML_SetBeltErrorReaction.....	222
ML_SetCylindricTransform.....	223
ML_SetRotaryTableConfiguration.....	225
ML_RobotSetMoveID.....	227
5.3.5 Kinematic/ Info.....	228
MC_GroupReadActualPosition.....	228
ML_RobotGetBeltConfiguration.....	229
ML_RobotGetBeltDesyncRelConfiguration.....	231
ML_RobotGetBeltErrorReaction.....	232
ML_RobotGetCmdInfo.....	233
ML_RobotGetMechanic.....	234
ML_RobotGetMoveID.....	236
ML_RobotGetPerformance.....	237
ML_RobotGetSafeZones.....	238
ML_RobotGetTrafoValues.....	239
ML_RobotMovePoint.....	241
ML_RobotReadCyclicChannel.....	242
ML_RobotSetPerformanceEnable.....	244
ML_RobotTransformPoint.....	245
ML_SwitchOnPath.....	246
5.3.6 Kinematic/ Movement.....	247
MC_GroupContinue.....	247
MC_GroupInterrupt.....	249
MC_GroupStop.....	250
MC_MoveCircularAbsolute.....	251
MC_MoveCircularRelative.....	254
MC_MoveDirectAbsolute.....	257
MC_MoveDirectRelative.....	259
MC_MoveLinearAbsolute.....	261
MC_MoveLinearRelative.....	264
ML_BeltDeSynchronisationWithLimits.....	266
ML_BeltSynchronisationWithLimits.....	267

Inhaltsverzeichnis

	Seite
ML_GroupJogCont.....	269
ML_GroupJogStep.....	271
ML_GroupWait.....	273
ML_RobotOpenCyclicChannel.....	275
ML_RobotOpenDirectCyclicChannel.....	276
ML_RobotWriteCyclicChannel.....	277
ML_RobotMoveJumpAbsolute.....	279
5.3.7 Kinematic/ Parameter.....	282
Zugriff auf Daten einer Kinematik.....	282
MB_GroupReadListParameter.....	283
MB_GroupReadParameter.....	284
MB_GroupReadRealParameter.....	285
MB_GroupReadSercosAttribute.....	286
MB_GroupReadSercosDataStatus.....	288
MB_GroupReadStringParameter.....	289
MB_GroupWriteListParameter.....	290
MB_GroupWriteParameter.....	291
MB_GroupWriteRealParameter.....	292
MB_GroupWriteStringParameter.....	293
5.3.8 Kinematic/ TeachIn.....	295
ML_GroupDeletePointFile.....	295
ML_GroupDeletePoints.....	296
ML_GroupLoadPointFile.....	297
ML_GroupReadNextPoint.....	298
ML_GroupReadPoint.....	299
ML_GroupSavePointFile.....	301
ML_GroupWritePoint.....	302
5.4 Datentypen.....	303
5.4.1 Datentypen, Übersicht.....	303
5.4.2 Data Types/ Config.....	305
MC_KIN_REF.....	305
MC_KIN_REF, Übersicht.....	305
MC_KIN_REF, Initialisation.....	305
MC_KIN_REF, Forward.....	306
MC_KIN_REF, Inverse.....	306
Fehlerhandling bei Interfacefunktionen.....	306
MC_KIN_REF_EXT.....	307
MC_KIN_REF_EXT, Übersicht.....	307
MC_KIN_REF_EXT, DetermineMemory.....	307
MC_KIN_REF_EXT, Forward.....	308
MC_KIN_REF_EXT, GetMechanic.....	308
MC_KIN_REF_EXT, GetUsedAxes.....	309
MC_KIN_REF_EXT, Initialisation.....	309
MC_KIN_REF_EXT, Inverse.....	309
Fehlerhandling bei Interfacefunktionen.....	309
ML_BELT_ERROR_REACTION_MODE.....	310

Inhaltsverzeichnis

	Seite
ML_KIN_AXES_CONFIG_DATA.....	310
ML_KIN_AXIS_CONFIG.....	311
ML_KIN_SAFE_ZONE_TYPE.....	311
ML_KIN_USED_AXES.....	311
5.4.3 Data Types/ Info.....	312
E_KIN_MECH_CMD.....	312
ML_KINEMATICS_DATA_SM.....	312
ML_KIN_CMD_INFO.....	313
ML_KIN_CS_TYPE.....	314
ML_KIN_INFO_TYPE.....	314
ML_KIN_MECHANIC_DATA.....	314
ML_KIN_PERF_DATA.....	315
ML_KIN_SAFE_ZONE_DATA.....	315
ML_POINTKIND.....	315
5.4.4 Data Types/ TeachIn.....	315
E_KIN_TEACH_IN_POINT_FILE_OPTIONS.....	315
5.4.5 Data Types/ Movement.....	317
ML_ARR_COORDS.....	317
ML_CIRCMODES.....	317
ML_COORDSYS.....	317
ML_CYCLIC_MODE.....	318
ML_SET_MODE.....	319
ML_SLOPE_TYPE.....	319
6 Kinematik-Interface.....	321
6.1 Einführung.....	321
6.2 Funktionsbeschreibung.....	321
6.2.1 Überblick.....	321
6.2.2 Programm zum Kinematik-Interface.....	323
6.2.3 Visualisierungen zum Kinematik-Interface.....	323
Überblick.....	323
Systemübersicht Visualisierung.....	324
Kinematikübersicht Visualisierung.....	327
Referenzierübersicht (Homing) Visualisierung.....	328
Positionierübersicht (Position) Visualisierung.....	329
Bandkonfiguration (ConveyorBelt) Visualisierung.....	331
Tipp-/Einrichtbetrieb (Setup) Visualisierung.....	332
Einlernbetrieb (TeachIn) Visualisierung.....	332
PCS-Konfiguration (ProductCoordSystem) Visualisierung.....	334
6.2.4 Betriebsarten.....	335
Überblick.....	335
Kinematik Bereit, Antriebe Bereit.....	336
Kinematik Bereit, Antriebe Halt.....	336
Kinematik Referenzieren.....	337
Absolutes PTP Positionieren.....	337
Relatives PTP Positionieren.....	338

Inhaltsverzeichnis

	Seite
Absolutes Lineares Positionieren.....	339
Relatives Lineares Positionieren.....	340
Absolutes Zirkulares Positionieren.....	341
Relatives Zirkulares Positionieren.....	342
Absolutes Jump Positionieren.....	343
Kinematik Stopp.....	344
Betriebsart "Interrupted".....	345
Betriebsart "Continue".....	345
Betriebsart "Externer Funktionsbaustein".....	346
Zyklische Positionssollwertvorgabe.....	346
6.2.5 Dialogbasierte Kinematikkommandierung.....	347
6.2.6 Kinematik-Interface Anwendererweiterung.....	350
Hinweise zur Implementation von Anwendererweiterungen.....	350
Erweiterung des MB_KinInterfaceType01.....	351
Erweitern der Kinematik-Interface-Struktur MB_KinInterfaceType01.....	351
Erweitern des Funktionsbausteins TE_KinematicsInterface.....	351
Erweiterung des MB_KinInterfaceType11.....	352
Bestehendes Interface erweitern.....	352
Neues Interface hinzufügen.....	352
6.2.7 Umstellung auf ereignisgesteuertes Kinematik-Interface (Typ 11).....	353
6.3 Kinematik-Interface HMI Template.....	353
6.3.1 Einleitung.....	353
6.3.2 KinInterface-Masken.....	355
Kinematikstatus.....	355
Kinematik vertippen.....	356
Punkte teachen.....	357
Kinematik positionieren.....	358
7 ML_KinTech.library.....	361
7.1 Einführung und Übersicht.....	361
7.2 KinInterface/Const.....	361
7.2.1 Global_MB_KinInterfaceVars.....	361
7.3 KinInterface/FB.....	363
7.3.1 ML_KinematicsInitType01.....	363
7.3.2 ML_KinematicsInterfaceType01.....	366
7.3.3 ML_KinematicsInterfaceType11.....	369
7.3.4 KinInterface/FB/DataTypes.....	372
Kin_Axis_Config_Elements.....	372
Kin_Axis_Type.....	372
7.4 KinInterface/Type.....	373
7.4.1 Kinematik-Interface-Strukturen, Überblick.....	373
7.4.2 Kinematik-Interface-Betriebsarten, Übersicht.....	373
7.4.3 MB_KINEMATICS_MODE_UNION.....	374
7.4.4 MB_KINEMATICS_MODE_BITS.....	374
7.4.5 MB_KINEMATICS_MODE.....	375
7.4.6 KinInterface/Type/Control.....	376

Inhaltsverzeichnis

	Seite
Ordner Kin_Interface/Type/Control, Übersicht.....	376
MB_KINEMATICS_CONTROL_TYPE01.....	377
MB_KINEMATICS_ADMINISTRATION.....	377
MB_KINEMATICS_CONVEYOR_BELT.....	380
MB_KINEMATICS_TEACH_IN.....	380
MB_KINEMATICS_TEACH_IN_MODE.....	382
MB_KINEMATICS_ADMIN_CHANGE_OPMODE.....	383
MB_KINEMATICS_PCS_CONFIG.....	383
MB_KINEMATICS_SETUP.....	383
MB_KINEMATICS_COORDINATED.....	385
ITF_KINEMATICS-Strukturen.....	387
7.4.7 KinInterface/Type/Status.....	388
Ordner Kin_Interface/Type/Status, Übersicht.....	388
MB_KINEMATICS_STATUS_TYPE01.....	388
MB_KINEMATICS_ADMIN_STATUS.....	389
MB_KINEMATICS_CONV_BELT_STATUS.....	391
MB_KINEMATICS_TEACH_IN_STATUS.....	392
MB_KINEMATICS_PCS_STATUS.....	392
MB_KINEMATICS_DIAGNOSIS.....	393
MB_KINEMATICS_SETUP_STATUS.....	394
7.5 KinJog.....	394
7.5.1 KinJog/FB.....	394
ML_KinJog.....	394
7.5.2 KinJog/Type.....	397
ML_JOGMODE.....	397
8 Anwendungsbeispiele.....	399
8.1 Anwendungsbeispiele, Übersicht.....	399
8.2 Einsatz von Koordinatensystemen.....	399
8.3 Bandsynchronität.....	400
8.4 Benutzerdefinierte Transformation.....	401
8.5 Erweiterte benutzerdefinierte Transformation.....	405
8.6 Erweiterung eines Punkt.....	410
8.7 Bandsynchronität mit "Adaptive Motion".....	411
8.8 Bestimmung der Kommandopuffergröße.....	412
8.9 Kinematik-Synchronisation mit zyklischer Sollwertvorgabe.....	414
8.10 Sequenzielle Programmierung mit dem Kinematik-Interface.....	417
8.11 Anmeldung einer benutzerdefinierten Transformation mit dem Kinematik-Interface.....	419
9 Parameter.....	421
9.1 Parameter, Allgemeines.....	421
9.2 Achsbezogene Parameter.....	421
9.2.1 Achsbezogene Parameter, Allgemeines.....	421
9.2.2 A-0-0028, Fahrbereichsgrenzschalter.....	421
9.2.3 A-0-0029, Positionspolaritäten.....	421

Inhaltsverzeichnis

	Seite
9.2.4	A-0-0030, Positionsgrenzwert, positiv..... 422
9.2.5	A-0-0031, Positionsgrenzwert, negativ..... 423
9.2.6	A-0-0032, Geschwindigkeitsgrenzwert, positiv..... 423
9.2.7	A-0-0033, Geschwindigkeitsgrenzwert, negativ..... 423
9.2.8	A-0-0034, Beschleunigungsgrenzwert, bipolar..... 423
9.2.9	A-0-0035, Geschwindigkeitsgrenzwert, bipolar..... 424
9.2.10	A-0-0036, Ruckgrenzwert, bipolar..... 424
9.2.11	A-0-0038, Drehmoment-/Kraftgrenzwert, bipolar..... 424
9.2.12	A-0-0039, Drehmoment-/Kraftgrenzwert, positiv..... 425
9.2.13	A-0-0040, Drehmoment-/Kraftgrenzwert, negativ..... 425
9.2.14	A-0-0048, Geschwindigkeitspolaritäten..... 425
9.2.15	A-0-0049, Drehmoment-/Kraft-Polaritäten..... 426
9.2.16	A-0-0224, Schnellhalt-Verzögerung..... 427
9.2.17	A-0-0225, Schnellhalt-Ruck..... 427
9.2.18	A-0-0240, Feedrate-Override..... 427
9.2.19	A-0-0241, Feedrate-Override-Steuerwort..... 427
9.3	Steuerungsbezogene Parameter..... 428
9.3.1	Steuerungsbezogene Parameter, Allgemeines..... 428
9.3.2	C-0-0123, Maximale Anzahl an Kinematiken..... 428
9.3.3	C-0-2480, Konfigurierte Kinematikenanzahl..... 428
9.3.4	C-0-2482, Liste der nicht verknüpften Achsen..... 428
9.3.5	C-0-2483, Liste der konfigurierten Kinematiken..... 428
9.3.6	C-0-2484, Kinematikkonfigurationsliste..... 428
9.4	Kinematikbezogene Parameter..... 429
9.4.1	Kinematikbezogene Parameter, Allgemeines..... 429
9.4.2	K-0-0001, Logische Kinematiknummer..... 429
9.4.3	K-0-0002, Kinematikbezeichnung..... 429
9.4.4	K-0-0003, Maximale Bahngeschwindigkeit..... 429
9.4.5	K-0-0005, Software-Endschalter, positiv, MCS..... 429
9.4.6	K-0-0006, Software-Endschalter, negativ, MCS..... 430
9.4.7	K-0-0007, Achsanzahl der Kinematik..... 430
9.4.8	K-0-0008, Achskonfigurationsliste..... 430
9.4.9	K-0-0009, Oszi: Liste aller oszilloskopierbaren K-Parameter..... 432
9.4.10	K-0-0010, Liste aller K-Parameter..... 432
9.4.11	K-0-0011, Liste aller ungültigen K-Parameter..... 432
9.4.12	K-0-0012, Prüfsumme aller K-Parameter..... 433
9.4.13	K-0-0013, IDN-Liste der K-Prüfsummenparameter..... 433
9.4.14	K-0-0016, Projektidentifikationsnummer..... 433
9.4.15	K-0-0019, Kinematiktyp..... 433
9.4.16	K-0-0020, Kinematikdiagnosemeldung..... 433
9.4.17	K-0-0021, Kinematikstatus..... 434
9.4.18	K-0-0022, Erweiterter Kinematikstatus..... 435
9.4.19	K-0-0023, Kinematikdiagnosenummer..... 435
9.4.20	K-0-0028, Geschwindigkeitslimit Hilfsachsen..... 436
9.4.21	K-0-0029, Beschleunigungslimit Hilfsachsen..... 436
9.4.22	K-0-0030, Rucklimit Hilfsachsen..... 437

Inhaltsverzeichnis

	Seite
9.4.23 K-0-0031, Transformationsschema.....	437
9.4.24 K-0-0034, Maximale Bahnbeschleunigung (bipolar).....	437
9.4.25 K-0-0035, Maximaler Bahnruck (bipolar).....	438
9.4.26 K-0-0037, Ruck für Move-Kommandos während Verzögerungsphase.....	438
9.4.27 K-0-0040, Transformation, Längen der Mechanik.....	438
9.4.28 K-0-0041, Transformation, Faktoren.....	438
9.4.29 K-0-0042, Verschiebung des BCS-Systems.....	438
9.4.30 K-0-0050, Distanzeinheiten MCS.....	439
9.4.31 K-0-0051, Geschwindigkeitseinheiten MCS.....	439
9.4.32 K-0-0052, Beschleunigungseinheiten MCS.....	440
9.4.33 K-0-0053, Ruckeinheiten MCS.....	441
9.4.34 K-0-0054, Maßeinheit für Kinematikdaten.....	441
9.4.35 K-0-0070, Bandanzahl der Kinematik.....	442
9.4.36 K-0-0090, Taskpriorität Satzvorbereitung.....	442
9.4.37 K-0-0095, Bewegungsidentifikationsnummer.....	442
9.4.38 K-0-0096, Höchstwert der Kommandopuffergröße.....	442
9.4.39 K-0-0097, Maximale Kommandopuffergröße.....	442
9.4.40 K-0-0098, Aktuelle Kommandopuffergröße.....	442
9.4.41 K-0-0099, Kinematikkonfiguration.....	442
9.4.42 K-0-0100, Aktuelle Position, Kartesische Maschinenkoordinaten.....	444
9.4.43 K-0-0102, Aktuelle Bahngeschwindigkeit.....	444
9.4.44 K-0-0103, Aktuelle Bahnbeschleunigung.....	444
9.4.45 K-0-0104, Aktueller Bahnruck.....	444
9.4.46 K-0-0150, Interpolierte Sollposition.....	445
9.4.47 K-0-0152, Interpolierte Bahngeschwindigkeit.....	445
9.4.48 K-0-0153, Interpolierte Bahnbeschleunigung.....	445
9.4.49 K-0-0154, Interpolierter Bahnruck.....	445
9.4.50 K-0-0223, Auswahl Fehlerreaktion.....	445
9.4.51 K-0-0224, Nothaltverzögerung.....	446
9.4.52 K-0-0225, Nothaltruck.....	446
9.4.53 K-0-0240, Feedrate-Override.....	446
9.4.54 K-0-0241, Feedrate-Override-Steuerwort.....	446
9.4.55 K-0-0242, Acceleration-Override.....	446
9.4.56 K-0-0243, Deceleration-Override.....	446
9.4.57 K-0-0250, Continuous Slope Vorhaltezeit.....	447
9.4.58 K-0-0815, Kilometerzähler.....	447
9.4.59 K-0-1000, Kommando: Basisparameter der Kinematik laden.....	447
9.4.60 K-0-2100, Liste der zu sichernden K-Parameter.....	447
10 Diagnosen.....	449
11 Anbindung von Robotik-Fremdsystemen.....	451
11.1 Allgemeine Information.....	451
11.2 uniVAL drive.....	451
11.2.1 Übersicht.....	451

Inhaltsverzeichnis

	Seite
11.2.2 Achsen anlegen.....	451
11.2.3 Achsen konfigurieren.....	452
11.2.4 Konfiguration der Kinematiktransformation.....	452
12 Umstieg von Robot-Control V1 auf Robot-Control V2.....	453
12.1 Überblick.....	453
12.2 Anpassung im IndraWorks-Projekt.....	453
12.3 Konvertierung von RCL-Befehlen.....	454
12.4 Konvertierung von RCL-Programmen.....	460
12.5 Konvertierung von IEC-Befehlen.....	464
13 Service und Support.....	467
Index.....	469

1 Zu dieser Dokumentation

1.1 Gültigkeit der Dokumentation

Zielgruppe Diese Dokumentation richtet sich an Anwender, die sich über die Robot-Control V2 der IndraMotion MLC und ihr Kinematik-Interface informieren möchten.

Erforderliche Qualifikationen: Person, die aufgrund ihrer fachlichen Ausbildung, Kenntnisse und Erfahrungen sowie Kenntnis der einschlägigen Normen und Bestimmungen die ihr übertragenen Arbeiten beurteilen und mögliche Gefahren erkennen kann.

Die Dokumentation unterstützt den Anwender in den Phasen:

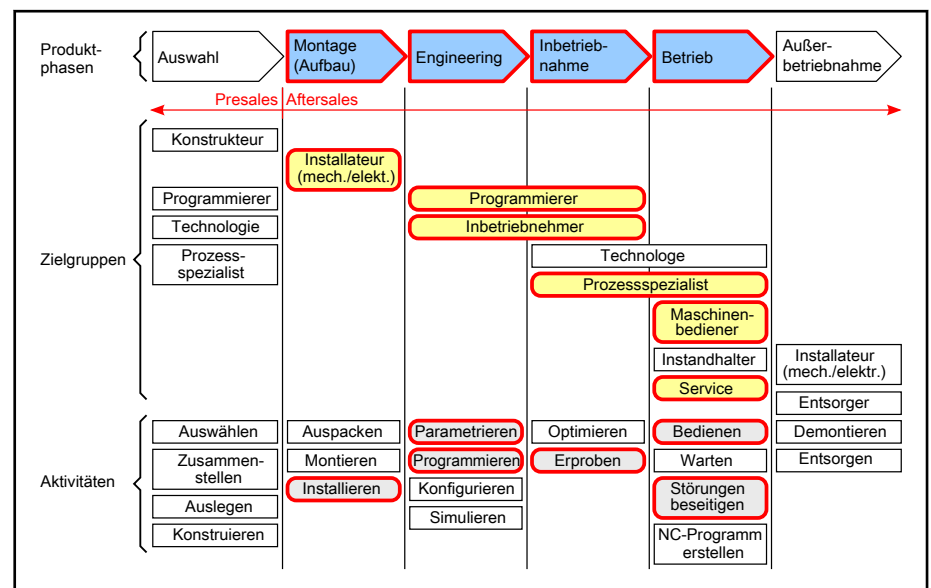


Abb. 1-1: Anwendungsphasen

1.2 Gliederung der Dokumentation

Robot-Control V2 basiert auf den Möglichkeiten der PLCopen / EN 61131-3.

Enthalten sind hierzu

- [Funktionsbeschreibung auf Seite 23](#), Funktionsumfang der Robot-Control V2
- [Oberflächenfunktionen auf Seite 87](#), Beschreibung der IndraWorks-Oberfläche der Komponente "Robot"
- [Transformationsschemen auf Seite 111](#), Beschreibung der Transformationen
- [Anwendungsbeispiele auf Seite 399](#)
- [Kinematik-Interface auf Seite 321](#) mit Bibliothek [ML_KinTech.library](#) auf [Seite 361](#)
- Programmierreferenzen zum detaillierten Nachschlagen

Für **beide Robot-Control Typen** ist die Beschreibung der gemeinsamen Bibliothek [ML_Robot.library](#) auf [Seite 185](#) enthalten.

Zu dieser Dokumentation

1.3 Gebrauch der Sicherheitshinweise

1.3.1 Aufbau der Sicherheitshinweise

Die Sicherheitshinweise sind wie folgt aufgebaut:

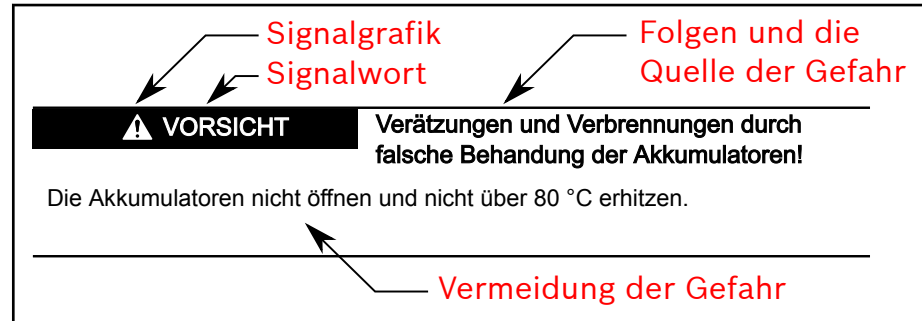


Abb. 1-2: Aufbau der Sicherheitshinweise

1.3.2 Erläuterung der Signalwörter und der Signalgrafik

Die Sicherheitshinweise in der vorliegenden Dokumentation beinhalten bestimmte Signalwörter (Gefahr, Warnung, Vorsicht, Hinweis) und gegebenenfalls eine Signalgrafik (nach ANSI Z535.6-2006).

Das Signalwort soll die Aufmerksamkeit auf den Sicherheitshinweis lenken und bezeichnet die Schwere der Gefährdung.

Die Signalgrafik (Warndreieck mit Ausrufezeichen), welche den Signalwörtern Gefahr, Warnung und Vorsicht vorangestellt wird, weist auf Gefährdungen für Personen hin.

⚠ GEFAHR

Bei Nichtbeachtung dieses Sicherheitshinweises **werden** Tod oder schwere Körperverletzung eintreten.

⚠ WARNUNG

Bei Nichtbeachtung dieses Sicherheitshinweises **können** Tod oder schwere Körperverletzung eintreten.

⚠ VORSICHT

Bei Nichtbeachtung dieses Sicherheitshinweises können mittelschwere oder leichte Körperverletzung eintreten.

HINWEIS

Bei Nichtbeachtung dieses Sicherheitshinweises können Sachschäden eintreten.

1.3.3 Verwendete Symbole

Fingerzeige werden wie folgt dargestellt:



Dies ist ein Hinweis.

Tipps werden wie folgt dargestellt:



Dies ist ein Tipp.

1.3.4 Erläuterung der Signalgrafik auf dem Gerät



Wenn dieses Symbol am Gerät angebracht ist, beachten Sie unbedingt die Dokumentation zu dem Gerät. In der jeweiligen Dokumentation finden Sie die Art der Gefährdung sowie die notwendigen Schritte zur Vermeidung der Gefährdung.

1.4 Erforderliche und ergänzende Dokumentationen XLC/MLC

Dokumentationstitel mit Typenschlüssel und Materialnummer

Systemübersicht
IndraLogic XLC IndraMotion MLC 14VRS Systemübersicht DOK-XLCMLC-SYSTEM**V14-PRRS-DE-P, R911341503 Diese Dokumentation stellt eine Übersicht der möglichen Hardware-/Softwarekomponenten der Automationssysteme IndraLogic XLC und IndraMotion MLC in der genannten Version dar. Die Dokumentation dient als Auswahlhilfe zur Zusammenstellung eines Systems.
<i>Tab. 1-1: XLC/MLC-Dokumentationsübersicht - Systemübersicht</i>
Erste Schritte
IndraMotion MLC 14VRS Erste Schritte DOK-MLC***-F*STEP**V14-CORS-DE-P, R911341516 Diese Dokumentation beschreibt die "ersten Schritte" mit der IndraMotion MLC und RobotControl. Beschrieben werden die Hard- und Softwarevoraussetzungen und das Erstellen eines Projekts.
IndraLogic XLC 14VRS Erste Schritte DOK-XLC***-F*STEP**V14-CORS-DE-P, R911341488 Diese Dokumentation beschreibt die "ersten Schritte" mit der IndraLogic XLC. Beschrieben werden die Hard- und Softwarevoraussetzungen und das Erstellen eines Projekts.
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Projektkonvertierung DOK-XLCMLC-PROCONV*V14-APRS-DE-P, R911341493 Diese Dokumentation beschreibt die Projektkonvertierung der IndraMotion MLC.

Tab. 1-2: XLC/MLC-Dokumentationsübersicht - Erste Schritte

Zu dieser Dokumentation

Engineering
IndraWorks 14VRS Software-Installation DOK-IWORKS-SOFTINS*V14-CORS-DE-P, R911344285 Die Dokumentation beschreibt den Installationsprozess von IndraWorks.
IndraWorks 14VRS Engineering DOK-IWORKS-ENGINEE*V14-APRS-DE-P, R911343564 Diese Dokumentation beschreibt die Verwendung von IndraWorks, in das sich die Rexroth Engineering-Tools integrieren. Beschrieben werden das Arbeiten mit IndraWorks und die Bedienung der Oszilloskop-Funktion.
IndraLogic XLC IndraMotion MLC 14VRS Funktionsbeschreibung DOK-XLCMLC-FUNC****V14-APRS-DE-P, R911341699 Die Dokumentation dient der Beschreibung der Wizards, Kontextmenüs, Dialoge, Steuerungsinbetriebnahme, Gerätekonfiguration, Funktionalitäten der IndraMotion MLC.

Tab. 1-3: XLC/MLC-Dokumentationsübersicht - Engineering

Diagnose und Service
IndraLogic XLC IndraMotion MLC 14VRS Diagnosen DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480 Diese Dokumentation enthält alle in den Steuerungssystemen IndraLogic XLC und IndraMotion MLC implementierten Steuerungsdiagnosen.
IndraLogic XLC IndraMotion MLC 14VRS Parameter DOK-XLCMLC-PARAM***V14-RERS-DE-P, R911341478 Diese Dokumentation beschreibt die Parameter der Systeme XLC/MLC, sowie das Wechselspiel zwischen Parametrieren und Programmieren. Beschrieben werden die Achs-Parameter, die Steuerungs-Parameter, die Kinematik-Parameter, die Messtaster- und die Nockenschaltwerks-Parameter.
IndraLogic XLC IndraMotion MLC 14VRS Inbetriebnahme DOK-XLCMLC-STARTUP*V14-CORS-DE-P, R911341500 Das vorliegende Handbuch beschreibt die Inbetriebnahme- und Servicearbeitsschritte für die Systeme IndraLogic XLC und IndraMotion MLC. Es beinhaltet Checklisten für häufig durchzuführende Arbeiten und eine ausführliche Beschreibung der Arbeitsschritte.

Zu dieser Dokumentation

Diagnose und Service
WebAssistant DOK-GENERL-WEB*ASSIST*-APRS-DE-P, R911381468 Diese Dokumentation beschreibt den WebAssistant. Der WebAssistant ist ein webbasiertes Diagnosewerkzeug, das zum Zugriff auf ein Steuerungssystem über eine Ethernet-Hochgeschwindigkeitsverbindung dient. Der WebAssistant ermöglicht OEMs, Endanwendern und Kundenbetreuern von überall auf ein System zuzugreifen und Ferndiagnosen zu stellen. Voraussetzung ist ein PC mit Internet Explorer Version 11 oder neuer, Firefox Version 45 oder neuer oder Google Chrome Version 54 oder neuer. Die folgenden Steuerungsvarianten werden unterstützt: • IndraMotion MLC L25/L45/L65/L75/XM2 • IndraLogic XLC L25/L45/L65/L75/VEP
IndraWorks IndraMotion Service Tool DOK-IWORKS-IMST*****-APRS-DE-P, R911341382 Diese Dokumentation beschreibt das IndraMotion Service Tool (IMST). Das IMST ist ein webbasiertes Diagnosewerkzeug, das zum Zugriff auf ein Steuerungssystem über eine Ethernet-Hochgeschwindigkeitsverbindung dient. Das IMST ermöglicht OEMs, Endanwendern und Kundenbetreuern von überall auf ein System zuzugreifen und Ferndiagnosen zu stellen. Voraussetzung ist ein PC mit Internet Explorer Version 8 oder Firefox 3.5 oder eine höhere Version. Die folgenden Steuerungs-Varianten werden unterstützt: • IndraMotion MLC L25/L45/L65/L75/XM12/XM2/VPx • IndraLogic XLC L25/L45/L65/L75

Tab. 1-4: XLC/MLC-Dokumentationsübersicht - Diagnose und Service

Firmware
Rexroth IndraDrive MPx-20, Funktionen ¹⁾ DOK-INDRV*-MP*-20VRS**-AP01-DE-P, R911345607 Diese Dokumentation beschreibt alle funktionellen Eigenschaften der IndraDrive-Firmware in den Varianten MPB-20, MPM-20, MPC-20 und MPE-20.
Rexroth IndraDrive MPx-16 bis MPx-20 und PSB, Parameter DOK-INDRV*-GEN1-PARA**-RERS-DE-P, R911328650 Diese Dokumentation enthält die Beschreibung aller, in der Firmware für Antriebsregelgeräte der IndraDrive-Familie implementierten Parameter. Sie unterstützt die Parametrierung der Antriebsregelgeräte.
Rexroth IndraDrive MPx-16 bis MPx-20 und PSB, Diagnosen DOK-INDRV*-GEN1-DIAG**-RERS-DE-P, R911326539 Diese Dokumentation enthält die Beschreibung aller in den folgenden Firmwares implementierten Diagnosen: • Antriebsregelgerät-Firmwares MPx-16 bis MPx-20 • Firmwares PSB der Versorgungsgeräte des Typs "HMU05" und "KMV03" • Firmwares der Versorgungsgeräte des Typs "HMOV01". Sie unterstützt den Maschinenbediener und den Anlagenprogrammierer bei der Fehlerbehebung.
Hardware
Rexroth IndraDrive Steuerteile CSB02, CSE02, CSH02, CDB02 DOK-INDRV*-Cxx02*****-PRRS-DE-P, R911338961
Rexroth IndraDrive Cs Antriebssysteme mit HCS01 DOK-INDRV*-HCS01*****-PRRS-DE-P, R911322209

Zu dieser Dokumentation

Rexroth IndraDrive Mi Antriebssysteme mit KCU02, KSM02, KMS02/03, KMV03

DOK-INDRV*-KCU02+KSM02-PRxx-DE-P, R911335702

Rexroth IndraDrive ML Antriebssysteme mit HMU05

DOK-INDRV*-Hxx05*****-PRxx-DE-P, R911344278

1) in Vorbereitung
Tab. 1-5: Übersicht der Dokumentationen für Antriebsregelgeräte

SPS

IndraWorks 14VRS SPS-Programmiersystem IndraLogic 2G

DOK-IWORKS-IL2GPRO*V14-APRS-DE-P, R911343570

Diese Dokumentation beschreibt das SPS-Programmierungswerkzeug IndraLogic 2G und seine Verwendung. Beschrieben werden die Basisverwendung, Erste Schritte, die Visualisierung, Menüpunkte und die Editoren.

IndraWorks 14VRS Basis-Bibliotheken IndraLogic 2G

DOK-IL*2G*-BASLIB**V14-LIRS-DE-P, R911343918

Das vorliegende Handbuch beschreibt die systemübergreifenden SPS-Bibliotheken.

IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken

DOK-XLCMLC-FUNLIB**V14-LIRS-DE-P, R911341490

Diese Dokumentation beschreibt die Funktionsbausteine, Funktionen und Datentypen der Bibliotheken RIL_CommonTypes, ML_Base und ML_PLCopen für die IndraLogic XLC/IndraMotion MLC. Die Dokumentation enthält ferner die Fehlerreaktion der Bausteine.

Tab. 1-6: XLC/MLC-Dokumentationsübersicht - SPS

Feldbusse

IndraWorks 14VRS Feldbusse

DOK-IWORKS-FB*****V14-APRS-DE-P, R911341484

Die Dokumentation dient der Beschreibung der von den Systemen IndraLogic XLC, IndraMotion MLC und IndraMotion MTX unterstützten Feldbus- und lokalen Peripherieanschlüssen. Der Schwerpunkt dieser Dokumentation liegt in der Konfiguration, Parametrierung, Inbetriebnahme und Diagnose der verschiedenen Peripherieanschlüssen. Sie ist die Grundlage für die Online-Hilfe.

IndraWorks 14VRS Feldbus-Bibliotheken

DOK-IWORKS-FB*LIB**V14-LIRS-DE-P, R911343574

Das vorliegende Handbuch beschreibt die Bibliotheken der Feldbusse: RIL_ProfibusDP_02, RIL_ProfibusDP Slave, RIL_ProfinetIO, RIL_ProfinetIODevice, RIL_EtherNetIPAdapter, RIL_MappingList, RIL_SERCOSIII, RIL_InLine, einschließlich deren Diagnosen, sowie die Fehlerreaktion der Bausteine.

Rexroth IndraWorks 12VRS FDT-Container

DOK-IWORKS-FDT*CON*V12-APRS-DE-P, R911334397

Diese Dokumentation beschreibt die FDT-Container-Funktionalität von IndraWorks. Beschrieben wird das Aktivieren der Funktionalität im Projekt und das Arbeiten mit DTMs.

Sercos Systemhandbuch für I/O-Geräte

DOK-CONTRL-ILS3*****-APxx-DE-P, R911333511

Diese Dokumentation beschreibt die Projektierung, Parametrierung, Inbetriebnahme und Diagnose von I/O-Geräten mit Sercos Schnittstelle.

xx entsprechender Ausgabestand
Tab. 1-7: XLC/MLC-Dokumentationsübersicht - Feldbusse

Zu dieser Dokumentation

HMI
IndraWorks 14VRS HMI DOK-IWORKS-HMI****V14-APRS-DE-P, R911343567 Diese Dokumentation beschreibt die Funktionen, die Konfiguration und die Bedienung der Bedienoberflächen IndraWorks HMI Engineering und IndraWorks HMI Operation.
IndraWorks 14VRS WinStudio 7.4 DOK-IWORKS-WINSTUD*V14-APRS-DE-P, R911341584 Dieses "Benutzerhandbuch und Technische Referenz" soll Sie dabei unterstützen, mit Ihrer Software "Rexroth WinStudio" optimale Ergebnisse zu erzielen. Das Dokument bietet technische Informationen und Schritt-für-Schritt-Anleitungen zum Anlegen von webfähigen HMI/SCADA-Programmen.
Rexroth IndraLogic XLC IndraMotion MLC 14VRS HMI-Anbindung DOK-XLCMLC-HMI****V14-APRS-DE-P, R911341496 Dieses Dokument beschreibt die von der IndraLogic XLC und IndraMotion MLC unterstützten Visualisierungssysteme sowie deren Anbindung.

Tab. 1-8: XLC/MLC-Dokumentationsübersicht - HMI

Technologie
IndraLogic XLC IndraMotion MLC 14VRS Generic Application Template DOK-XLCMLC-TF*GAT**V14-APRS-DE-P, R911341486 Dieses Handbuch stellt dem IndraLogic SPS-Programmierer ein strukturiertes Template, welches zum Hinzufügen und Bearbeiten des SPS-Programmierscodes verwendet werden kann, zur Verfügung. Beschrieben werden das Template selbst, der Template Wizard und Beispielapplikationen.
IndraMotion MLC 14VRS Technologie-Bibliotheken DOK-MLC***-TF*LIB**V14-LIRS-DE-P, R911341510 Diese Dokumentation beschreibt die Funktionsbausteine, Funktionen und Datentypen der Bibliotheken "ML_TechInterface.library", "ML_TechMotion.library", "RMB_TechCam.library" und "ML_TechBase.library" sowie Bibliotheken zur Wicklerfunktionalität, Registerreglerfunktionalität und Querschneiderfunktionalität.
IndraWorks 14VRS Energy Efficiency Management DOK-IWORKS-4EE****V14-APRS-DE-P, R911339228 Diese Dokumentation beschreibt die HMI-Anbindung des Energiemanagement-Systems und die Funktionsbausteine, Funktionen und Datentypen der Bibliothek "RIL_4EE ". Die Dokumentation enthält ferner die Fehlerreaktion der Bausteine.
Rexroth IndraMotion MLC 14VRS RegisterControl (Bibliothek) DOK-MLC***-REGI*CO*V14-LIRS-DE-P, R911341508 Diese Dokumentation beschreibt die Ein- und Ausgänge der einzelnen Funktionsbausteine und gibt Hinweise für deren Verwendung.
IndraMotion MLC 14VRS RegisterControl (Anwendungsbeschreibung) DOK-MLC***-REGI*CO*V14-APRS-DE-P, R911341506 Dieses Dokument beschreibt die Anwendung der integrierten Registerregelung für eine Tiefdruckmaschine. Beschrieben werden die Komponenten des Markenstromsensors, die HMI-Anwendung und die Fehlerbehebung. Diese Anleitung dient dazu, die Registerregelung zu bedienen und auf Fehler zu reagieren sowie Diagnosen abzufragen. Die Zielgruppe für diese Dokumentation sind Maschineneinrichter und Maschinenbediener.
Rexroth IndraMotion MLC 14VRS Anwendung Wicklerfunktionen DOK-MLC***-TF*WIND*V14-APRS-DE-P, R911341512 Diese anwendungsbezogene Systemdokumentation beschreibt die Anwendung der Wickler-Technologiefunktionen.

Zu dieser Dokumentation

Technologie
Rexroth IndraWorks 13VRS CamBuilder DOK-IWORKS-CAMBUIL*V13-APRS-DE-P, R911336290 Diese Dokumentation dient zur Beschreibung der Grundlagen und der Bedienung des CamBuilder, des Werkzeugs zur Kurvenscheibenbearbeitung.
IndraMotion MLC 14VRS Robot-Control V2 DOK-MLC***-ROCO****V14-RERS-DE-P, R911341586 Die Dokumentation informiert über die Robot-Control V2. Die Programmierung über PLCopen steht hierbei im Vordergrund. Beschrieben werden die Programmstruktur, Variablen, Funktionen, Bewegungsanweisungen sowie die notwendigen Parameter des Systems. Die Dokumentation enthält ferner die Beschreibung der Bibliothek "ML_Robot". Robot-Control V2 wird unterstützt durch ein Kinematik-Interface. Die Komponenten dieses Interfaces werden in der Bibliothek "ML_KinTech" beschrieben.

Tab. 1-9: XLC/MLC-Dokumentationsübersicht - Technologie

Open Core Engineering - OCE
IndraLogic XLC IndraMotion MLC 14VRS Erste Schritte MLPI (Onlinehilfe) DOK-XLCMLC-MLPI****V14-CORS-EN-E, R911341033 Dieses Dokument beschreibt die Installation und Inbetriebnahme der MLPI-Schnittstelle.
IndraLogic XLC IndraMotion MLC 14VRS Automation Interface (Onlinehilfe) DOK-XLCMLC-AUT*INT*V14-APRS-EN-E, R911341499 Diese Dokumentation beschreibt den script-basierten Zugriff auf IndraWorks-Projektdateien über die Schnittstelle des Automation Interfaces.

Tab. 1-10: XLC/MLC-Dokumentationsübersicht - Open Core Engineering - OCE

SafeLogic
IndraWorks SafeLogic 14VRS Die ersten Schritte DOK-IWORKS-SL*STEP*V14-CORS-DE-P, R911341519 Dieses Dokument beschreibt die Erstinbetriebnahme des Safety-Funktionsmoduls und dessen externer Safety-Peripherie. Am Beispiel eines Projektes werden alle notwendigen Schritte zur Inbetriebnahme einer einfachen Safety-Anwendung durchgeführt.
IndraLogic XLC IndraMotion MLC 14VRS SafeLogic Systemübersicht DOK-XLCMLC-SL**SYS*V14-PRRS-DE-P, R911341695 Dieses Dokument beschreibt die Projektierung, die Erstellung und die Inbetriebnahme von sicherheitstechnischen Vorrichtungen.
IndraWorks 14VRS SafeLogic - Projektkonfiguration, Stand 1.2.0.0. DOK-IWORKS-SL**PRJ*V14-APRS-DE-P, R911341693 Dieses Dokument beschreibt die Projektierung von Safety-Anwendungen.
IndraControl CFL01.1-F1 Safety-Funktionsmodul DOK-CTRL-SL**FM*****-ITRS-DE-P, R911336575 Dieses Dokument liegt der Hardware bei und beschreibt die Inbetriebnahme, die Installation, die Außerbetriebnahme und Entsorgung der Hardware.

Tab. 1-11: XLC/MLC-Dokumentationsübersicht - SafeLogic

Zu dieser Dokumentation

Hardware
Rexroth IndraControl L45/L65/L85 Steuerung DOK-CTRL-ICL45L65L85-PRxx-DE-P, R911332115 Diese Dokumentation dient der Beschreibung der Steuerungen IndraControl L45/L65/L85.
Rexroth IndraControl L25 DOK-CTRL-IC*L25****-PRxx-DE-P, R911328473 Diese Dokumentation dient der Beschreibung der Steuerungen IndraControl L25.
Rexroth IndraControl Lxx 14VRS Funktionsmodule DOK-CTRL-FM*LXX**V14-APRS-DE-P, R911341582 Diese Dokumentation dient der Beschreibung aller Funktionsmodule der Lxx Steuerungen inkl. Engineering und Diagnose.
IndraControl XM21, XM22 Steuerungen DOK-CTRL-IC*XM2****-ITRS-DE-P, R911340666 Diese Dokumentation dient der Beschreibung der Steuerungen IndraControl XM21/XM22 Steuerungen.
IndraControl XFE 01.1 Erweiterungsmodule Profibus, RT-Ethernet, Sercos, CAN DOK-CTRL-XFE**EXTMOD-ITRS-DE-P, R911345569 Diese Dokumentation dient der Beschreibung aller Erweiterungsmodule der XM-Steuerungen.

xx entsprechender Ausgabestand
Tab. 1-12: XLC/MLC-Dokumentationsübersicht - Hardware

Hydraulik
IndraMotion MLC 14VRS Hydraulische Funktionen DOK-MLC***-TF*HMOT*V14-LIRS-DE-P, R911341905 Diese Dokumentation beschreibt die Funktionsbausteine, Funktionen und Datentypen der Bibliotheken MH_TechHydrBase und MH_TechHydrMotion. Die Dokumentation enthält ferner die Fehlerreaktion der Bausteine.
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Sequentielle Programmierung DOK-XLCMLC-SEQPROG*V14-LIRS-DE-P, R911341908 Diese Dokumentation beschreibt die Möglichkeiten der Sequentiellen Programmierung, auch in Verbindung mit GAT, und die Bibliothek MH_HydrControlProg.

Tab. 1-13: XLC/MLC-Dokumentationsübersicht - Hydraulik

1.5 Bezeichnungen und Abkürzungen

Begriff	Erklärung
ACS	Achs-Koordinatensystem = Axes Coordinate System
BCS	Basis-Koordinatensystem = Base Coordinate System
GAT	Generic Application Template, Applikationsprogramme für Motion und Logic Applikationen
MCS	Maschinen-Koordinatensystem = Machine Coordinate System
PCS	Produkt-Koordinatensystem = Product Coordinate System
PTP	Punkt zu Punkt = Point To Point
RCL	Programmiersprache Robot Control Language (Robot-Control V1)
TCP	Werkzeug-Arbeitspunkt = Tool Center Point
MLPI	Motion Logic Programming Interface

Tab. 1-14: Verwendete Bezeichnungen und Abkürzungen

Zu dieser Dokumentation

1.6 Kundenfeedback

Anregungen, Wünsche oder Verbesserungen von unseren Kunden haben bei uns einen hohen Stellenwert. Senden Sie uns Ihre Anmerkungen zu den Dokumentationen per E-Mail an Feedback.Documentation@boschrexroth.de. Sie können direkt im elektronischen PDF-Dokument Kommentare einfügen und uns die PDF-Datei zusenden.

2 Funktionsbeschreibung

2.1 Systemübersicht

Die IndraMotion MLC stellt neben Einzelachsbewegungen auch koordinierte Multiachsbewegungen im dreidimensionalen Raum zur Verfügung.

Mit Einzelachsbewegungen sind nur koordinierte Bewegungen mehrerer Achsen möglich, indem aus der Position einer Masterachse die Positionen der Slaveachsen berechnet werden.

Mehrdimensionale Bewegungen gehen über diesen Punkt hinaus. Hier werden Achsen zu einer Kinematik zusammengefasst. Eine zusätzliche Masterachse ist nicht notwendig. Für diese Kinematik wird eine Bahn im Raum definiert.

Auf diese Weise ist eine flexiblere Bahnplanung möglich, als es durch einfache Master-Slave-Beziehungen möglich wäre. Weiterhin können bei Master-Slave-Beziehungen Probleme auftreten, wenn ein Fehler bei einer Slave-Achse auftritt und die anderen Achsen keine Kenntnis darüber haben und weiter fahren. Eine Kinematik hat Kenntnis darüber, welche Achsen zur Gruppe gehören und daher eine bessere Basis für eine Fehlerreaktion.



Die Robot-Control-Funktionalität steht nur bei einer IndraMotion MLC zur Verfügung.

Die IndraMotion MLC ist in der Lage, mehrere Kinematiken gleichzeitig zu berechnen.

Zahl der Kinematiken, abhängig vom Steuerungstyp:

- IndraMotion MLC L25: 4 Kinematiken
- IndraMotion MLC L45: 16 Kinematiken
- IndraMotion MLC L65: 16 Kinematiken
- IndraMotion MLC L75: 16 Kinematiken

Zu einer Kinematik können bis zu 16 Achsen gehören.



Es existieren zwei Versionen der Robot-Control:

- Robot-Control V1
- Robot-Control V2

Bis einschließlich Version 13VRS der MLC-Firmware wird Robot-Control V1 unterstützt.

Ab Version 13VRS der MLC-Firmware wird Robot-Control V2 unterstützt.

Bei neuen Projekten sollte nur noch Robot-Control V2 verwendet werden. Robot-Control V1 ist in Version 13VRS nur aus Kompatibilitätsgründen vorhanden.

Für den Umstieg von Robot-Control V1 auf Robot-Control V2 siehe [Umstieg von Robot-Control V1 auf Robot-Control V2 auf Seite 453](#).

Funktionsbeschreibung

2.2 Koordinatensysteme und Transformationen

2.2.1 Koordinatensysteme

Koordinatensysteme, Übersicht

Die Position einer Kinematik im Raum ist über verschiedene Koordinatensysteme darstellbar. Zu jeder Position muss immer das zugehörige Koordinatensystem angegeben werden.

Koordinatensystem	Beschreibung
Achs-Koordinatensystem (ACS) auf Seite 26	Bezug zur Achse
Basis-Koordinatensystem (BCS) auf Seite 26	Bezug zur Kinematik
Maschinen-Koordinatensystem (MCS) auf Seite 26	Bezug zur Maschinenanlage
Produkt-Koordinatensystem (PCS) auf Seite 27	Bezug zum Produkt bzw. Werkstück

Tab. 2-1: Übersicht der Koordinatensysteme

Koordinatensystemtypen

Kartesisches Koordinatensystem

Koordinatensysteme können in folgende Typen unterteilt werden:

Das kartesische Koordinatensystem hat folgende Eigenschaften:

- **Kartesisch:** Die drei Koordinatenachsen stehen orthogonal (rechtwinklig) aufeinander
- **Dreidimensional:** Es existieren drei Koordinatenachsen. Ein Punkt im Raum wird durch drei Werte (X, Y, Z) im Raum angegeben
- **Rechtshändig:** Spannt man mit der rechten Hand ein dreidimensionales kartesisches Koordinatensystem auf, so zeigt der Daumen in Richtung der X-Achse, der Zeigefinger in Richtung der Y-Achse und der Mittelfinger in Richtung der Z-Achse

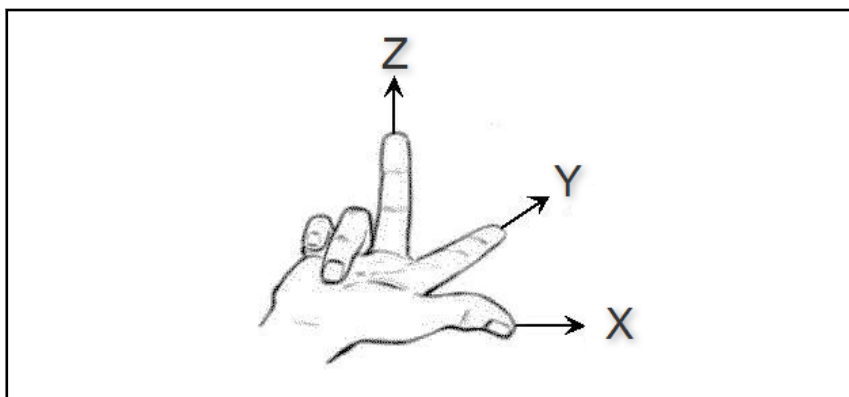


Abb. 2-1: Rechtshändiges Koordinatensystem

Neben der Position im Raum wird die Orientierung eines Objekts durch die Verdrehung um die einzelnen Achsen angegeben.

Hierbei ist A die Verdrehung um die X-Achse, B die Verdrehung der aus der Verdrehung um die X-Achse resultierende Y-Achse und C die Verdrehung der aus der Verdrehung um die X-Achse und Y-Achse resultierende Z-Achse.

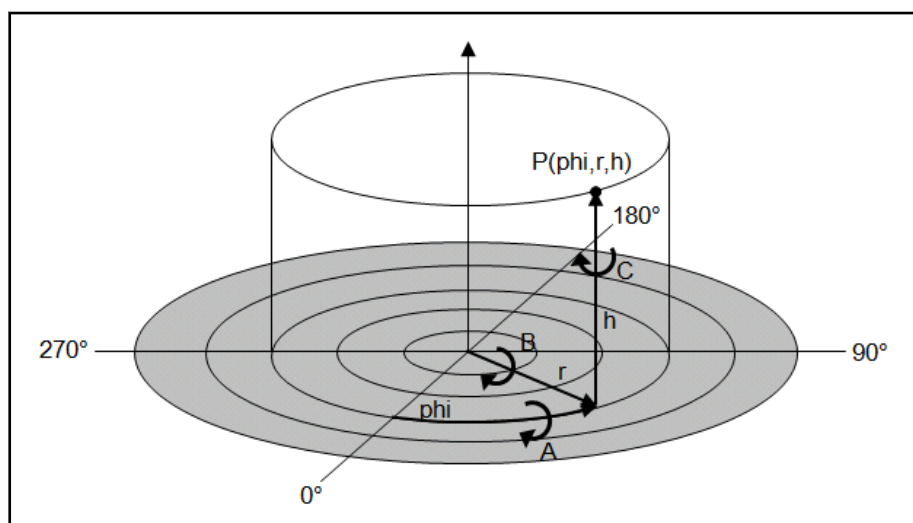
Die Position eines Objektes wird folglich durch das 6-Tupel (X, Y, Z, A, B, C) angegeben.

Zylindrisches Koordinatensystem

In einem zylindrischen Koordinatensystem wird die Position eines Objekts durch drei Koordinaten angegeben. Jedoch liegen die Koordinatenachsen

Funktionsbeschreibung

nicht orthogonal aufeinander wie beim kartesischen Koordinatensystem. Die Position eines Objekts im Zylinder-Koordinatensystem ist definiert durch (ϕ, r, h) .



- phi** Winkel des Objekts im mathematisch positiven Drehsinn (linksdrehend) von der 0°-Gerade (Winkelkoordinate)
- r** Abstand des Objekts von der Höhenachse (Radialkoordinate)
- h** Höhe des Objekts von der Ebene, die durch die phi- und r-Koordinatenachse aufgespannt wird (Höhenkoordinate)
- A** Verdrehung um die Winkelachse
- B** Verdrehung um die Radialachse
- C** Verdrehung um die Höhenachse

Abb. 2-2: Zylindrisches Koordinatensystem

Die Position eines Objektes wird folglich durch das 6-Tupel (ϕ, r, h, A, B, C) angegeben.

Achsbezogenes Koordinatensystem

In einem achsbezogenen Koordinatensystem wird die Position und die Orientierung eines Objektes durch die Stellung der Achsen angegeben.

Für jede Achse einer Kinematik muss eine Koordinate angegeben werden.

Beispiel:

Achsbezogenes Koordinatensystem einer 2-Achs-Kinematik

Die Position einer 2-Achs-Kinematik mit rotatorischen Achsen wird mit den Winkeln α und β als Koordinaten (α, β) angegeben.

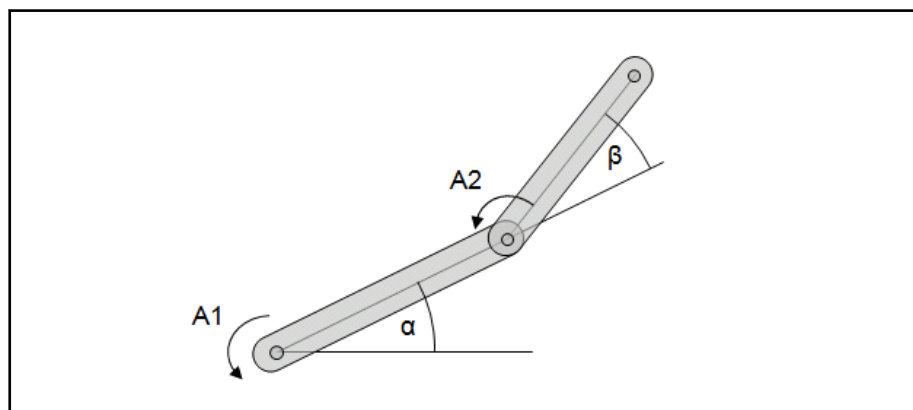


Abb. 2-3: Achsbezogenes Koordinatensystem einer 2-Achs-Kinematik

Funktionsbeschreibung

Achs-Koordinatensystem (ACS)

Das Achs-Koordinatensystem (ACS = **A**xes **C**oordinate **S**ystem) ist ein achs-bezogenes Koordinatensystem.

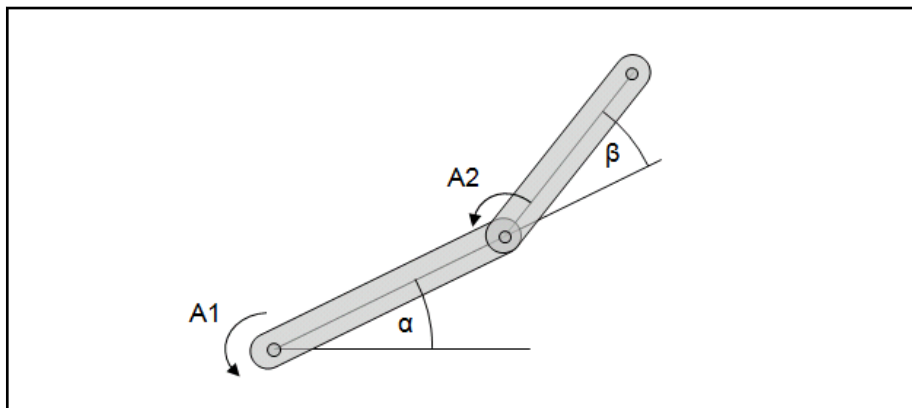


Abb. 2-4: ACS

Basis-Koordinatensystem (BCS)

Das Basis-Koordinatensystem (BCS = **B**ase **C**oordinate **S**ystem) ist das Koordinatensystem der Kinematik.

Das BCS ist ein kartesisches Koordinatensystem.

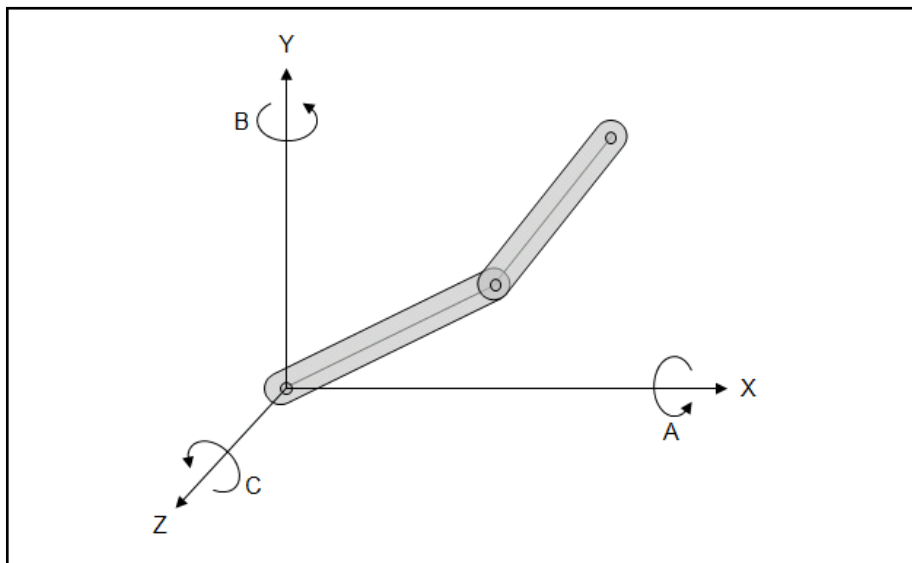


Abb. 2-5: BCS

Maschinen-Koordinatensystem (MCS)

Das Maschinen-Koordinatensystem (MCS = **M**achine **C**oordinate **S**ystem) ist das Koordinatensystem der Maschinenanlage.

Das MCS ist das gemeinsame Koordinatensystem aller Kinematiken (BCS).

Das MCS ist ein kartesisches Koordinatensystem.

Funktionsbeschreibung

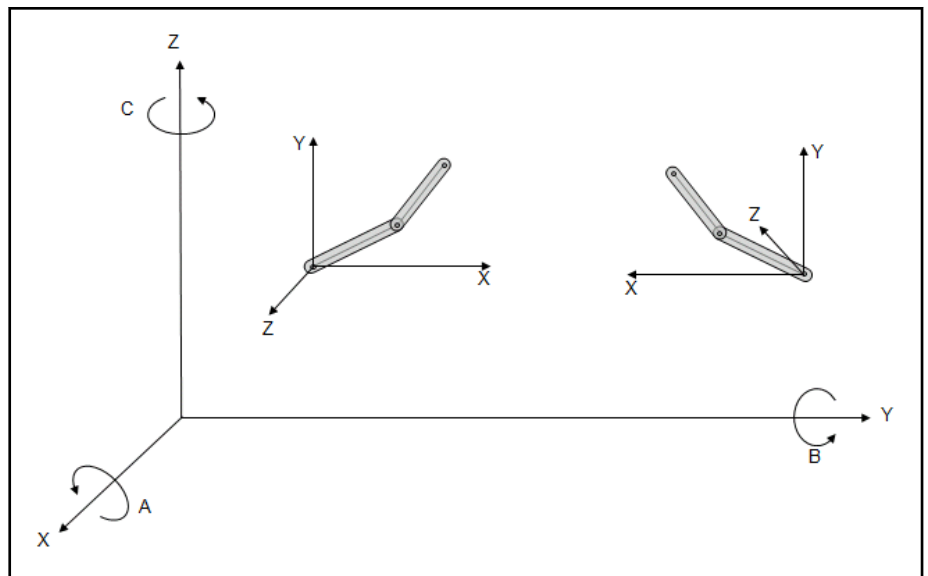


Abb. 2-6: MCS

Produkt-Koordinatensystem (PCS)

Das Produkt-Koordinatensystem (PCS = **P**roduct **C**oordinate **S**ystem) ist das Koordinatensystem des Produktes.

Ein bandsynchroner Arbeitsbereich wird ebenfalls durch ein PCS definiert (siehe [Bandsynchrone Bewegung, Allgemeines auf Seite 67](#)).

Pro Kinematik können 16 PCS angegeben werden.

Das PCS ist entweder ein kartesisches oder ein zylindrisches Koordinatensystem.

Der Einsatz von Produkt-Koordinatensystemen im Zusammenhang mit Bandsynchronität ist beschrieben in [Bandsynchronität auf Seite 400](#).

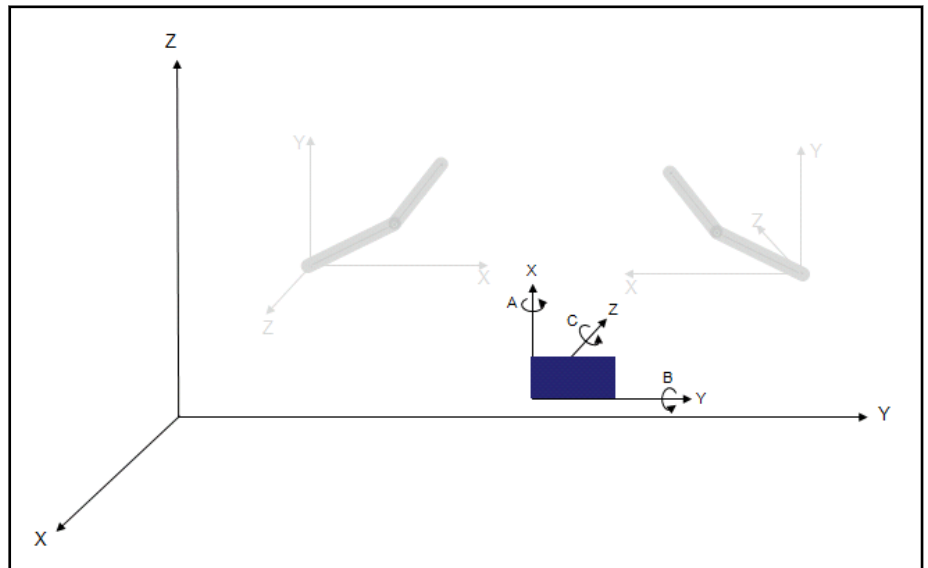


Abb. 2-7: PCS (kartesisch)

Funktionsbeschreibung

2.2.2 Transformationen

Transformationen, Übersicht

Positionen können von einem Koordinatensystem in ein anderes Koordinatensystem mittels Transformationen umgerechnet werden.

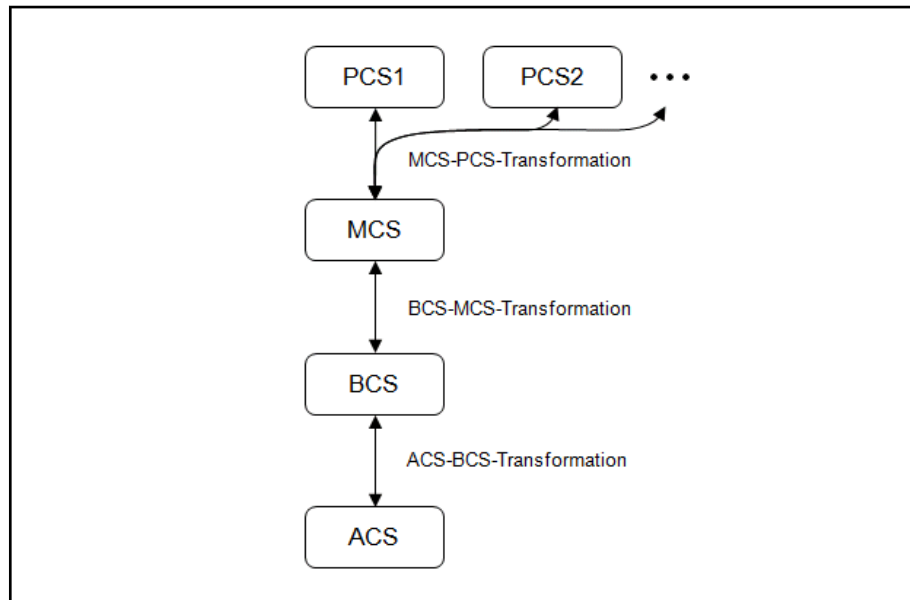


Abb. 2-8: Übersicht der Transformationen

ACS-BCS-Transformation

Die ACS-BCS-Transformation rechnet Positionen zwischen dem ACS und BCS um.

Das Transformationsschema der ACS-BCS-Transformation wird im Parameter **K-0-0031** (Transformationsschema) angegeben (z. B. Karteser, Scara,...) und im Parameter **K-0-0040** (Transformation, Längen der Mechanik) und **K-0-0041** (Transformation, Faktoren) konfiguriert.

Eine Liste der verfügbaren Transformationsschemen befindet sich in: [Transformationsschemen auf Seite 111](#).

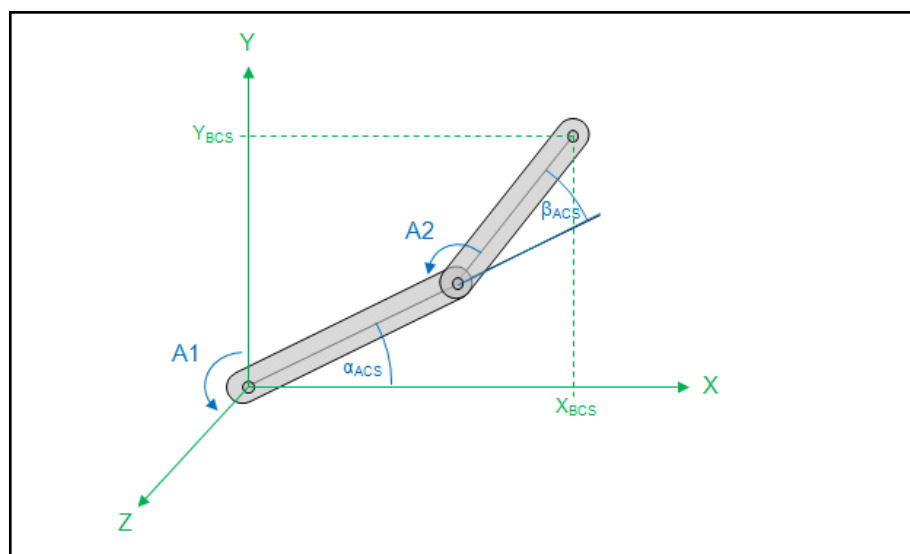


Abb. 2-9: ACS-BCS

Beispiel:

K-0-0031 = 9 (Scara)

K-0-0040 = 100, 200 (Armlänge L1, L2)

ACS: $(\alpha_{ACS}, \beta_{ACS}) = (90^\circ, -90^\circ)$

BCS: $(X_{BCS}, Y_{BCS}, Z_{BCS}, A_{BCS}, B_{BCS}, C_{BCS}) = (200, 100, 0, 0, 0, 0)$

BCS-MCS-Transformation

Die BCS-MCS-Transformation rechnet Positionen zwischen dem BCS und MCS um.

Die Verdrehung und Verschiebung zwischen BCS und MCS wird im Parameter **K-0-0042** (Verschiebung des BCS-Systems) konfiguriert.

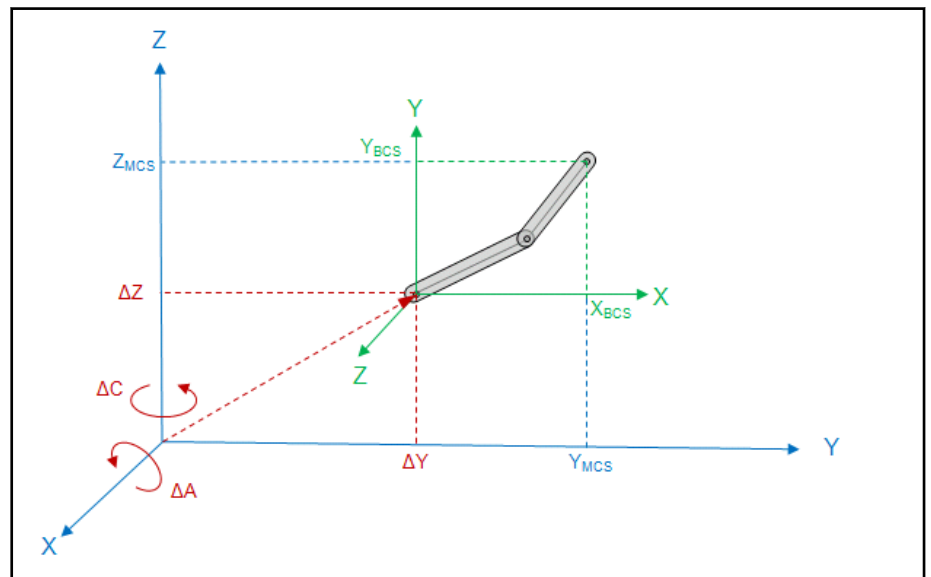


Abb. 2-10: BCS-MCS

Beispiel:

K-0-0042 = $(\Delta X, \Delta Y, \Delta Z, \Delta A, \Delta B, \Delta C) = (0, 111, 499, 0^\circ, 0^\circ, 90^\circ)$

BCS: $(X_{BCS}, Y_{BCS}, Z_{BCS}, A_{BCS}, B_{BCS}, C_{BCS}) = (200, 100, 0, 0^\circ, 0^\circ, 0^\circ)$

MCS: $(X_{MCS}, Y_{MCS}, Z_{MCS}, A_{MCS}, B_{MCS}, C_{MCS}) = (-100, 311, 499, 0^\circ, 0^\circ, 90^\circ)$



Die Verdrehung und Verschiebung zwischen BCS und MCS kann mit dem Kommando [Get Trafo Values auf Seite 38](#) ausgelesen werden.

MCS-PCS-Transformation

Die MCS-PCS-Transformation rechnet Positionen zwischen dem MCS und PCS um.

Die Verdrehung und Verschiebung zwischen MCS und PCS wird über die nachstehenden Kommandos konfiguriert:

- [Set Cartesian Transform auf Seite 38](#), bei einem kartesischen PCS
- [Set Cylindric Transform auf Seite 38](#), bei einem zylindrischen PCS

Funktionsbeschreibung

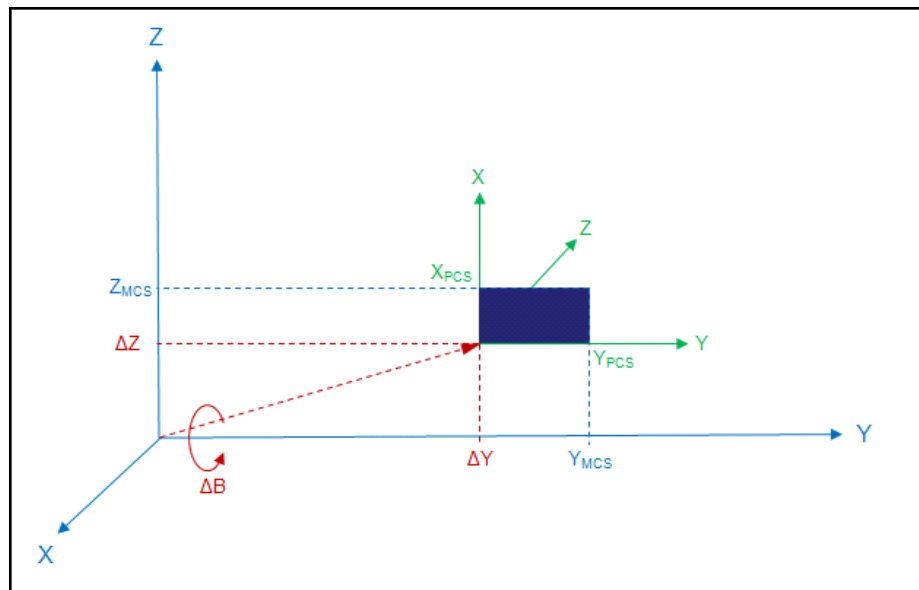


Abb. 2-11: MCS-PCS

Beispiel:

Set Cartesian Transform = $(\Delta X, \Delta Y, \Delta Z, \Delta A, \Delta B, \Delta C) = (0, 111, 499, 0^\circ, -90^\circ, 0^\circ)$

MCS: $(X_{MCS}, Y_{MCS}, Z_{MCS}, A_{MCS}, B_{MCS}, C_{MCS}) = (0, 211, 699, 0^\circ, 0^\circ, 0^\circ)$

PCS: $(X_{PCS}, Y_{PCS}, Z_{PCS}, A_{PCS}, B_{PCS}, C_{PCS}) = (200, 100, 0, 0^\circ, 90^\circ, 0^\circ)$



Die Verdrehung und Verschiebung zwischen MCS und PCS können mit dem Kommando [Get Trafo Values](#) auf Seite 38 ausgelesen werden.

2.3 Achstypen

2.3.1 Achstypen, Überblick

Die MLC Robot-Control unterscheidet die folgenden Achstypen:

- Positionierungsachsen
- Orientierungsachsen
- Hilfsachsen
- Bandachsen

2.3.2 Positionierungsachsen

Positionierungsachsen bestimmen die Position im Raum. Durch sie werden die Koordinaten X, Y, Z bestimmt.

Sie bestimmen maßgeblich die Bahninterpolation im Raum.

Es können maximal 3 Positionierungsachsen pro Kinematik angelegt werden.

2.3.3 Orientierungsachsen

Orientierungsachsen bestimmen die Orientierung im Raum. Durch sie werden die Verdrehungen A, B, C bestimmt.

Sie bestimmen maßgeblich die Orientierungsinterpolation im Raum.

Es können maximal 3 Orientierungsachsen pro Kinematik angelegt werden.

2.3.4 Hilfsachsen

Es gibt 2 Arten von Hilfsachsen:

- die Teil der Transformation sind. Eine Positionsänderung dieser Achse kann Einfluss auf die Position und/oder Orientierung im Raum haben.
- die nicht Teil der Transformation sind. Eine Positionsänderung dieser Achse hat keinen Einfluss auf die Position und/oder Orientierung im Raum.

Beide Arten werden vollsynchron zur Bewegung im Raum mitgeführt (siehe auch [Bewegungsarten auf Seite 39](#)).

Es können maximal 10 Hilfsachsen pro Kinematik angelegt werden.



Ob eine Hilfsachse verwendet wird, kann im [Transformations-schema auf Seite 111](#) nachgeschaut werden.

2.3.5 Bandachsen

Über eine Bandachse kann ein Band einer Kinematik zugeordnet werden.

Über die Funktion "[Bandsynchrone Bewegung auf Seite 67](#)" kann eine Kinematik auf ein Band aufsynchronisiert werden.

Es können maximal 16 Bandachsen pro Kinematik angelegt werden.

2.4 Punkt

2.4.1 Punkt, Übersicht

Ein Punkt liefert alle Informationen zu einer Position einer Kinematik. Die Koordinaten eines Punktes haben die Reihenfolge Positionierachsen, Orientierungsachsen, Hilfsachsen und Bandachsen. Ein Punkt hat somit den maximalen Aufbau (X, Y, Z, A, B, C, H1, ..., H10, B1, ..., B16).

Für die Darstellung eines Punktes gibt es folgende Varianten:

- Darstellung eines Punktes
Die Darstellung eines Punktes ist abhängig von der verwendeten Kinematik. Nur die verwendeten Achsen einer Kinematik werden im Punkt dargestellt. Koordinaten von nicht verwendeten Achsen werden nicht dargestellt.
- Zuschalten aller Positionierungsachsen
Es werden immer alle Positionen (X, Y, Z) unabhängig von der verwendeten Kinematik dargestellt.
- Zuschalten aller Orientierungsachsen
Es werden immer alle Orientierungen (A, B, C) unabhängig von der verwendeten Kinematik dargestellt.



Das Zuschalten der Positionierungsachsen und Orientierungsachsen hat keinen Einfluss auf die Darstellung von Punkten im ACS.

Nicht verwendete Hilfs- und Bandachsen werden im Punkt nicht dargestellt.

Beispiel:

Darstellung eines Punktes für eine Scara-Kinematik

Eine Scara-Kinematik besteht aus einem Band B1, zwei Positionsachsen X und Y und einer Hilfsachse H1. Ein Punkt für diese Kinematik besteht aus 4 Koordinaten: (X, Y, H1, B1). Diese Zusammensetzung eines Punktes gilt für

Funktionsbeschreibung

alle Koordinatensysteme. Die in der Transformation nicht enthaltene Z-Koordinate und Orientierungen A, B, C werden nicht dargestellt und müssen nicht programmiert werden.

2.4.2 Konfiguration der Punktdarstellung

Folgende Optionen können für die Punktdarstellung genutzt werden:

- **K-0-0099** Bit 7: Zuschalten aller Positionierungsachsen
- **K-0-0099** Bit 8: Zuschalten aller Orientierungsachsen

Durch diese Optionen verschieben sich die Koordinaten im Punkt.



GEFAHR

Durch Verwendung dieser Optionen verschieben sich die Koordinaten im Punkt

Es ist notwendig alle Punkte im Anwenderprogramm zu überprüfen und anzupassen.

Beispiel:

Koordinatenposition bei verschiedenen Punktdarstellungen

	Punkt	Zuschalten Positionierungsachsen	Zuschalten Orientierungsachsen	Zuschalten von Positionierungsachsen und Orientierungsachsen
Punktdarstellung	(X, Z, B, H1, B1)	(X, Y, Z, B, H1, B1)	(X, Z, A, B, C, H1, B1)	(X, Y, Z, A, B, C, H1, B1)
Koordinatenposition für H1	4	5	6	7

Abhängige Parameter

Die Konfiguration der Punktdarstellung hat Einfluss auf folgende Parameter:

- **K-0-0100, K-0-0150** (Positionsdaten):
Die Kinematik-Positionen werden um die zusätzlichen Koordinaten erweitert.
- **K-0-0050, K-0-0051, K-0-0052, K-0-0053** (Einheiten):
Für die neu hinzugefügten Koordinaten im Punkt werden die Einheiten hinzugefügt. Die Einheiten werden je nach Art der zusätzlichen Achse ergänzt.
- **K-0-0007** (Achszahl):
Die Achszahl erhöht sich um die jeweiligen zusätzlichen Koordinaten im Punkt.

Ein Anwendungsbeispiel findet man unter [Erweiterung eines Punkt auf Seite 410](#).

Beispiel:

Veranschaulichung im MCS

Transformations- schema	Punkt	Zuschalten Positionierungsachsen	Zuschalten Orientierungsachsen	Zuschalten von Positionierungsachsen und Orientierungsachsen
FlexStamper (39) mit 1 Band	(X, Y, H1, B1)	(X, Y, Z, H1, B1)	(X, Y, A, B, C, H1, B1)	(X, Y, Z, A, B, C, H1, B1)
4-Achs-Scara (10) mit einer Hilfsachse	(X, Y, Z, H1)	(X, Y, Z, H1)	(X, Y, Z, A, B, C, H1)	(X, Y, Z, A, B, C, H1)
3-Achs Canfield (81)	(Z, B, C, H1)	(X, Y, Z, B, C, H1)	(Z, A, B, C, H1)	(X, Y, Z, A, B, C, H1)

2.5 Zustandsdiagramm einer Kinematik

Wie eine Einzelachse besitzt auch eine Kinematik ein Zustandsdiagramm.

Das Zustandsdiagramm beschreibt die Zustände und Übergänge zwischen den Zuständen einer Kinematik.

Der aktive Zustand einer Kinematik wird im Parameter [K-0-0022](#), Erweiterter Kinematikstatus dargestellt.

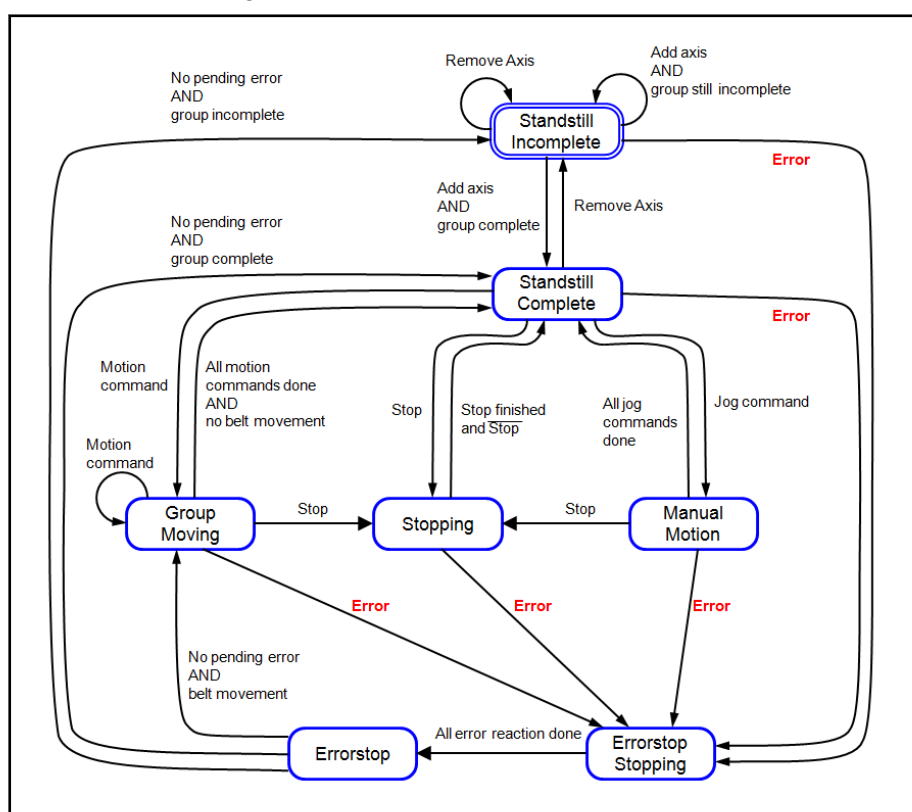


Abb. 2-12: Zustandsdiagramm einer Kinematik

Zustände einer Kinematik

- **Standstill Incomplete**

Der Zustand "Standstill Incomplete" ist der Ausgangszustand.

Funktionsbeschreibung

Im Zustand "Standstill Incomplete" sind noch nicht alle im Parameter [K-0-0008](#) konfigurierten Achsen der Kinematik hinzugefügt.

Mit den Kommandos [Group Axis auf Seite 38](#) und [Group All Axes auf Seite 38](#) lassen sich Achsen der Kinematik hinzufügen. Sind alle konfigurierten Achsen der Kinematik hinzugefügt, wird der Zustand "Standstill Complete" aktiv.

- **Standstill Complete**

Im Zustand "Standstill Complete" sind alle im Parameter [K-0-0008](#) konfigurierten Achsen der Kinematik hinzugefügt.

Mit den Kommandos [Remove Axis auf Seite 38](#) und [Ungroup All Axes auf Seite 38](#) lassen sich Achsen aus der Kinematik entfernen. Wird eine Achse aus der Kinematik entfernt, wird der Zustand "Standstill Incomplete" aktiv.

- **Group Moving**

Im Zustand "Group Moving" ist mindestens ein Bewegungskommando (Motion command) (siehe [Liste aller Bewegungskommandos und Funktionen auf Seite 37](#)) oder eine bandsynchrone Bewegung (siehe [Bandsynchrone Bewegung auf Seite 67](#)) aktiv.

Sind alle Bewegungskommandos abgearbeitet (done) und ist keine bandsynchrone Bewegung aktiv, wird der Zustand "Standstill Complete" aktiv.

- **Stopping**

Im Zustand "Stopping" ist ein "Stop"-Kommando aktiv (siehe [Stop auf Seite 37](#)).

Hat die Kinematik Stillstand erreicht und wird das Stoppen beendet, wird der Zustand "Standstill Complete" aktiv.

- **Manual Motion**

Im Zustand "Manual Motion" ist mindestens ein Handbetrieb-Kommando (Jog command) ("[Jog Step auf Seite 37](#)" oder "[Jog Cont auf Seite 37](#)") aktiv.

Sind alle Handbetrieb-Kommandos abgearbeitet (done), wird der Zustand "Standstill Complete" aktiv.

- **Errorstop Stopping**

Im Zustand "Errorstop Stopping" werden die Fehlerreaktionen der Kinematik (siehe [Fehlerreaktion der kommandierten Bewegung auf Seite 82](#)) und der Bänder (siehe [Bandspezifische Fehlerreaktion auf Seite 82](#)) ausgeführt.

Der Zustand "Errorstop Stopping" wird aktiv, wenn bei einer Kinematik ein Fehler auftritt.

Sind alle Fehlerreaktionen abgeschlossen, wird der Zustand "Errorstop" aktiv.

- **Errorstop**

Im Zustand "Errorstop" ist ein Fehler bei einer Kinematik aufgetreten und die Fehlerreaktionen sind abgeschlossen.

Sind die aufgetretenen Fehler mittels "Fehler löschen" quittiert, wird je nach erfüllten Bedingungen der Zustand "Standstill Incomplete", "Standstill Complete" oder "Group Moving" aktiv.

2.6 Robot-Control, Kommandos und Funktionen

2.6.1 Robot-Control, Kommandos und Funktionen, Überblick

Es können zwei Arten von Aktionen bezüglich einer Kinematik unterschieden werden:

- Kommandos
- Funktionen

Kommandos

Die Abarbeitung eines Kommandos findet asynchron statt. Dies bedeutet, dass mit der Kommandierung eines Kommandos nur der Auftrag erteilt wird, eine Aktion auszuführen. Die eigentliche Aktion findet später statt. Dazu wird das Kommando in den sogenannten Kommandopuffer gestellt. Dieser Kommandopuffer arbeitet nach dem FIFO-Prinzip (First In First Out). Kommandos, die zuerst kommandiert werden, werden vor Kommandos abgearbeitet, die später kommandiert wurden. Ein Kommando wird dabei zuerst vorbereitet. Diese Vorbereitung kann je nach Auslastung des Systems mehrere Motion-Takte benötigen. Wenn ein Kommando vorbereitet ist, steht es zur Ausführung zur Verfügung.

Die maximale Anzahl von Kommandos im Kommandopuffer kann über den Parameter [K-0-0097](#), Maximale Kommandopuffergröße eingestellt werden.

Die aktuelle Anzahl von Kommandos im Kommandopuffer ist im Parameter [K-0-0098](#), Aktuelle Kommandopuffergröße abgebildet.

Die größte Anzahl von Kommandos, die jemals im Kommandopuffer standen, steht im Parameter [K-0-0096](#), Höchstwert der Kommandopuffergröße.



Ein Anwendungsbeispiel für die Auswirkungen unterschiedlicher Kommandopuffer-Einstellungen findet man unter [Bestimmung der Kommandopuffergröße auf Seite 412](#).

Um Überschleifen von Bewegungen durchführen zu können, ist eine Pufferung von Kommandos nötig (siehe [Überschleifen auf Seite 54](#)).

Der Zustand eines Kommandos sagt aus, in welchem Abarbeitungsschritt sich das Kommando befindet (siehe [Zustand eines Kommandos auf Seite 36](#)).

Funktionen

Funktionen arbeiten im Gegensatz zu Kommandos nicht über den Kommandopuffer.

Die Abarbeitung von Funktionen findet direkt bei Kommandierung der Funktion statt (synchrone Abarbeitung). Dies bedeutet, dass direkt nach der Abarbeitung der Funktion das Ergebnis vorliegt.

Arten von Kommandos und Funktionen

	Beschreibung
Liste aller Bewegungskommandos und Funktionen auf Seite 37	Bewegung einer Kinematik im Raum
Liste aller Konfigurationskommandos auf Seite 38	Konfiguration einer Kinematik
Liste aller Informationsfunktionen auf Seite 38	Informationen über eine Kinematik
Liste aller Teach-in-Kommandos und -Funktionen auf Seite 39	Positionen einer Kinematik als Punkte speichern und wiederverwenden

Tab. 2-2: Arten von Kommandos und Funktionen

Bewegungskommandos

Die Kommandierung eines Bewegungskommandos auf eine Kinematik führt zur Bewegung des "Tool Center Point" TCP im Raum.

Funktionsbeschreibung

Der TCP ist ein ausgezeichnete Punkt der Kinematik, der sich meist auf die Spitze der Kinematik bezieht. Der Anwender kommandiert dabei die neue Zielposition des TCP. Die Art des Bewegungskommandos legt dabei die Bahn fest, auf dem die neue TCP-Position erreicht werden soll.



Die neue Zielposition kann in jedem Koordinatensystem kommandiert werden.

Das Koordinatensystem hat keinen Einfluss auf die Bahn, über die der Punkt erreicht wird.

Siehe [Koordinatensysteme, Übersicht auf Seite 24](#)

Die Position des TCP kann mit Hilfe der Funktion [Read Pos auf Seite 37](#) ermittelt werden. Dabei wird zwischen zwei Positionen unterschieden.

- **Prepare:** Position des TCP vom zuletzt vorbereiteten Motion-Kommando
- **Execute:** Zuletzt dem Roboter kommandierte Position

Die Prepare-Position springt nach der Vorbereitung eines Motion-Kommandos zur nächsten Zielposition. Die Execution-Position hat hingegen einen kontinuierlichen Verlauf.

Ein Bewegungskommando setzt sich aus folgenden Teilen zusammen:

- **Endpunkt und/oder Zwischenpunkte:** Punkte auf der Bahn der Bewegung
- **Maximalwerte:** Maximalwerte für das Geschwindigkeitsprofil, wie z. B. maximale Geschwindigkeit und maximale Beschleunigung
- **Bewegungsart:** Die Art der Bahn, auf der die Zwischenpunkte und der Endpunkt erreicht werden sollen
Siehe [Bewegungsarten auf Seite 39](#)
- **Überschleif- bzw. Pufferart:** Die Art, in dem der Satz gestartet bzw. überschleift werden soll

Siehe [Überschleifen auf Seite 54](#)

2.6.2 Zustand eines Kommandos

Der Zustand eines Kommandos sagt aus, in welchem Abarbeitungsschritt sich das Kommando im Bezug zum Kommandopuffer befindet. Ein Kommando kann sich in folgenden Zuständen befinden:

Zustand eines Kommandos

Zustand	Bedeutung
InTransmission	Das Kommando wurde abgesetzt und wird in den Kommandopuffer geschrieben
InBuffer	Das Kommando wurde vorbereitet, es befindet sich im Kommandopuffer und wartet darauf aktiv zu werden
Active	Das Kommando ist aktiv
Done	Das Kommando wurde erfolgreich beendet
Aborted	Das Kommando wurde abgebrochen
Error	Das Kommando hat einen Fehler

Tab. 2-3: Zustand eines Kommandos

2.6.3 Liste aller Kommandos und Funktionen

Die gleiche Funktion/das gleiche Kommando kann über verschiedene Schnittstellen aufgerufen werden. Die folgende Tabelle zeigt den Aufrufnamen im jeweiligem Interface zum Referenznamen der Funktionsbeschreibung.

Die Schnittstellen sind im einzelnen:

- **SPS:** Funktionsbausteinname der SPS
- **MLPI core:** Funktionsname des C/C++ Motion Logic Programming Interface
- **MLPI COM:** Funktionsname des COM Motion Logic Programming Interface
- **MLPI Java:** Funktionsname des Java Motion Logic Programming Interface

Liste aller Bewegungskommandos und -Funktionen

Referenz Beschreibung	SPS	MLPI core	MLPI COM MLPI Java
Move Linear Absolute	MC_MoveLinearAbsolute	mlpiRobotMoveLinearAbs	MoveLinearAbsolute
Move Linear Relative	MC_MoveLinearRelative	mlpiRobotMoveLinearRel	MoveLinearRelative
Move Circular Absolute	MC_MoveCircularAbsolute	mlpiRobotMoveCircularAbs	MoveCircularAbsolute
Move Circular Relative	MC_MoveCircularRelative	mlpiRobotMoveCircularRel	MoveCircularRelative
Move Direct Absolute	MC_MoveDirectAbsolute	mlpiRobotMoveDirectAbs	MoveDirectAbsolute
Move Direct Relative	MC_MoveDirectRelative	mlpiRobotMoveDirectRel	MoveDirectRelative
Move Jump Absolute	ML_RobotMoveJumpAbsolute	mlpiRobotMoveJumpAbs	MoveJumpAbsolute
Synchronization with Limits	ML_BeltSynchronisationWithLimits	mlpiRobotSyncOnWithLimits	SyncOnWithLimits
Desynchronization with Limits	ML_BeltDeSynchronisationWithLimits	mlpiRobotSyncOffWithLimits	SyncOffWithLimits
Jog Cont	ML_GroupJogCont	mlpiRobotJogCont	JogCont
Jog Step	ML_GroupJogStep	mlpiRobotJogStep	JogStep
Open Cyclic Position	ML_RobotOpenCyclicChannel	mlpiRobotOpenCyclicChannel	
Open Direct Cyclic Channel	ML_RobotOpenDirectCyclicChannel		
Write Cyclic Position (Funktion)	ML_RobotWriteCyclicChannel	mlpiRobotWriteCyclicChannel	
Stop	MC_GroupStop	mlpiRobotStop	Stop
Interrupt	MC_GroupInterrupt	mlpiRobotInterrupt	Interrupt
Continue	MC_GroupContinue	mlpiRobotContinue	Continue
Wait	ML_GroupWait	mlpiGroupWait	Wait

Tab. 2-4: Liste aller Bewegungskommandos und Funktionen

Funktionsbeschreibung

Liste aller Konfigurationskommandos

Referenz Beschreibung	SPS	MLPI core	MLPI COM MLPI Java
Group Reset	MC_GroupReset	mlpiRobotReset	Reset
Group Axis	MC_AddAxisToGroup	mlpiMotionAddAxisToGroup	
Remove Axis	MC_RemoveAxisFromGroup	mlpiMotionRemAxisFrom-Group	
Group All Axes	ML_GroupAllAxes	mlpiRobotAddAllAxisToGroup	AddAllAxisToGroup
Ungroup All Axes	MC_UngroupAllAxes	mlpiRobotRemAllAxisFrom-Group	RemAllAxisFromGroup
Set Kinematik Transformation	MC_SetKinTransform		
Set Kinematik Transform Extended	MC_SetKinTransformExt		
Set Cartesian Transformation	MC_SetCartesianTransform	mlpiRobotSetCartesianTransform	SetCartesianTransform
Set Cylindric Transformation	ML_SetCylindricTransform	mlpiRobotSetCylindricTransform	SetCylindricTransform
Set Belt Configuration	ML_SetBeltConfiguration	mlpiRobotSetBeltConfiguration	SetBeltConfiguration
Set Belt Desync Relative Configuration	ML_SetBeltDesyncRelConfiguration	mlpiRobotSetBeltDesyncRelConfiguration	SetBeltDesyncRelConfiguration
Set Rotary Table Configuration	ML_SetRotaryTableConfiguration	mlpiRobotSetRotaryTableConfiguration	SetRotaryTableConfiguration
Set Belt Error Reaction	ML_SetBeltErrorReaction	mlpiRobotSetBeltErrorReaction	SetBeltErrorReaction
Set Safe Zone	ML_GroupSetSafeZone	mlpiRobotSetSafeZone	SetSafeZone
Set Move ID	ML_RobotSetMoveID	mlpiRobotSetMoveId	SetMoveId

Tab. 2-5: Liste aller Konfigurationskommandos

Liste aller Informationsfunktionen

Referenz Beschreibung	SPS	MLPI core	MLPI COM MLPI Java
Read Pos	MC_GroupReadActualPosition	mlpiReadPos	ReadPos
Switch On Path	ML_SwitchOnPath		
Read Cyclic Position	ML_RobotReadCyclicChannel	mlpiRobotReadCyclicChannel	
Move Point	ML_RobotMovePoint	mlpiRobotMovePoint	MovePoint
Transform Point	ML_RobotTransformPoint	mlpiTransformPoint	TransformPoint
Set Performance Enable	ML_RobotSetPerformanceEnable	mlpiRobotSetPerformanceEnable	SetPerformanceEnable
Get Performance	ML_RobotGetPerformance	mlpiRobotGetPerformanceResult	GetPerformanceResult
Get Mechanic	ML_RobotGetMechanic	mlpiRobotGetMechanicalData	GetMechanicData

Funktionsbeschreibung

Referenz Beschreibung	SPS	MLPI core	MLPI COM MLPI Java
Get Trafo Values	ML_RobotGetTrafoValues	mlpiRobotGetTrafoValues	GetTrafoValues
Get Belt Configuration	ML_RobotGetBeltConfiguration	mlpiRobotGetBeltConfigurati- on	GetBeltConfiguration
Get Belt Desync Relative Con- figuration	ML_RobotGetBeltDesyncRel- Configuration	mlpiRobotGetBeltDesyncRel- Configuration	GetBeltDesyncRelCon- figuration
Get Belt Error Reaction	ML_RobotGetBeltErrorreaction	mlpiRobotGetBeltErrorReacti- on	GetBeltErrorReaction
Get Safe Zone	ML_RobotGetSafeZones	mlpiRobotGetSafeZones	GetSafeZones
Get Move ID	ML_RobotGetMoveID	mlpiRobotGetMoveId	GetMoveId

Tab. 2-6: Liste aller Informationsfunktionen

Liste aller Teach-in-Kommandos und -Funktionen

Referenz Beschreibung	SPS	MLPI core	MLPI COM MLPI Java
Write Point	ML_GroupWritePoint	mlpiRobotTeachInWritePoint	
Read Point	ML_GroupReadPoint	mlpiRobotTeachInReadPoint	
Read Next Point	ML_GroupReadNextPoint	mlpiRobotTeachInReadNext- Point	
Delete Points	ML_GroupDeletePoints	mlpiRobotTeachInDeletePo- ints	
Save Point File	ML_GroupSavePointFile	mlpiRobotTeachInSavePoint- File	
Load Point File	ML_GroupLoadPointFile	mlpiRobotTeachInLoadPoint- File	
Delete Point File	ML_GroupDeletePointFile	mlpiRobotTeachInDeletePoint- File	

Tab. 2-7: Liste aller Teach-in-Kommandos und -Funktionen

2.7 Bewegungsarten

2.7.1 Bewegungsarten, Überblick

Die MLC Robot-Control unterstützt das Verfahren einer Kinematik in vier Bewegungsarten:

- Linear (Gerade im Raum), [Bewegungsart Linear auf Seite 40](#)
- Circular (Kreisabschnitt im Raum), [Bewegungsart Circular auf Seite 41](#)
- Punkt zu Punkt (PTP), [Bewegungsart Punkt zu Punkt auf Seite 42](#)
- Jump, [Bewegungsart Jump auf Seite 42](#)
- Jump, [Bewegungsart HalfJump auf Seite 45](#)



Bei der Interpolation werden Orientierungsbewegungen vollsyn-
 chron mit angesteuert.

Funktionsbeschreibung

Weiterhin wird zwischen "[Absolute Positionierung auf Seite 48](#)" und "[Relative Positionierung auf Seite 48](#)" unterschieden.

Bewegungsarten bei absoluter und relativer Positionierung

	Absolut	Relativ
Linear (Gerade im Raum)	Move Linear Absolut	Move Linear Relativ
Circular (Kreisabschnitt im Raum)	Move Circular Absolut	Move Circular Relativ
Punkt zu Punkt (PTP)	Move Direct Absolut	Move Direct Relativ
Jump	Move Jump Absolut	
HalfJump	Move Jump Absolut	

Tab. 2-8: Überblick über Bewegungsarten

Alle Bewegungsarten werden durch Bewegungsgrenzwerte definiert. Dazu gehören:

- Maximale Geschwindigkeit - V
- Maximale Beschleunigung / Verzögerung - A
- Maximaler Ruck - J

2.7.2 Bewegungsart Linear

Die Kinematik fährt den Zielpunkt auf einer Geraden an.

Die Gerade ist geometrisch durch zwei Punkte definiert. Da der Steuerung die momentane Position P1 der Kinematik bekannt ist, genügt die Angabe eines Zielpunktes P2.

Die Bewegungen der Einzelachse ergeben sich aus der Bahn und der inversen Transformation der Kinematik.

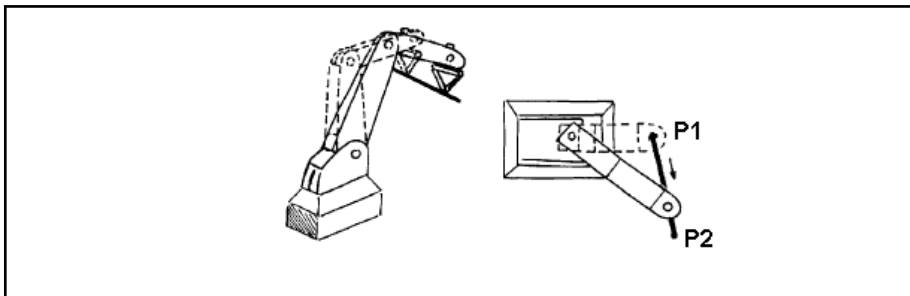


Abb. 2-13: Interpolationsart Linear

Folgende Bewegungskommandos können für diese Bewegungsart verwendet werden:

- [Move Linear Absolut auf Seite 37](#)
- [Move Linear Relativ auf Seite 37](#)

Hilfsachsen

Hilfsachsen werden vollsynchron zur Bewegung im Raum mitgeführt. Dies bedeutet, dass nach x% der Strecke im Raum auch x% der Hilfsachsen-Bewegung erreicht ist. Dies hat zur Folge, dass auf der Bahn im Raum zeitsynchron mit der Hilfsachsen-Bewegung beschleunigt, mit konstanter Geschwindigkeit gefahren und abgebremst wird.

Weiterhin werden die Bewegungsgrenzwerte auf der Bahn so begrenzt, dass die Grenzwerte der Hilfsachsen nicht überschritten werden. Dadurch kann es vorkommen, dass die programmierten Bewegungsgrenzwerte der Bewegung im Raum nicht erreicht werden.

Funktionsbeschreibung

Die Grenzwerte der Hilfsachsen bestimmen sich aus dem Minimum der Einzelachsgrenzwerte und der Limits der Hilfsachsen:

- Grenzwert_V = Minimum(Einzelachsgrenzwert_V (A-0-0032, A-0-0033, A-0-0035), Limit_V (K-0-0028))
- Grenzwert_A = Minimum(Einzelachsgrenzwert_A (A-0-0034), Limit_A (K-0-0029))
- Grenzwert_J = Minimum(Einzelachsgrenzwert_J (A-0-0036), Limit_J (K-0-0030))

Wird nur eine Hilfsachsenbewegung ohne Bewegung im Raum kommandiert, so wird diese Bewegung mit den Grenzwerten der Hilfsachse ausgeführt.

2.7.3 Bewegungsart Circular

Die Kinematik führt den Zielpunkt auf einem Kreisabschnitt im Raum an.

Ein Kreis ist geometrisch definiert durch drei Punkte. Zusätzlich zum Zielpunkt P2 müssen Sie also noch einen Zwischenpunkt "PZ" angeben, damit die Steuerung den Kreisabschnitt eindeutig berechnen kann.

Der Punkt P1 ist der letzte angefahrene Punkt und somit der Steuerung bekannt. Daher genügt die Angabe des Zwischenpunkts PZ und des Zielpunkts P2.

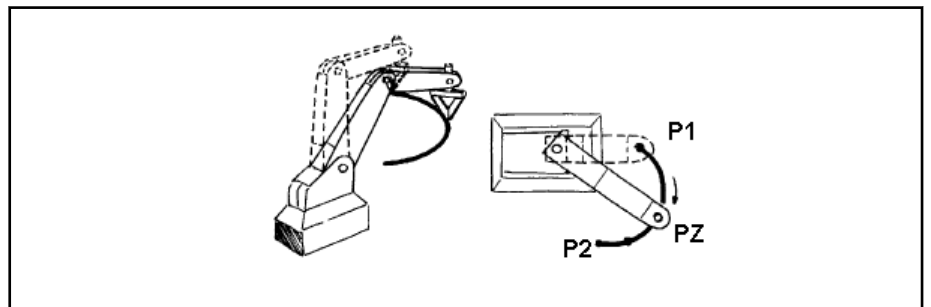


Abb. 2-14: Interpolationsart Circular

Folgende Bewegungskommandos können für diese Bewegungsart verwendet werden:

- [Move Circular Absolut auf Seite 37](#)
- [Move Circular Relative auf Seite 37](#)

Hilfsachsen

Hilfsachsen werden vollsynchron zur Bewegung im Raum mitgeführt. Dies bedeutet, dass nach x% der Strecke im Raum auch x% der Hilfsachsen-Bewegung erreicht ist. Dies hat zur Folge, dass auf der Bahn im Raum zeitsynchron mit der Hilfsachsenbewegung beschleunigt, mit konstanter Geschwindigkeit gefahren und abgebremst wird.

Weiterhin werden die Bewegungsgrenzwerte auf der Bahn so begrenzt, dass die Grenzwerte der Hilfsachsen nicht überschritten werden. Dadurch kann es vorkommen, dass die programmierten Bewegungsgrenzwerte der Bewegung im Raum nicht erreicht werden.

Die Grenzwerte der Hilfsachsen bestimmen sich aus dem Minimum der Einzelachsgrenzwerte und der Limits der Hilfsachsen:

- Grenzwert_V = Minimum(Einzelachsgrenzwert_V (A-0-0032, A-0-0033, A-0-0035), Limit_V (K-0-0028))
- Grenzwert_A = Minimum(Einzelachsgrenzwert_A (A-0-0034), Limit_A (K-0-0029))
- Grenzwert_J = Minimum(Einzelachsgrenzwert_J (A-0-0036), Limit_J (K-0-0030))

Funktionsbeschreibung

Wird nur eine Hilfsachsenbewegung ohne Bewegung im Raum kommandiert, so wird diese Bewegung mit den Grenzwerten der Hilfsachse ausgeführt.

Entartete Kreise *Entartete Kreise entstehen wenn:*

- Start-, Zwischen- und Endpunkt auf einer Geraden liegen
- Start- und Endpunkt der gleiche Punkt sind
- Start- und Zwischenpunkt der gleiche Punkt sind
- Zwischen- und Endpunkt der gleiche Punkt sind

In allen diesen Fällen wird statt eines Kreises eine Gerade zwischen Startpunkt und Endpunkt verfahren. Der Zwischenpunkt wird ignoriert.



Der Zwischenpunkt wird auch ignoriert, wenn der Start-, Zwischen- und Endpunkt auf einer Geraden liegen und der Zwischenpunkt hinter dem Endpunkt liegt.

In diesem Fall wird der Zwischenpunkt nicht erreicht.

2.7.4 Bewegungsart Punkt zu Punkt

Mit dieser Bewegungsart wird die kommandierte Position so schnell wie möglich angefahren.

Dies wird dadurch erreicht, dass jede einzelne Achse auf dem kürzesten Weg vom Start- zum Zielpunkt verfährt.

Das Geschwindigkeitsprofil der Bewegung wird durch die Bewegungsgrenzwerte ([A-0-0032](#), [A-0-0033](#), [A-0-0034](#), [A-0-0036](#)) der Einzelachsen bestimmt.

Die Bahn im Raum und die Bahngeschwindigkeit spielen keine Rolle. Es ist sichergestellt, dass die Bewegungen aller Einzelachsen gleichzeitig enden. Dabei werden alle Abschnitte der Bewegung (Beschleunigung, konstante Geschwindigkeit, Abbremsung) für alle Achsen gleichzeitig gestartet und beendet. Dies wird allgemein auch als Vollsynchron-PTP bezeichnet.

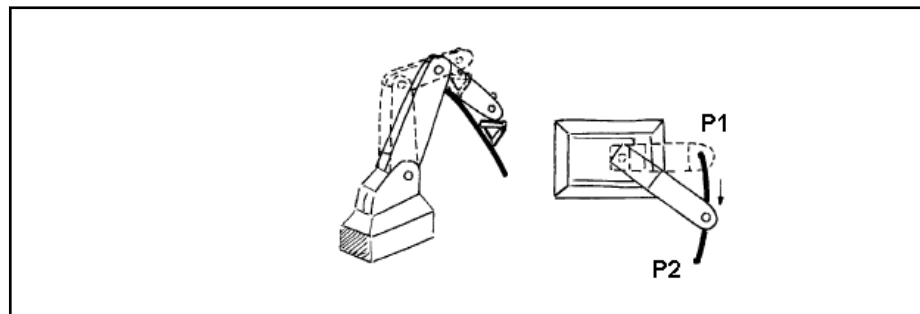


Abb. 2-15: *Interpolationsart Direct*

Folgende Bewegungskommandos können für diese Bewegungsart verwendet werden:

- [Move Direct Absolute auf Seite 37](#)
- [Move Direct Relativ auf Seite 37](#)

2.7.5 Bewegungsart Jump

Die Bewegungsart fährt den Zielpunkt durch eine senkrechte Bewegung in Z-Richtung, eine bogenförmige Bewegung über den Endpunkt P2 und eine senkrechte Bewegung in Z-Richtung an. Dabei wird die Bewegung mit optimiertem Geschwindigkeitsprofil ohne mehrmalige Beschleunigungs- bzw. Verzögerungsrampen angefahren.

Funktionsbeschreibung

Die Bewegung ist geometrisch definiert durch Startpunkt, Zielpunkt, Starthöhe, Endhöhe und Maximalhöhe. Der Punkt P1 ist die letzte kommandierte Position und somit der Steuerung bekannt. Daher genügt die Angabe des Endpunkts P2 sowie der Starthöhe, Endhöhe und Maximalhöhe.

Die Starthöhe definiert die minimale Höhe vor Beginn der horizontalen Bewegung. Die Endhöhe definiert die minimale Höhe zum Ende der horizontalen Bewegung. Die Maximalhöhe definiert die maximale Höhe, welche über den gesamten Bewegungsablauf erreicht werden darf.

Start- und Endhöhe werden relativ zum Start- bzw. Endpunkt in Z-Richtung angegeben. Die Maximalhöhe muss relativ zum tieferen (kleineren Z-Wert) der beiden Punkte (P1 und P2) in Z-Richtung angegeben werden.



Die Bewegungsart kann sowohl in positive und negative Z-Richtung verfahren. Dadurch kann die MoveJump-Funktion auch in einem PCS kommandiert werden, welches zum MCS verdreht ist. Für den negativen Fall muss die Maximalhöhe relativ zum höheren (größeren Z-Wert) der beiden Punkte (P1 und P2) in Z-Richtung angegeben werden.

Um mögliche Kollisionen mit Verpackungshülsen oder ähnlichem zu vermeiden, wird vor Erreichen der Starthöhe bzw. nach Erreichen der Endhöhe (nach der horizontalen Bewegung) eine rein senkrechte Bewegung durchgeführt. Für bandsynchrone Bewegungsabläufe wird während der senkrechten Bewegung die Synchronität nicht geändert. Der Um-, Auf- oder Absynchronisiervorgang findet ausschließlich während der horizontalen Bewegung statt. Änderungen in der Orientierung finden ebenfalls ausschließlich während der horizontalen Bewegung statt.

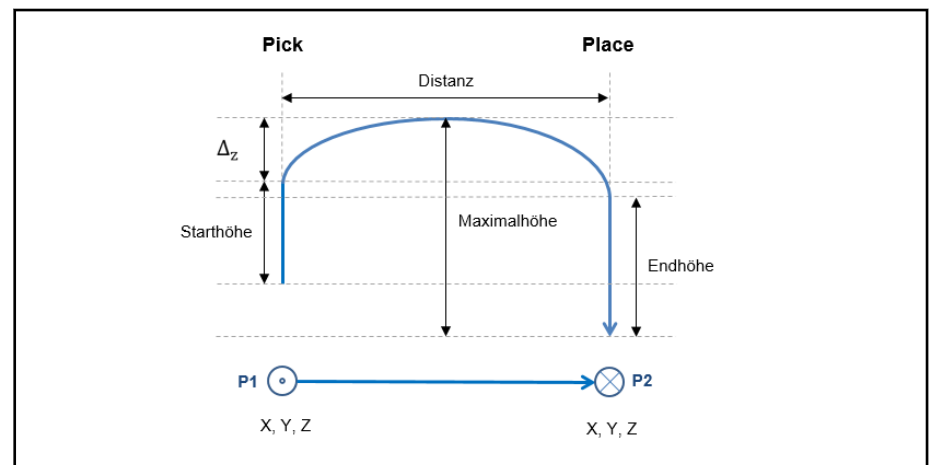


Abb. 2-16: Bewegungsablauf und benötigte Parameter

Beschränkungen:

- Für Jump in positive Z-Richtung: Start-, Maximal- und Endhöhe dürfen keine negativen Werte annehmen
- Für Jump in negative Z-Richtung: Start-, Maximal- und Endhöhe dürfen keine positiven Werte annehmen
- Die Maximalhöhe muss absolut größer sein als Start- und Endhöhe, da Δ_z größer null sein muss
- Das Verhältnis von Δ_z zur Distanz darf 1:50 nicht überschreiten, da sonst die Kurve im Raum nicht berechnet werden kann
- Start- und Endpunkt dürfen nicht der gleiche Punkt sein

Funktionsbeschreibung



Die senkrechte Bewegung zu Beginn und Ende der Bewegung erfolgt immer in Z-Richtung. Daher werden Start-, Maximal- und Endhöhe ebenfalls immer in Z-Richtung interpretiert und müssen relativ zum Start- bzw. Endpunkt angegeben werden.

MoveJump über Hindernis

Mit der MoveJump-Funktion können statische Hindernisse zwischen dem Pick- und Placeband automatisch überwunden werden. Dazu wird ein lineares, horizontales Segment an der höchsten Stelle der Bewegung eingefügt. Dieses Segment muss nicht in der Mitte der Bewegung liegen und ist so lang, wie das Hindernis breit. Ein Hindernis, dessen Höhe fast der Maximalhöhe entspricht, kann so ohne Kollision übersprungen werden.



Liegt eine schmale Schutzzone nicht in der Mitte der Bewegung, so wird das Linear-Segment bis zur Mitte der Bewegung ausgeweitet.

Hindernisse werden mit Schutzzonen realisiert und mit dem Kommando Set Safe Zone definiert. An der MoveJump-Funktion sind dazu keine Eingaben am Interface nötig. Sobald die Schutzzonen eine imaginäre Linie zwischen dem Ende der Start- und Endhöhe schneiden, wird ein zusätzliches Segment in den MoveJump eingefügt.

Der Anwender muss sicherstellen, dass die Maximalhöhe der Bewegung über der Schutzzone liegt. Bei Verletzung der Schutzzone durch den Pfad, wird eine Fehlermeldung ausgegeben. Lesen Sie dazu das Kapitel [Verfahrensbereichsgrenzen auf Seite 79](#).

Schutzzonen werden als statische Hindernisse im Raum programmiert. Bei Pick- und Place-Anwendungen ändern sich die Abstände zu den Hindernissen dynamisch. Die MoveJump-Funktion reagiert automatisch auf diese veränderten Abstände und passt die Länge und Lage des Linearsegments an.

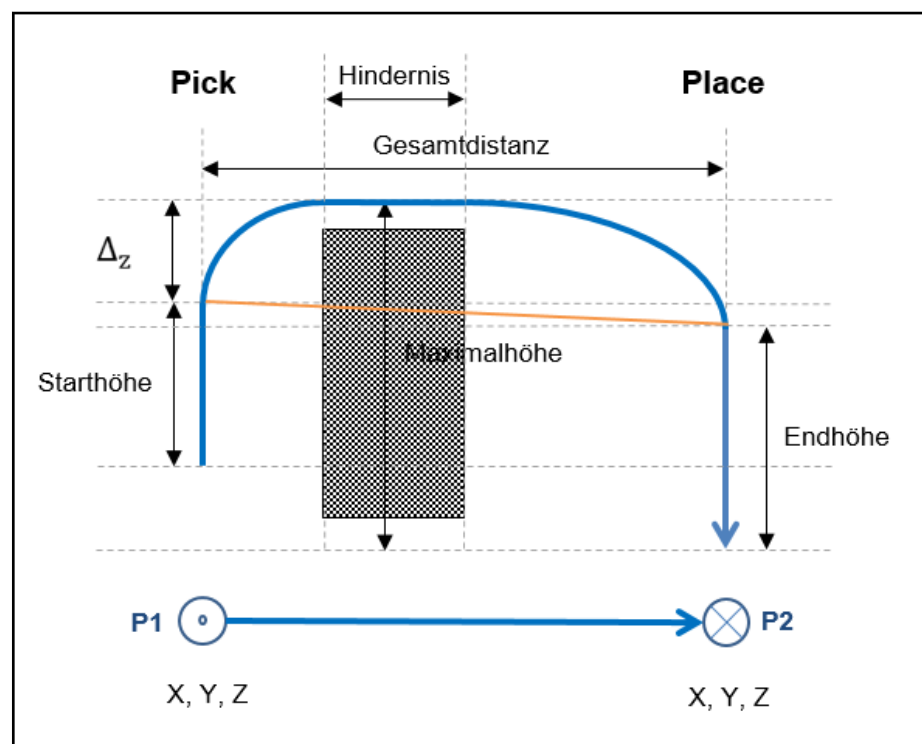


Abb. 2-17: Bewegungsablauf über Hindernis

2.7.6 Bewegungsart HalfJump

HalfJumpUp Die Bewegungsart fährt den Zielpunkt P2 durch eine senkrechte Bewegung in Z-Richtung und eine bogenförmige Bewegung an.

Die Bewegung ist geometrisch definiert durch Startpunkt, Zielpunkt und die Starthöhe. Der Punkt P1 ist die letzte kommandierte Position und somit der Steuerung bekannt. Daher genügt die Angabe des Endpunkts P2 sowie der Starthöhe.



Da die gleiche Schnittstelle wie bei Bewegungsart MoveJump verwendet wird, muss zur Unterscheidung als Maximalhöhe 0 angegeben werden. Die Endhöhe wird in diesem Fall ignoriert.

Die Bewegungsart kann, genau wie der Jump, sowohl in positive und negative Z-Richtung verfahren.

Die Starthöhe definiert die minimale Höhe vor Beginn der horizontalen Bewegung.

Die Starthöhe wird relativ zum Startpunkt in Z-Richtung angegeben.

Um mögliche Kollisionen mit Verpackungshülsen oder ähnlichem zu vermeiden, wird vor Erreichen der Starthöhe eine rein senkrechte Bewegung durchgeführt. Für bandsynchrone Bewegungsabläufe wird während der senkrechten Bewegung die Synchronität nicht geändert. Der Um-, Auf- oder Absynchronisiervorgang findet ausschließlich während der horizontalen Bewegung statt. Änderungen in der Orientierung finden ebenfalls ausschließlich während der horizontalen Bewegung statt.

Funktionsbeschreibung

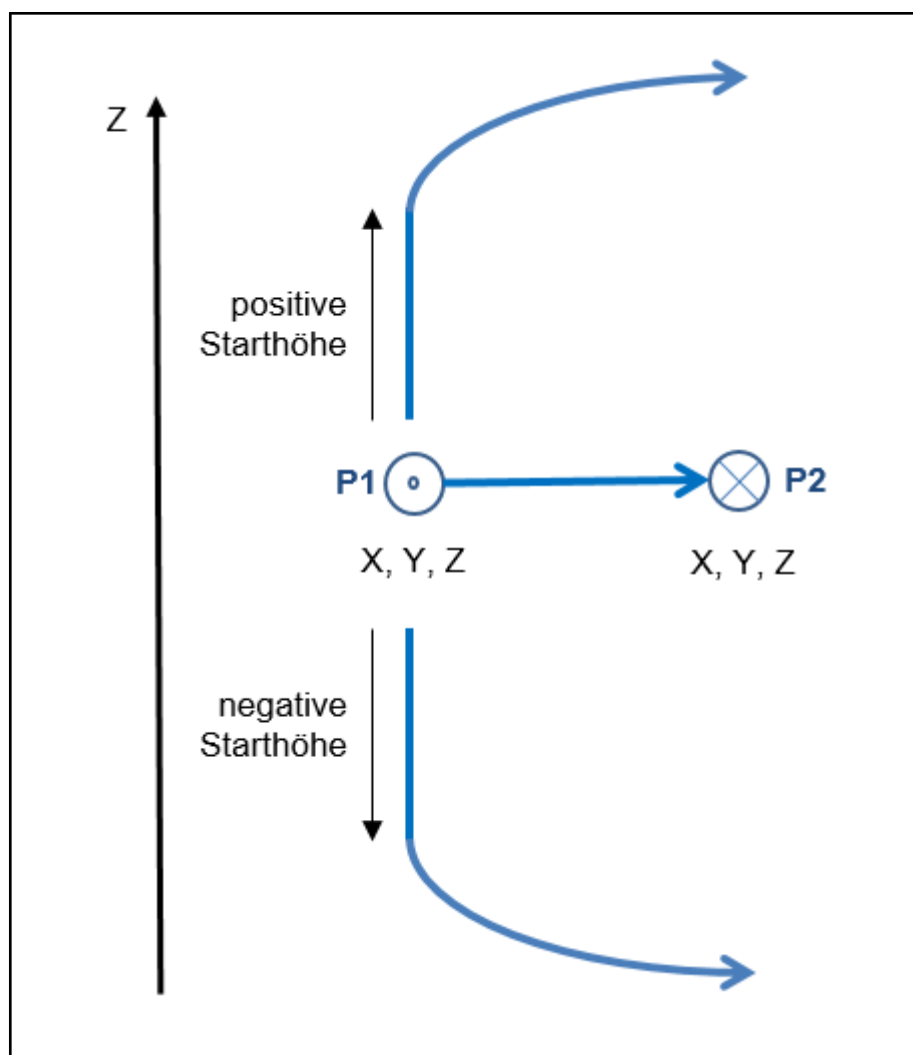


Abb. 2-18: Bewegungsablauf und benötigte Parameter

Beschränkungen:

- Für positiven HalfJumpUp: Starthöhe darf keine negativen Werte annehmen
- Für negativen HalfJumpUp: Starthöhe darf keine positiven Werte annehmen
- Start- und Endpunkt dürfen nicht der gleiche Punkt sein



Die senkrechte Bewegung zu Beginn erfolgt immer in Z-Richtung. Daher wird die Starthöhe ebenfalls immer in Z-Richtung interpretiert und muss relativ zum Startpunkt angegeben werden.

HalfJumpDown

Die Bewegungsart fährt den Zielpunkt P2 durch bogenförmige Bewegung und eine senkrechte Bewegung in Z-Richtung an.

Die Bewegung ist geometrisch definiert durch Startpunkt, Zielpunkt und die Endhöhe. Der Punkt P1 ist die letzte kommandierte Position und somit der Steuerung bekannt. Daher genügt die Angabe des Endpunkts P2 sowie der Endhöhe.

Funktionsbeschreibung



Da die gleiche Schnittstelle wie bei Bewegungsart MoveJump verwendet wird, muss zur Unterscheidung als Maximalhöhe 0 angegeben werden. Die Starthöhe wird in diesem Fall ignoriert.

Die Bewegungsart kann, genau wie der Jump, sowohl in positive und negative Z-Richtung verfahren.

Die Endhöhe definiert die minimale Höhe vor Beginn der horizontalen Bewegung.

Die Endhöhe wird relativ zum Endpunkt in Z-Richtung angegeben.

Um mögliche Kollisionen mit Verpackungshülsen oder ähnlichem zu vermeiden, wird bei Erreichen der Endhöhe eine rein senkrechte Bewegung durchgeführt. Für bandsynchrone Bewegungsabläufe wird während der senkrechten Bewegung die Synchronität nicht geändert. Der Um-, Auf- oder Absynchronisiervorgang findet ausschließlich während der horizontalen Bewegung statt. Änderungen in der Orientierung finden ebenfalls ausschließlich während der horizontalen Bewegung statt.

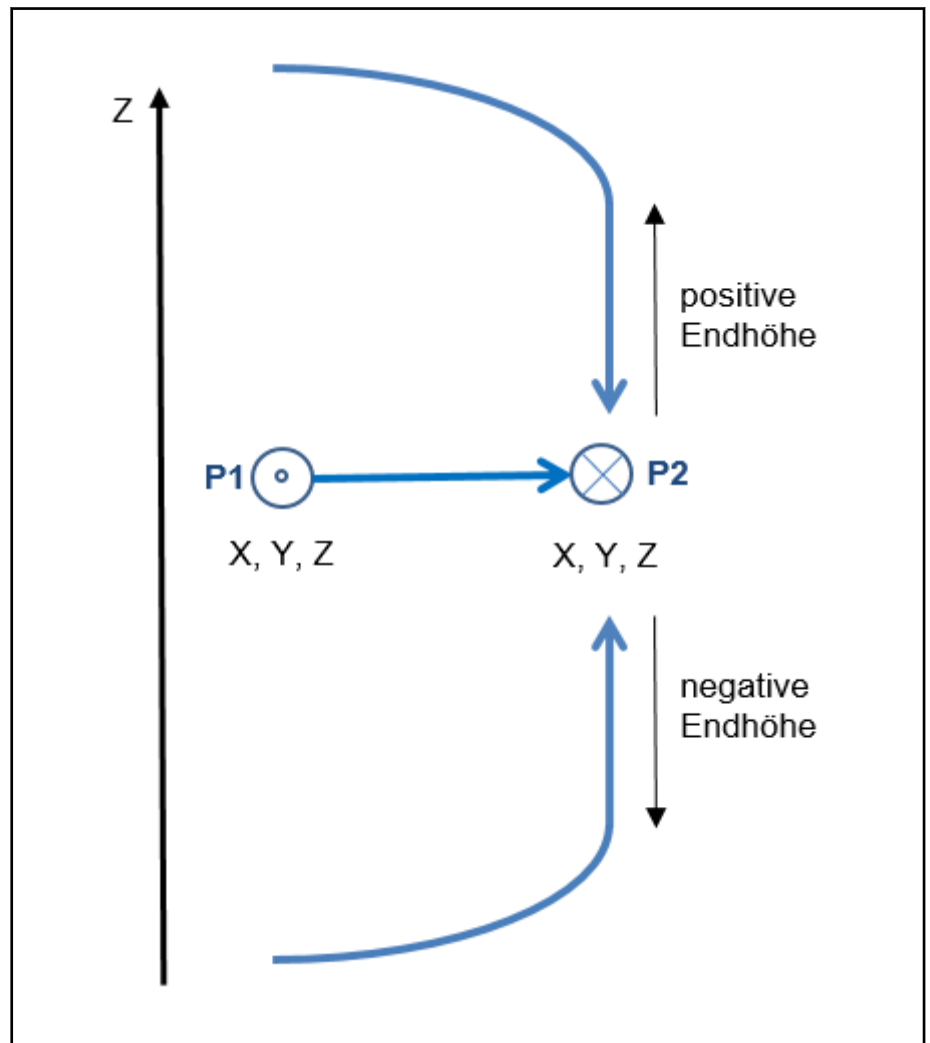


Abb. 2-19: Bewegungsablauf und benötigte Parameter

Beschränkungen:

- Für positiven HalfJumpDown: Endhöhe darf keine negativen Werte annehmen

Funktionsbeschreibung

- Für negativen HalfJumpDown: Endhöhe darf keine positiven Werte annehmen
- Start- und Endpunkt dürfen nicht der gleiche Punkt sein



Die senkrechte Bewegung zu Beginn erfolgt immer in Z-Richtung. Daher wird die Endhöhe ebenfalls immer in Z-Richtung interpretiert und muss relativ zum Endpunkt angegeben werden.

2.7.7 Absolute Positionierung

Bei absoluter Positionierung wird eine Position vorgegeben. Die vorgegebene Position wird absolut im Raum angefahren.

2.7.8 Relative Positionierung

Bei relativer Positionierung wird eine Distanz vorgegeben. Die vorgegebene Distanz wird relativ zu der letzten kommandierten Position verfahren, d. h. die letzte kommandierte Position addiert zu der zu verfahrenen Distanz ergibt die neue absolute Endposition.

2.7.9 Kommandieren von modulo-gewichteten Achsen

Bei modulo-gewichteten Achsen kann über die Kommandierung der Zielposition für diese Achse die Anzahl der Umdrehungen und die Richtung angegeben werden. Die relative Distanz für die Bewegung ist dabei immer der kommandierte Wert minus die aktuelle modulo-behaftete Position der Achse. Daraus ergibt sich:

- Kommandierung zwischen 0 und dem Modulo-Wert der Achse:
Die Achse fährt die Zielposition an, als ob die Achse absolut-gewichtet wäre
- Kommandierung kleiner 0:
Die Achse fährt in negativer Richtung
- Kommandierung größer Modulo-Wert der Achse:
Die Achse fährt in positive Richtung

Beispiel:

Kommandierung einer modulo-gewichteten Achse mit einem Modulo-Wert von 360°

Aktuelle modulo-behaftete Position	Kommandierter Wert	Relative Bewegung der Achse	Modulo-behaftete Position nach Bewegung
100°	50°	-50°	50°
50°	350°	+300°	350°
350°	-20°	-370°	340°
340°	-370°	-710°	350°
350°	370°	+20°	10°
10°	370°	+360°	10°
10°	750°	+740°	30°

Tab. 2-9: Kommandierung einer modulo-gewichteten Achse

2.8 Handbetrieb

2.8.1 Handbetrieb, Allgemein

Um eine Kinematik im Handbetrieb zu verfahren, wird einer der folgenden Kommandotypen verwendet:

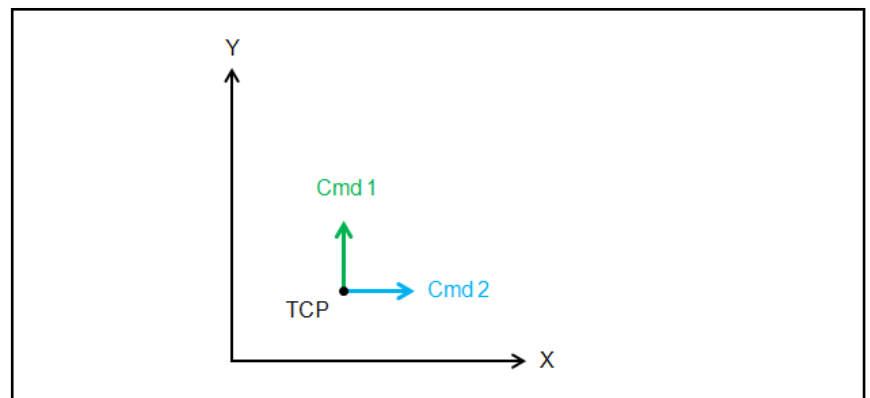
- [Schrittbetrieb auf Seite 50](#)
- [Kontinuierlicher Betrieb auf Seite 50](#)

Im Handbetrieb werden die folgenden Bewegungsarten (siehe [Bewegungsarten auf Seite 39](#)) anhand des angegebenen Koordinatensystems unterstützt:

- **PTP (ACS):**
Wird das Koordinatensystem ACS angegeben, wird mit der Bewegungsart PTP verfahren. Angegebene, prozentuale Werte für Geschwindigkeit, Beschleunigung, Verzögerung und Ruck beziehen sich immer auf die Achse mit dem größeren "Abstand" bzw. "Richtung".
- **Linear (BCS, MCS, PCS):**
Werden die Koordinatensysteme BCS, MCS oder PCS angegeben, wird mit der Bewegungsart Linear verfahren

Die Handbetrieb-Kommandos haben folgende Eigenschaften:

- Mehrere Handbetrieb-Kommandos in **verschiedene** Richtungen können gleichzeitig ausgeführt werden

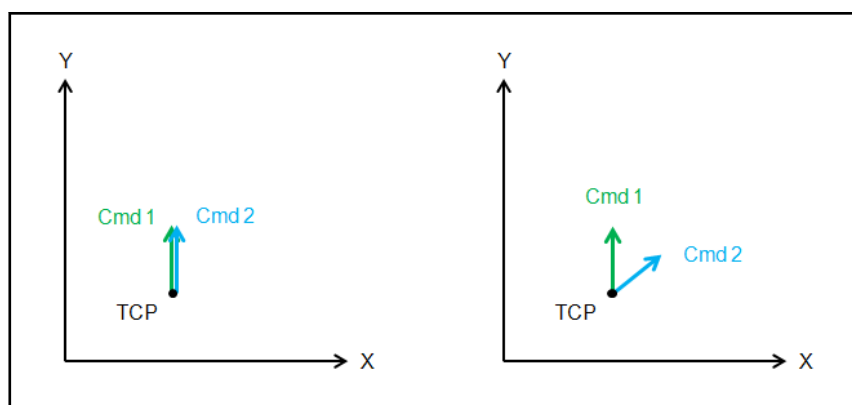


Cmd 1 Handbetrieb-Kommando in Y-Richtung
Cmd 2 Handbetrieb-Kommando in X-Richtung

Abb. 2-20: Mehrere Handbetrieb-Kommandos in verschiedene Richtungen

- Mehrere Handbetrieb-Kommandos in die **gleiche oder teilweise gleiche** Richtung können nicht gleichzeitig ausgeführt werden

Funktionsbeschreibung



Cmd 1 Handbetrieb-Kommando in Y-Richtung
Cmd 2 Handbetrieb-Kommando in Y-Richtung und teilweise in X-Richtung

Abb. 2-21: Mehrere Handbetrieb-Kommandos in dieselbe oder teilweise in dieselbe Richtung

- Mehrere Handbetrieb-Kommandos mit **unterschiedlichen Koordinatensystemen** (z. B. MCS und BCS) können nicht gleichzeitig ausgeführt werden
- Mehrere Handbetrieb-Kommandos mit **unterschiedlichen Kommandotypen** (z. B. Schrittbetrieb und Kontinuierlicher Betrieb) können nicht gleichzeitig ausgeführt werden

2.8.2 Schrittbetrieb

Um eine Kinematik im Handbetrieb um eine bestimmte Schrittweite zu verfahren, wird das Kommando [Jog Step auf Seite 37](#) verwendet.

Die Schrittweite ist die angegebene Distanz im angegebenen Koordinatensystem.

2.8.3 Kontinuierlicher Betrieb

Um eine Kinematik im Handbetrieb kontinuierlich in eine bestimmte Richtung zu verfahren, wird das Kommando [Jog Cont auf Seite 37](#) verwendet.

Solange das Kommando aktiv ist, verfährt die Kinematik in die angegebene Richtung im angegebenen Koordinatensystem.

Wird das Kommando beendet, wird die Kinematik mit den angegebenen Maximalwerten gestoppt.

2.9 Teach-in

2.9.1 Teach-in, Allgemein

Mit der Funktionalität Teach-in lassen sich Positionen einer Kinematik als Punkte speichern und wiederverwenden.

Die Namen der Punkte haben folgende Konvention:

- Namen haben eine maximale Länge von 20 Zeichen
- Namen dürfen keine Sonderzeichen " \ ; , * " enthalten
- Groß- und Kleinschreibung wird unterschieden

Um Punkte in einer Kinematik (im RAM) zu speichern, zu löschen oder zu laden werden folgende Funktionen verwendet:

- [Schreibe Punkt auf Seite 52](#)
- [Lese Punkt auf Seite 52](#)

- [Lese nächsten Punkt auf Seite 52](#)
- [Lösche Punkte auf Seite 53](#)

Punkte werden in einer Kinematik nicht remanent gespeichert, d. h. wenn eine Kinematik gelöscht wird oder die Steuerung ausgeschaltet wird, werden die Punkte gelöscht.

Um Punkte aus einer Kinematik in eine Punktedatei zu speichern oder Punkte aus einer Punktedatei in eine Kinematik zu laden, werden folgende Kommandos verwendet:

- [Speichere Punktedatei auf Seite 53](#)
- [Lade Punktedatei auf Seite 54](#)

Punkte werden in einer Punktedatei remanent gespeichert. Die Punktedateien werden auf der Compact Flash-Karte im Verzeichnis "<User-Partition> / PointFile" abgelegt.

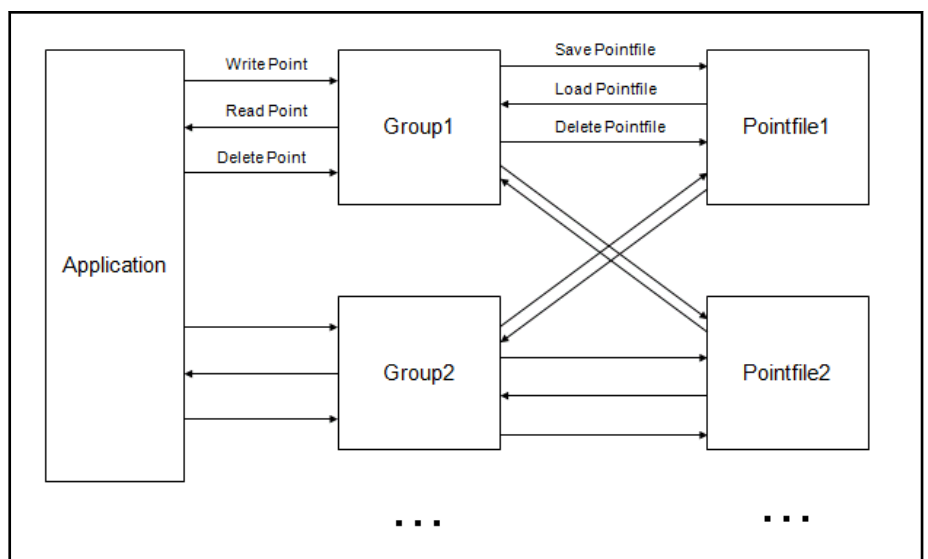
Um eine Punktedatei zu löschen, wird folgendes Kommando verwendet:

- [Lösche Punktedatei auf Seite 54](#)



Es können Punkte aus jeder Punktedatei in jede Kinematik geladen werden.

Es liegt in der Verantwortung des Anwenders zu prüfen, ob die geladenen Punkte von der gewählten Kinematik verfahren werden können.



Application	Anwender-Applikation
Group	Kinematik
Pointfile	Punktedatei
Write Point	Funktion "Schreibe Punkt"
Read Point	Funktion "Lese Punkt" oder "Lese nächsten Punkt"
Delete Point	Funktion "Lösche Punkte"
Save Pointfile	Kommando "Speichere Punktedatei"
Load Pointfile	Kommando "Lade Punktedatei"

Abb. 2-22: Übersicht der Funktionalität "Punkte teachen"

2.9.2 Aufbau der Punktedatei

Die Punktedatei hat das Format ".txt".

Pro Zeile ist ein Datensatz gespeichert.

Beispiel:

Sortierte Punktnamen einer Kinematik

1. Point1
2. Point10
3. Point2
4. point1

Wird der Punktnamen "Point1" übergeben, wird der Punkt "Point10" geladen.

2.9.6 Lösche Punkte

Um Punkte einer Kinematik zu löschen, wird die Funktion [Delete Points auf Seite 39](#) verwendet.

Mit dem Zeichen "*" kann eine Wildcard als letztes Zeichen angegeben werden.

Beispiel:

Punktnamen einer Kinematik

- Point11
- Point22
- point3

Wird der Punktnamen "Point11" übergeben, wird der Punkt "Point11" gelöscht.

Wird der Punktnamen "Point*" angegeben, werden die Punkte "Point11" und "Point22" gelöscht.

Wird der Punktnamen "*" angegeben, werden alle Punkte gelöscht.

2.9.7 Speichere Punktedatei

Um Punkte aus einer Kinematik in eine Punktedatei zu speichern, wird das Kommando [Save Point File auf Seite 39](#) verwendet.

Beim Speichern der Punktedatei gibt es folgende Optionen:

- Ist die angegebene Punktedatei noch nicht vorhanden, wird sie neu angelegt
- Ist die angegebene Punktedatei schon vorhanden, gibt es folgende Optionen:
 - **Neu**
Der Inhalt der vorhandenen Punktedatei wird gelöscht.
Die Punkte der Kinematik werden in die Punktedatei eingetragen.
 - **Hinzufügen**
Der Inhalt der vorhandenen Punktedatei bleibt erhalten, d. h. die vorhandenen Punkte behalten ihre Werte.
Punkte, die nicht in der vorhandenen Punktedatei enthalten sind, werden aus der Kinematik der Punktedatei hinzugefügt.
 - **Hinzufügen und Überschreiben**
Punkte, die in der vorhandenen Punktedatei und in der Kinematik enthalten sind, werden mit den Werten aus der Kinematik überschrieben.
Der Inhalt der anderen Punkte in der vorhandenen Punktedatei bleibt erhalten.

Funktionsbeschreibung

Die anderen Punkte der Kinematik werden der Punktedatei hinzugefügt.

2.9.8 Lade Punktedatei

Um Punkte aus einer Punktedatei in eine Kinematik zu laden, wird das Kommando [Load Point File auf Seite 39](#) verwendet.

Beim Laden der Punktedatei gibt es folgende Optionen:

- **Neu**
Der Inhalt der Kinematik wird gelöscht.
Die Punkte der Punktedatei werden in die Kinematik eingetragen.
- **Hinzufügen**
Der Inhalt der Kinematik bleibt erhalten, d. h. die vorhandenen Punkte behalten ihre Werte.
Punkte, die nicht in der Kinematik enthalten sind, werden aus der Punktedatei der Kinematik hinzugefügt.
- **Hinzufügen und Überschreiben**
Punkte, die in der Kinematik und in der vorhandenen Punktedatei enthalten sind, werden mit den Werten aus der Punktedatei überschrieben.
Der Inhalt der anderen Punkte in der Kinematik bleibt erhalten.
Die anderen Punkte der vorhandenen Punktedatei werden der Kinematik hinzugefügt.

2.9.9 Lösche Punktedatei

Um eine Punktedatei zu löschen, wird die Funktion [Delete Point File auf Seite 39](#) verwendet.

2.10 Überschleifen

2.10.1 Überschleifen, Allgemeines

Bei der koordinierten Bewegung einer Kinematik gibt es zwei Möglichkeiten den Endpunkt eines Bewegungskommandos anzufahren:

- **Endpunkt direkt anfahren (TO):**
Der Endpunkt P1 wird direkt angefahren, die Geschwindigkeit auf Null abgebremst. Erst dann wird das nächste Bewegungskommando (P1→P2) gestartet.

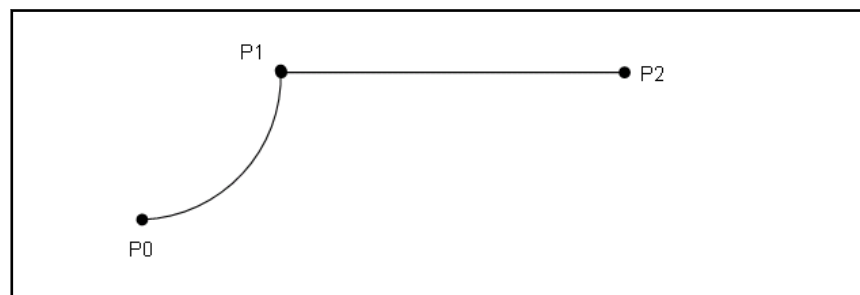


Abb. 2-23: TO

- **Endpunkt überschleifen (VIA):**
Der Endpunkt P1 wird mit dem nächsten Bewegungskommando überschleifen. Der Endpunkt wird nicht erreicht und die programmierte Bahn wird verlassen, um eine kontinuierliche Bewegung zu erreichen.

Funktionsbeschreibung

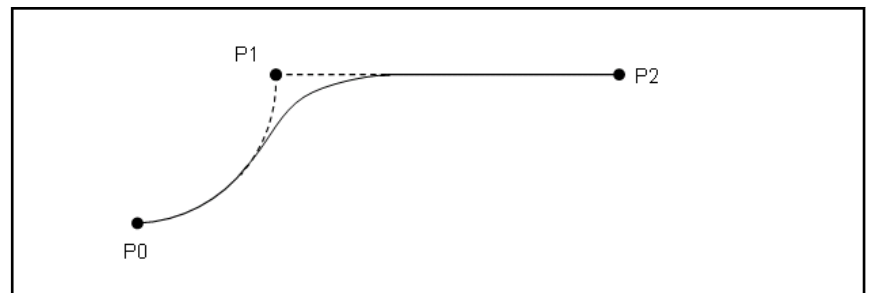


Abb. 2-24: VIA

Gründe für den Einsatz des Überschleifens sind:

- Durch eine kontinuierliche Bewegung wird die Mechanik geschont
- Durch eine kontinuierliche Bewegung werden Schwingungen an Greifer und Last reduziert
- Die Prozesszykluszeit wird verringert
- Ein applikativ erforderlicher, konstanter Geschwindigkeitsverlauf kann besser realisiert werden, z.B. zum Klebstoff auftragen, zum Lackieren, ...

2.10.2 Überschleifradius

Der programmierte Radius R_{Prog}

Mit dem Überschleifradius R wird angegeben, ab welcher Distanz zum Endpunkt eines Bewegungskommandos der Überschleifvorgang ausgelöst wird.

R_{Prog} ergibt sich bei der Bewegungsart PTP aus dem programmierten Prozentsatz multipliziert mit der Distanz im Koordinatensystem ACS.

Bei den Bewegungsarten Linear und Circular wird R_{Prog} als Distanz im Raum angegeben.

2.10.3 Slope-Arten

Mit der Wahl einer Slope-Art kann das Verhalten im Überschleifbereich festgelegt werden.

Slope-Arten	Vorteile	Nachteile
Block_Slope (verfrühtes Starten)	Schnelle Reaktionen möglich	Bahngeschwindigkeit und -beschleunigung ergeben sich aus der überlagerten Bewegung Die Bahn ergibt sich aus der überlagerten Bewegung Die Bahn ist abhängig von den Maximalwerten für das Geschwindigkeitsprofil
Cont_Slope (Spline)	Konstante Bahngeschwindigkeit Die Bahn ist unabhängig von Maximalwerten für das Geschwindigkeitsprofil	Keine schnelle Reaktion möglich wegen Splineberechnung

Tab. 2-10: Übersicht Slope-Arten

2.10.4 Block_Slope

Bei der Slope-Art "Block_Slope" wird das Überschleifen durch das verfrühte Starten des nächsten Bewegungskommandos realisiert.

Durch das verfrühte Starten überlagern sich die Bewegungskommandos. Dadurch ergibt sich im Überschleifbereich eine Bahn sowie eine Bahngeschwindigkeit und Bahnbeschleunigung.

Funktionsbeschreibung

Diese Bahn ist abhängig von den Maximalwerten für das Geschwindigkeitsprofil der Bewegungskommandos.

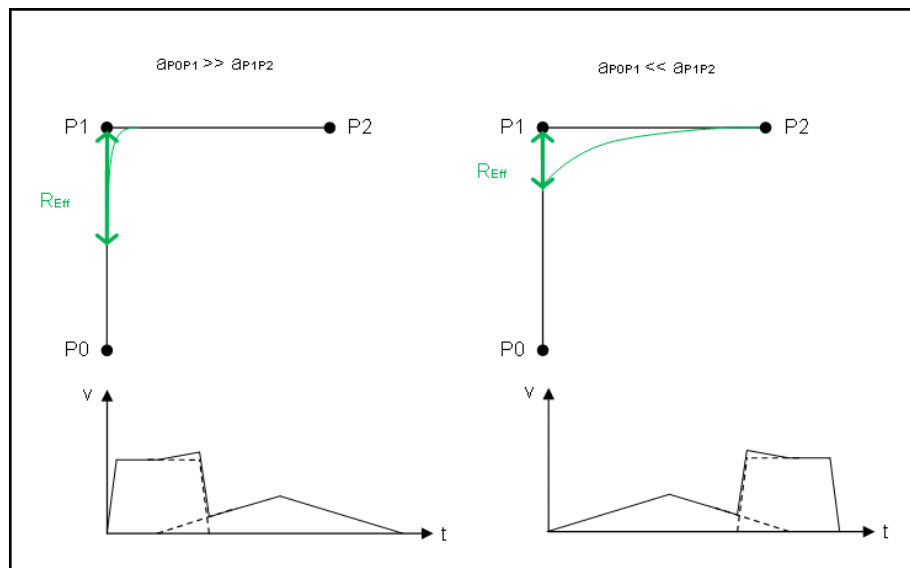


Abb. 2-25: Bahn in Abhängigkeit der Maximalwerte für das Geschwindigkeitsprofil

Der wirksame Radius R_{Eff}

Der wirksame (effective) Radius R_{Eff} ergibt sich beim Block_Slope je nach gewählter Option im Parameter [K-0-0099](#), Kinematikkonfiguration:

- **Überschleifen ohne Begrenzung:**

R_{Eff} wird nicht begrenzt. R_{Eff} entspricht somit dem programmierten Radius R_{Prog} .

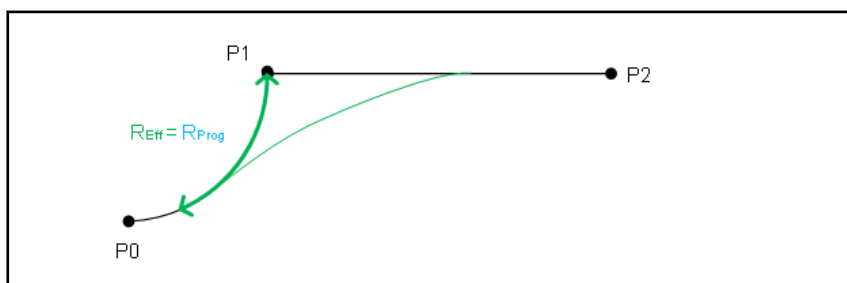


Abb. 2-26: Überschleifen ohne Begrenzung

- **Überschleifen mit Begrenzung auf halbe Bahndistanz:**

R_{Eff} wird begrenzt auf das Minimum von R_{Prog} , der Hälfte der Bahndistanz von P0 nach P1 und der Hälfte der Bahndistanz von P1 nach P2.

$$R_{Eff} = \min \left(R_{Prog}, \frac{1}{2} * \overline{P_0 P_1}, \frac{1}{2} * \overline{P_1 P_2} \right)$$

Funktionsbeschreibung

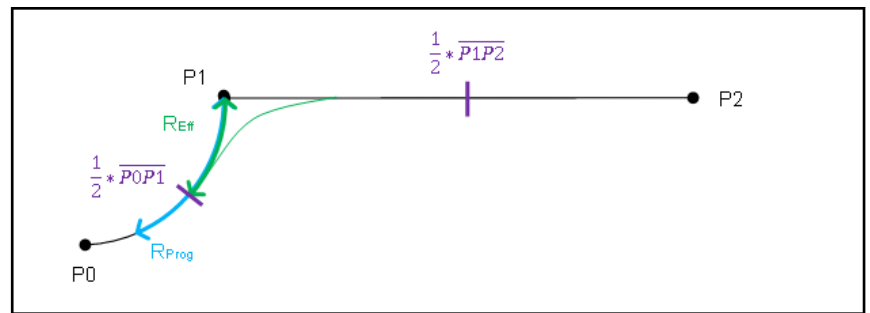
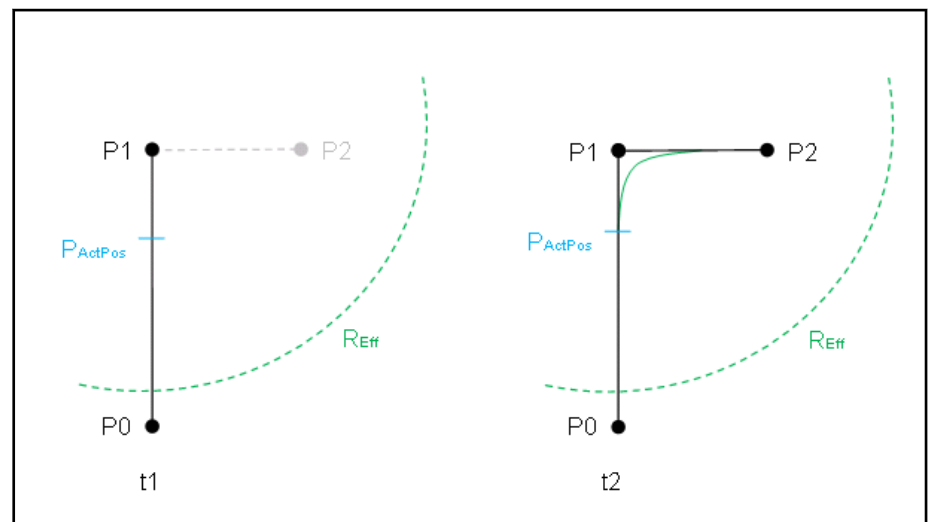


Abb. 2-27: Überschleifen mit Begrenzung auf halbe Bahndistanz

Spätes Überschleifen

Wenn ein Bewegungskommando, dessen Endpunkt mit VIA angefahren wird, schon aktiv ist und ein weiteres Bewegungskommando kommandiert wird, werden die beiden Bewegungskommandos spät überschleift.

Beim späten Überschleifen mit Block_Slope muss berücksichtigt werden, dass das zweite Bewegungskommando vorbereitet werden muss.

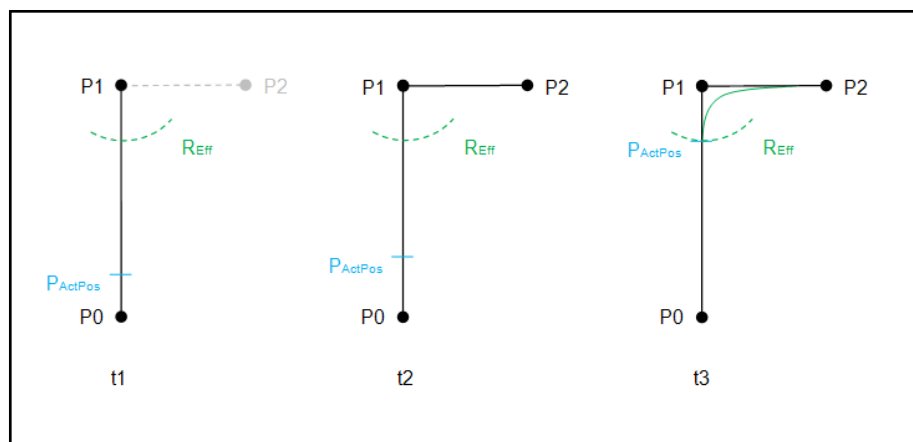


- t1 Zum Zeitpunkt t1 wird das Bewegungskommando nach P2 abgesetzt
- t2 Zum Zeitpunkt t2 ist das Bewegungskommando nach P2 vorbereitet

Abb. 2-28: Spätes Überschleifen Block_Slope

Wird das zweite Bewegungskommando kommandiert während sich die aktuelle Position noch nicht im Überschleifbereich R_{Eff} befindet, wird erst beim Eintritt in R_{Eff} überschleift.

Funktionsbeschreibung



- t1** Zum Zeitpunkt t1 wird das Bewegungskommando nach P2 abgesetzt
- t2** Zum Zeitpunkt t2 ist das Bewegungskommando nach P2 vorbereitet
- t3** Zum Zeitpunkt t3 ist die aktuelle Position in R_{Eff} eingetreten

Abb. 2-29:

Maximale Anzahl überlagerter Bewegungen

Die maximale Anzahl überlagerter Bewegungen ist abhängig von der gewählten Option im Parameter K-0-0099:

- Maximal 2 Bewegungskommandos können überlagert werden.
- Unbegrenzt viele Bewegungskommandos können überlagert werden.

2.10.5 Cont_Slope

Bei der Slope-Art "Cont_Slope" wird das Überschleifen durch das Verbinden zweier Bewegungskommandos mit einem Spline realisiert.

Durch den Spline wird ein konstanter Geschwindigkeitsverlauf gewährleistet. Der Spline wird dem ersten der beiden zu überschleifenden Bewegungskommandos zugeordnet und übernimmt dessen Maximalwerte für das Geschwindigkeitsprofil.

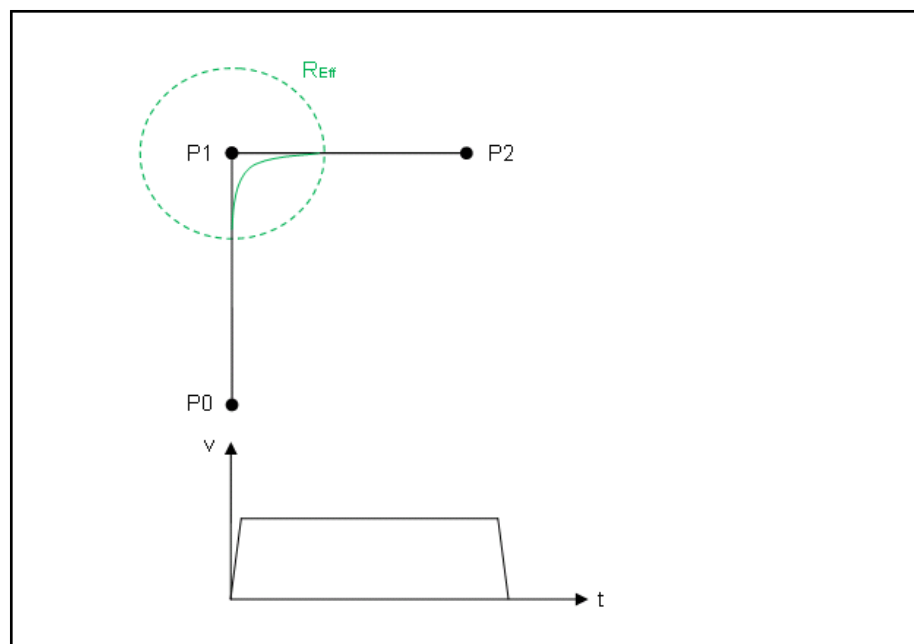


Abb. 2-30: Überschleifen mit Cont_Slope

Funktionsbeschreibung

Der wirksame Radius R_{Eff}

Der wirksame (effective) Radius R_{Eff} entspricht beim Cont_Slope dem programmierten Radius R_{Prog} .

R_{Eff} -Begrenzung bei TO, VIA

Wird auf ein Bewegungskommando überschiffen bei dem der Endpunkt mit TO angefahren wird, wird R_{Eff} nicht begrenzt.

Wird auf ein Bewegungskommando überschiffen bei dem der Endpunkt mit VIA angefahren wird, wird R_{Eff} auf die Hälfte der Distanz des Bewegungskommandos begrenzt. R_{Eff} wird begrenzt, da ein mit VIA angefahrener Endpunkt ebenfalls überschiffen werden kann.

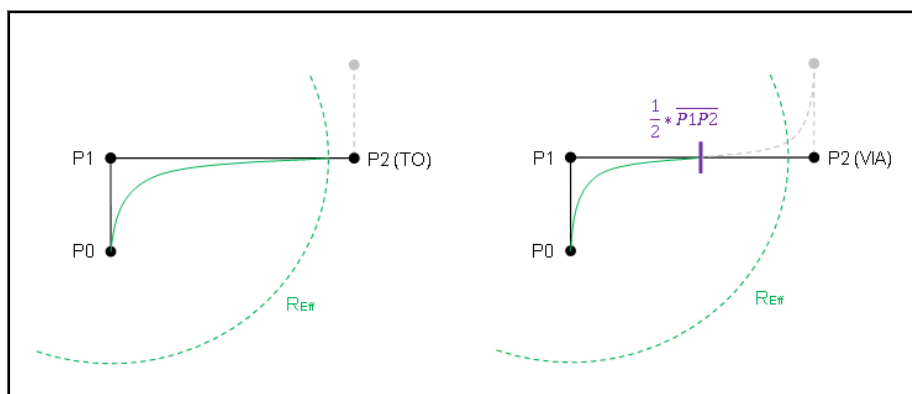
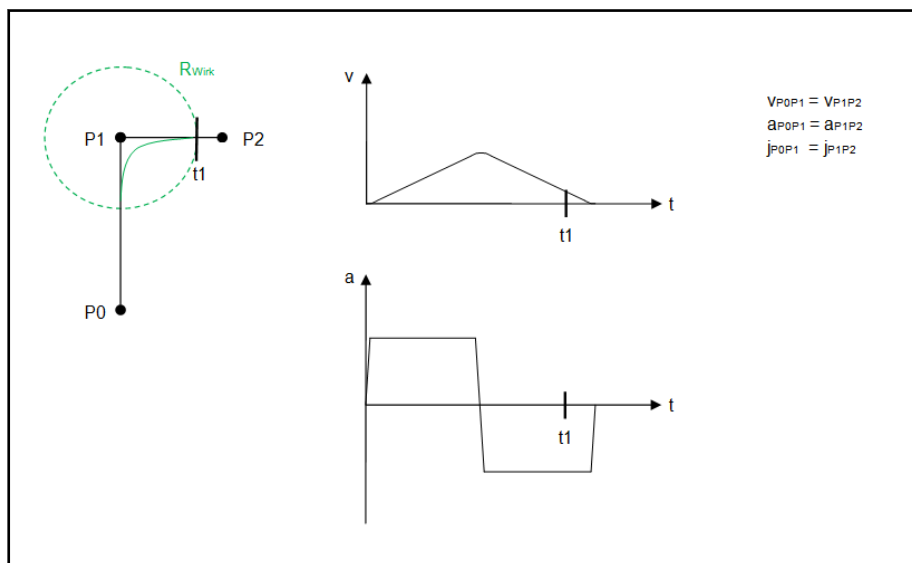


Abb. 2-31: R_{Eff} -Begrenzung bei TO, VIA

Übergangsbeschleunigungen

Haben beide Bewegungskommandos die gleichen Maximalwerte für das Geschwindigkeitsprofil, kann am Übergang der Kommandos mit einer Übergangsbeschleunigung überschiffen werden.

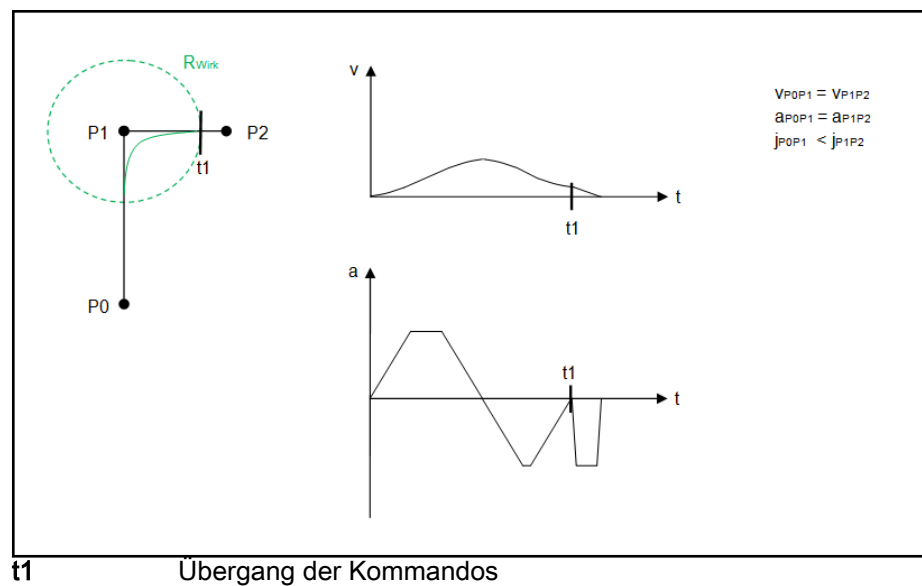


t1 Übergang der Kommandos

Abb. 2-32: Übergangsbeschleunigung - Gleiche Maximalwerte für das Geschwindigkeitsprofil

Haben beide Bewegungskommandos unterschiedliche Maximalwerte für das Geschwindigkeitsprofil, wird am Übergang der Kommandos die Beschleunigung auf Null gesetzt.

Funktionsbeschreibung



Übergang der Kommandos

Abb. 2-33: Übergangsbeschleunigung Null - Unterschiedliche Maximalwerte für den Ruck j

Spätes Überschleifen

Wenn ein Bewegungskommando, dessen Endpunkt mit VIA angefahren wird, schon aktiv ist und ein weiteres Bewegungskommando kommandiert wird, werden die beiden Bewegungskommandos spät überschleifen.

Beim späten Überschleifen mit Cont_Slope muss berücksichtigt werden, dass sowohl das zweite Bewegungskommando, als auch der Spline vorbereitet werden müssen. Diese Vorbereitungszeit muss über eine, im Parameter [K-0-0250](#) hinterlegte Vorhaltezeit (lead time) kompensiert werden. Die Vorhaltezeit muß im Rahmen der Inbetriebnahme für den jeweiligen Anwendungsfall angepasst werden.

Der Startpunkt des Splines ergibt sich aus der Vorhaltezeit multipliziert mit der maximalen Bahngeschwindigkeit und der Addition mit der aktuellen Position.

$$P_{\text{StartSpline}} = (t_{\text{LeadTime}} \cdot v_{\text{Max}}) + P_{\text{ActPos}}$$

Zum Zeitpunkt t_1 wird das Bewegungskommando nach P2 abgesetzt.

Zum Zeitpunkt t_2 ist der Spline und das Bewegungskommando nach P2 vorbereitet.

Funktionsbeschreibung

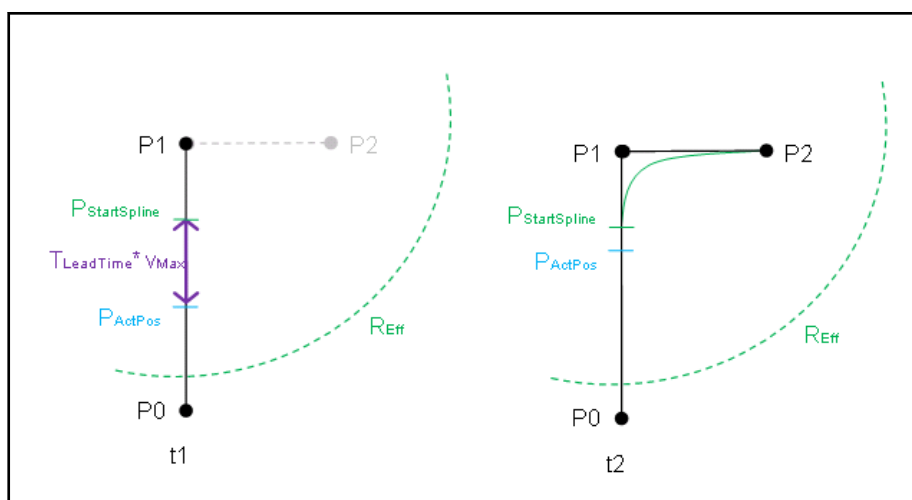


Abb. 2-34: Spätes Überschleifen Cont_Slope

Ist nach der Vorbereitung des zweiten Bewegungskommandos und des Splines der Startpunkt des Splines schon überschritten, ist das Überschleifen mit Cont_Slope fehlgeschlagen (Siehe Umschalten auf Block_Slope).

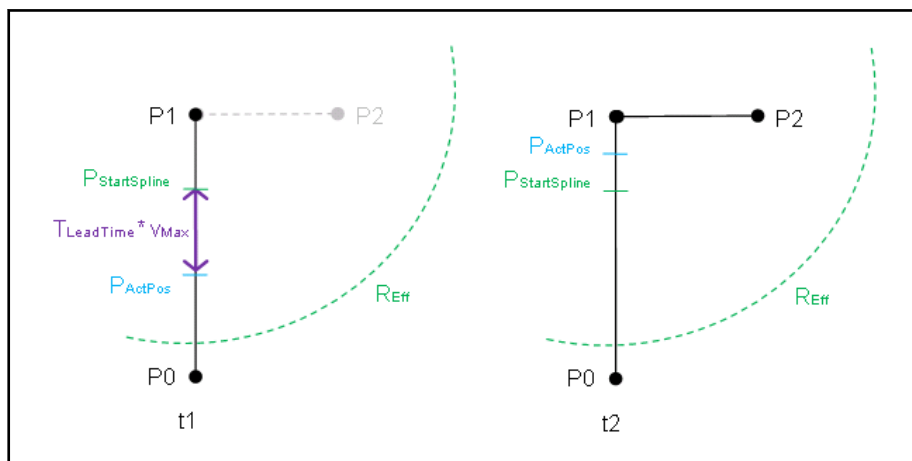


Abb. 2-35: Spätes Überschleifen Cont_Slope fehlgeschlagen

Umschalten auf Block_Slope

Ist das Überschleifen mit Cont_Slope fehlgeschlagen, ist das weitere Verhalten abhängig von der gewählten Option im Parameter [K-0-0099](#), Kinematikkonfiguration:

- Der Endpunkt wird mit TO angefahren
- Das Überschleifen wird mit der Slope-Art Block_Slope durchgeführt

Splineberechnung

Im Parameter [K-0-0021](#) (Kinematikstatus) wird mit einem Bit signalisiert, ob gerade eine Splineberechnung durchgeführt wird. Mit einem anderen Bit wird signalisiert, ob der Spline gültig ist.

2.10.6 Überschleifmöglichkeiten bei verschiedenen Bewegungsarten

Die folgenden Tabellen zeigen an, bei welchen Bewegungsarten überschleifen werden kann. Ist ein Überschleifen nicht möglich, wird der Endpunkt mit "TO" angefahren.

Block_Slope

		2. Kommando		
		PTP	Linear	Circular

Funktionsbeschreibung

1. Kommando	PTP	x	-	-
	Linear	-	x	x
	Circular	-	x	x

Tab. 2-11: Überschleifmöglichkeiten bei Block_Slope

Cont_Slope

		2. Kommando		
		PTP	Linear	Circular
1. Kommando	PTP	-	-	-
	Linear	-	x	x
	Circular	-	x	x

Tab. 2-12: Überschleifmöglichkeiten bei Cont_Slope

2.11 Robot Control, Move ID

Die Move ID ist ein numerischer Wert, der die aktuelle Position des Roboters auf der Bahn angibt.

Der ganzzahlige Teil ist eine Nummer der Bewegungsanweisung. Diese Nummer wird mit jeder neuen Bewegungsanweisung automatisch erhöht.

Der Dezimalteil zeigt den Fortschritt der Bewegung in Prozent an.

Beispiel:

Move ID einer Bewegung

- Get Move ID: Wurde noch kein Bewegungskommando abgesetzt, ist die Move ID 0
- Move Linear Absolute: Es wird ein Bewegungskommando abgesetzt
- Get Move ID: Nach Absetzen eines Kommandos liefert die Funktion Get Move ID z.B. den Wert 0.12. Dies bedeutet, dass die Bewegung zu 12% abgeschlossen ist
- Get Move ID: Am Ende einer Bewegung liefert die Funktion Get Move ID den Wert 1.00. Dies bedeutet, dass die Bewegung komplett abgeschlossen ist und kein weiteres Kommando aktiv ist

2.11.1 Robot Control, Move ID, Setzen

Die Move ID kann über die Funktion [Set Move ID auf Seite 39](#) gesetzt werden. Die neue Move ID gilt für alle Kommandos, die nach dem Setzen der Move ID kommandiert werden. Sätze, die sich bereits im Satzpuffer befinden, werden nicht beeinflusst.

Beispiel:

Setzen der Move ID

- Move Linear Absolute: Es wird ein Bewegungskommando abgesetzt
- Get Move ID: Nach Absetzen eines Kommandos liefert die Funktion Get Move ID z.B. den Wert 0.12. Dies bedeutet, dass die Bewegung zu 12% abgeschlossen ist
- Set Move ID (15): Für neue Kommandos wird die Move ID 15 gesetzt
- Move Linear Absolute: Es wird ein zweites Bewegungskommando abgesetzt

Funktionsbeschreibung

- Get Move ID: Die Funktion Get Move ID liefert z.B. den Wert 0.75. Dies bedeutet, dass die erste Bewegung zu 75% abgeschlossen ist
- Get Move ID: Die Funktion Get Move ID liefert z.B. den Wert 15.25. Dies bedeutet, dass die zweite Bewegung zu 25% abgeschlossen ist

2.11.2 Robot Control, Move ID, Überschleifen

Block Slope Das Block Slope Verfahren überschleift durch verfrühtes Starten des Folgesatzes.

Die Move ID ergibt sich aus der Addition des Fortschritts der parallel laufenden Einzelsätze.

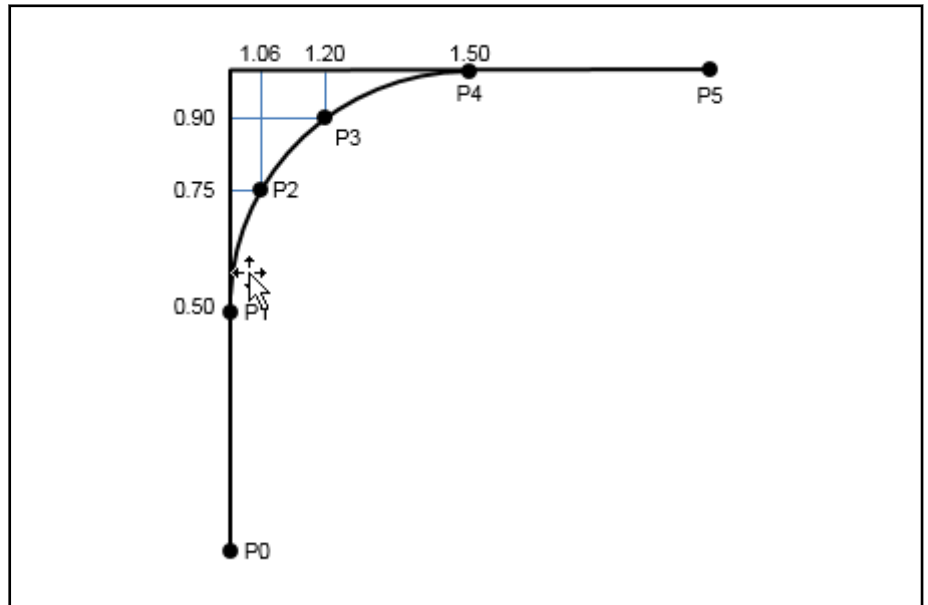


Abb. 2-36: Move ID Block Slope

- P0: An diesem Punkt startet der erste Satz. Die Move ID ist 0.0
- P1: An diesem Punkt startet der zweite Satz. Der erste Satz ist zu 50% abgearbeitet. Die Move ID ist 0.5
- P2: An diesem Punkt befindet ist der erste Satz zu 75% und der zweite Satz ist zu 6% abgearbeitet. Die Move ID ist somit 0.81
- P3: An diesem Punkt befindet ist der erste Satz zu 90% und der zweite Satz ist zu 20% abgearbeitet. Die Move ID ist somit 1.1
- P4: An diesem Punkt befindet ist der erste Satz komplett und der zweite Satz ist zu 50% abgearbeitet. Die Move ID ist somit 1.5
- P5: An diesem Punkt sind beide Sätze komplett abgearbeitet. Die Move ID ist somit 2.0

Cont Slope Das Cont Slope Verfahren überschleift, indem im Überschleifbereich die ursprünglichen Sätze gekürzt und durch einen Spline ersetzt werden. Die Move ID im Überschleifbereich ergibt sich durch den Abarbeitungsfortschritt des Splines.

Funktionsbeschreibung

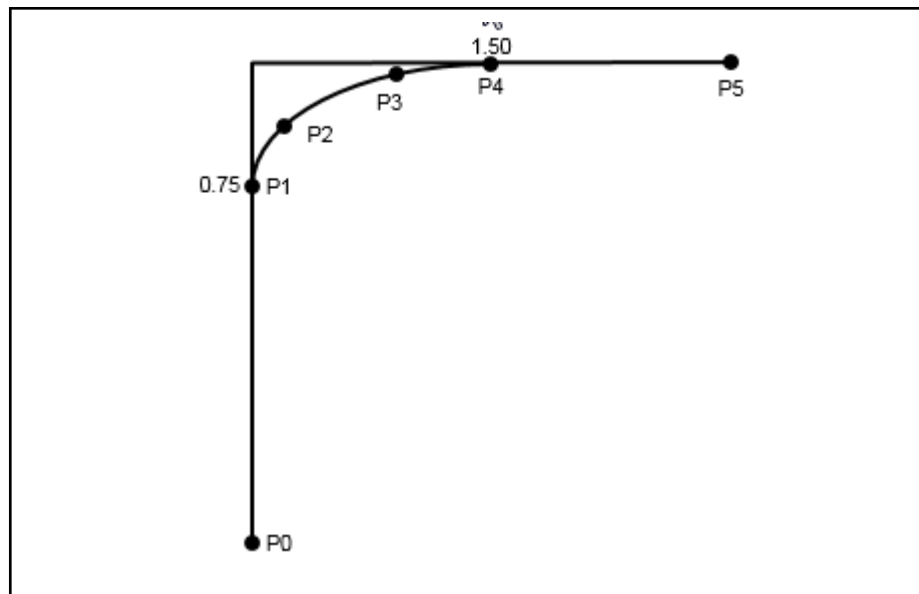


Abb. 2-37: Move ID Cont Slope

- P0: An diesem Punkt startet der erste Satz. Die Move ID ist 0.0
- P1: An diesem Punkt startet der Überschleifbereich. Der erste Satz ist zu 75% abgearbeitet. Die Move ID ist 0.75
- P2: An diesem Punkt ist der Spline zu 20% abgearbeitet. Da der Spline zwischen 0.75 und 1.5 abgearbeitet wird, ergibt sich eine Move ID von 0.9 ($0.75 + 20\%(1.5 - 0.75)$)
- P3: An diesem Punkt ist der Spline zu 80% abgearbeitet. Da der Spline zwischen 0.75 und 1.5 abgearbeitet wird, ergibt sich eine Move ID von 1.35 ($0.75 + 80\%(1.5 - 0.75)$)
- P4: An diesem Punkt ist der Spline komplett abgearbeitet. Die Move ID ist somit 1.5
- P5: An diesem Punkt ist der zweite Satz komplett abgearbeitet. Die Move ID ist somit 2.0

2.12 Stoppen, Unterbrechen, Wiederaufnahme von Bewegungen

2.12.1 Stoppen von Bewegungen

Um alle Bewegungen einer Kinematik zu stoppen, wird das Kommando [Stop auf Seite 37](#) verwendet.

Der Stopp-Vorgang hat folgende Eigenschaften:

- die Kinematik wird kontrolliert gestoppt
- alle aktiven Bewegungskommandos werden mit den angegebenen Werten für Verzögerung und Ruck auf der programmierten Bahn gestoppt



Sind die angegebenen Werte für Verzögerung und Ruck kleiner als die jeweiligen Werte des Bewegungskommandos, so werden die Werte des Bewegungskommandos für den Stopp-Vorgang verwendet.

- alle Kommandos im Satzpuffer werden abgebrochen
- vorhandene Bandsynchronität wird über die angegebene Verzögerung und Ruck abgebaut



Sind die angegebene Verzögerung oder der Ruck 0, so werden die Verzögerung oder der Ruck der bandspezifischen Fehlerreaktion verwendet (siehe [Bandspezifische Fehlerreaktion auf Seite 69](#)).

- der Inhalt des Satzpuffers wird geleert

Das folgende Diagramm veranschaulicht den Geschwindigkeitsverlauf bei einem Stop während einer Aufsynchonisierungsbewegung.

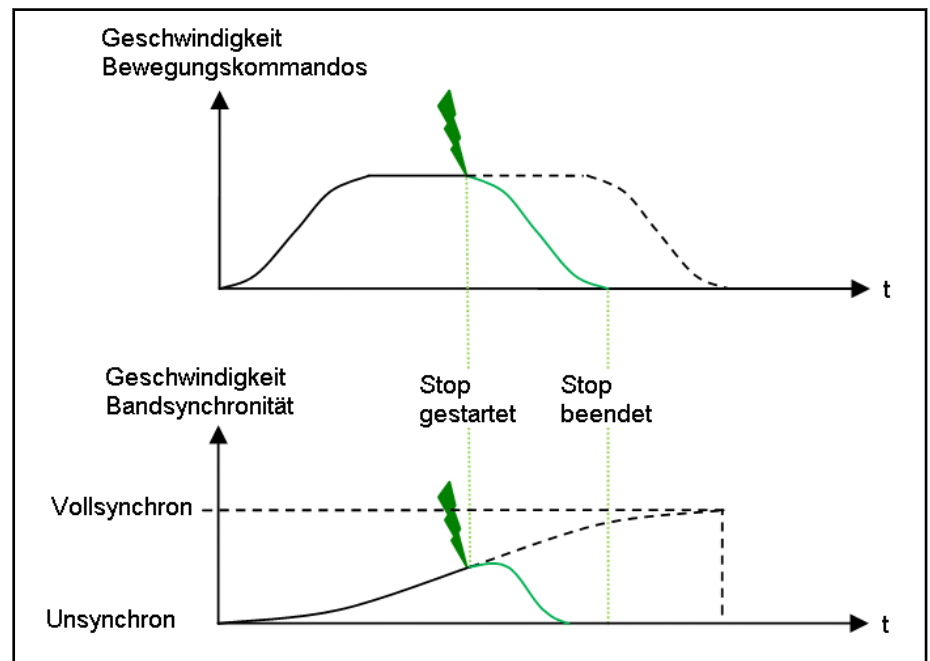


Abb. 2-38: Geschwindigkeitsverlauf bei einem Stop

Die Zustandsmaschine der Kinematik befindet sich während des Stopp-Vorgangs im Zustand "Stopping". Ist der Stopp-Vorgang beendet, wird die Zustandsmaschine in den Zustand "Standstill Complete" geschaltet.

Im Zustand "Stopping" kann kein Kommando abgesetzt werden.

Der Stopp-Vorgang ist beendet wenn:

1. die Kinematik Stillstand erreicht hat
2. das Stoppen durch den Anwender beendet wurde

2.12.2 Unterbrechen von Bewegungen

Um alle Bewegungen einer Kinematik zu unterbrechen wird das Kommando [Interrupt auf Seite 37](#) verwendet.

Der Interrupt-Vorgang hat folgende Eigenschaften:

- die Kinematik wird kontrolliert gestoppt
- alle aktiven Bewegungskommandos werden mit angegebenen Werten für Verzögerung und Ruck auf ihrer Bahn gestoppt

Funktionsbeschreibung



Sind die angegebenen Werte für Verzögerung und Ruck kleiner als die jeweiligen Werte des Bewegungskommandos, so werden die Werte des Bewegungskommandos für den Stopp-Vorgang verwendet.

- vorhandene Bandsynchronität bleibt erhalten
- der Inhalt des Satzpuffers bleibt erhalten

Das folgende Diagramm veranschaulicht den Geschwindigkeitsverlauf bei einem Interrupt während einer Aufsynchronisierungsbewegung.

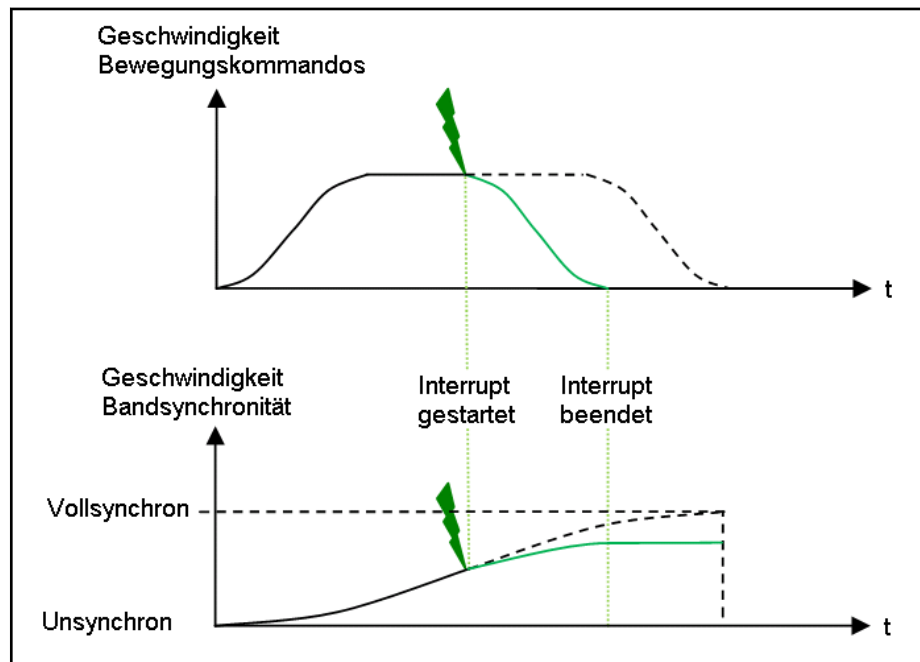


Abb. 2-39: Geschwindigkeitsverlauf bei einem Interrupt

Die Zustandsmaschine behält ihren momentanen Zustand bei.

Während des Interrupt-Vorgangs können weitere Kommandos abgesetzt werden.

Der Interrupt-Vorgang ist beendet wenn:

1. die Bewegung mit dem Kommando [Continue auf Seite 37](#) wiederaufgenommen wird
2. die Bewegung mit dem Kommando [Stop auf Seite 37](#) gestoppt wird

2.12.3 Wiederaufnahme von Bewegungen

Um die durch das Kommando [Interrupt auf Seite 37](#) unterbrochene Bewegungen einer Kinematik wieder aufzunehmen, wird das Kommando [Continue auf Seite 37](#) verwendet.

Der Continue-Vorgang hat folgende Eigenschaften:

- alle Bewegungskommandos werden mit ihrer ursprünglichen Verzögerung und Ruck weiterverfahren
- der Interrupt-Vorgang wird abgebrochen

Die Zustandsmaschine behält ihren momentanen Zustand bei.

Während des Continue-Vorgangs können weitere Kommandos abgesetzt werden.

Der Continue-Vorgang ist beendet wenn:

1. der Interrupt-Vorgang beendet wurde
2. alle Bewegungskommandos ihre Bewegungen wiederaufgenommen haben

2.12.4 Warten

Mit dem Kommando [Wait auf Seite 37](#) kann eine wählbare Zeit zwischen zwei anderen Kommandos gewartet werden. Das Wait-Kommando verzögert die Ausführung aller weiteren Kommandos um die angegebenen Zeit. Dadurch ist es möglich sofort nach Kommandieren des Wait-Kommandos weitere Kommando in den Kommandopuffer zu schreiben. Diese weiteren Kommandos werden dann aber erst verzögert gestartet.



Zwei Motion-Kommandos, die durch ein Wait-Kommando getrennt sind, werden nicht überschliffen.

2.13 Bandsynchrone Bewegung

2.13.1 Bandsynchrone Bewegung, Allgemeines

Die Funktion "Bandsynchron" ermöglicht, dass die Kinematikbewegung zu einem Montage- oder Förderband hinsichtlich Position und Orientierung synchronisiert werden kann. Hierbei kann das Band beliebig vor- oder rückwärts bewegt werden, seine Geschwindigkeit ändern oder stoppen. Je nach Konfiguration muss das Band einen geradlinigen oder kreisförmigen Verlauf haben und kann beliebig im Raum liegen. Die Maximalanzahl der verfügbaren Bänder pro Kinematik entspricht 16, abzüglich Kinematikachsen (Positionier-/Orientierungs-/Hilfsachsen).

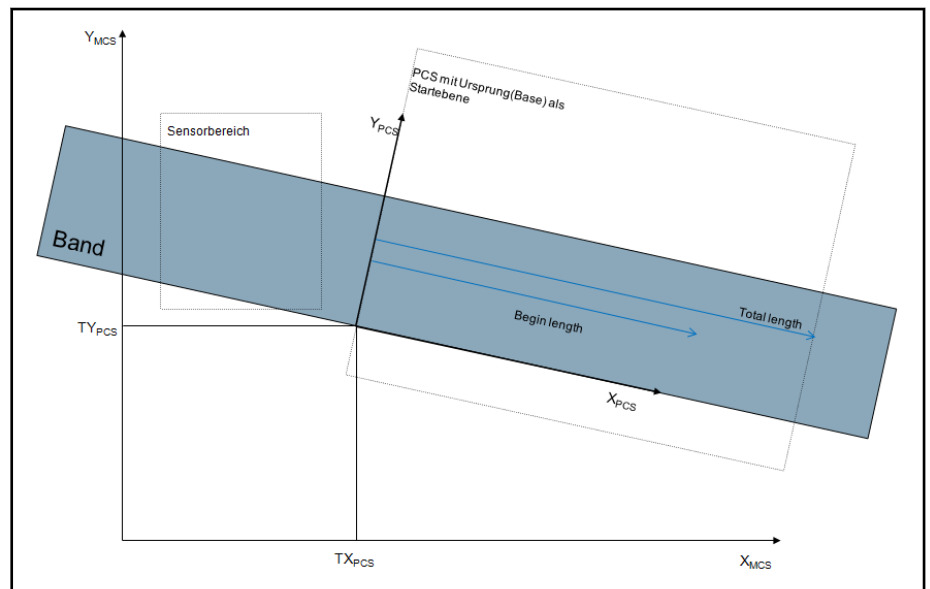


Abb. 2-40: Übersicht Bandsynchron - geradliniges Band

Das im Bild dargestellte, geradlinige Band wird durch ein Produktkoordinatensystem ([Produkt-Koordinatensystem \(PCS\) auf Seite 27](#)) repräsentiert. " TX_{PCS} " und " TY_{PCS} " stellen hierbei die Verschiebung der Startebene im Maschinenkoordinatensystem dar. "Begin length" und "Total length" definieren den Abstand zwischen Startebene und dem Punkt des spätesten Aufsynchronisierens bzw. der Bandlänge und liegen beide in Richtung der X-Koordinate des PCS.

Funktionsbeschreibung

Desweiteren kann auch der, vor der Startebene befindliche, Sensorbereich durch ein eigenes Produktkoordinatensystem realisiert werden, wodurch die Kommandierung der Kinematik erleichtert wird.

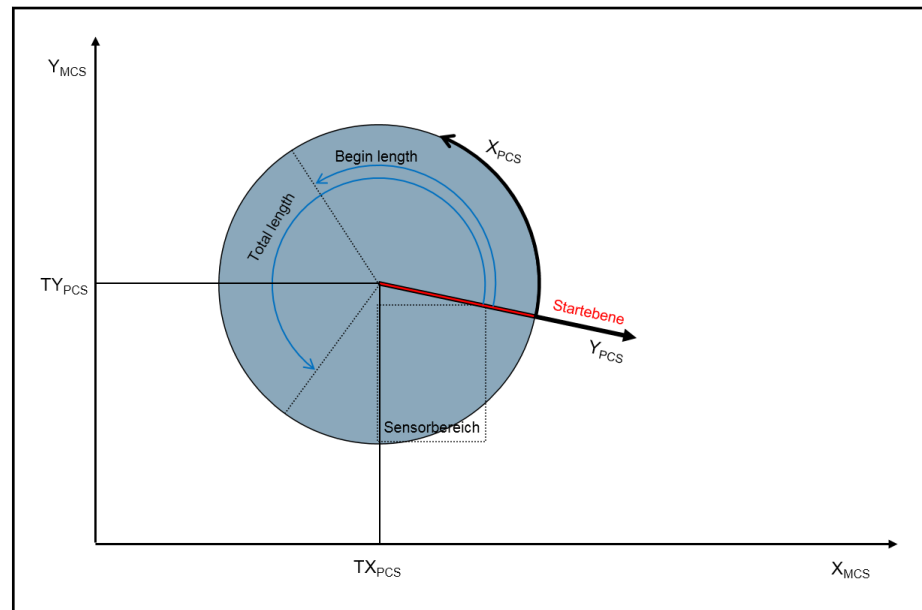


Abb. 2-41: Übersicht Bandsynchron - Drehtisch

Der im Bild dargestellte Drehtisch wird, äquivalent zum geradlinigen Band, ebenfalls durch ein Produktkoordinatensystem (**Produkt-Koordinatensystem (PCS)** auf Seite 27) repräsentiert. Auch hier stellen " TX_{PCS} " und " TY_{PCS} " die Verschiebung der Startebene im Maschinenkoordinatensystem dar. "Begin length" und "Total length" hingegen definieren die Winkelverdrrehung des Bandes zur X-Achse des PCS.

Unabhängig der verwendeten Bandart wird die Bewegung des Bandes in der Kinematik automatisch intern erfasst. Aktuelle Bandwerte können über den Parameter **K-0-0100** (Aktuelle Position, Kartesische Maschinenkoordinaten oder über die KinData-Schnittstelle gelesen werden.

2.13.2 Konfiguration eines Bandes

Die Konfiguration eines Bandes besteht aus folgenden Teilen:

- [Anlegen eines Bandes im K-0-0008 auf Seite 68](#)
- [Grundkonfiguration auf Seite 68](#)
- [Bandspezifische Fehlerreaktion auf Seite 69](#)

Anlegen eines Bandes im K-0-0008

Ein Band muss im Parameter **K-0-0008** (Achskonfigurationsliste) angelegt werden. Dies kann entweder beim Hinzufügen einer Kinematik (Hinzufügen einer neuen Kinematik, Schritt 2: Achszuordnung, Typ Bandachse) oder durch direktes Beschreiben des Parameters erfolgen. Erst danach kann das Band verwendet werden.

Grundkonfiguration

Vor dem bandsynchronen Verfahren einer Kinematik muss das Band/die Bänder einmalig konfiguriert werden. Hierzu verwendet man in Abhängigkeit der Bandart (lineares Band oder Drehtisch) das Kommando [Set Belt Configuration auf Seite 38](#) oder [Set Rotary Table Configuration auf Seite 38](#). Somit wird automatisch ein Produktkoordinatensystem, beginnend beim PCS1, lückenlos nummeriert angelegt (z. B.: Band 4 entspricht PCS 4). Ein zuvor konfiguriertes PCS wird von diesen Kommandos automatisch überschrieben. Somit müssen PCS, die keine Bänder darstellen, dahinter konfiguriert werden.

Funktionsbeschreibung

Will man die aktuelle Konfiguration eines Bandes ermitteln, so kann man das Kommando [Get Belt Configuration auf Seite 38](#) verwenden. Desweiteren ist es möglich mittels des Kommandos [Get Trafo Values auf Seite 38](#) das hinter dem Band stehenden Produktkoordinatensystems auszulesen.

Die Grundkonfiguration besteht aus folgenden Teilen:

- **Lage im Raum (Origin):** Die Lage der Y-Z-Ebene des Produktkoordinatensystems definiert die Startebene. Die Lage der X-Achse definiert die Bandrichtung.
- **Begin Length:** Definiert den letztmöglichen Start der Aufsynchronisierbewegung
- **Total Length:** Definiert das Ende des bandsynchronen Arbeitsbereichs



Der bandsynchrone Arbeitsbereich wird durch die Startebene und "Total Length" begrenzt. Nur in diesem Bereich kann eine Kinematik bandsynchron verfahren.

Die "Begin Length" im bandsynchronen Arbeitsbereich definiert den letztmöglichen Start der Aufsynchronisierbewegung. Diese Länge ist so zu wählen, dass die komplette bandsynchrone Bewegung inklusive Absynchronisierbewegung bis zur "Total Length" abgeschlossen ist.

Bandspezifische Fehlerreaktion

Zusätzlich zu der vorher beschriebenen Grundkonfiguration besteht die Möglichkeit, eine bandspezifische Fehlerreaktion für jedes konfigurierte Band festzulegen.

Um eine Fehlerreaktion zu konfigurieren, verwendet man das Kommando [Set Belt Error Reaction auf Seite 38](#)

Die konfigurierte Fehlerreaktion kann mit "[Get Belt Error Reaction auf Seite 38](#)" ausgelesen werden.



Wird keine Fehlerreaktion über "Set Belt Error Reaction" definiert, wird "Absynchronisieren über Grenzwerte" als Bandfehlerreaktion gesetzt. Das bedeutet, dass die Kinematik im Fehlerfall die Bandsynchronität mit den Standardwerten für Verzögerung (1000 mm/s²) und Ruck (10000 mm/s³) abbaut.

2.13.3 Bandwerte und Punkte auf dem Band

Punkte können mit der Bewegung von Bändern gekoppelt werden:

Bewegt sich das Band, so bewegt sich auch der Punkt in gleicher Weise. Die Kopplung eines Punktes mit einem Band erfolgt durch den sogenannten Bandwert. Der Bandwert ist dabei Teil der Koordinaten jedes Punktes. Dabei besitzt jedes Band seinen eigenen Bandwert. Die Position des Bandwerts innerhalb der Koordinaten legt dabei die Achskonfiguration der Kinematik fest:

Beispiel:

3-Achs-Delta mit einer Orientierungsachse und 2 Bändern

Es wurde ein 3-Achs-Delta mit einer Orientierungsachse und 2 Bändern konfiguriert. Somit ist die fünfte Koordinate jedes Punktes der Bandwert für das erste Band und die sechste Koordinate der Bandwert für das zweite Band.

Punkte die nicht mit einem Band gekoppelt sind, haben in der entsprechenden Koordinate einen Bandwert von 0. Punkte, die mit einem Band gekoppelt sind, haben in der entsprechenden Koordinate einen Bandwert ungleich 0.

Neben dem Bandwert eines Punktes, gibt es einen aktuellen Bandwert eines Bandes, der die aktuelle Position des Bandes repräsentiert. Der aktuelle

Funktionsbeschreibung

Bandwert verändert sich in gleicher Weise, wie sich das Band bewegt. Die aktuellen Bandwerte können aus der aktuellen Position der Kinematik ([K-0-0100](#)) ausgelesen werden.



Der aktuelle Bandwert eines Bandes ist nicht gleich des Positionswerts der Achse, die dem Band zugeordnet ist. Die beiden Werte verändern sich zwar in der gleichen Weise, jedoch kann ein Versatz bzw. eine andere Modulation existieren.

Auch für den aktuellen Bandwert gilt:

Die Position des Bandwerts innerhalb des Parameter [K-0-0100](#) wird durch die Achskonfiguration der Kinematik festgelegt:

Beispiel:

3-Achs-Delta mit einer Orientierungsachse und 2 Bändern

Es wurde ein 3-Achs-Delta mit einer Orientierungsachse und 2 Bändern konfiguriert. Somit ist die fünfte Koordinate im Parameter [K-0-0100](#) der Bandwert für das erste Band und die sechste Koordinate im Parameter [K-0-0100](#) der Bandwert für das zweite Band:

K100 sei $P_{MCS} = (X, Y, Z, \phi, b1, b2)$

- b1: Bandwert für erstes Band
- b2: Bandwert für zweites Band

Soll ein Punkt mit der Bewegung eines Bandes gekoppelt werden, muss der aktuelle Bandwert des entsprechenden Bandes zum Zeitpunkt der Punktaufnahme gelesen werden und der aufgenommene Punkt zugeordnet werden, der gekoppelt werden soll. Dazu muss der aktuelle Bandwert des Bandes aus der entsprechenden Koordinate aus dem Parameter [K-0-0100](#) gelesen werden und in die entsprechende Koordinate des Punktes kopiert werden. Der Punkt kann sich dabei in einem beliebigen Koordinatensystem befinden. Der Bandwert ist in jedem Koordinatensystem gleich.

Beispiel:

3-Achs-Delta mit einer Orientierungsachse und 2 Bändern

Für diese Kamera wurde ein Produktkoordinatensystem PCS3 definiert und so gegenüber des MCS verdreht und verschoben, dass sich die Erfassungsebene der Kamera in gleicher Höhe mit dem Band befindet. Es wurde ein Produkt von der Kamera erkannt. Die Kamera liefert als Koordinaten des Produkts $P_{PCS3} = (5, 8, 0, 45, 0, 0)$. Der Punkt liegt an $X = 5$, $Y = 8$ und ist um 45 Grad gedreht. Zum Zeitpunkt des Fotografierens wurde die aktuelle Position des Roboters ([K-0-0100](#)) erfasst. Diese sei $P_{MCS} = (10, 10, -700, 0, 20, 30)$. Befindet sich das Produkt auf dem ersten Band, so ist der mit dem Band gekoppelte Punkt $P_{PCS3} = (5, 8, 0, 45, 20, 0)$.



Es ist sicher gestellt, dass der aktuelle Bandwert im Parameter [K-0-0100](#) immer einen Wert ungleich 0 besitzt. Dadurch ist nach Kopieren des aktuellen Bandwerts eines Bands aus dem Parameter [K-0-0100](#) in einen Punkt der Punkt immer mit einem Band gekoppelt.

Mit diesen Angaben ist es dem System möglich die aktuelle Position eines, mit einem Band gekoppeltem Punktes zu jedem Zeitpunkt zu bestimmen. Dazu wird intern der aufgenommene Bandwert mit dem aktuellen Bandwert verrechnet (siehe Anwendungsbeispiel "Bandsynchronität" [Bandsynchronität auf](#)

Seite 400). Für die Verschiebung eines Punktes auf einen Punkt hinter die Startebene gilt folgende allgemeine Abbildungsvorschrift:

$$P'_{x(n)} = P_{x(n-1)\_PCS(k)} - B_{P(n)\_PCS(k)} + B_{ref}$$

$$P'_{n\_PCS(k)} = (P'_{x(n)}, P_y, \dots, B_{ref})$$

k	Bandnummer
P'_{x(n)}	X-Wert des Punktes zum Zeitpunkt n verschoben auf die aktuelle Bandposition
P_{x(n-1)_PCS(k)}	X-Wert des Punktes zum Zeitpunkt (n-1) im Produkt-Koordinatensystem k
B_{Ref}	Bandwert des Punktes P zum Zeitpunkt der Punkterfassung
B_{P(n)_PCS(k)}	Bandwert des Punktes P zum Zeitpunkt n im Produkt-Koordinatensystem k
P'_{n_PCS(k)}	Punkt hinter der Startebene zum Zeitpunkt n im Produkt-Koordinatensystem k

Abb. 2-42: Abbildung eines Punktes P zum Zeitpunkt n-1 auf einen Punkt hinter der Startebene zum Zeitpunkt n

Zum einen werden gekoppelte Punkte beim Aufsynchronisieren und vollsynchrone Fahrt automatisch von der Steuerung verfolgt, zum anderen kann der Anwender eigene Punkte mittels der Funktion [Move Point auf Seite 38](#) beispielsweise für Anzeigezwecke auf dem Band verfolgen. Diese Funktion verschiebt einen gekoppelten Punkt auf den aktuellen Bandwert und rechnet die Position im Raum auf den neuen Bandwert um. Da das Band immer entlang der X-Koordinate eines Band-PCS ausgerichtet ist, wird zur Verschiebung des Punktes auf dem Band lediglich seine X-Koordinate im Band-PCS verschoben. Durch Transformation in andere Koordinatensysteme ergeben sich durch Bandverdrehungen oder durch die Verwendung einer Kreisscheibe Änderungen in anderen Koordinaten.

Beispiel:

Verfolgen von Punkten

Es ist ein zweiachsiger Kartheser mit einem Band (PCS1) konfiguriert. Das Band (PCS1) ist gegenüber dem MCS nicht verschoben und um 90 Grad verdreht. Das Band bewegt sich in X-Richtung im PCS1 bzw. in Y-Richtung im MCS. Es wurde ein Punkt $P_{MCS} = (3, 5)$ mit diesem Band gekoppelt. Zu diesem Zeitpunkt hatte das Band den Bandwert 10. Der gekoppelte Punkt hat somit die Koordinaten $P_{MCS} = (3, 5, 10)$ entspricht $P_{PCS1} = (5, -3, 10)$. Bewegt sich das Band nun um 100 weiter, so ist der neue Bandwert 110. Wird die Funktion Move Point mit $P_{MCS} = (3, 5, 10)$ aufgerufen, so liefert die Funktion $P'_{MCS} = (3, 105, 110)$ zurück. Transformiert man diesen Punkt in das Koordinatensystem PCS1, so ergibt sich der Punkt $P'_{PCS1} = (105, -3, 110)$.

Das Band besitzt einen Modulowert, welcher unabhängig von der Bandachse moduliert. Standardmäßig moduliert der Bandwert bei 10000.

Im Parameter [K-0-0099](#) (Kinematikkonfiguration) kann optional der Band-Modulowert gleich dem Achs-Modulowert gesetzt werden.



Die "Total Length" muss so gewählt werden, dass sie immer kleiner oder gleich der Hälfte des Bandmodulo-Wertes ist. Diese Bedingung gewährleistet die sichere Erkennung des Bandmodulo-Überlaufs.

Funktionsbeschreibung

2.13.4 Betrieb eines Bandes

Der laufende Betrieb teilt sich in folgende Phasen auf:

- [Aufsynchronisieren auf Seite 72](#)
- [Vollsynchrones Verfahren auf Seite 73](#)
- [Absynchronisieren auf Seite 73](#)

Die folgende Übersicht stellt mögliche Bewegungskommandos in den einzelnen Phasen dar:

Referenz Beschreibung	Aufsynchronisieren	Vollsynchرون verfahren	Absynchronisieren
Move Direct Absolute	x	x	x
Move Direct Relative		x	x
Move Linear Absolute	x	x	x
Move Linear Relative		x	x
Move Circular Absolute	x	x	x
Move Circular Relative		x	x
Synchronization with Limits	x		
Desynchronization with Limits			x

Tab. 2-13: Übersicht der Kommandierungsmöglichkeiten für den bandsynchronen Betrieb

Aufsynchronisieren

Für das Aufsynchronisieren mittels [Move Linear Absolute auf Seite 37](#), [Move Direct Absolute auf Seite 37](#) oder [Move Circular Absolute auf Seite 37](#) muss lediglich ein mit einem Band gekoppelter Punkt kommandiert werden.

Falls dieser Punkt vor der Startebene liegt, wartet die Kinematik standardmäßig bis der Endpunkt die Startebene durchdrungen hat und fährt anschließend den Endpunkt synchron zur Bandbewegung an. Eine Alternative wird in [Adaptive Motion auf Seite 74](#) beschrieben.

Mit Erreichen des kommandierten Punktes ist die Kinematik vollsynchرون zum Band, d. h. die Bandgeschwindigkeit wird durch die Kinematik über die Satzdistanz aufgebaut. Je größer die Satzdistanz ist, desto kleiner sind die Überhöhungen der Satzgrenzwerte während des Aufsynchronisierkommandos. Im Umkehrschluss baut die Kinematik bei sehr kleinen Distanzen, z. B. Bahndistanzen nahe "0", die Bandgeschwindigkeit sprunghaft auf.



Punkte, die zum Zeitpunkt des Kommandierens noch vor der Startebene liegen, werden intern auf die Startebene verschoben.

Entspricht der auf die Startebene verschobene Aufsynchronisierungspunkt der Warteposition der Kinematik, kommt das oben beschriebene Verhalten des sprunghaften Bandgeschwindigkeitsaufbaus zustande.

Die Aufsynchronisierdistanz geht in diesem Fall gegen "0".

Aus diesem Grunde wurde die Möglichkeit des Aufsynchronisierens mit Maximalwerten für Beschleunigung und Ruck [Synchronization with Limits auf Seite 37](#) geschaffen.

Hier wird kein konkreter Endpunkt zum Aufsynchronisieren vorgegeben, sondern Maximalwerte für Beschleunigung und Ruck, um die entsprechende Bandgeschwindigkeit in Bandrichtung kontrolliert aufzubauen.

Funktionsbeschreibung

Liegt der TCP außerhalb des bandsynchronen Arbeitsbereichs, wird das Aufsynchronisieren abgebrochen. Die Endposition auf dem Band ist erst nach dem Aufsynchronisieren bekannt. In einem zweiten Schritt kann anschließend die Position auf dem Band synchron angefahren werden.



Sind für den Aufsynchronisierungsmodus "Synchronization with Limits" die Maximalwerte sehr klein gewählt, nimmt die Zeit zum Aufsynchronisieren zu. Es kann bei einer hohen Banddynamik der Fall auftreten, dass die Kinematik auf Grund der geringen Maximalwerte nicht aufsynchronisieren kann.

Vollsynchrones Verfahren

Das vollsynchrone Verfahren mittels [Move Linear Absolute auf Seite 37](#), [Move Direct Absolute auf Seite 37](#) oder [Move Circular Absolute auf Seite 37](#) funktioniert auf gleiche Art und Weise wie oben unter dem Stichwort "Aufsynchronisieren" beschrieben. Es muss lediglich ein gekoppelter Punkt kommandiert werden.

Zusätzlich besteht die Möglichkeit mittels [Move Linear Relative auf Seite 37](#), [Move Direct Relative auf Seite 37](#) oder [Move Circular Relative auf Seite 37](#) relativ auf dem Band zu verfahren. Statt einem konkreten Endpunkt wird hier eine Distanz angegeben, welche relativ zum Band verfahren wird. Dieses Verhalten wird erreicht, wenn neben der kommandierten Distanz, auch eine Banddistanz als Bandwert angegeben wird.

Beispiel:

Zweiachsiger Kartheser mit einem Band

Sie wollen 50 mm in Bandrichtung verfahren. Sie kommandieren den relativen Punkt P = (0, 0, 50) mit dem Kommando Move Linear Relative.



Im Fall einer linearen Bewegung ist der Pfad relativ zum Band eine Gerade. Dies ist auch so, wenn das Band während dieser Bewegung beschleunigt oder abbremsst.

Im Fall einer direkten Bewegung (PTP) ist der Pfad auf dem Band undefiniert. In diesem Fall wird nur garantiert, dass der Endpunkt der jeweiligen Bewegung vollsynchron zum Band ist (Endpunktsynchrone Bewegung).

Absynchronisieren

Zum Absynchronisieren mit [Move Linear Absolute auf Seite 37](#), [Move Direct Absolute auf Seite 37](#) oder [Move Circular Absolute auf Seite 37](#) muss ein nicht gekoppelter Punkt kommandiert werden (Bandwert "0"). Mit Erreichen des kommandierten Endpunktes ist die Kinematik unsynchron, d. h. die Bandgeschwindigkeit wird durch die Kinematik über die Satzdistanz abgebaut. Je größer die Satzdistanz ist, desto kleiner sind die Überhöhungen der Satzgrenzwerte während des Absynchronisierungskommandos. Im Umkehrschluss baut die Kinematik bei sehr kleinen Distanzen, z. B. Bahndistanzen nahe "0", die Bandgeschwindigkeit sprunghaft ab.

Werden der Bandwert "1000000" an die Position der Bandachse geschrieben und das Kommando [Move Linear Relative auf Seite 37](#), [Move Direct Relative auf Seite 37](#) oder [Move Circular Relative auf Seite 37](#) abgesetzt, erfolgt ein relatives Absynchronisieren, d. h. die Kinematik verfährt von ihrer derzeitigen Position um die in der Punktedefinition angegebene Distanz und ist nach dieser Distanz absynchronisiert. Das relative Absynchronisieren unterteilt sich dabei in drei Abschnitte:

- Vollsynchron:
Relative Bewegung vollsynchron zum Band

Funktionsbeschreibung

- Absynchronisieren:
Relative Bewegung mit gleichzeitigem Absynchronisieren
- Unsynchron:
Relative unsynchrone Bewegung

Die Länge der einzelnen Abschnitte der relativen Absynchronisierbewegung kann prozentual mit dem Kommando [Set Belt Desync Relative Configuration auf Seite 37](#) konfiguriert werden. In der Standardeinstellung ist der vollsynchrone Abschnitt 0 %, der Abschnitt zum Absynchronisieren 50 % und der unsynchrone Abschnitt 50 %. Der absolute Endpunkt im Raum ist bei diesem Mechanismus erst nach dem Absynchronisierungsabschnitt bekannt.

Weiterhin ist das Absynchronisieren mittels des Kommandos [Desynchronization with Limits auf Seite 37](#) möglich. Hier wird kein konkreter Endpunkt zum Absynchronisieren vorgegeben, statt dessen gibt der Anwender Maximalwerte für Ruck und Verzögerung an.

Diese Maximalwerte werden dazu verwendet, die entsprechende Bandgeschwindigkeit kontrolliert abzubauen.

2.13.5 Adaptive Motion

Grundsätzlich erfolgen bandsynchrone Bewegungen erst, wenn der programmierte Zielpunkt (der kommandierte Punkt wird intern mit dem Bandwert bewegt) im Bandsynchronen-Arbeitsbereich liegt. Ist dies nicht der Fall, wartet die Kinematik an der letzten kommandierten Endposition, bis dieser die Startebene durchdrungen hat. Ist diese Warte- bzw. Stillstandszeit nicht erwünscht, kann über den Parameter [K-0-0099 Bit 6](#), die Option "adaptive Motion" aktiviert werden.

Die Aktivierung hat zur Folge, dass die Bewegung bereits startet wenn der gekoppelte Punkt kommandiert wird, auch wenn sich der Punkt vor der Startebene befindet. Der Zielpunkt der Bewegung ist dabei der verschobene gekoppelte Punkt in die Startebene. Der Fortschritt der Bewegung hängt dabei direkt am Fortschritt des Bandes. Die Bewegung ist abgeschlossen, sobald das Produkt (der gekoppelte Punkt) die Startebene erreicht hat.

Dies hat folgende Vorteile:

- Zeitersparnis: Der gekoppelte Punkt wird bereits erreicht, wenn dieser Punkt die Startebene erreicht. Ohne "adaptive Motion" würde zu diesem Zeitpunkt erst die Bewegung starten.
- Schonung der Mechanik: Die Wartezeit wird für die Bewegung benutzt. Dadurch sind die Bewegungen in der Regel langsamer und damit mechanikschonender.
- Energieersparnis: Durch langsamere Bewegungen mit geringeren Beschleunigungs- und Verzögerungswerten wird Energie gespart.

Eigenschaften von "adaptiven Motion":

- Im Fehlerfall wird wie gewohnt auf dem Pfad mit den bekannten Fehlerreaktionen angehalten.
- Die Stop-Funktionalität steht wie gewohnt zur Verfügung.
- Der Override-Mechanismus steht für den Fall der adaptiven Interpolation nicht zur Verfügung, da die Bewegung des Zielpunktes maßgebend für die Interpolation ist und dadurch der Bezug zum Produkt verloren gehen würde. Der Override gleich "0" ist jedoch jederzeit möglich.

Möchte man die Kinematik anhalten, muss die Bandachse gestoppt werden, da diese Bandachse die Referenz zur Interpolation bildet.

Funktionsbeschreibung

Hier gilt: Stoppt das Band und somit der kommandierte Zielpunkt, stoppt auch die Kinematik.

- Interrupt und Continue stehen ebenso wie der Override nicht zur Verfügung.
- Die Grenzwerte auf dem Pfad werden auch mit dieser Interpolationsart eingehalten. Unter Umständen hat dieser Fakt zur Folge, dass der Zielpunkt erst hinter der Startebene erreicht wird.
- Bei "adaptive Motion" kann der Pfad auch Punkte vor der Startebene erreichen. Es ist allerdings sichergestellt, dass sich der Pfad immer im mechanischen Arbeitsbereich der Kinematik befindet.

Generell gilt: Die Funktion ist dynamisch zu jeder Zeit zuschaltbar. Wird also vor einer bandsynchronen Bewegung die adaptive Interpolation aktiviert, so besteht ab der nächsten bandsynchronen Bewegung die Möglichkeit adaptiv zu interpolieren. Hier entscheidet die Steuerung selbstständig, wann eine adaptive Interpolation einer bandsynchronen Bewegung stattfindet.

Weiterhin kann die Funktionalität auch jederzeit im laufenden Betrieb deaktiviert werden.

Die gesamte Funktionalität hat keine Auswirkung auf die Kommandierung der Kinematik, d. h. vorhandene Programme müssen nicht geändert werden, um diese Funktionalität nutzen zu können.



Befindet sich der gekoppelte Punkt nicht nur vor dem Bandbereich, sondern auch vor dem Arbeitsbereich der Kinematik, so bewegt sich die Kinematik auch in diesem Fall direkt zum Schnittpunkt von gekoppeltem Punkt und Startebene. Das Aufsynchronisieren startet allerdings, wenn sich der gekoppelte Punkt im Arbeitsbereich der Kinematik befindet.

Weitere Informationen zu diesem Thema befinden sich unter [Bandsynchronität mit Adaptive Motion auf Seite 411](#).

2.13.6 Überschleifen und Bandsynchron

Das Überschleifen der einzelnen Bewegungssätze ist in allen Fällen so gewährleistet, wie in [Überschleifen auf Seite 54](#) beschrieben. Das bandsynchrone Verfahren ist an keine bestimmte Überschleifart gebunden, sondern unabhängig von den Bewegungskriterien zu sehen.

Bei folgenden Kommandos ist das Überschleifen eingeschränkt:

- [Synchronization with Limits auf Seite 37](#)

Aufgrund der Baddynamik kann ein genaues Ende der Aufsynchronisierungsbewegung (Endpunkt) nicht vorherbestimmt werden. Somit ist der Startpunkt der Folgebewegung erst bekannt und der Folgesatz kann erst gestartet werden, wenn die volle Synchronität zum jeweiligen Band erreicht ist und "Synchronization with Limits" beendet ist. Es kann nicht überschleifen werden.

- Relatives Absynchronisieren

Es ist nur möglich, den unsynchronen Abschnitt des relativen Absynchronisierens zu überschleifen.

2.13.7 Zusammenfassung

Ein Band wird durch ein **PCS** ([Produkt-Koordinatensystem \(PCS\) auf Seite 27](#)) repräsentiert, das gegenüber dem **MCS** ([Maschinen-Koordinatensystem \(MCS\) auf Seite 26](#)) verschoben und verdreht sein kann. Es kann einen geradlinigen oder kreisförmigen Verlauf haben.

Funktionsbeschreibung

Ein Anwendungsbeispiel für ein geradliniges Band findet man unter [Bandsynchronität auf Seite 400](#).

Ein Punkt auf dem Band wird durch seine Koordinaten und dem sogenannten Bandwert dargestellt. Der Bandwert definiert die absolute Position des Punktes in einem beliebigen Koordinatensystem auf dem Band zum Zeitpunkt der Punktaufnahme.

Das Überschleifen der einzelnen Bewegungssätze während der Bandsynchronität ist in allen Fällen nahtlos gewährleistet.

Die einzige Ausnahme bildet [Synchronization with Limits auf Seite 37](#). Hier kann aufgrund der Banddynamik ein genaues Ende der Aufsynchronisierbewegung (Endpunkt) nicht vorherbestimmt werden.

Jedes Band besitzt eine bandspezifische Fehlerreaktion siehe dazu [Band-spezifische Fehlerreaktion auf Seite 82](#).

2.14 Schalten auf der Bahn

Funktionsbeschreibung

In vielen Anwendungsfällen ist es notwendig, Ereignisse (Events), die von der Bahn abhängig sind, zu erfassen und in deren Abhängigkeit Schaltvorgänge auszulösen.

Nachstehend wird das Funktionsprinzip mit und ohne Vorhaltezeit dargestellt.

Funktionsprinzip

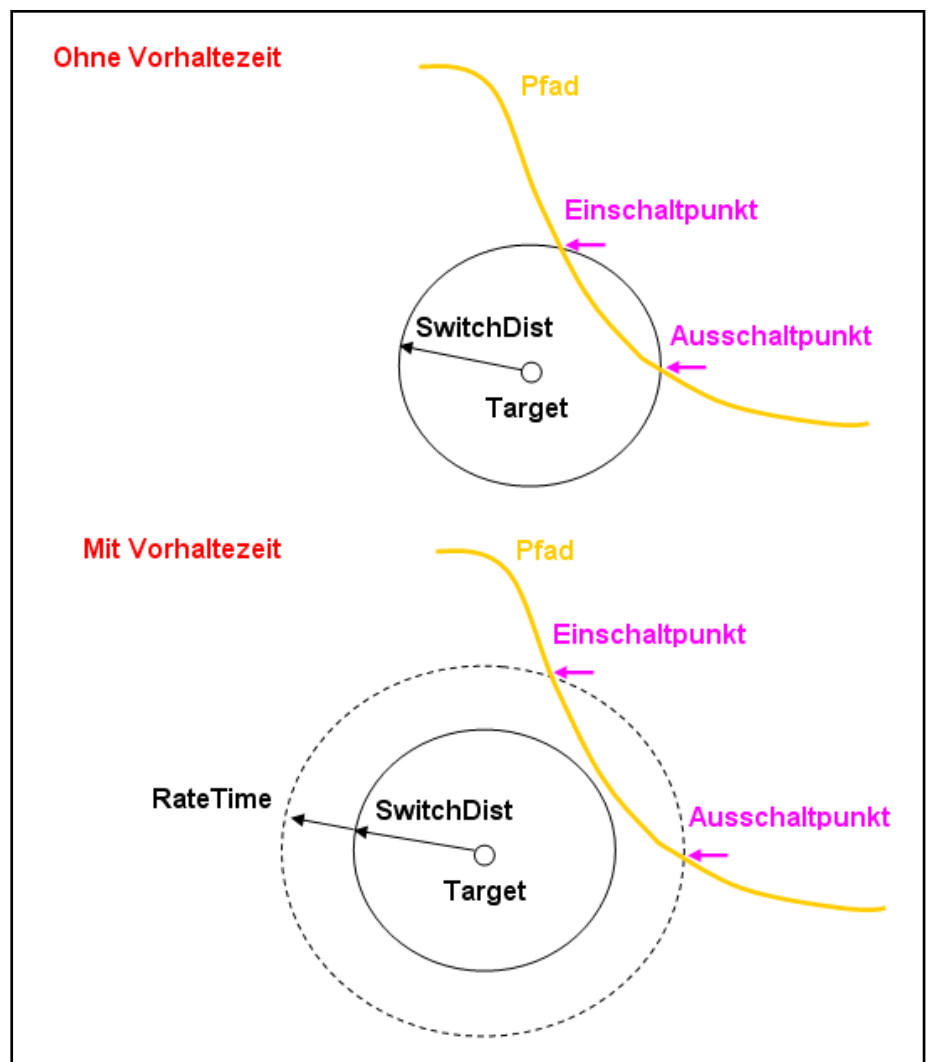


Abb. 2-43: Funktionsprinzip

Der Schalterpunkt ist durch "Target" und "SwitchDist" festgelegt (Kugel um "Target"). Über "RateTime" ist eine Vorhaltezeit einstellbar.

Diese Zeit vergrößert die Kugel um "Target" in Abhängigkeit von der aktuellen Geschwindigkeit der Kinematik.

Kann die Kinematik die Kugel um "Target" auf direktem Weg zum Target mit der aktuellen Geschwindigkeit in der angegebenen "RateTime" erreichen, so ist "InRange" TRUE.

Funktionsbeschreibung

Beispiel

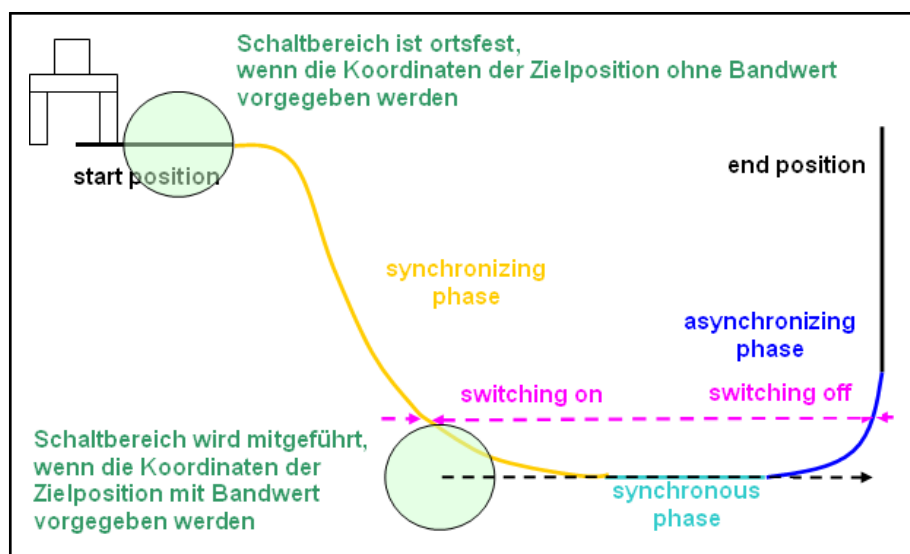


Abb. 2-44: Beispiel

Wenn die Koordinaten von "Target" ohne Bandwerte vorgegeben werden, ist der Schaltbereich des Funktionsbausteins als ortsfeste Kugel aufzufassen.

Wird hingegen ein Bandwert angegeben, so wird die Kugel um den Target-Punkt in Abhängigkeit des aktuellen Bandzählerwertes in Bandrichtung mitgeführt.

Zeitverhalten

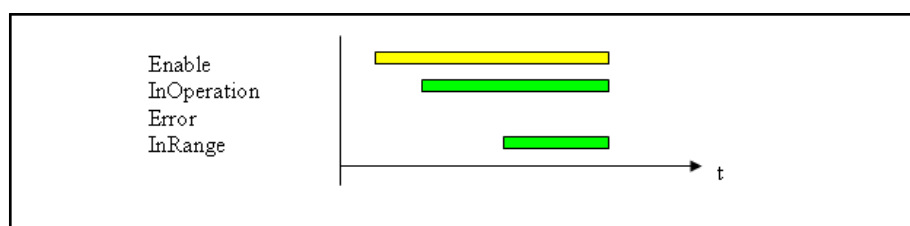


Abb. 2-45: Zeitverhalten

Der Funktionsbaustein ML_SwitchOnPath, ML_Robot, beginnt zustandsgesteuert mit steigender Flanke an "Enable" seine Logik abzuarbeiten.

Nachdem alle vorbereitenden Befehle durchlaufen wurden, wechselt der FB in den Zustand "InOperation". Änderungen an "SwitchDist" und "RateTime" werden so lange übernommen, wie "Enable" TRUE ist.

Der Ausgang "InRange" wird gesetzt, wenn die Kinematikposition innerhalb des Kugelradius liegt und die "RateTime" erfüllt ist, ansonsten wird er zurückgesetzt.

Bei Problemen wird der Zustand "Error" eingenommen, die Ausgänge "InRange" und "InOperation" werden zurückgesetzt. Die Ausgänge "ErrorID" und "ErrorIdent" enthalten weitere Informationen über den Fehler.

Wenn "Enable" zurückgesetzt wird, werden "InRange", "InOperation" und "Error" zurückgesetzt und der FB wartet in seinem internen Grundzustand wieder auf eine steigende Flanke an "Enable".

Schaltzeitpunkt

Befindet sich der TCP der Kinematik auf einer Position kleiner oder gleich der "SwitchDist", wird der FB-State auf "InRange" gesetzt.

Vorhaltezeit

"RateTime" bestimmt, wie viele Millisekunden vor dem Erreichen der eigentlichen Schaltposition der Ausgang "InRange" gesetzt werden soll. Hierzu wird die aktuelle Geschwindigkeit aus der globalen Variablenliste gelesen und die restliche Zeit bis zum Schaltzeitpunkt berechnet (ohne Berücksichtigung von Beschleunigung, Verzögerung und Ruck).

Funktionsbeschreibung

Wenn die verbleibende Restzeit kleiner oder gleich der Vorhaltezeit "Rate-Time" ist, wird der Ausgang "InRange" sofort gesetzt. Andernfalls wird die aktuelle Position neu eingelesen und die Berechnung erneut durchgeführt.

Anwendungshinweise

Der Baustein arbeitet nur dann richtig, wenn realistische Werte für die Eingänge "SwitchDist" und "RateTime" vorgegeben werden:

- Bei SPS-Zykluszeit der Motion-Task von 10 ms ist die Angabe von kleineren Vorhaltezeiten als 10 ms nicht sinnvoll
- "SwitchDist" muss bei großen Geschwindigkeiten groß genug sein, um rechtzeitig schalten zu können

Bei einer Geschwindigkeit von 5000 mm/s und einer Zykluszeit von 10 ms kann der FB alle 50 mm die Restdistanzen bzw. Restzeiten bis zum Schalterpunkt berechnen. Daher müssen Distanz und Vorhaltezeit groß genug gewählt oder die Zykluszeit heruntergesetzt werden.

2.15 Verfahrbereichsgrenzen

Die Kinematik unterstützt verschiedene Verfahrbereichsgrenzen.

Das folgende Bild zeigt eine Übersicht über alle möglichen Verfahrbereichsgrenzen und ihre Wirksamkeit.

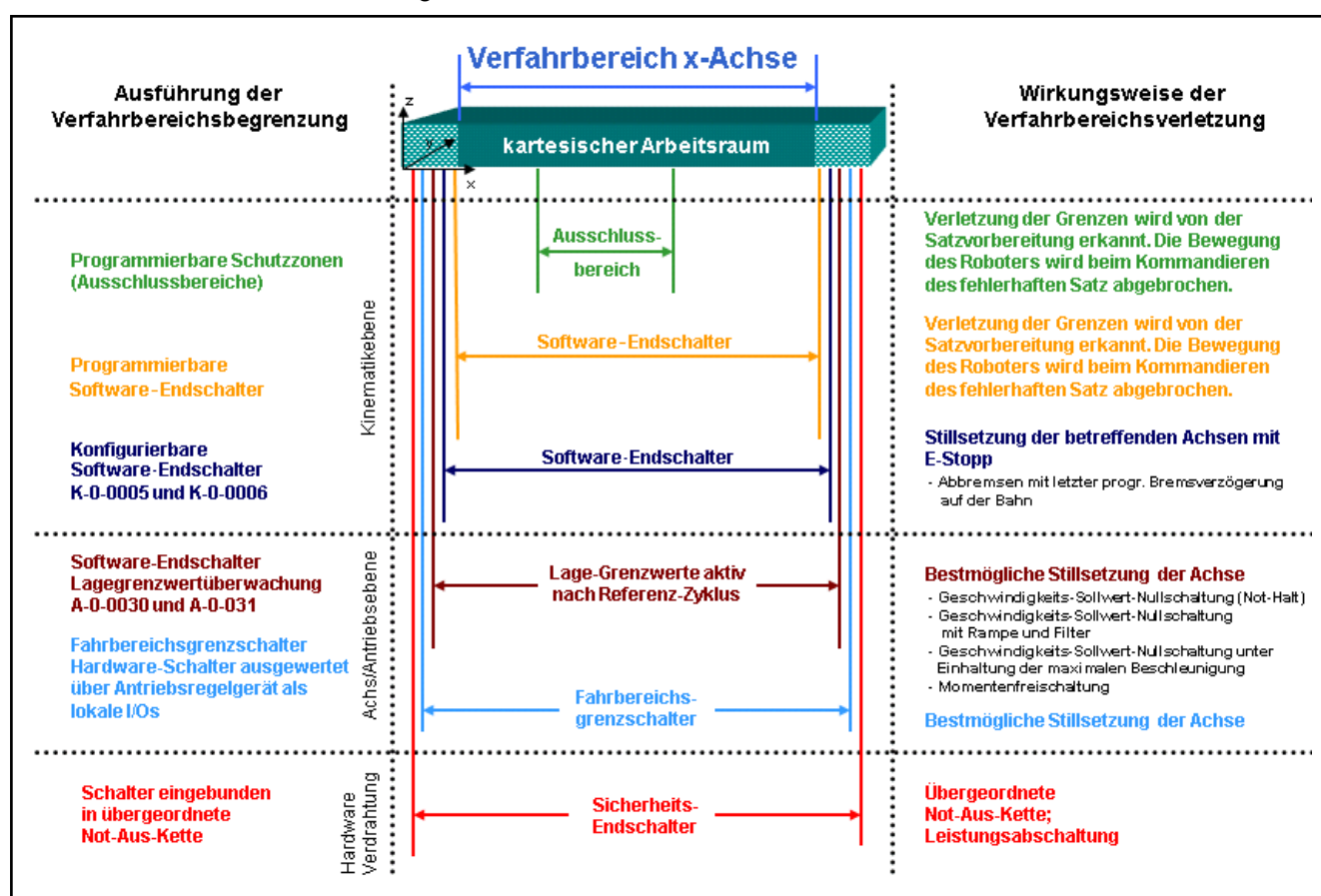


Abb. 2-46: Übersicht Verfahrbereichsgrenzen

Programmierbare Schutzzeiten

Programmierbare Schutzzeiten können mit Hilfe des Kommandos **Set Safe Zone** auf Seite 38 definiert werden.

Es stehen wiederum 32 Schutzzeiten zur Verfügung.

Funktionsbeschreibung

Eine Schutzzone wird mit zwei Punkten im Maschinen-Koordinatensystem angegeben, die einen Quader aufspannen. Der Bereich **innerhalb** dieses Quaders definiert die Schutzzone. Die Grenzen der Schutzzone, wie beispielsweise der minimale Punkt S1 (0,0,0), gehören ebenfalls zur Schutzzone.

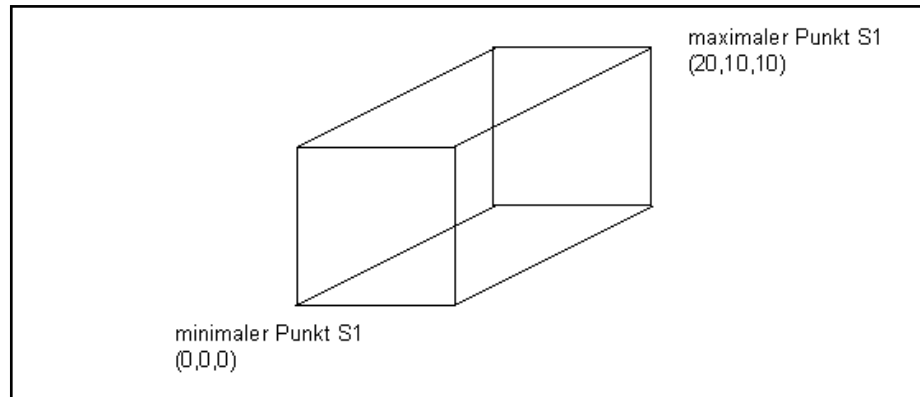


Abb. 2-47: Maximaler und minimaler Punkt einer Schutzzone

Die Schutzzone wird beim Kommandieren eines neuen Bewegungskommandos in der Satzvorbereitung überprüft. Dabei wird überprüft, ob die lineare Strecke zwischen Startpunkt (A) und Endpunkt (B) der Bewegung einen Schnitt- oder Berührungspunkt mit einer Seite der Schutzzone besitzt.

Erkennt die Satzvorbereitung eine Verletzung einer Schutzzone, so wird die Fehlermeldung F12C0433, Arbeitsraumverletzung (Robot-Control V1) ausgegeben.

Die Berührungsüberprüfung zwischen Gerade und Schutzzone findet nicht statt, wenn sich der Startpunkt der Geraden in der Schutzzone befindet. Ist z. B. die Position des TCP's auf der Grenze der Schutzzone, so steht der TCP der Kinematik innerhalb der definierten Schutzzone. Die Kinematik kann somit, ohne dass eine Verletzung einer Schutzzone gemeldet wird, auf einen Punkt außerhalb der Schutzzone kommandiert werden.

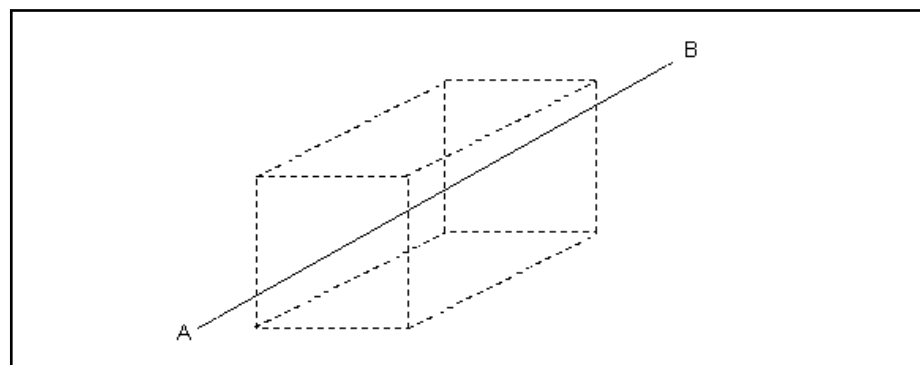


Abb. 2-48: Schutzzone mit Bewegung von A nach B

Bei der Wirksamkeit der Schutzzone müssen die folgenden Punkte beachtet werden:

- Jede Kinematik berücksichtigt immer nur die eigenen Schutzzonen
- Die Schutzzonen berücksichtigen nur den TCP der Kinematik, nicht die Mechanik
- Bei PTP-Interpolation ist die Überwachung der Schutzzonen stark eingeschränkt, da der zu verahrende Weg nicht definiert ist. Abweichungen von der linearen Strecke AB werden nicht erkannt

Funktionsbeschreibung

- Bei CIRCULAR-Interpolation ist die Überwachung der Schutzzonen stark eingeschränkt. Die Prüfung der Schutzzonen beschränkt sich hier auch auf die lineare Strecke zwischen Start- und Zwischenpunkt und zwischen Zwischen- und Endpunkt. Der Kreisabschnitt selbst wird nicht geprüft
- Beim Überschleifen mit VIA oder APPROX werden Abweichungen von der linearen Strecke AB nicht erkannt. Abweichungen durch Überschleifradien werden nicht berücksichtigt
- Der maximale Punkt muss immer größer sein als der minimale Punkt, da sonst die Prüfung der Schutzzone inaktiv ist
- Da die Prüfung der Schutzzonen in der Satzvorbereitung durchgeführt wird, sollten Schutzzonen nicht dynamisch angepasst werden. Werden Schutzzonen geändert, wirken diese erst für nicht vorbereitete Motion-Kommandos. Daher wird empfohlen die programmierbaren Schutzzonen vor Beginn von Bewegungskommandos zu programmieren

Programmierbare Software-Endschalter

Programmierbare Software-Endschalter können mit Hilfe des Kommandos [Set Safe Zone auf Seite 38](#) definiert werden.

Die Arbeitsweise ist gleich der Arbeitsweise der Schutzzonen mit dem Unterschied, dass der minimale und maximale Punkt einen Quader aufspannt, der den erlaubten Bereich einer Kinematik darstellt.

Weiterhin wird der Fehler F12C0885, Arbeitsraum überschritten ausgegeben.

Konfigurierbare Software-Endschalter

Konfigurierbare Software-Endschalter werden über die Parameter [K-0-0005](#), Software-Endschalter, positiv, MCS und [K-0-0006](#), Software-Endschalter, negativ, MCS eingestellt.

Sie verhalten sich wie programmierbare Software-Endschalter. Zusätzlich wird die kommandierte Position der Kinematik zyklisch überprüft. Liegt diese außerhalb des konfigurierbarem Software-Endschalter so wird auch die Fehlermeldung F12C0885, Arbeitsraum überschritten ausgegeben.

Befindet man sich außerhalb des konfigurierbaren Software-Endschalters, so führen nur kommandierte Positionen zum Fehler, bei denen man sich weiter weg vom Arbeitsraum bewegt.

Dabei wird jede Koordinate des Maschinen-Koordinatensystem (MCS) separat geprüft:

Befindet sich z. B. die Kinematik über dem konfigurierten Software-Endschalter in X- und Y-Achse, so sind nur Bewegungen in Richtung kleinerer X- und kleinerer Y-Werte erlaubt, Bewegungen in Richtung kleinerer X- aber größerer Y-Werte führen zum Fehler.

Software-Endschalter Lagegrenzwertüberwachung [A-0-0030](#) und [A-0-0031](#)

Die Lagegrenzwerte der Achsen werden von der Kinematik überwacht.

Es ist nicht möglich Punkte zu kommandieren, deren Ziel- oder Zwischenpunkt (bei CIRCULAR-Interpolation) in Achskoordinaten die Lagegrenzwerte überschreitet. Diese Prüfung findet in der Satzvorbereitung statt.

Weiterhin wird die kommandierte Position der Kinematik zyklisch auf die Lagegrenzwerte der Achsen überprüft. Im Unterschied zu Robot-Control V1 werden Änderungen in den Parametern [A-0-0030](#), Positionsgrenzwert, positiv und [A-0-0031](#), Positionsgrenzwert, negativ direkt übernommen.

2.16 Fehler und Fehlerreaktion

2.16.1 Fehler und Fehlerreaktion, Allgemeines

Es können verschiedene Meldungen, Warnungen und Fehler auftreten.

Funktionsbeschreibung

Unterstützt werden:

Fehlerkürzel	Bedeutung	Fehlerreaktion
A0xxxxxx	Meldung	Keine
E0xxxxxx	Warnung	Keine
F0xxxxxx	F0-Fehler	Keine
Fyxxxxxx (y > 0)	Fehler mit Fehlerreaktion der Kinematik	Fehlerreaktion der Kinematik

Tab. 2-14: Diagnosen und Fehlerreaktionen

Die Fehlerreaktion der Kinematik setzt sich aus folgenden Teilen zusammen:

- Fehlerreaktion der kommandierten Bewegung auf Seite 82
- Bandspezifische Fehlerreaktion auf Seite 82

Die Fehlerreaktion der Kinematik ist abgeschlossen, wenn beide Teile der Fehlerreaktion abgeschlossen sind.

Während eine Kinematik eine Fehlerreaktion ausführt, befindet sie sich im Zustand "Errorstop Stopping" (siehe [Zustandsdiagramm einer Kinematik auf Seite 33](#)). In diesem Zustand ist keine Kommandierung der Kinematik möglich.

Nach Abschluss der Fehlerreaktion werden alle nicht abgeschlossenen Bewegungskommandos abgebrochen (siehe [Bewegungskommandos auf Seite 35](#)).

2.16.2 Fehlerreaktion der kommandierten Bewegung

Die Fehlerreaktion der Kinematik ist das Anhalten auf der programmierten Bahn, soweit dies möglich ist. Welche Werte für Verzögerung und Ruck dabei verwendet werden, kann im Parameter "K-0-0223, Auswahl Fehlerreaktion" konfiguriert werden:

- **Verwendung der Werte der Kommandos:**
Es werden die Maximalwerte zum Anhalten verwendet, die am jeweiligem Kommando angegeben wurden (siehe [Bewegungskommandos auf Seite 35](#))
- **Verwendung der Nothaltverzögerung und des Nothaltrucks:**
Es werden die Nothaltverzögerung ([K-0-0224](#)) und der Nothaltruck ([K-0-0225](#)) verwendet.
Dabei gilt: Es werden die jeweils größeren Werte von Nothaltverzögerung und kommandierter Verzögerung bzw. Nothaltruck und kommandiertem Ruck verwendet.

2.16.3 Bandspezifische Fehlerreaktion

Zusätzlich zur Fehlerreaktion der gesamten Kinematik ist es möglich, für jedes konfigurierte Band eine spezifische Fehlerreaktion zu aktivieren. Diese wird über [Set Belt Error Reaction auf Seite 38](#) konfiguriert. Ist eine Band-Fehlerreaktion konfiguriert, wird diese im Fehlerfall immer in Bandrichtung und unabhängig von der Kinematik-Fehlerreaktion erfolgen. Insgesamt stehen für jedes Band drei Reaktionsmöglichkeiten zur Verfügung:

Funktionsbeschreibung

Bandspezifische Fehlerreaktionsmöglichkeiten	Beschreibung
Keine	Als Fehlerreaktion ist nur die Fehlerreaktion der Kinematik aktiv. Im Fehlerfall bleibt die eventuell vorhandene Synchronität zum jeweiligen Band bestehen
Absynchronisieren über Zeit	Im Fehlerfall wird die vorhandene Synchronität über die konfigurierte Zeit abgebaut
Absynchronisieren über Grenzwerte	Im Fehlerfall wird die vorhandene Synchronität über die konfigurierten Maximalwerte für Verzögerung und Ruck abgebaut

Tab. 2-15: Bandspezifische Fehlerreaktionsmöglichkeiten



Ist eine Band-Fehlerreaktion konfiguriert, wird diese im Fehlerfall immer in Bandrichtung und unabhängig von der Kinematik-Fehlerreaktion erfolgen.

Die Dauer der Band-Fehlerreaktion kann je nach gewählten Parametern variieren.

2.17 Zyklische Sollwertvorgabe

2.17.1 Zyklische Sollwertvorgabe, Allgemeines

In der Robotik ist es aktuell möglich, den Tool Center Point (TCP) satzweise auf Punkte im Raum zu bewegen (Satzbetriebsart). Weiterhin besteht die Möglichkeit, zyklisch (in jedem Motion-Takt) Soll-Positionswerte im Raum vorzugeben (zyklische Betriebsart). Es kann flexibel zwischen den beiden Betriebsarten gewechselt werden, ohne dass beim Wechsel der Betriebsart sich der Kinematik im Stillstand befinden muss.

Mit Hilfe der zyklischen Sollwertvorgabe kann einer Kinematik zyklisch ein neuer Positions-, Geschwindigkeits- oder Distanzsollwert vorgegeben werden.

Kurvenscheiben, Flexprofile oder andere Kinematiken können verwendet werden, um den Sollwert zu generieren.

2.17.2 Start und Betrieb der zyklischen Sollwertvorgabe

Die Zyklische Sollwertvorgabe wird mit Hilfe von [Open Cyclic Channel](#) und [Open Cyclic Direct Channel](#) auf Seite 37 gestartet. Beim Kommando Open Cyclic Channel erfolgt das Aufaddieren der Sollwerte im MCS. Beim Kommando Open Direct Cyclic Channel erfolgt das aufaddieren der Sollwerte im ACS (PTP-Bewegung). Dadurch sind auch Armigkeitswechsel während der zyklischen Sollwertvorgabe möglich.

Ein Überschleifen auf den zyklischen Sollwertbereich ist möglich, siehe [Abb. 2-50 "Schematische Darstellung der zyklischen Sollwertvorgabe mit Überschleifen"](#) auf Seite 84, wenn während des Öffnens des zyklischen Kanals noch ein Verfahrbefehl aktiv ist und beide Kommandos nicht explizit das Überschleifen mit **TO** verhindern. In diesem Fall muss zum Zeitpunkt des Startens des Überschleifens um Punkt P1 mit der zyklischen Sollwertvorgabe am Punkt P1 begonnen werden. Das Überschleifen zwischen PTP und Linearbewegungen ist nicht möglich.

Weiterhin wird in [Abb. 2-49 "Schematische Darstellung der zyklischen Sollwertvorgabe ohne Überschleifen"](#) auf Seite 84 die zyklische Sollwertvorgabe ohne Überschleifen dargestellt.

Funktionsbeschreibung

Anschließend **muss** mit der Funktion [Write Cyclic Channel auf Seite 37](#) einmal im Motionzyklus ein neuer Sollwert an die Kinematik vorgegeben werden.



Die Funktion [Write Cyclic Channel auf Seite 37](#) muss **zwingend** im Motionzyklus aufgerufen werden, da es sonst zu unerwarteten Fehlermeldungen oder ruckenden Achsen kommen kann!

Der Sollwert kann in einem beliebigen Koordinatensystem angegeben werden.

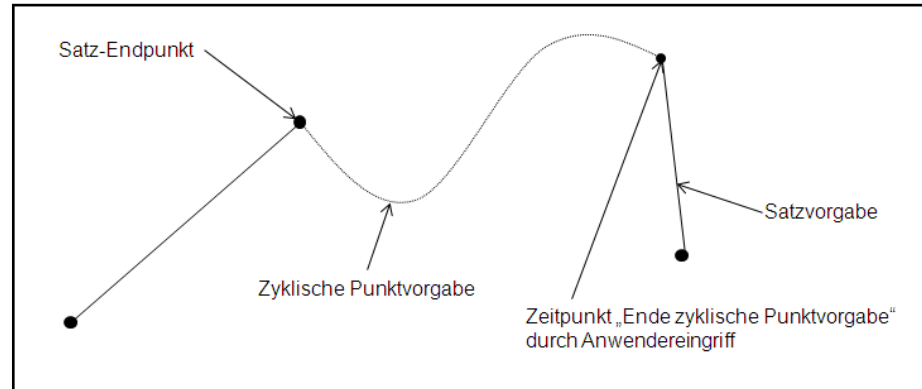


Abb. 2-49: Schematische Darstellung der zyklischen Sollwertvorgabe ohne Überschleifen

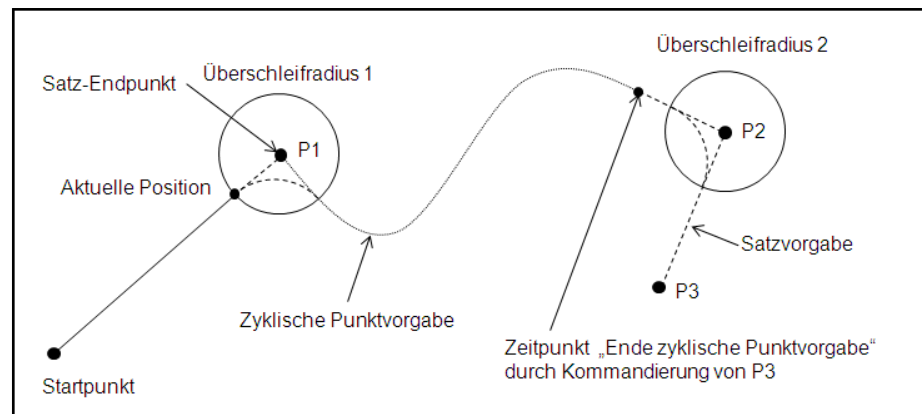


Abb. 2-50: Schematische Darstellung der zyklischen Sollwertvorgabe mit Überschleifen

2.17.3 Die zyklische Sollwertvorgabe

Innerhalb des zyklischen Sollwertkanals kann der Sollwert in drei verschiedenen Formaten vorgeben werden:

Positionssollwertmodus absolut

Die zyklische Sollwertvorgabe erfolgt als absolute Position.

Positionssollwertmodus relativ

Die zyklische Sollwertvorgabe erfolgt als relative Distanz zur letzten kommandierten Position aus dem letzten Takt.

Geschwindigkeitssollwertmodus

Die zyklische Sollwertvorgabe erfolgt als Geschwindigkeit. Die vorgegebene Geschwindigkeit wird intern zu einer Positionsdivergenz umgerechnet, welche relativ zur letzten kommandierten Position addiert wird.

Die zyklische Sollwertvorgabe sollte mit der letzten kommandierten Position der Satzvorbereitung initialisiert werden, die mit Hilfe der Funktion [Read Pos auf Seite 38](#) ausgelesen werden kann.

Darauf aufbauend können zyklisch Sollwerte vorgegeben werden (siehe [Liste aller Bewegungskommandos und Funktionen auf Seite 37](#)).

HINWEIS

Die Kinematik folgt dem Sollwert!

Vorgegebene Sollwertsprünge können zu einem Fehler führen, weil die Kinematik z. B. aufgrund mangelnder Dynamik einzelner Achsen dem Sollwert nicht folgen kann.

2.17.4 Ende der zyklischen Sollwertvorgabe

Eine zyklische Sollwertvorgabe wird beendet wenn:

- Eine Fehlerreaktion gestartet wird
- Ein Stop oder Interrupt kommandiert wird
- Der Geschwindigkeits-Override auf 0 gesetzt wird
- Ein neuer Verfahrbefehl kommandiert wird

Wird die zyklische Sollwertvorgabe mittels eines Verfahrkommandos beendet, so ist auf diesen Verfahrbefehl ebenfalls ein Überschleifen möglich, siehe [Abb. 2-50 "Schematische Darstellung der zyklischen Sollwertvorgabe mit Überschleifen" auf Seite 84](#). Die mit der zyklischen Sollwertvorgabe erreichte Geschwindigkeit wird dann in Richtung des letzten, zyklisch vorgegebenen Bewegungsvektors mit den zuvor definierten Maximalwerten abgebaut. Hierbei wird die gesamte Strecke berechnet, welche bis zum Abbau der letzten, zyklisch aufgebauten Geschwindigkeit benötigt wird. Mit Hilfe dieser Strecke wird ein Punkt im Raum bestimmt werden, welcher den Startpunkt für das folgende Verfahrkommando bildet, siehe Punkt P2 in [Abb. 2-50 "Schematische Darstellung der zyklischen Sollwertvorgabe mit Überschleifen" auf Seite 84](#). Somit ist ein Überschleifen auf folgende Verfahrkommandos durch verfrühtes Starten des Kommandos immer gegeben.



Ist die Kinematik auf Grund der vorgegeben Sollwerte bereits in einem Abbremsvorgang und sind die Werte für Ruck und Verzögerung größer als die Werte beim Öffnen des zyklischen Kanals, werden die im Vorfeld definierten Werte unter bestimmten Umständen angepasst.

Insbesondere wenn der gewählte Ruck sehr klein ist, reicht dieser nicht aus, die hohe aktuelle Verzögerung abzubauen und Geschwindigkeit "0" zu erreichen. Man würde zurückfahren.

Durch die interne Ruckanpassung wird gewährleistet, dass eine vorhandene Verzögerung beim Verlassen der Sollwertvorgabe immer abgebaut wird.

2.18 Performancemessung

Die Performancemessung der Kinematik ermöglicht die Messung der Zeitdauer und der Anzahl der Aufrufe spezieller Teilbereiche des Kinematik-Kerns. Diese Informationen können herangezogen werden, um Anwendungen zu optimieren.

Die Performancemessung wird mit der Funktion [Set Performance Enable auf Seite 38](#) gestartet und gestoppt.

Beim erneuten Starten der Messung werden zuvor aufgezeichnete Daten gelöscht.

Die Ergebnisse können mit der Funktion [Get Performance auf Seite 38](#) gelesen werden, die eine Liste von Performancemessungen zurückgibt.

Jeder Eintrag dieser Liste hat folgenden Aufbau:

Funktionsbeschreibung

Name	Einheit	Bedeutung
name	Text	Name der Performancemessung
count	Anzahl	Anzahl der Messwerte
min	µs	Minimaler Messwert
max	µs	Maximaler Messwert
avg	µs	Durchschnittlicher Messwert

Tab. 2-16: Einträge der Performancemessung

Folgende Messungen stehen zur Verfügung (Auszug):

Name des Eintrags	Bedeutung
Shared Memory	In der SPS steht ein Shared Memory, ML_Robot, für jede Kinematik zur Verfügung Dieses Shared Memory wird in jedem Motion-Takt bedient
Adapt	Istwerterfassung der Kinematik
Execute	Sollwertgenerierung der Kinematik
SetCmd	Absetzen von Kommandos und Funktionen vom Anwender in den Kern (siehe Robot-Control, Kommandos und Funktionen, Überblick auf Seite 35)
GetCmd	Lesen von Statusinformationen von Kommandos (siehe Zustand eines Kommandos auf Seite 36)
prepare	Satzvorbereitung
MoveExecute	Zyklische Abarbeitung eines Move-Kommandos

Tab. 2-17: Verfügbare Messungen (Auszug)

3 Oberflächenfunktionen

3.1 Oberflächenfunktionen, Allgemeines

IndraWorks Engineering unterstützt die Arbeit mit Kinematiken im Ordner "Robot".

Dieser Ordner erscheint bei einer

- IndraMotion MLC, Hardwarebasis IndraControl CML 25/45/65/75 oder
- IndraMotion MLC, Hardwarebasis IndraControl XM21/XM22 mit Funktionspaket "Robot Control".

Im Ordner "Robot" können Kinematiken und sofern noch nicht vorhanden die Globale Variablenliste "Globale Daten" hinzugefügt werden (Kontextmenü des Ordners "Robot" bzw. Hauptmenüpunkt "Robot").

Für jede im Ordner "Robot" angelegte Kinematik entsteht ein eigener Ordner mit dem Namen der Kinematik.

3.2 Ordner Robot

3.2.1 Ordner Robot, Allgemeines

Durch Anwahl des Ordners "Robot" im Projekt wird der Menüpunkt **Robot** im IndraWorks-Hauptmenü aktiviert.

Dieser Menüpunkt enthält die folgenden Untermenüpunkte:

- **Hinzufügen:** Hier kann eine Kinematik hinzugefügt werden
Siehe auch [Hinzufügen einer neuen Kinematik auf Seite 87](#)
- **Kinematikinbetriebnahme:** Hier kann der Kinematikinbetriebnahmedialog geöffnet werden
Siehe auch [Kinematikinbetriebnahme auf Seite 89](#)
- **PathViewer:** Hier kann der PathViewer geöffnet werden
Siehe auch [PathViewer auf Seite 98](#)
- **Parameter:** Hier kann der Parameterexport bzw. -import für alle Kinematiken aufgerufen werden
- **Exportieren:** Hier kann die Kinematikkonfiguration exportiert werden
- **Importieren:** Hier kann eine exportierte Kinematikkonfiguration importiert werden
- **Online-Vergleich:** Hier können die Online- mit den Offline-Gerätedaten verglichen werden
- **Vergleichen:** Hier können zwei Projekte miteinander verglichen werden

Die gleichen Untermenüpunkte finden Sie auch im Kontextmenü des Knotens "Robot"

Hier gibt es zusätzlich noch die folgenden Untermenüpunkte:

- **Elementsuche:** Hier kann innerhalb der Kinematikkonfiguration gesucht werden
- **Neue Ansicht:** Hier kann die Kinematikkonfiguration in einem eigenen Projektextplorerfenster angezeigt werden

3.2.2 Hinzufügen einer neuen Kinematik

Über den Menüpunkt **Hinzufügen** unter dem Knoten "Robot" kann eine neue Kinematik angelegt werden.

Oberflächenfunktionen

Ein Assistent unterstützt das Anlegen der neuen Kinematik:

Schritt 1: Eingabe von Gerätenamen, Kommentar und Ersteller

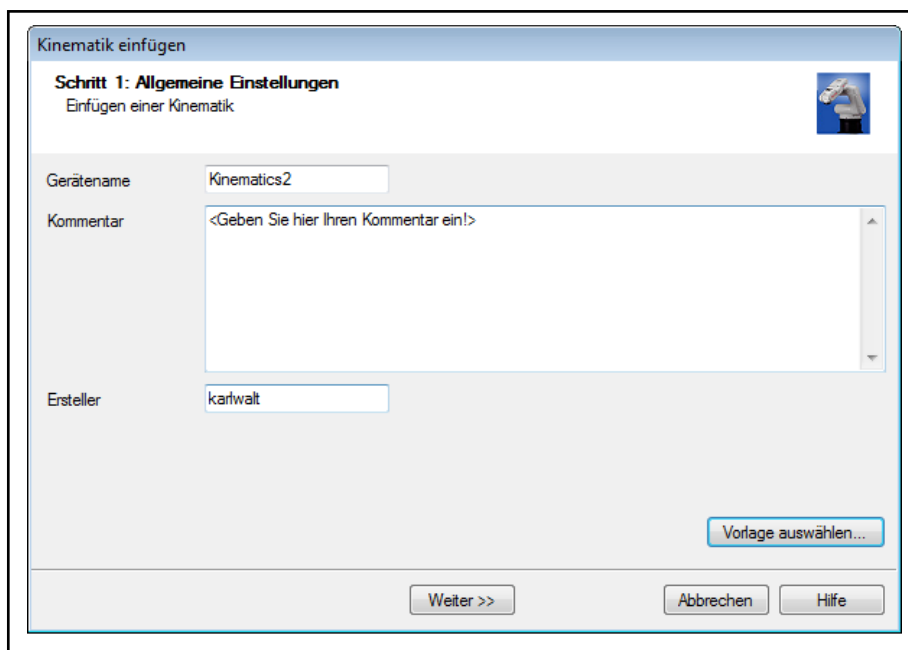


Abb. 3-1: Kinematik anlegen, Schritt 1

Schritt 2: Eingabe von RobotControl-Version, Kinematik-Nummer und Kinematik-Typ

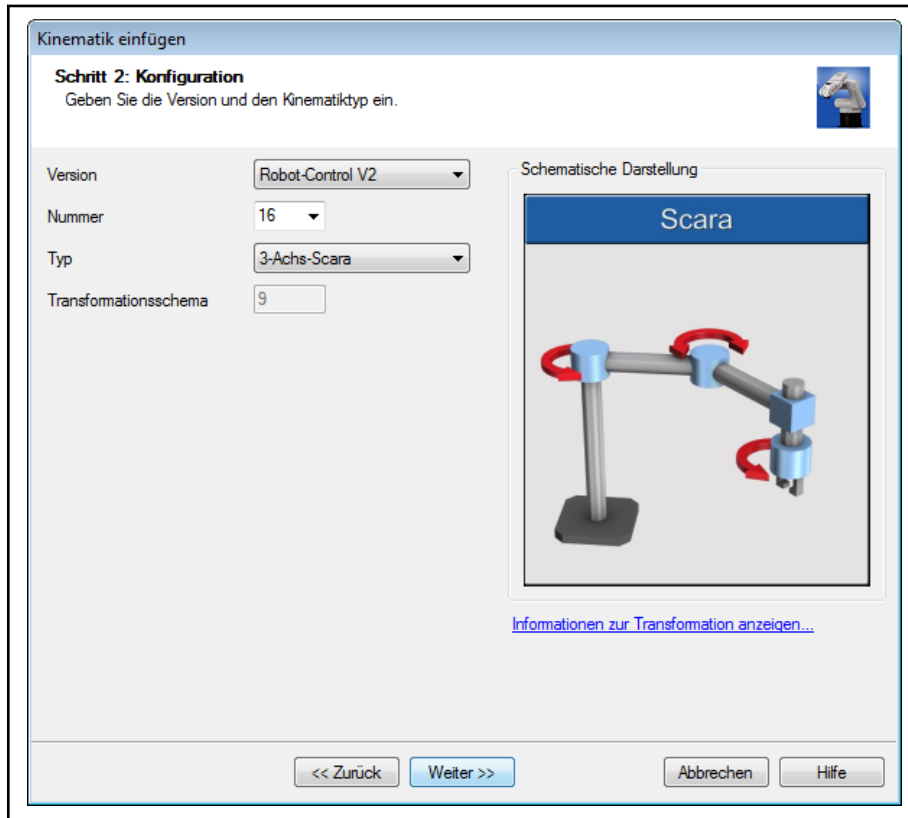


Abb. 3-2: Kinematik anlegen, Schritt 2

Schritt 3: Achsen zuordnen

Kinematik einfügen

Schritt 3: Achszuordnung
 Ordnen Sie die für die Kinematik benötigten Achsen zu.

Position	Typ	Steuerung	Achse
1	Positionierungsachse	lokal	vAxis3
2	Positionierungsachse	lokal	vAxis4
3	---	---	---
4	---	---	---
5	---	---	---
6	---	---	---
7	---	---	---
8	---	---	---
9	---	---	---
10	---	---	---
11	---	---	---
12	---	---	---
13	---	---	---
14	---	---	---
15	---	---	---
16	---	---	---

[Informationen zur Transformation anzeigen...](#)

<< Zurück Fertigstellen Abbrechen Hilfe

Abb. 3-3: Kinematik anlegen, Schritt 3

Die hier eingetragenen Achsen (Positionierachsen, Orientierungsachsen und Bandachsen) werden in den Parameter **K-0-0008** (Achskonfigurationsliste) übernommen.

Nachdem die Kinematik komplett angelegt ist, erscheint unter dem Knoten "Robot" ein neuer Eintrag mit den Namen der Kinematik.

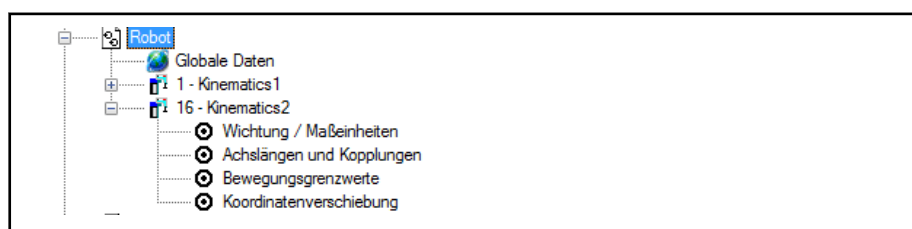


Abb. 3-4: Neuer Eintrag "Kinematics2"

3.2.3 Kinematikinbetriebnahme

Übersicht

Mit Hilfe des Dialogs "Kinematikinbetriebnahme" im Kontextmenü des Robot-Knotens können Kinematiken bewegt werden, ohne dafür ein SPS-Programm schreiben zu müssen. Dies ist insbesondere in der Inbetriebnahmephase nützlich, wenn das SPS-Programm noch nicht vollständig ist oder eine Kinematik probenhalber, anders als im SPS-Programm vorgesehen, bewegt werden soll.



Dieser Dialog ist nur im Online-Modus auswählbar. Des Weiteren können Kinematiken nur in der Kommunikationsphase P4 (BB) betrieben werden.

Oberflächenfunktionen

Der Dialog gliedert sich in einen Übersichts- und Detailbereich.

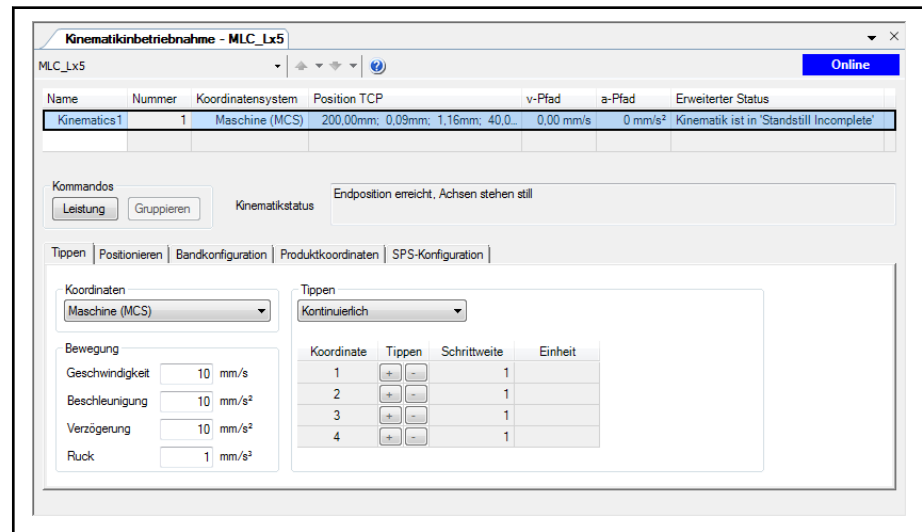


Abb. 3-5: Dialog Kinematikbetriebnahme

Übersichtsbereich

Im Übersichtsbereich werden die wichtigsten Daten der Kinematik angezeigt. Durch Auswahl einer konfigurierten Kinematik in der Spalte "Name" werden die weiteren Spalten mit Daten gefüllt. Ebenfalls kann das Koordinatensystem, in welchem die Position des TCP angezeigt wird, ausgewählt werden.

Neben dem Kinematiknamen wird weiterhin angezeigt:

- Nummer: Kinematiknummer
- Position TCP: Aktuelle Position des TCP im gewählten Koordinatensystem
- v-Pfad: Aktuelle Pfadgeschwindigkeit
- a-Pfad: Aktuelle Pfadbeschleunigung
- Erweiterter Status: Erweiterter Kinematikstatus (entspricht dem dekodierten Inhalt des Parameters [K-0-0022](#))

Zum Löschen einer Kinematik kann entweder der Leereintrag in der Spalte "Name" ausgewählt oder die <Entf>-Taste der Tastatur (nach Auswahl der entsprechenden Zeile) genutzt werden.

Detailbereich

Im Detailbereich können Bewegungskommandos ausgelöst, Band- und Produktkoordinatensysteme konfiguriert sowie SPS-Konfigurationsbausteine erzeugt werden. Die Bewegungskommandos orientieren sich hierbei am Zustandsdiagramm der PLCopen (siehe Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Kapitel "Zustandsdiagramm einer Kinematik") und entsprechen den durch einen äquivalenten Funktionsbaustein erzeugten Kommandos.

Alle Kommandos und Werte beziehen sich auf die im Übersichtsbereich ausgewählte Kinematik. Über die Registerkarten können die entsprechenden Kommandos ausgewählt und Sollwerte vorgegeben werden. Die Felder innerhalb einer Registerkarte sind reine Eingabefelder, d. h. es wird kein Status ausgegeben. Wird ein falscher Wert eingegeben, werden das Eingabefeld und dessen Registerkarte rot eingefärbt und im Tooltip des Eingabefeldes erscheint eine Fehlermeldung. In diesem Fall kann kein Kommando abgesetzt werden, d. h. die entsprechenden Buttons sind deaktiviert.

Oberflächenfunktionen

Desweiteren kann im Übersichtsbereich die Leistung aller der Kinematik zugewiesenen Achsen zugeschaltet sowie die Kinematik gruppiert werden. Über das Feld "Kinematikstatus" wird ebenfalls der aktuelle Status der Kinematik dargestellt, welches dem dekodierten Inhalt des Parameters [K-0-0021](#) entspricht.

Sicherheitsfunktionen

Bevor der eigentliche Inbetriebnahmedialog angezeigt wird, muss bestätigt werden, dass alle notwendigen Maßnahmen, die für einen sicheren Betrieb der Kinematik notwendig sind, durchgeführt wurden.



Abb. 3-6: Warnhinweis beim Öffnen des Kinematikinbetriebnahmedialogs

Während der Inbetriebnahmedialog geöffnet ist, können mittels des Fensters "Kinematiken AUS" alle Achsen aller Kinematiken im Übersichtsbereich sofort gestoppt werden.

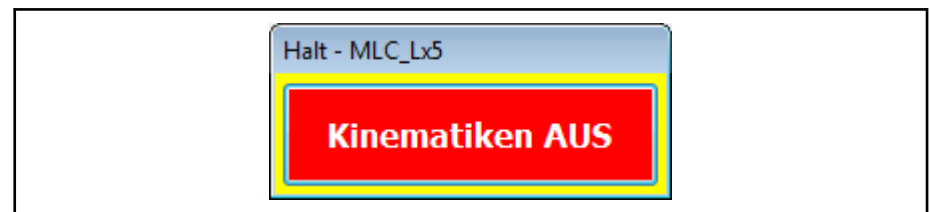


Abb. 3-7: Achsen aller Kinematiken anhalten

Kinematiken dürfen nur bewegt werden, so lange diese Kinematiken im Dialog angezeigt werden. Aus diesem Grund werden alle Kinematiken gestoppt, sobald der Dialog geschlossen wird. Außerdem kann eine laufende Kinematik im Übersichtsbereich nicht gelöscht oder geändert werden.

Um Konflikte mit dem SPS-Programm zu vermeiden, können Kinematiken nur unter einer der folgenden Bedingungen bewegt werden:

- Alle SPS Anwender-Applikationen sind gestoppt
- Alle laufenden SPS Anwender-Applikationen enthalten die Variable "AllowKinematicsCommissioning" mit dem Wert TRUE in der "UserVarGlobal" bzw. "UserVarAppGlobal" falls eine Geräteapplikation verwendet wird. Damit Achsen bei laufender SPS bewegt werden können, müssen Sie im Knoten "UserVarGlobal" bzw. "UserVarAppGlobal" eine boolesche Variable namens "AllowKinematicsCommissioning" anlegen, die

Oberflächenfunktionen

Variable in die Symbolkonfiguration der SPS aufnehmen und deren Wert auf TRUE setzen. Auf diese Weise können die SPS-Programme steuern, ob bzw. wann das Bewegen von Achsen über den Dialog erlaubt ist

Werden Kinematiken über den Inbetriebnahmedialog bewegt und die vorgeannten Bedingungen sind nicht mehr erfüllt, werden alle Kinematiken sofort gestoppt und weitere Bewegungskommandos gesperrt.



Beim Schließen des Dialogs wird ein Stopp-Kommando an alle noch laufenden Kinematiken geschickt und auf den Stillstand gewartet. Benötigt eine Kinematik länger als zwei Sekunden bis zum Stillstand, wird der Dialog trotzdem geschlossen und eine Warnung in der Statuszeile von IndraWorks ausgegeben. In diesem Fall wird die entsprechende Kinematik zwar weiterhin gestoppt, verbleibt dann aber im Zustand "Stopping". Um die Kinematik endgültig zu stoppen (Zustand "Standstill Incomplete") kann der Dialog jederzeit wieder geöffnet und der Kinematik nochmals ein "Stopp"-Kommando gegeben werden. Alternativ kann die Kinematik auch über die SPS gestoppt werden.



Verhalten bei Verbindungsunterbrechung zwischen IndraWorks-Oberfläche und Steuerung.

Wird die Verbindung zwischen PC und der Steuerung unterbrochen (z. B. Netzwerkabel defekt bzw. abgezogen), löst die Steuerung den Fehler "F536000E MLPI: WatchDog Fehler" aus, wodurch alle eventuell kommandierten Kinematiken automatisch gestoppt werden.

Gruppieren einer Kinematik

Um eine Kinematik bewegen zu können, muss vorher die Leistung der zugehörigen Achsen zugeschaltet und diese Achse der Kinematik zugewiesen werden. Hierbei entspricht die Leistungszuschaltung der Funktionalität des Funktionsbausteins "MC_Power" (siehe Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken") und die Zuweisung der Achsen zur Kinematik der des Funktionsbausteins ML_GroupAllAxes. Bei erfolgreicher Leistungszuschaltung aller der Kinematik zugewiesenen Achsen wird der Button **Leistung** grün hinterlegt. Bei erfolgreicher Gruppierung der Kinematikachsen wird der Button **Gruppieren** grün hinterlegt.

Ein Degruppierung bzw. Abschaltung der Leistung ist durch nochmalige Betätigung des entsprechenden Buttons möglich. Beide Fälle (Zuschaltung der Leistung und Gruppierung bzw. Degruppierung und Abschalten der Leistung) sind nur in der richtigen Reihenfolge möglich. So kann eine Kinematik nicht gruppiert werden, wenn die Leistung der Achsen nicht zugeschaltet ist. Ebenfalls kann die Leistung der Achsen nicht abgeschaltet werden, wenn die Kinematik gruppiert ist.

Damit die Leistung der Achsen zugeschaltet und die Kinematik gruppiert werden kann, müssen folgende Voraussetzungen erfüllt sein:

- Der Sercos Bus ist in P4 (Betriebsmodus)
- Alle zugehörigen Achsen sind nicht in P2 (Parametriermodus)
- Steuer- und Leistungsteile der zugehörigen Achsen sind betriebsbereit (d. h. Bit 14 im Parameter A-0-0021 ist gesetzt)
- An der Kinematik oder den zugehörigen Achsen steht kein Fehler an

Oberflächenfunktionen

Trifft eine der vorgenannten Voraussetzungen nicht zu, ist die Leistungszuschaltung nicht möglich und die fehlenden Voraussetzungen werden neben den Buttons als Tooltip angezeigt.

Bei virtuellen Achsen ist es nicht explizit notwendig die Leistung zuzuschalten. Diese Achsen sind automatisch freigegeben und können sofort gruppiert werden, insofern alle anderen genannten Voraussetzungen erfüllt sind.

Tippen

In der Registerkarte "Tippen" kann eine gruppierte Kinematik in einem beliebigen Koordinatensystem kontinuierlich oder inkrementell vertippt werden. Hierzu ist im Bereich "Koordinatensystem" das gewünschte Koordinatensystem zu wählen und im Bereich "Bewegung" die Werte für Geschwindigkeit, Beschleunigung, Verzögerung und Ruck anzugeben. Im Bereich "Tippen" kann anschließend gewählt werden, ob kontinuierlich oder inkrementell vertippt werden soll. Bei der Wahl des inkrementellen Tippens ist es weiterhin möglich die Schrittweite anzugeben. Mittels der Buttons + und - kann das Kommando für die entsprechende Koordinate/Achse ausgeführt werden.

Abb. 3-8: Kinematik in Betriebnahme, Registerkarte "Tippen"

Die Funktionalität entspricht der der Funktionsbausteine "ML_GroupJogStep" und "ML_GroupJogCont". Durch die Wahl der Art des Tippens wird implizit der entsprechende Funktionsbaustein aktiviert. Die Bedeutung der Eingabefelder ist analog zu den Eingangsparametern des entsprechenden Funktionsbausteins:

Inkrementelles Tippen

Eingabefeld im Dialog	Eingangsparameter des Funktionsbausteins "ML_GroupJogStep"
Bereich "Koordinatensystem"	CoordSystem
Bewegung/Geschwindigkeit	Velocity
Bewegung/Beschleunigung	Acceleration
Bewegung/Verzögerung	Deceleration
Bewegung/Ruck	Jerk
Tippen/Schrittweite	Distance

Tab. 3-1: Zuordnung der Eingabefelder im Dialog auf die Eingänge des Funktionsbausteins "ML_GroupJogStep"

Oberflächenfunktionen

Kontinuierliches Tippen

Eingabefeld im Dialog	Eingangsparameter des Funktionsbausteins "ML_GroupJogCont"
Bereich "Koordinatensystem"	CoordSystem
Bewegung/Geschwindigkeit	Velocity
Bewegung/Beschleunigung	Acceleration
Bewegung/Verzögerung	Deceleration
Bewegung/Ruck	Jerk

Tab. 3-2: Zuordnung der Eingabefelder im Dialog auf die Eingänge des Funktionsbausteins "ML_GroupJogCont"

Positionieren

In der Registerkarte "Positionieren" kann eine gruppierte Kinematik in einem beliebigen Koordinatensystem positioniert werden. Hierzu sind im Bereich "Koordinatensystem" das gewünschte Koordinatensystem zu wählen sowie die Zielposition und, insofern die Positionierart "Zirkular" gewählt ist, die Zwischenposition einzugeben. Im Bereich "Bewegung" kann die gewünschte Positionierart ausgewählt und Werte für Geschwindigkeit, Beschleunigung, Verzögerung und Ruck angegeben werden. Mittels des Buttons **Start** kann das Kommando ausgeführt und mit **Stopp** gestoppt werden.

Abb. 3-9: Kinematikinbetriebnahme, Registerkarte "Positionieren"

Die Funktionalität entspricht der der Funktionsbausteine "MC_MoveLinearAbsolute", "MC_MoveDirectAbsolute" und "MC_MoveCircularAbsolute". Durch die Wahl der Positionierart wird implizit der entsprechende Funktionsbaustein ausgewählt. Die Bedeutung der Eingabefelder ist analog zu den Eingangsparametern des Funktionsbausteins:

Eingabefeld im Dialog	Eingangsparameter der Funktionsbausteine "MC_MoveLinearAbsolute" und "MC_MoveDirectAbsolute"
Koordinaten/Koordinatensystem	CoordSystem
Koordinaten/Position TCP	Position
Bewegung/Geschwindigkeit	Velocity
Bewegung/Beschleunigung	Acceleration

Oberflächenfunktionen

Eingabefeld im Dialog	Eingangsparameter der Funktionsbausteine "MC_MoveLinearAbsolute" und "MC_MoveDirectAbsolute"
Bewegung/Verzögerung	Deceleration
Bewegung/Ruck	Jerk

Tab. 3-3: Zuordnung der Eingabefelder im Dialog auf die Eingänge der Funktionsbausteine "MC_MoveLinearAbsolute" und "MC_MoveDirectAbsolute"

Eingabefeld im Dialog	Eingangsparameter des Funktionsbausteins "MC_MoveCircularAbsolute"
Koordinaten/Koordinatensystem	CoordSystem
Koordinaten/Position TCP	EndPoint
Koordinaten/Zwischenposition	AuxPoint
Bewegung/Geschwindigkeit	Velocity
Bewegung/Beschleunigung	Acceleration
Bewegung/Verzögerung	Deceleration
Bewegung/Ruck	Jerk

Tab. 3-4: Zuordnung der Eingabefelder im Dialog auf die Eingänge des Funktionsbausteins "MC_MoveCircularAbsolute"

Bandkonfiguration

In der Registerkarte "Bandkonfiguration" können Bänder konfiguriert werden, welche der gewählten Kinematik zugehörig sind. Besitzt die Kinematik mehrere Bandachsen, kann im Bereich "Achse" die zu konfigurierende Bandachse ausgewählt werden. In den Bereichen "Band" und "Koordinatenverschiebung" sind der Bandtyp, die Start- und Gesamtlänge sowie die Verschiebung und Verdrehung des Bandes gegenüber dem Maschinenkoordinatensystem anzugeben. Im Bereich "Fehlerreaktion" ist die Reaktion des Bandes im Fehlerfall zu definieren. Hierzu müssen je nach gewählter Art der Reaktion die Desynchronisationszeit oder Verzögerung und Ruck definiert werden. Mittels des Buttons **Konfiguration schreiben** wird das Band in der Steuerung konfiguriert.

Abb. 3-10: Kinematikinbetriebnahme, Registerkarte "Bandkonfiguration"

Die Funktionalität entspricht der der Funktionsbausteine "ML_SetBeltConfiguration", "ML_SetRotaryTableConfiguration" und "ML_SetBeltErrorReaction". Durch die Wahl des Bandtyps wird implizit der entsprechende Funktionsbaustein ausgewählt. Die Bedeutung der Eingabefelder ist analog zu den Eingangsparametern des Funktionsbausteins:

Oberflächenfunktionen

Eingabefeld im Dialog	Eingangsparameter des Funktionsbausteins "ML_SetBeltConfiguration"
Bereich "Achse"	ConveyorBelt
Band/Startlänge	BeginLength
Band/Gesamtlänge	TotalLength
Bereich "Koordinatenverschiebung"	ConveyorBeltOrigin

Tab. 3-5: Zuordnung der Eingabefelder im Dialog auf die Eingänge des Funktionsbausteins "ML_SetBeltConfiguration"

Eingabefeld im Dialog	Eingangsparameter des Funktionsbausteins "ML_SetRotaryTableConfiguration"
Bereich "Achse"	ConveyorBelt
Band/Startlänge	BeginLength
Band/Gesamtlänge	TotalLength
Bereich "Koordinatenverschiebung"	RotaryTableOrigin

Tab. 3-6: Zuordnung der Eingabefelder im Dialog auf die Eingänge des Funktionsbausteins "ML_SetRotaryTableConfiguration"

Eingabefeld im Dialog	Eingangsparameter des Funktionsbausteins "ML_SetBeltErrorReaction"
Fehlerreaktion/Art der Reaktion	ErrorReactionType
Fehlerreaktion/Desynchronisationszeit	DeSyncTime
Fehlerreaktion/Verzögerung	Deceleration
Fehlerreaktion/Ruck	Jerk

Tab. 3-7: Zuordnung der Eingabefelder im Dialog auf die Eingänge des Funktionsbausteins "ML_SetBeltErrorReaction"

Produktkoordinaten

In der Registerkarte "Produktkoordinaten" können Produktkoordinatensysteme definiert werden. Im Bereich "Koordinaten" können hierzu das Koordinatensystem, welches definiert werden soll, sowie der Koordinatentyp (kartesisch, zylindrisch) ausgewählt werden. Im Bereich "Koordinatenverschiebung" sind die Verschiebung und Verdrehung des Koordinatensystems gegenüber dem Maschinenkoordinatensystem anzugeben. Mittels des Buttons **Konfiguration schreiben** wird das Produktkoordinatensystem in der Steuerung konfiguriert.

Oberflächenfunktionen

Abb. 3-11: Kinematik inbetriebnahme, Registerkarte "Produktkoordinaten"

Die Funktionalität entspricht der der Funktionsbausteine "MC_SetCartesianTransform" und "ML_SetCylindricTransform". Durch die Wahl des Koordinatentyps wird implizit der entsprechende Funktionsbaustein ausgewählt. Die Bedeutung der Eingabefelder ist analog zu den Eingangsparametern des Funktionsbausteins:

Eingabefeld im Dialog	Eingangsparameter der Funktionsbausteine "MC_SetCartesianTransform" und "ML_SetCylindricTransform"
Koordinaten/Koordinatensystem	ConveyorBelt
Bereich "Koordinatenverschiebung"	ConveyorBeltOrigin
Koordinatenverschiebung/Verschiebung X	TransX
Koordinatenverschiebung/Verschiebung Y	TransY
Koordinatenverschiebung/Verschiebung Z	TransZ
Koordinatenverschiebung/Verdrehung X	RotAngle1
Koordinatenverschiebung/Verdrehung Y	RotAngle2
Koordinatenverschiebung/Verdrehung Z	RotAngle3

Tab. 3-8: Zuordnung der Eingabefelder im Dialog auf die Eingänge der Funktionsbausteine "MC_SetCartesianTransform" und "ML_SetCylindricTransform"

SPS-Konfiguration

Nach einem Neustart der Steuerung verliert die Steuerung jegliche Konfiguration von Band- und Produktkoordinatensystemen. Um die Notwendigkeit einer erneuten Konfiguration zu umgehen, erlaubt die Registerkarte "SPS-Konfiguration" die dauerhafte Speicherung dieser Konfiguration in einem Funktionsbaustein. Somit besteht die Möglichkeit durch den Aufruf des Funktionsbausteins, beispielsweise in der Initialisierungsphase seiner Applikation, die Konfiguration schnell und einfach automatisch wiederherzustellen.

Es besteht die Möglichkeit alle oder nur die aktuell konfigurierten Bänder und Produktkoordinatensysteme zu sichern. Hierzu muss im Bereich "Produktkoordinatensysteme" zwischen den beiden Möglichkeiten gewählt werden.

Oberflächenfunktionen

Mittels der Schaltfläche **SPS Baustein erzeugen** wird der Funktionsbaustein erzeugt.

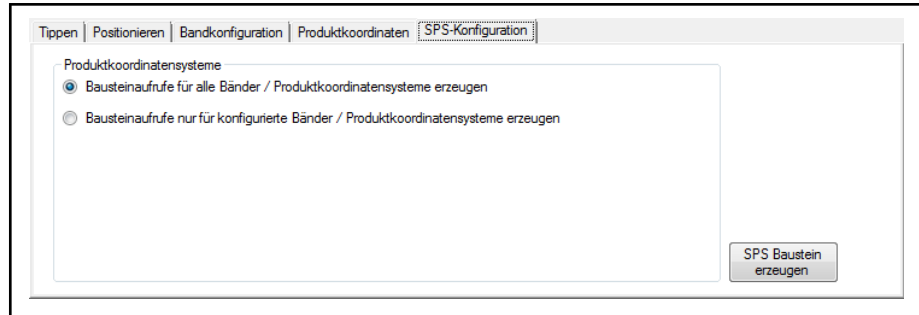


Abb. 3-12: Kinematik in betriebsnahme, Registerkarte "SPS-Konfiguration"

Durch Betätigung des Buttons **SPS Baustein erzeugen** wird unter dem Applikationsknoten ein neuer Ordner namens "KinematicsInitialConfig" erzeugt. In diesem Ordner wird der Funktionsbaustein angelegt, dessen Namen sich aus dem Namen der Kinematik und der Endung "...InitialConfig" zusammensetzt. So kann für jeder konfigurierte Kinematik ein zugehöriger Funktionsbaustein erzeugt werden, welcher beim Aufruf die Konfiguration der Band- und Produktkoordinatensysteme durchführt. Falls bereits ein Funktionsbaustein für die gewählte Kinematik existiert, wird dieser Funktionsbaustein ersetzt.

3.2.4 PathViewer

Übersicht

Der Dialog "PathViewer" im Kontextmenü des Robot-Knotens stellt eine Projektierungshilfe für Kinematik-Anwendungen dar. Mithilfe dessen können alle konfigurierten Kinematiken in Form von dreidimensionalen Stäben und Kugeln dargestellt werden, wobei die Perspektive frei wählbar ist. Dies kann insbesondere in der Projektierungsphase des SPS-Programms nützlich sein, wenn eine reale Anlage nicht verfügbar ist und die Kinematik probehalber bewegt werden soll.



Dieser Dialog ist nur im Online-Modus auswählbar. Des Weiteren werden Kinematiken nur in der Kommunikationsphase P4 (BB) dargestellt.

Die Übertragung der Daten zwischen der Steuerung und dem PathViewer erfolgt über die Ethernet-Verbindung, weshalb keine Echtzeit gewährleistet werden kann. Aus diesem Grund erhebt die angezeigte TCP-Spur keinen Anspruch auf Genauigkeit und entspricht unter Umständen nicht zu 100% der verfahrenen Bahnkurve. Bei entsprechender Auslastung der Verbindung oder auch hohen Bahngeschwindigkeiten kann diese Abweichung extrem werden.

Der PathViewer ist über das Kontextmenü des Robot-Knotens oder über den Ordner "Robot" im IndraWorks-Hauptmenü zu erreichen. Nach dem Öffnen werden alle in der Steuerung konfigurierten Kinematiken sowie deren Koordinatensysteme im Raum angezeigt.



Benutzerdefinierte Kinematiken (Transformationsnummer 100 bzw. 111) werden nur angezeigt, wenn hierfür die mechanischen Informationen in der Methode "GetMechanic" (siehe [Mechanische Informationen auf Seite 183](#)) zur Verfügung stehen.

Oberflächenfunktionen

Folgende Abbildung zeigt eine beispielhafte Darstellung eines Deltas, Scaras und Parallelogramms.

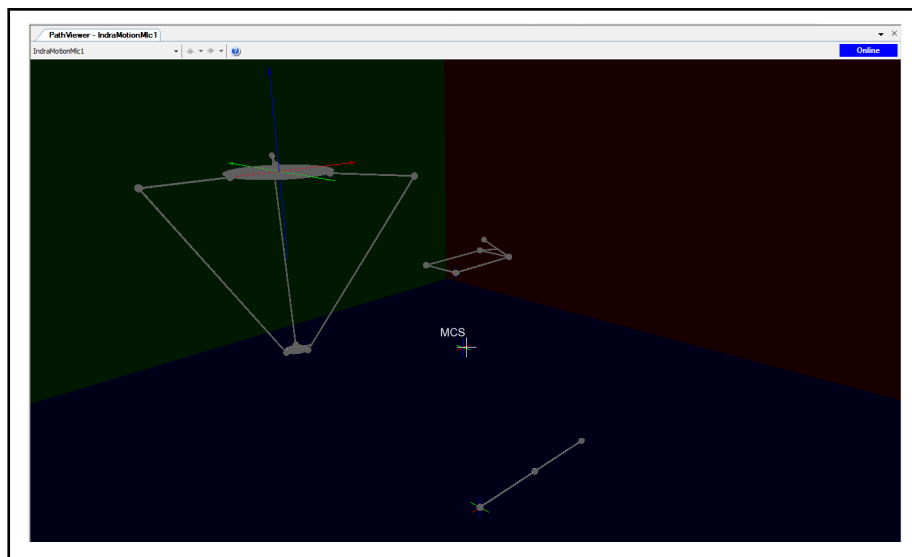
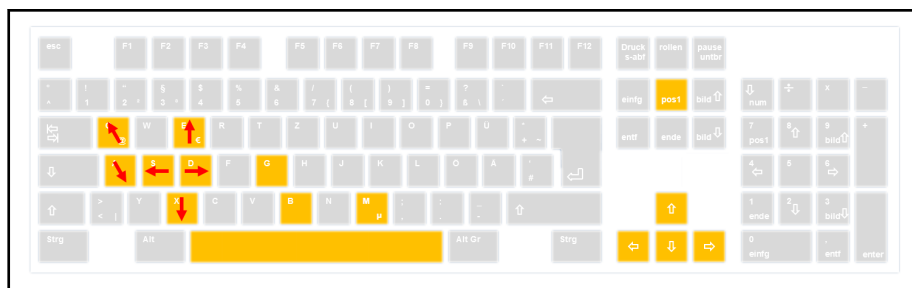


Abb. 3-13: Dialog PathViewer

Navigation und Bedienung

Eine Bewegung innerhalb des PathViewers erfolgt mittels des Zusammenspiels von Maus und Tastatur, wobei ebenfalls eine Bedienung mit nur einem der beiden Eingabegeräte möglich ist.

Folgende Übersicht soll die Navigation durch den Raum im PathViewer mittels Tastatur verdeutlichen.



Q	vorwärts bewegen
A	rückwärts bewegen
S	nach links bewegen
D	nach rechts bewegen
E	nach oben bewegen
X	nach unten bewegen
G	gehe zur Kinematik, welche im Fadenkreuz angezeigt wird
B	zeige die Begrenzungen der einzelnen Elemente
pos1	gehe zum MCS
Pfeil links	nach links drehen
Pfeil rechts	nach rechts drehen
Pfeil hoch	nach unten drehen
Pfeil runter	nach oben drehen
M	höhere Geschwindigkeit für die Bewegung der Ansicht
Leertaste	niedrigere Geschwindigkeit für die Bewegung der Ansicht

Abb. 3-14: Tastaturbelegung PathViewer

Neben der Bedienung per Tastatur kann auch die Maus für die Bewegung im Raum genutzt werden.

Oberflächenfunktionen

Folgende Tabelle zeigt die Reaktion auf die Bewegung der Maus in Verbindung mit verschiedenen Aktionen.

Reaktion auf Maus-Aktionen

Aktion	Reaktion	Tastatur
Linke Maustaste halten und ziehen	Ansicht verdrehen (3D)	Pfeiltasten
Rechte Maustaste halten und ziehen	Ansicht verschieben (2D)	S, D, E, X
Mausrad nach vorne drehen	Vorwärts bewegen	Q
Mausrad nach hinten drehen	Rückwärts bewegen	A
Mit linker Maustaste auf Kinematik klicken	Gehe zum BCS der Kinematik	G
Mauscursor auf Kinematik halten	Name der Kinematik wird im Tooltip angezeigt	
Mit rechter Maustaste klicken	Öffnen des Kontextmenüs des PathViewers	

Tab. 3-9: PathViewer-Bedienung per Maus

Die beiden Tastaturtasten "Shift" und "Strg" haben ebenfalls Einfluss auf die Scrollgeschwindigkeit der Mauseinschaltungen.

Kontextmenü des PathViewers

Durch einen Rechtsklick mit der Maus in die PathViewer-Oberfläche öffnet sich das Kontextmenü. Darin sind folgende Menüpunkte enthalten:

- **Position:** Hier kann die Ansicht gespeichert, wiederhergestellt oder auf eine vorhandene Kinematik zentriert werden
- **Aufzeichnung:** Hier können Kinematikbewegungen aufgezeichnet werden
- **TCP-Spuren löschen:** Alle TCP-Spuren werden gelöscht
- **Einstellungen...:** Öffnet die Optionen zum PathViewer

Aufzeichnung

Im Menüpunkt "Aufzeichnungen" können die Bewegungen der vorhandenen Kinematiken ähnlich einem Video aufgezeichnet und im späteren beliebig oft angesehen werden. Das Starten der Aufzeichnung erfolgt mithilfe des Untermenüpunktes "Aufzeichnung starten". Ist dieser Punkt aktiv werden die Bewegungen der Kinematiken im Zwischenspeicher gespeichert. Die maximale Aufzeichnungslänge ist hierbei auf 10 Minuten begrenzt. Ist dieses Limit erreicht, wird die laufende Aufzeichnung automatisch gestoppt.

Durch ein erneutes Betätigen des Untermenüpunktes "Aufzeichnung starten" wird dieser deaktiviert und die Aufzeichnung gestoppt. Die Daten stehen nun solange im Zwischenspeicher zur Verfügung, bis eine erneute Aufzeichnung gestartet oder der PathViewer geschlossen wurde.

Die Wiedergabe einer aufgezeichneten Bewegung ist mittels des Untermenüpunktes "Aufzeichnung ansehen" möglich. Durch Betätigung dieses Untermenüpunktes öffnet sich ein neues Fenster, welches der Steuerung zur Wiedergabe dient. Hier kann die Aufzeichnung beliebig oft gestartet, gestoppt und mittels Slider manuell bedient werden. Betätigt man diesen Menüpunkt während einer Aufzeichnung, so wird diese Aufzeichnung automatisch gestoppt.

Einstellungen

Der Menüpunkt "Einstellungen" stellt einige Optionen zum PathViewer bereit.

Oberflächenfunktionen

Folgende Tabelle zeigt die Bedeutung der einzelnen Möglichkeiten im Reiter "Allgemein".

Optionen im Reiter "Allgemein"

Funktion		Bedeutung
TCP-Spur		
	TCP-Spur anzeigen	Durch Aktivierung/Deaktivierung dieser Checkbox wird die Anzeige der TCP-Spur aktiviert/deaktiviert.
	Länge der TCP-Spur	Hier kann die maximale Länge der angezeigten TCP-Spur gesteuert werden. Die Zeitangabe entspricht der Anzeigedauer eines Punktes der TCP-Spur. Ältere Punkte werden gelöscht, auch bei stehender Kinematik!
	TCP-Spur als Punkte anzeigen	Bei deaktivierter Checkbox wird die TCP-Spur als linear interpolierte Linie zwischen den einzelnen Punkten dargestellt. Da die Punkte über die Ethernet-Verbindung und somit nicht in Echtzeit übertragen werden, entspricht die als Linie dargestellte TCP-Spur unter Umständen nicht zu 100% der verfahrenen Bahnkurve. Durch Aktivierung dieser Checkbox wird die TCP-Spur als Punktwolke dargestellt, ohne Interpolation. Durch Angabe einer Punktgröße kann diese individuell angepasst werden. Diese Darstellungsart hat den Vorteil, dass die übermittelten Punkte exakt die Position des TCP repräsentieren. So kann bei einer schnellen periodischen Bewegung des TCPs die Bahnkurve sehr genau analysiert werden.
Tastatursteuerung		
	Y-Achse spiegeln	Durch Aktivierung/Deaktivierung dieser Checkbox wird die Bedeutung der Tastatur-Taste "Pfeil hoch" und "Pfeil runter" vertauscht.
Raumgröße		
	Raumgröße manuell vorgeben	Bei deaktivierter Checkbox berechnet sich die Raumgröße automatisch aus den Abständen der Kinematiken zueinander. Ist nur eine Kinematik vorhanden wird eine vordefinierte Raumgröße erstellt. Durch Aktivierung dieser Checkbox kann die Raumgröße manuell vorgegeben werden. Die Ausdehnung der Koordinatensysteme passt sich hierbei ebenfalls automatisch an.

Tab. 3-10: Kontextmenü "Einstellungen" im PathViewer

Über den zweiten Reiter "Farben" können die Farben der TCP-Spur, des Roboters, der Wände sowie des Fadenkreuzes angepasst werden.

3.3 Ordner Kinematik

3.3.1 Ordner Kinematik, Allgemeines

Jede im Ordner "Robot" angelegte Kinematik verfügt im Projektextplorer über einen eigenen, namensgleichen Ordner mit

- Kontextmenü, siehe [Kinematik - Kontextmenü auf Seite 102](#) und
- Dialogen, siehe [Kinematik - Dialoge auf Seite 107](#)

Oberflächenfunktionen

3.3.2 Kinematik - Kontextmenü

Kinematik - Kontextmenü bzw. Menüpunkt Kinematik, Allgemeines

Die folgenden Menüpunkte sind im Kontextmenü einer Kinematik **und** im Hauptmenüpunkt **Kinematik** enthalten:

- **Erstinbetriebnahme** - Inbetriebnahmezyklus Kinematik, siehe [Erstinbetriebnahme auf Seite 102](#)
- **Parameter** - Bearbeiten von Parametern
- **Diagnose** - Diagnosefunktionen der Kinematik, siehe [Diagnose auf Seite 105](#)
- **Exportieren...** - Abspeichern der Robot-Elemente in einer Datei
- **Importieren...** - Importieren von Robot-Elementen aus einer Datei
- **Vergleichen...** - Vergleichen und Zusammenführen von Objekten

Die folgenden Menüpunkte stehen **nur** im Kontextmenü der Kinematik zur Verfügung:

- **Löschen** - Löschen der angewählten Kinematik
- **Umbenennen** - Umbenennen der angewählten Kinematik
- **Elementsuche...** - Suchen nach Elementen innerhalb des Projekts
- **Druckvorschau...** - liefert eine Vorschau der Kinematikdaten
- **Drucken...** - ermöglicht den Ausdruck der aktuellen Kinematikdaten
- **Neue Ansicht** - Kinematik in separatem Fenster anzeigen
- **Eigenschaften...** - liefert Informationen zur Kinematik (z. B. Name, Typ usw.)

Erstinbetriebnahme

Die Erstinbetriebnahme einer Kinematik wird durch einen Wizard in folgenden Schritten unterstützt:

1. Basisparameter laden

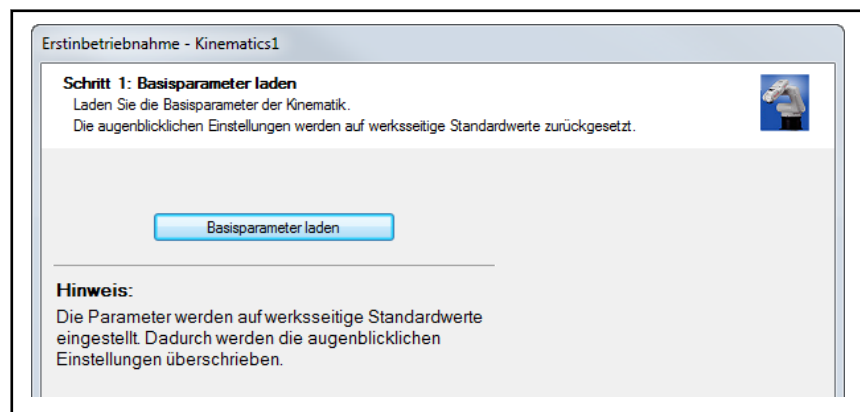


Abb. 3-15: Dialog Basisparameter der Kinematik laden

Die Parameter werden auf die werksseitigen Standardwerte eingestellt. Alle augenblicklichen Werte werden damit überschrieben. Auf dieser Basis können nun in den Folgeschritten die gewünschten Einstellungen vorgenommen werden.

2. Wichtungen/Maßeinheiten

Erstinbetriebnahme - Delta_virt

Schritt 2: Wichtung / Maßeinheiten
Legen Sie die Wichtungen und Maßeinheiten der Kinematik fest.

Maßeinheit für Kinematikdaten

☒ Kinematik in Millimeter [mm] wichten
☐ Kinematik in Inch [in] wichten

Wichtung der zugeordneten Achsen

Achsnamen	Wichtung der Lagedaten	Format der Lagedaten
vAxis1	Rotatorisch	Absolut
vAxis2	Rotatorisch	Absolut
vAxis3	Rotatorisch	Absolut
vAxis4	Rotatorisch	Modulo

Achswichtungen anpassen

Abb. 3-16: Dialog Wichtungen/Maßeinheiten der Kinematik

Konfiguriert werden die Parameter

- K-0-0054, Maßeinheit für Kinematikdaten
- A-0-0053, Wichtungsart für Beschleunigungsdaten
- A-0-0056, Wichtungsart für Geschwindigkeitsdaten
- A-0-0059, Wichtungsart für Positionsdaten

Durch diese Parameter wird zum einen die Maßeinheit für Kinematikdaten festgelegt und zum anderen die Wichtung aller, der Kinematik zugewiesenen Achsen angepasst.

Siehe auch [Wichtung/Maßeinheiten auf Seite 107](#).

3. Achslänge und Kopplung

Erstinbetriebnahme - Kinematics1

Schritt 3: Achslänge und Kopplung
Legen Sie die Achslängen und die Achskopplungen fest.

Achslängen			Achskopplungen	
Name	Wert	Einheit	Name	Wert
L1	0	mm	K1	0
L2	0	mm	K2	0
L3	0	mm	K3	0
L4	0	mm	K4	0
L5	0	mm	K5	0
L6	0	mm	K6	0
L7	0	mm	K7	0
L8	0	mm	K8	0
L9	0	mm	K9	0
L10	0	mm	K10	0
L11	0	mm	K11	0
L12	0	mm	K12	0
L13	0	mm	K13	0
L14	0	mm	K14	0
L15	0	mm	K15	0
L16	0	mm	K16	0

Kinematiktyp

Diagramm: Kinematiktyp (Delta-Arm) mit Achsen A1, A2, A3, Längen L1 bis L5 und TCP (Tool Center Point).

Abb. 3-17: Dialog Achslänge und Kopplung der Kinematik

Konfiguriert werden die Parameter

- K-0-0040, Transformation, Längen der Mechanik
- K-0-0041, Transformation, Faktoren

Diese Parameter werden zur Transformation von Achskoordinaten in kartesische Koordinaten benötigt.

Oberflächenfunktionen

Siehe auch [Achslängen und Kopplung auf Seite 108](#).

4. Bewegungsgrenzwerte

Schritt 4: Bewegungsgrenzwerte
Legen Sie die Werte für Bewegungsgrenzen fest.

Maximale Bahngeschwindigkeit: mm/s

Maximale Bahnbeschleunigung: mm/s²

Maximaler Bahnruddruck: mm/s³

Software-Endschalter für Maschinen-Koordinaten (MCS)

	Positiv		Negativ	
Koordinate 1:	999999	mm	-999999	mm
Koordinate 2:	999999	mm	-999999	mm
Koordinate 3:	999999	mm	-999999	mm

Abb. 3-18: Dialog Bewegungsgrenzwerte der Kinematik

Konfiguriert werden die Parameter

- K-0-0003, Maximale Bahngeschwindigkeit
- K-0-0034, Maximale Bahnbeschleunigung (bipolar)
- K-0-0035, Maximaler Bahnruddruck (bipolar)
- K-0-0005, Software-Endschalter, positiv, MCS
- K-0-0006, Software-Endschalter, negativ, MCS

Diese Parameter werden zur Definition der Bewegungsgrenzwerte der Kinematik benötigt.

Siehe auch [Bewegungsgrenzwerte auf Seite 108](#).

5. Koordinatenverschiebung

Schritt 5: Koordinatenverschiebung
Legen Sie die Werte für die Verschiebung bzw. Verdrehung des Koordinatensystems fest.

Verschiebung

Diagramm: 3D-Koordinatensystem mit Achsen X, Y, Z.

X: mm

Y: mm

Z: mm

Verdrehung

Diagramm: 3D-Koordinatensystem mit Achsen X, Y, Z und Rotationsachsen A, B, C.

A: Grad

B: Grad

C: Grad

Abb. 3-19: Dialog Koordinatenverschiebung der Kinematik

Konfiguriert wird der Parameter

- K-0-0042, Verschiebung des BCS-Systems

Dieser Parameter gestattet es, das Basis-Koordinatensystem (BCS) der Kinematik gegenüber dem Maschinen-Koordinatensystem (MCS) zu verdrehen und zu verschieben.

Siehe auch [Koordinatenverschiebung auf Seite 109](#).

Diagnose

Dieses Dialogfenster zeigt die Statusinformationen der jeweiligen Kinematik in grafisch aufbereiteter Form.

In der oberen Hälfte des Fensters werden allgemeine Informationen der Kinematik dargestellt. Der mittlere Teil zeigt die kinematikspezifischen Informationen aller Achsen einer Kinematik in tabellarischer Form.

Die Informationen beinhalten die

- Verschiebung und Verdrehung des gewählten Koordinatensystems zum Maschinen-Koordinatensystem
- Aktuelle Position und programmierte Endposition im gewählten Koordinatensystem

In der unteren Hälfte des Fensters werden Details des Kinematikstatus dargestellt.

Status - Kinematics2

Kinematics2

Kinematiknummer: **10** Kinematikname: **Kinematics2** Kinematiktyp: **Kartheser**

Zustand: Kinematik ist in 'StandStill Complete'

Koordinatensysteme: Maschine (MCS)

Koordinatenverschiebung zum MCS

X	Y	Z	A	B	C	Typ
0	0	0	0	0	0	Cartesian

Nr.	Koordinatenname	Position	Endposition	Einheit
1	Koordinate 1	0	0	mm
2	Koordinate 2	0	0	mm
3	Koordinate 3	0	0	mm
4	Koordinate 4	0	0	Grad
5	Koordinate 5	0	0	Grad
6	Koordinate 6	0	0	Grad

<< Details

Bewegung: Kinematik ist im Zustand "Standstill Complete"

Achsen:

- ☒ Alle Achsen betriebsbereit ("AH" oder "AF")
- ☒ Zielposition erreicht
- ☒ Alle Achsen der Kinematik stehen
- ☐ Homing erforderlich

Fehler / Warnungen:

- ☐ es steht eine Warnung an
- ☐ es steht ein Fehler an
- ☐ Fehlerreaktion aktiviert

Meldungen:

- ☐ Überschleifen
- ☐ Bandsynchron
- ☐ Spline wird berechnet
- ☐ Spline ist gültig

Abb. 3-20: Diagnose einer Kinematik

Koordinatensysteme

Die Einstellung unter "Koordinatensysteme" legt fest, in welchem Koordinatensystem die Positionswerte angezeigt werden sollen.

Es kann zwischen Maschinen-Koordinatensystem (MCS), Achs-Koordinatensystem (ACS), Basis-Koordinatensystem (BCS) sowie jedem konfigurierten Produkt-Koordinatensystem (PCS) gewählt werden:

Oberflächenfunktionen

- **Maschinen-Koordinatensystem (MCS)**= Anzeige der Position im Raum, bezogen auf die Maschinenanlage

Koordinatensysteme

Maschine (MCS)

Koordinatenverschiebung zum MCS

X	Y	Z	A	B	C	Typ
0	0	0	0	0	0	Cartesian

Nr.	Koordinatenname	Position	Endposition	Einheit
1	Koordinate 1	0	0	mm
2	Koordinate 2	0	0	mm
3	Koordinate 3	0	0	mm
4	Koordinate 4	0	0	Grad
5	Koordinate 5	0	0	mm
6	Koordinate 6	0	0	mm

Abb. 3-21: Anzeige der Maschinen-Koordinaten (MCS)

- **Achs-Koordinatensystem (ACS)**= Anzeige der Positionen der einzelnen Achsen der Kinematik

Koordinatensysteme

Achse (ACS)

Koordinatenverschiebung zum MCS

X	Y	Z	A	B	C	Typ
--	--	--	--	--	--	--

Achsnr.	Achsname	Position	Endposition	Einheit
9	vAxis9	0	0	mm
10	vAxis10	0	0	mm
11	vAxis11	0	0	mm
12	vAxis12	0	0	Grad
17	vAxis17	0	0	mm
18	vAxis18	0	0	mm

Abb. 3-22: Anzeige der Achs-Koordinaten (ACS)

- **Basis-Koordinatensystem (BCS)**= Anzeige der Position im Raum, bezogen auf die Basis der Kinematik

Koordinatensysteme

Basis (BCS)

Koordinatenverschiebung zum MCS

X	Y	Z	A	B	C	Typ
0	0	0	0	0	0	Cartesian

Nr.	Koordinatenname	Position	Endposition	Einheit
1	Koordinate 1	0	0	mm
2	Koordinate 2	0	0	mm
3	Koordinate 3	0	0	mm
4	Koordinate 4	0	0	Grad
5	Koordinate 5	0	0	mm
6	Koordinate 6	0	0	mm

Abb. 3-23: Anzeige der Basis-Koordinaten (BCS)

- **Produkt-Koordinatensystem (PCS)** = Anzeige der Position im Raum, bezogen auf den Nullpunkt des Bandes bzw. Produktes

Oberflächenfunktionen

Koordinatensysteme		Koordinatenverschiebung zum MCS						
Produkt 1 (PCS1)		X	Y	Z	A	B	C	Typ
		15	10	0	0	0	0	Cartesian
Nr.	Koordinatenname	Position			Endposition			Einheit
1	Koordinate 1	-15			-15			mm
2	Koordinate 2	-10			-10			mm
3	Koordinate 3	0			0			mm
4	Koordinate 4	0			0			Grad
5	Koordinate 5	0			0			mm
6	Koordinate 6	0			0			mm

Abb. 3-24: Anzeige der Produkt-Koordinaten (PCS)

Liste mit Achskoordinaten

Die Liste der Koordinaten zeigt:

- Nummer der Achse/Koordinate
- Name der Achse/Koordinate
- Aktuelle Position der Achse/Koordinate
- Programmierte Endposition der Achse/Koordinate

3.3.3 Kinematik - Dialoge

Dialoge im Überblick

Beim Öffnen des Ordners der Kinematik im Projekt-Explorer bieten sich einzelne Dialoge an.

Diese Dialoge sind zunächst inaktiv und werden mit dem Onlineschalten des Projektes aktiv.

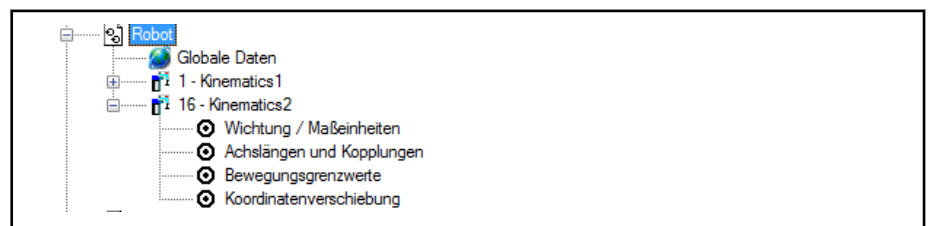


Abb. 3-25: Kinematikdialoge (Robot-Control V2)

Dialoge im Überblick

- [Kap. "Wichtung/Maßeinheiten" auf Seite 107](#)
- [Kap. "Achslängen und Kopplung" auf Seite 108](#)
- [Kap. "Bewegungsgrenzwerte" auf Seite 108](#)
- [Kap. "Koordinatenverschiebung" auf Seite 109](#)

Wichtung/Maßeinheiten

Über diesen Dialog kann die Darstellung der Einheiten zwischen Millimeter [mm] und Inch [in] umgeschaltet werden. Desweiteren können die Achswichtungen aller, der Kinematik zugewiesenen, Achsen in einem Schritt angepasst werden. Diese Anpassungen sind abhängig von der verwendeten Transformation.

Oberflächenfunktionen

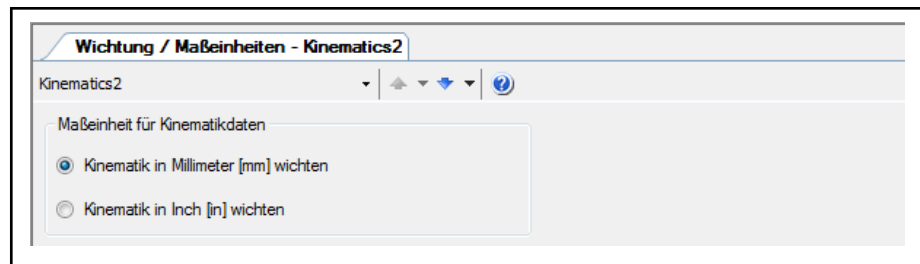


Abb. 3-26: Umschaltung Maßeinheiten und Achswichtungen
Nachstehende Parameter werden ausgewertet:

- [K-0-0054](#), Maßeinheit für Kinematikdaten
- A-0-0053, Wichtungsart für Beschleunigungsdaten
- A-0-0056, Wichtungsart für Geschwindigkeitsdaten
- A-0-0059, Wichtungsart für Positionsdaten

Achslängen und Kopplung

Dieser Dialog gestattet die Einstellung der Transformationslängen und Transformationsfaktoren, die für die Transformation von Achs-Koordinaten in kartesische Koordinaten benötigt werden.

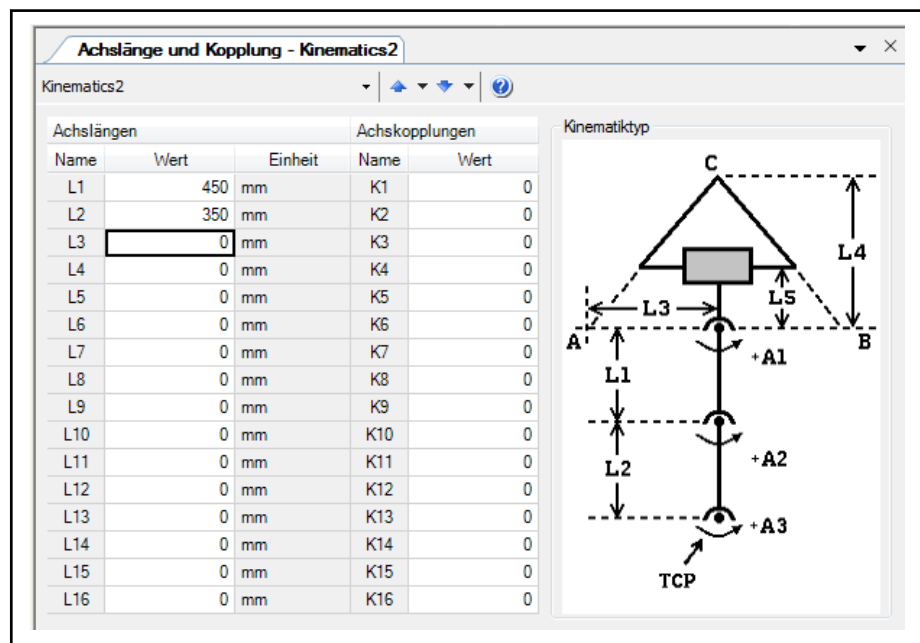


Abb. 3-27: Festlegen der Achslängen und der Achskopplungen
Nachstehende Parameter werden ausgewertet:

- [K-0-0040](#), Transformation, Längen der Mechanik
- [K-0-0041](#), Transformation, Faktoren

Bewegungsgrenzwerte

Es können die nachstehenden Grenzwerte eingegeben werden:

- Maximale Bahngeschwindigkeit
- Maximale Bahnbeschleunigung (bipolar)
- Maximaler Bahnruddruck (bipolar)
- Software-Endschalter, positiv, MCS
- Software-Endschalter, negativ, MCS

Oberflächenfunktionen

Das Dialogfenster gibt einen Überblick über die voreingestellten Bewegungsgrenzwerte der Kinematik.

Software-Endschalter für Maschinen-Koordinaten (MCS)			
	Positiv		Negativ
Koordinate 1:	999999 mm		-999999 mm
Koordinate 2:	999999 mm		-999999 mm

Abb. 3-28: Dialog Bewegungsgrenzwerte

Nachstehende Parameter werden ausgewertet:

- K-0-0003, Maximale Bahngeschwindigkeit
- K-0-0034, Maximale Bahnbeschleunigung (bipolar)
- K-0-0035, Maximaler Bahnrudder (bipolar)
- K-0-0005, Software-Endschalter, positiv, MCS
- K-0-0006, Software-Endschalter, negativ, MCS

Koordinatenverschiebung

Dieser Dialog gestattet es, das Basis-Koordinatensystem (BCS) einer Kinematik gegenüber dem Maschinen-Koordinatensystem (MCS) zu verdrehen und zu verschieben.

Die hier angegebenen Werte stellen somit die Position und Verdrehung der Kinematik im Raum dar.

Das Dialogfenster gibt einen Überblick über die voreingestellte Koordinatenverschiebung der Kinematik.

Abb. 3-29: Übersicht Koordinatenverschiebung

Oberflächenfunktionen

Nachstehender Parameter wird ausgewertet:

- [K-0-0042](#), Verschiebung des BCS-Systems

4 Transformationsschemen

4.1 Transformationsschemen, Überblick

Die ACS-BCS-Transformation rechnet Positionen zwischen Achs-Koordinatensystem (ACS) und Basis-Koordinatensystem (BCS) um (siehe auch [ACS-BCS-Transformation auf Seite 28](#)).

Grundvoraussetzungen

Die Grundvoraussetzungen für eine korrekte Funktionsweise der ACS-BCS-Transformation sind:

- **Grundstellung:** Exakte Einstellung der Achs-Nullpunkte
- **Justierung:** Exakte Justierung der Achswinkel
- **Wichtung:** Korrekte Wichtung der Achsen
- **Drehrichtung:** Korrekte Drehrichtung der Achsen



Ist eine dieser Grundvoraussetzungen nicht erfüllt, so kann keine korrekte Bewegung garantiert werden.

Dies kann zu falschen bzw. unerwarteten Bewegungen führen.

Beteiligte Parameter

Folgende Parameter sind bei der Konfiguration einer ACS-BCS-Transformation beteiligt:

- **K-0-0008:** Achskonfigurationsliste
- **K-0-0031:** Transformationsschema
- **K-0-0040:** Transformation, Längen der Mechanik
- **K-0-0041:** Transformation, Faktoren

Transformations- schema	Name
0	Kartesis auf Seite 112
7	2-Achs-Delta mit optional 1 Orientierungsachse auf Seite 113
8	4-Achs Scara mit linearer Hubachse auf Seite 115
9	3-Achs-Scara mit optional 1 Orientierungsachse auf Seite 119
10	4-Achs-Scara mit optional 1 Orientierungsachse auf Seite 123
12	2-Achs-Parallelogramm auf Seite 127
13	2-Achs-Parallelogramm invers auf Seite 129
14	2-Achs Presse auf Seite 130
18	Umschlingungskinematik auf Seite 132
19	2-Achs-Parallelogramm V1 auf Seite 133
20	3-Achs-Delta mit optional 1 Orientierungsachse auf Seite 135
21	2-Achs-Parallelogramm V2 auf Seite 137
22	3-Achs-Delta Linear mit optional 1 Orientierungsachse auf Seite 138
23	5-Achs-Delta auf Seite 141
30	Adabot auf Seite 145
32	2-Achs-Delta mit asymmetrischem TCP auf Seite 148
38	3-Achs-Twisting Crane auf Seite 149

Transformations-schemen

Transformations-schema	Name
39	3-Achs-Flex Stamper auf Seite 151
40	3-Achs-Delta mit Linearachsen mit optional 1 Orientierungsachse auf Seite 152
45	2-Achs-Delta Slide auf Seite 156
46	3-Achs-Parallelogramm V3 auf Seite 157
47	6-Achs-Karteser auf Seite 158
56	3-Achs-Delta mit 90 Grad Winkeln auf Seite 160
61	4-Achs-Speedpicker auf Seite 163
66	Stewart-Plattform auf Seite 165
73	4-Achs-Scara mit Kurbel auf Seite 169
81	3-Achs Canfield auf Seite 173
100	Benutzerdefinierte Transformation auf Seite 176
111	Erweiterte Benutzerdefinierte Transformation auf Seite 178

Tab. 4-1: Übersicht ACS-BCS-Transformationsschemen

4.2 Transformationsschemen

4.2.1 Karteser

Zur Verwendung dieser Transformation ist die Transformationsnummer 0 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Grundstellung

Beim Karteser (Transformationsschema 0) findet eine Eins-zu-Eins-Transformation zwischen dem Achs-Koordinatensystem (ACS) und dem Basis-Koordinatensystem (BCS) statt.

Aus diesem Grund sind [Move Direct-Bewegungen auf Seite 37](#) in Bezug auf die Bahnbewegung identisch mit [Move Linear-Bewegungen auf Seite 37](#).

Zur Verwendung dieser Transformation ist die Transformationsnummer 0 im Parameter "K-0-0031, Transformationsschema" einzutragen.

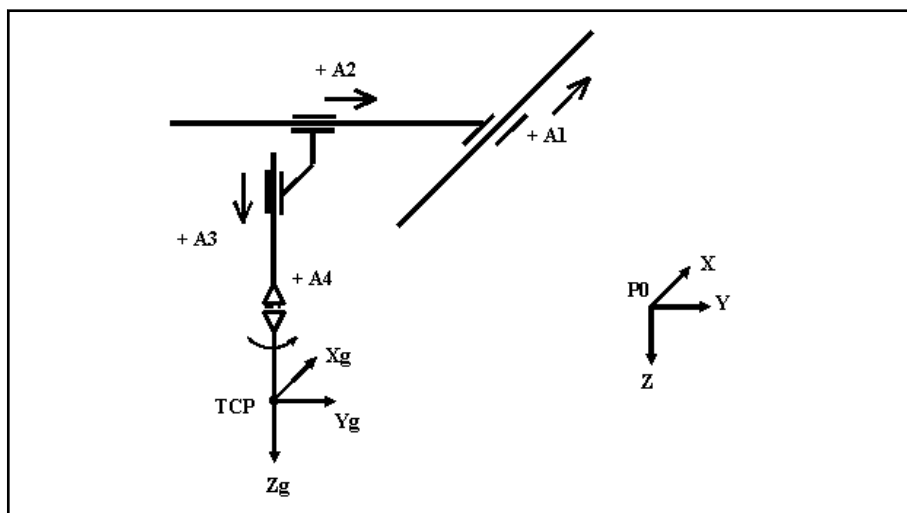


Abb. 4-1: Schematische Ansicht Karteser

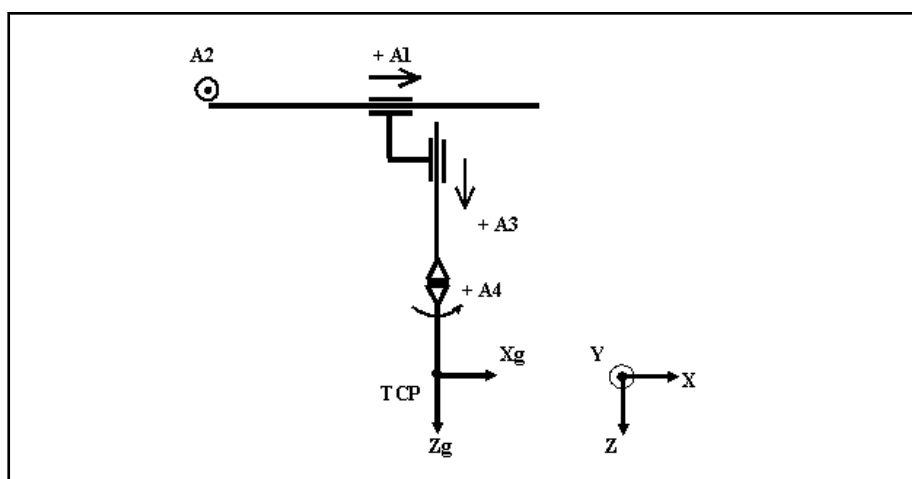


Abb. 4-2: Seitenansicht: ein typisches X-Y-Z-C kartesisches System

Achsdrehrichtungen und Nullstellungen

Die Nullstellungen aller Achsen ist beim Karteser der Ursprungspunkt des Basiskoordinatensystems.

Die Verfahrrichtungen der bis zu 3 Positionierachsen bilden immer ein rechtshändiges kartesisches Koordinatensystem.

Die Nullpunkte und die Verfahrrichtungen der Orientierungsachsen sind die des rechtshändigen kartesischen Koordinatensystems.

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung
A1:	Positionierungsachse	translatorisch absolut
A2:	Positionierungsachse (optional) Orientierungsachse (optional)	translatorisch absolut rotatorisch
A3:	Positionierungsachse (optional) Orientierungsachse (optional)	translatorisch absolut rotatorisch
A4:	Orientierungsachse (optional)	rotatorisch

Tab. 4-2: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

4.2.2 2-Achs-Delta mit optional 1 Orientierungsachse

Zur Verwendung dieser Transformation ist die Transformationsnummer 7 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Transformationsschemen

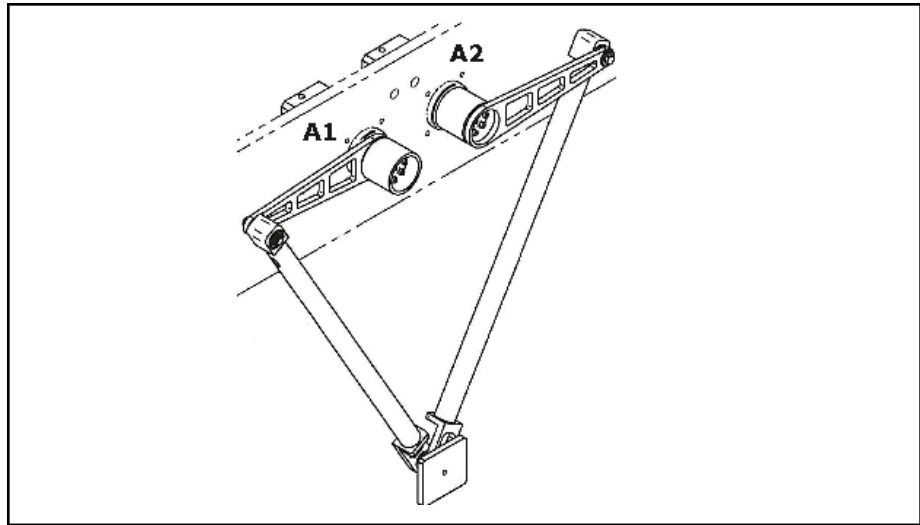


Abb. 4-3: Robotertyp 07, 2-Achs-Delta

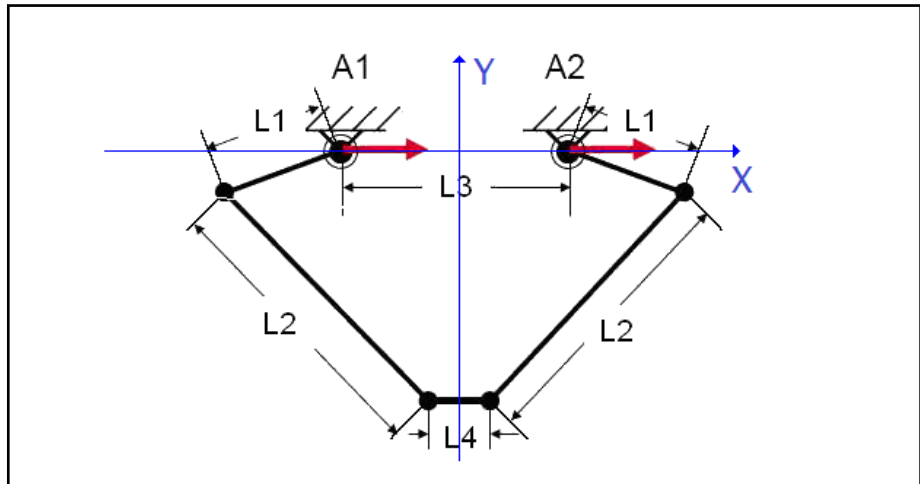


Abb. 4-4: Schematische Ansicht 2-Achs-Delta (X-Y Ansicht)

Achsen-Drehrichtungen und Nullstellungen

Abb. 4-3 "Robotertyp 07, 2-Achs-Delta" auf Seite 114 zeigt einen Bipod-Roboter in 3D-Darstellung mit den Achspositionen $A1=180^\circ$ und $A2 = 0^\circ$.

In der Nullstellung stehen beide Positionierachsen A1 und A2 waagrecht in Richtung der X-Achse (siehe rote Pfeile in Abb. 4-4 "Schematische Ansicht 2-Achs-Delta (X-Y Ansicht)" auf Seite 114).

- Die dritte Achse rotiert um die Z-Achse. Der Nullpunkt der 3. Achse (Orientierungsachse) ist in positive Richtung der X-Achse
- Die Armlänge L_1 ist gemessen vom Rotationsmittelpunkt bis zum Mittelpunkt der Aufnahme der Stäbe
- Die Stablänge L_2 ist gemessen von den beidseitigen Aufnahmepunkten
- L_3 ist der Abstand der Rotationsmittelpunkte der beiden Arme
- L_4 ist die Länge der Werkzeugplatte gemessen an den Lagerungen der Stäbe
- Die Drehrichtung der Positionierachsen A1 und A2 ist positiv entgegen dem Uhrzeigersinn
- Der kartesische Nullpunkt des Bipod-Roboters liegt in der Mitte der beiden Armdrehpunkte A1 und A2

Transformationsschemen

Koordinaten eines Punktes Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	+ 90 .. 360 Grad
A2:	Positionierungsachse	rotatorisch absolut	+ 90 .. -180 Grad
A3:	Orientierungsachse	rotatorisch abs./modulo	optional

Tab. 4-3: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms (Schenkel) am Motor	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des Stabs zur Werkzeugplatte	mm bzw. inch (siehe K-0-0054)	>0
L3:	Abstand der beiden Rotationspunkte der Arme	mm bzw. inch (siehe K-0-0054)	>0
L4:	Abstand der Stäbe an der Grundplatte	mm bzw. inch (siehe K-0-0054)	>0

Tab. 4-4: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.3 4-Achs Scara mit linearer Hubachse

Zur Verwendung dieser Transformation ist die Transformationsnummer 8 im Parameter "K-0-0031, Transformationsschema" einzutragen.

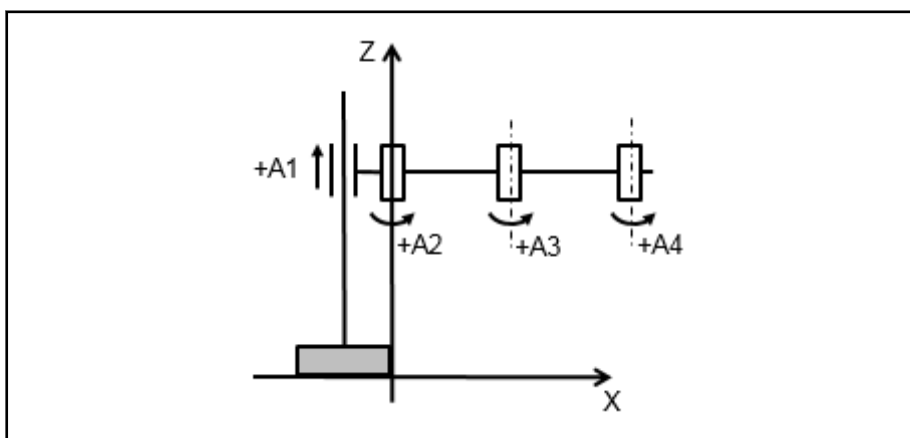


Abb. 4-5: Seitenansicht 4-Achs Scara mit linearem Hub

Transformationsschemen

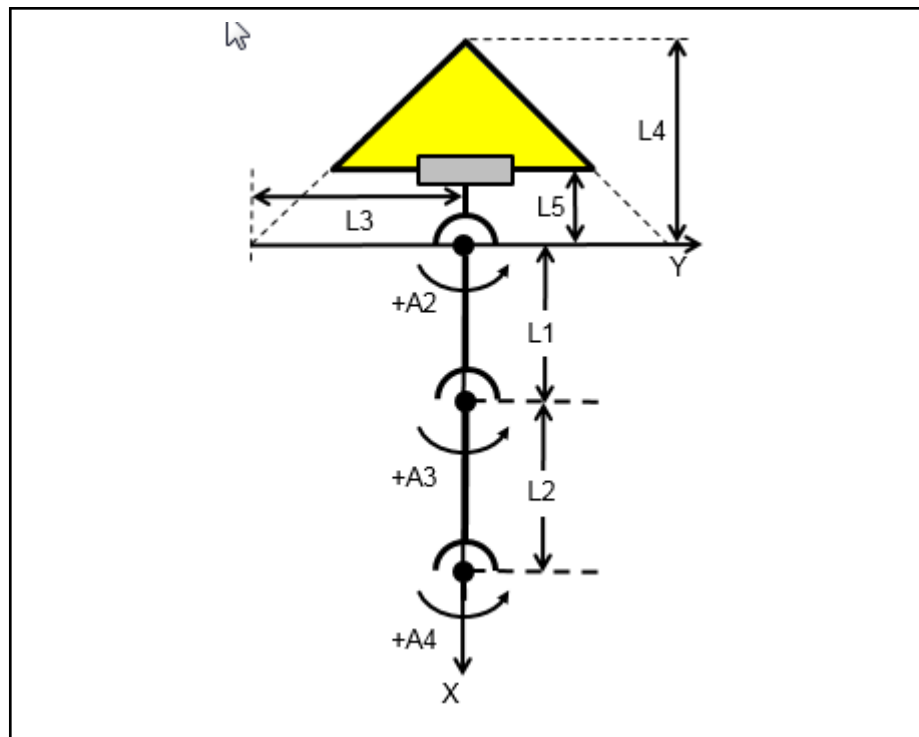


Abb. 4-6: Draufsicht 4-Achs Scara mit linearem Hub

Achsen-Drehrichtungen und Nullstellungen

- Nimmt der Roboter die in den Abbildungen dargestellte Stellung ein, so müssen sämtliche Achsen in Maschinenkoordinaten (MCS) Null anzeigen.
Dies ist durch eine entsprechende Einstellung der Referenzpunkt-Istwerte zu gewährleisten
- In der Nullstellung stehen die Achsen 2, 3 und 4 nach vorn
- Der Nullpunkt der 1. Achse ist frei wählbar
- Die Armlängen L_1 , L_2 und die positiven Drehrichtungen der Achsen sind ebenfalls dargestellt
- Mit Hilfe der Längen L_3 , L_4 , L_5 lässt sich ein verbotener Bereich definieren, in den der Tool-Center-Point (TCP) nicht eindringen kann
- Dies dient der Kollisionsvermeidung, wenn die Verfahrbereiche der Achsen 2, 3 so groß sind, dass der Roboter gegen sein eigenes Stativ fahren könnte.
Dieser verbotene Bereich ist in der Draufsicht gelb dargestellt
- Ist die Länge $L_3=0$, so ist keine Bereichsüberwachung aktiv
- Die Längen L_1 , L_2 , L_3 , L_4 , L_5 sind immer positiv einzugeben, wobei L_1 , L_2 ungleich Null sein müssen
- K_1 gibt die mechanische Kopplung der Achse A_2 mit der Achse A_3 an
Eine solche Kopplung liegt vor, wenn sich beim Verfahren der Achsen A_2 die Achse A_3 mit bewegt, ohne dass das Messsystem der Achse A_3 diese Bewegung erfasst. Dies ist in der Regel dann der Fall, wenn der Motor der Achse A_3 nicht direkt auf der Achse sitzt, sondern auf dem Stativ. Der Kopplungsfaktor K_1 ist invers zum Kopplungsfaktor K_2 definiert.

Beispiel:

Kopplungsfaktor K_1

$K_1 = 0$: Eine Bewegung der Achse A_2 hat Auswirkungen auf die Stellung des zweiten Arms im Raum: Der Winkel zwischen dem ersten Arm und dem zweiten Arm verändert sich nicht, wenn Achse A_2 bewegt wird. Die Stellung des zweiten Arms bezüglich der X-Achse verändert sich im gleichen Maße wie Achse A_1 bewegt wird.

$K_1 = 1$: Eine Bewegung der Achse A_2 hat keine Auswirkung auf die Stellung des zweiten Arms im Raum: Der Winkel zwischen dem ersten Arm und dem zweiten Arm verändert sich im gleichen Maße wie Achse A_2 bewegt wird. Die Stellung des zweiten Arms bezüglich der X-Achse bleibt konstant.

- K_2 gibt die mechanische Kopplung der Achsen A_2 und A_3 mit Achse A_4 an.

Eine solche Kopplung liegt vor, wenn sich beim Verfahren der Achsen A_2 und A_3 die Achse A_4 mit bewegt, ohne dass das Messsystem der Achse A_4 diese Bewegung erfasst. Dies ist in der Regel dann der Fall, wenn der Motor der Achse A_4 nicht direkt auf der Achse, sondern auf dem Stativ sitzt.

Beispiel:

Kopplungsfaktor K_2

$K_2 = 0$: Eine Bewegung der Achse A_2 oder A_3 hat Auswirkungen auf die Orientierung im Raum. Die Bewegung muss von Achse A_4 ausgeglichen werden.

$K_2 = 1$: Eine Bewegung der Achse A_2 oder A_3 hat keine Auswirkungen auf die Orientierung im Raum. Die Bewegung muss nicht von Achse A_4 ausgeglichen werden.



Der Verfahrbereich der Achsen A_2 und A_3 darf maximal ± 179 Grad betragen. Der Verfahrbereich der Achse A_1 und A_4 ist unbeschränkt.

Armigkeitswahl bei PTP-Bewegungen

Wird eine Bewegung [Punkt zu Punkt auf Seite 42](#) Kap. 2.7.4 "Bewegungsart Punkt zu Punkt" auf Seite 42 ausgeführt und der anzufahrende Punkt nicht im ACS angegeben, kann dieser Punkt auf zwei verschiedene Weisen angefahren werden.

Transformationsschemen

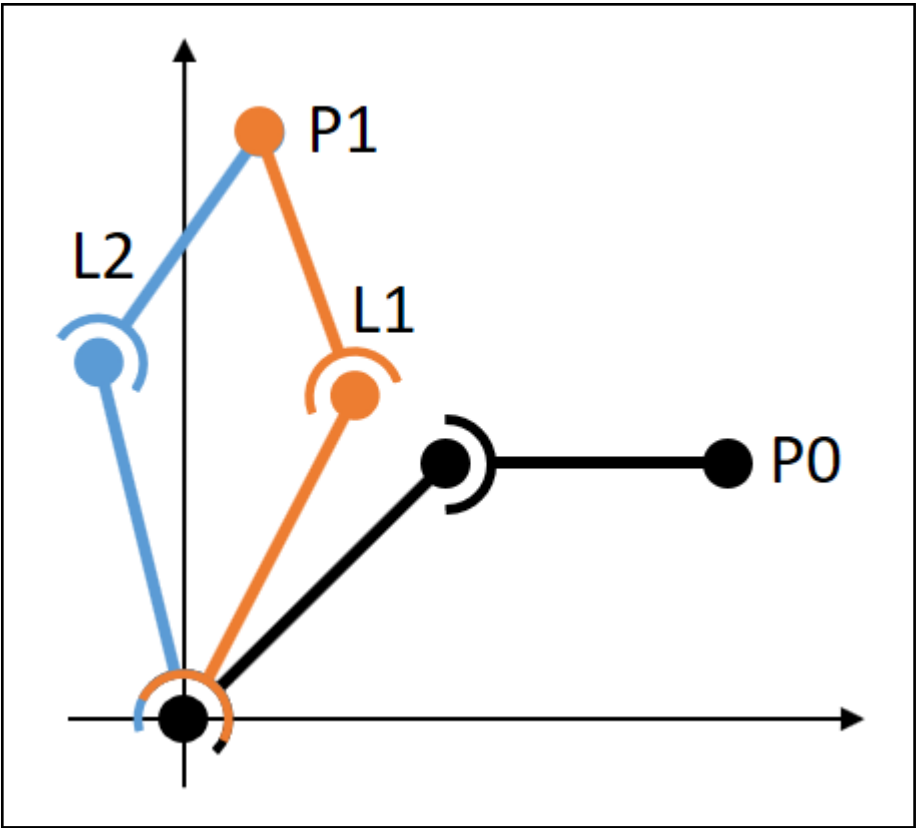


Abb. 4-7: Verschiedene Armigkeiten bei einem Scara Roboter

Von eine Punkt P0 kann durch eine Punkt zu Punkt Bewegung der Punkt P1, der nicht im ACS angegeben wurde, auf zwei verschiedene Weisen angefahren werden (L1 und L2).

Mittels K_4 kann das Verhalten der Armwahl konfiguriert werden:

- 0: Es wird der Zielpunkt so angefahren, dass die Armigkeit zum Startpunkt nicht geändert wird
- 1: Es wird der Zielpunkt so angefahren, dass die Bewegung des ersten Arms vom Start- zum Zielpunkt minimiert wird. Dabei wird die Armigkeit so gewählt, dass Achse 1 die Grenzen von $\pm 179^\circ$ nicht überschreitet

Beispiel:

Konfiguration K_4

- $K_4 = 0$: Punkt P1 wird mit L2 angefahren. Die Armigkeit ändert sich nicht während der Bewegung
- $K_4 = 1$: Punkt P1 wird mit L2 angefahren. L2 hat die geringere Bewegung von Arm 1 zur Folge

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	Translatorisch absolut	
A2:	Positionierungsachse	Rotatorisch absolut	+/- 179 Grad

Transformationsschemen

A3:	Positionierungsachse	Rotatorisch absolut	+/- 179 Grad
A4:	Orientierungsachse	Rotatorisch	Optional

Tab. 4-5: K-0-0008, Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des 1. Arms	mm bzw. Inch (siehe K-0-0054)	>0
L2:	Länge des 2. Arms	mm bzw. Inch (siehe K-0-0054)	>0
L3:	Abstand vom Nullpunkt bis zum Schnittpunkt des Dreiecks mit der Y-Achse	mm bzw. Inch (siehe K-0-0054)	Optional, siehe (1) >=0
L4:	Abstand vom Nullpunkt bis zur Spitze des Dreiecks	mm bzw. Inch (siehe K-0-0054)	Optional, siehe (1) >=0
L5:	Abstand vom Nullpunkt zum Stativ	mm bzw. Inch (siehe K-0-0054)	Optional, siehe (1) >=0

Tab. 4-6: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

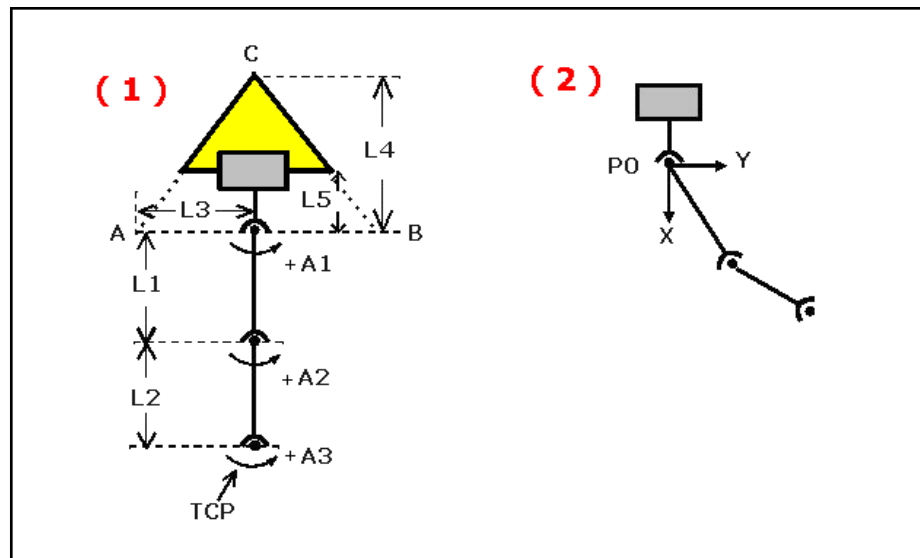
Trafo-Faktor	Achskonfiguration	Maßeinheit	Achskopplung
K1:	Mechanische Kopplung zwischen Achse 2 und 3	Faktor	Optional
K2:	Mechanische Kopplung zwischen Achsen 2, 3 und 4	Faktor	Optional
K4:	Armigkeitswahl bei PTP-Bewegungen und Punktvorgabe im nicht ACS • 0: Armigkeit wird beibehalten • 1: Armigkeit wird so gewählt, dass die Bewegung des ersten Arms minimiert wird	Faktor	Optional

Tab. 4-7: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.4 3-Achs-Scara mit optional 1 Orientierungsachse

Zur Verwendung dieser Transformation ist die Transformationsnummer 9 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Transformationsschemen



(1) Draufsicht mit Nullstellungen aller Achsen, gelb: verbotener Bereich

(2) Nullpunkt-Bezug aller Achsen

Abb. 4-8: Schematische Ansicht 3-Achs-Scara mit optional 1 Orientierungsachse

Achs-Drehrichtungen und Nullstellungen

Abb. 4-8 "Schematische Ansicht 3-Achs-Scara mit optional 1 Orientierungsachse" auf Seite 120: (1) zeigt in einer Draufsicht die Nullstellungen aller Achsen.

- Nimmt der Roboter die in den Abbildungen dargestellte Stellung ein, so müssen sämtliche Achsen in Maschinenkoordinaten (MCS) Null anzeigen

Dies ist durch eine entsprechende Einstellung der Referenzpunkt-Istwerte zu gewährleisten

- Die Armlängen L_1 , L_2 und die positiven Drehrichtungen der Achsen sind ebenfalls in den Abbildungen dargestellt
- Mit Hilfe der Längen L_3 , L_4 , L_5 lässt sich ein verbotener Bereich definieren, in den der Tool-Center-Point (TCP) nicht eindringen kann. Dies dient der Kollisionsvermeidung, wenn die Verfahrbereiche der Achsen 1, 2 so groß sind, dass der Roboter gegen sein eigenes Stativ fahren könnte. Dieser verbotene Bereich ist in (1) gelb dargestellt
- Ist die Länge $L_3=0$, so ist keine Bereichsüberwachung aktiv
- Die Längen L_1 , L_2 , L_3 , L_4 , L_5 sind immer positiv einzugeben, wobei L_1 , L_2 ungleich Null sein müssen
- K_1 gibt die mechanische Kopplung der Achsen A_1 und A_2 mit Achse A_3 an.

Eine solche Kopplung liegt vor, wenn sich beim Verfahren der Achsen A_1 und A_2 die Achse A_3 mit bewegt, ohne dass das Messsystem der Achse A_3 diese Bewegung erfasst. Dies ist in der Regel dann der Fall, wenn der Motor der Achse A_3 nicht direkt auf der Achse, sondern auf dem Stativ sitzt

Beispiel:

Kopplungsfaktor K_1

$K_1 = 0$: Eine Bewegung der Achse A_1 oder A_2 hat Auswirkungen auf die Orientierung im Raum. Die Bewegung muss von Achse A_3 ausgeglichen werden.

$K_1 = 1$: Eine Bewegung der Achse A_1 oder A_2 hat keine Auswirkungen auf die Orientierung im Raum. Die Bewegung muss nicht von Achse A_3 ausgeglichen werden.

- K_2 gibt die mechanische Kopplung der Achse A_1 mit der Achse A_2 an.
Eine solche Kopplung liegt vor, wenn sich beim Verfahren der Achsen A_1 die Achse A_2 mit bewegt, ohne dass das Messsystem der Achse A_2 diese Bewegung erfasst. Dies ist in der Regel dann der Fall, wenn der Motor der Achse A_2 nicht direkt auf der Achse, sondern auf dem Stativ sitzt. Der Kopplungsfaktor K_2 ist invers zum Kopplungsfaktor K_1 definiert.

Beispiel:

Kopplungsfaktor K_2

$K_2 = 0$: Eine Bewegung der Achse A_1 hat Auswirkungen auf die Stellung des zweiten Arms im Raum: Der Winkel zwischen dem ersten Arm und dem zweiten Arm verändert sich nicht, wenn Achse A_1 bewegt wird. Die Stellung des zweiten Arms bezüglich der X-Achse verändert sich im gleichen Maße wie Achse A_1 bewegt wird.

$K_2 = 1$: Eine Bewegung der Achse A_1 hat keine Auswirkung auf die Stellung des zweiten Arms im Raum: Der Winkel zwischen dem ersten Arm und dem zweiten Arm verändert sich im gleichen Maße wie Achse A_1 bewegt wird. Die Stellung des zweiten Arms bezüglich der X-Achse bleibt konstant.



Der Verfahrbereich der Achsen 1 und 2 darf maximal +/-179 Grad betragen. Der Verfahrbereich der 3. Achse ist unbeschränkt.

Das zugehörige Basis-Koordinatensystem (BCS) ist in [Abb. 4-8 "Schematische Ansicht 3-Achs-Scara mit optional 1 Orientierungsachse"](#) auf Seite 120 (2) durch die 2 Hauptachsen (x, y) gekennzeichnet.

Der Basiskoordinaten-Nullpunkt liegt im Fußpunkt P_0 des Roboters.

Armigkeitswahl bei PTP-Bewegungen

Wird eine Bewegung [Punkt zu Punkt auf Seite 42](#)Kap. 2.7.4 "Bewegungsart Punkt zu Punkt" auf Seite 42 ausgeführt und der anzufahrende Punkt nicht im ACS angegeben, kann dieser Punkt auf zwei verschiedene Weisen angefahren werden.

Transformationsschemen

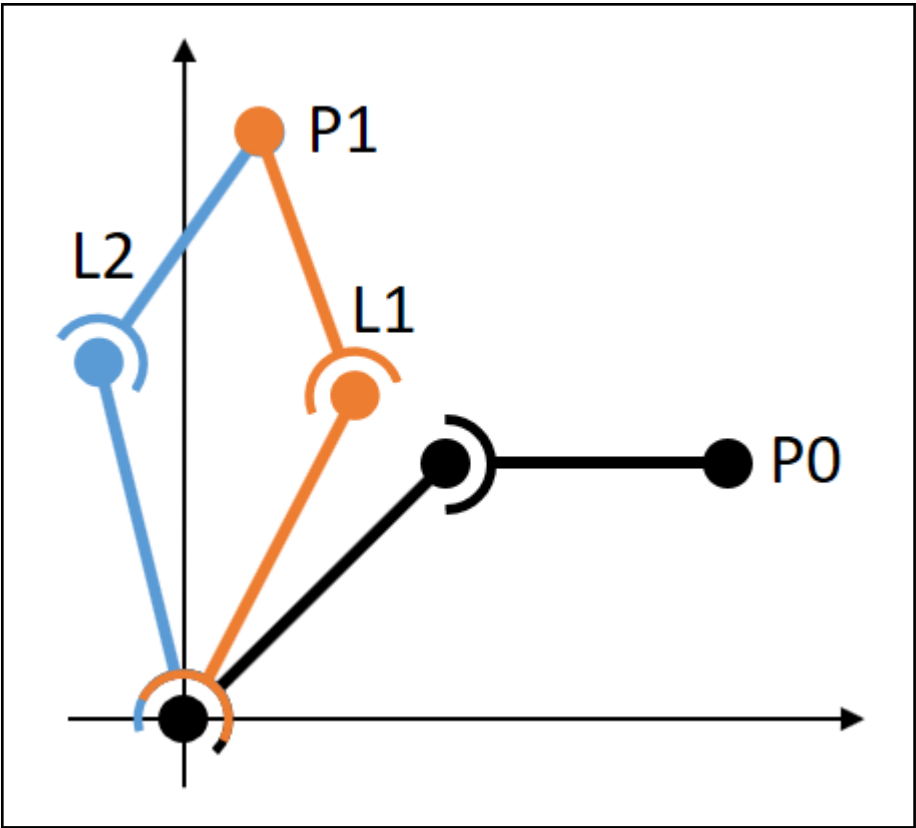


Abb. 4-9: Verschiedene Armigkeiten bei einem Scara Roboter
Von eine Punkt P0 kann durch eine Punkt zu Punkt Bewegung der Punkt P1, der nicht im ACS angegeben wurde, auf zwei verschiedene Weisen angefahren werden (L1 und L2).

- Mittels K_4 kann das Verhalten der Armwahl konfiguriert werden:
- 0: Es wird der Zielpunkt so angefahren, dass die Armigkeit zum Startpunkt nicht geändert wird
 - 1: Es wird der Zielpunkt so angefahren, dass die Bewegung des ersten Arms vom Start- zum Zielpunkt minimiert wird. Dabei wird die Armigkeit so gewählt, dass Achse 1 die Grenzen von $\pm 179^\circ$ nicht überschreitet

Beispiel:

Konfiguration K_4

- $K_4 = 0$: Punkt P1 wird mit L2 angefahren. Die Armigkeit ändert sich nicht während der Bewegung
- $K_4 = 1$: Punkt P1 wird mit L2 angefahren. L2 hat die geringere Bewegung von Arm 1 zur Folge

Koordinaten eines Punktes Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	Rotatorisch absolut	+/- 179 Grad

Transformationsschemen

A2:	Positionierungsachse	Rotatorisch absolut	+/- 179 Grad
A3:	Orientierungsachse	Rotatorisch	Optional

Tab. 4-8: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des 1. Arms	mm bzw. Inch (siehe K-0-0054)	>0
L2:	Länge des 2. Arms	mm bzw. Inch (siehe K-0-0054)	>0
L3:	Abstand vom Nullpunkt bis zum Schnittpunkt der Geraden AC	mm bzw. Inch (siehe K-0-0054)	>=0
L4:	Abstand vom Nullpunkt bis zur Spitze C des Dreiecks	mm bzw. Inch (siehe K-0-0054)	>=0
L5:	Abstand vom Nullpunkt zur Geraden AB	mm bzw. Inch (siehe K-0-0054)	>=0

Tab. 4-9: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

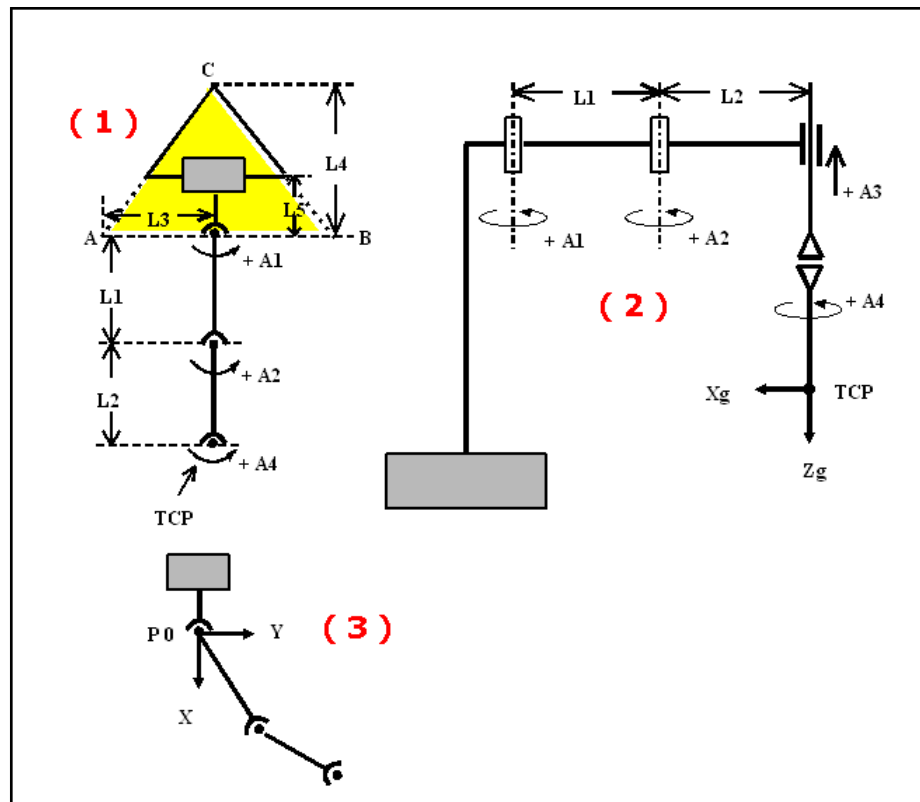
Trafo-Faktor	Achskonfiguration	Maßeinheit	Achskopplung
K1:	Mechanische Kopplung zwischen Achse 1, 2 und 3	Faktor	Optional
K2:	Mechanische Kopplung zwischen Achse 1 und 2	Faktor	Optional
K4:	Armigkeitswahl bei PTP-Bewegungen und Punktvorgabe im nicht ACS • 0: Armigkeit wird beibehalten • 1: Armigkeit wird so gewählt, dass die Bewegung des ersten Arms minimiert wird	Faktor	Optional

Tab. 4-10: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.5 4-Achs-Scara mit optional 1 Orientierungsachse

Zur Verwendung dieser Transformation ist die Transformationsnummer 10 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Transformationsschemen



- (1) Draufsicht mit Nullstellungen aller Achsen, gelb: verbotener Bereich
 (2) Seitenansicht mit Nullstellungen aller Achsen
 (3) Nullpunkt-Bezug aller Achsen

Abb. 4-10: Schematische Ansicht 4-Achs-Scara mit optional 1 Orientierungsachse

Achs-Drehrichtungen und Nullstellungen

Abb. 4-10 "Schematische Ansicht 4-Achs-Scara mit optional 1 Orientierungsachse" auf Seite 124: (1) zeigt in einer Draufsicht und (2) in einer Seitenansicht die Nullstellungen aller Achsen.

- Nimmt der Roboter die in den Abbildungen dargestellte Stellung ein, so müssen sämtliche Achsen in Maschinenkoordinaten (MCS) Null anzeigen.
Dies ist durch eine entsprechende Einstellung der Referenzpunkt-Istwerte zu gewährleisten.
- In der Nullstellung stehen die Achsen 1, 2, 4 nach vorne.
- Der Nullpunkt der 3. Achse ist frei wählbar.
- Die Armlängen L_1 , L_2 und die positiven Drehrichtungen der Achsen sind ebenfalls dargestellt.
- Mit Hilfe der Längen L_3 , L_4 , L_5 lässt sich ein verbotener Bereich definieren, in den der Tool-Center-Point (TCP) nicht eindringen kann.
- Dies dient der Kollisionsvermeidung, wenn die Verfahrbereiche der Achsen 1, 2 so groß sind, dass der Roboter gegen sein eigenes Stativ fahren könnte.
Dieser verbotene Bereich ist in (1) gelb dargestellt.
- Ist die Länge $L_3=0$, so ist keine Bereichsüberwachung aktiv.
- Die Längen L_1 , L_2 , L_3 , L_4 , L_5 sind immer positiv einzugeben, wobei L_1 , L_2 ungleich Null sein müssen.

Transformationsschemen

- Die Transformation kann eine mechanische Kopplung der Achsen 1, 2 mit der 4. Achse berücksichtigen.

Eine solche Kopplung liegt vor, wenn sich beim Verfahren der Achsen 1, 2 die 4. Achse mit bewegt, ohne dass das Meßsystem der 4. Achse diese Bewegung erfaßt. Dies ist in der Regel dann der Fall, wenn der Motor der 4. Achse nicht direkt auf der Achse, sondern auf dem Stativ sitzt.

Diese Kopplung ist über den Parameter [K-0-0041](#) einzustellen (siehe oben).

- Desweiteren kann die Transformation eine mechanische Kopplung der 4. Achse mit der 3. Achse berücksichtigen.

Eine solche Kopplung liegt vor, wenn sich beim Verfahren der Achse 4 die 3. Koordinate (Z-Achse) verändert, ohne dass das Meßsystem der 3. Achse diese Bewegung erfaßt.

Dies ist beispielsweise dann der Fall, wenn es sich bei den Achsen 3 und 4 um eine Hub-, Drehspindel handelt.

Der Kopplungsfaktor ist folgendermaßen definiert:

- $K_{43} = \Delta z / \Delta A_4$

Dabei bezeichnet ΔA_4 den vom Meßsystem erfaßten Achswinkel und Δz die Änderung der Koordinate.



Der Verfahrbereich der Achsen 1 und 2 darf maximal ± 179 Grad betragen.

Der Verfahrbereich der Achsen 3 und 4 ist unbeschränkt.

Das zugehörige Basiskoordinaten-System ist durch die 3 Hauptachsen (x, y, z) gekennzeichnet.

Der Basiskoordinaten-Nullpunkt liegt im Fußpunkt P_0 des Roboters.

Die Transformation kann auch für einen 2- bzw. 3- Achs-Scara verwendet werden.

Armigkeitswahl bei PTP-Bewegungen

Wird eine Bewegung [Punkt zu Punkt auf Seite 42](#) [Kap. 2.7.4 "Bewegungsart Punkt zu Punkt" auf Seite 42](#) ausgeführt und der anzufahrende Punkt nicht im ACS angegeben, kann dieser Punkt auf zwei verschiedene Weisen angefahren werden.

Transformationsschemen

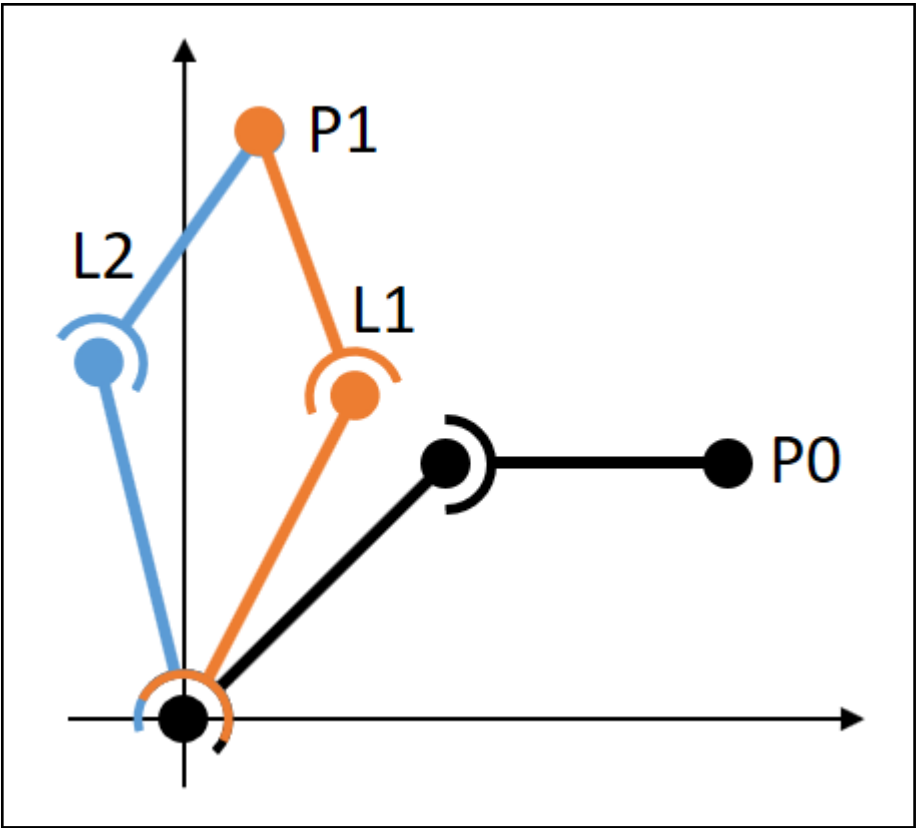


Abb. 4-11: Verschiedene Armigkeiten bei einem Scara Roboter

Von eine Punkt P0 kann durch eine Punkt zu Punkt Bewegung der Punkt P1, der nicht im ACS angegeben wurde, auf zwei verschiedene Weisen angefahren werden (L1 und L2).

Mittels K_4 kann das Verhalten der Armwahl konfiguriert werden:

- 0: Es wird der Zielpunkt so angefahren, dass die Armigkeit zum Startpunkt nicht geändert wird
- 1: Es wird der Zielpunkt so angefahren, dass die Bewegung des ersten Arms vom Start- zum Zielpunkt minimiert wird. Dabei wird die Armigkeit so gewählt, dass Achse 1 die Grenzen von $\pm 179^\circ$ nicht überschreitet

Beispiel:

Konfiguration K_4

- $K_4 = 0$: Punkt P1 wird mit L2 angefahren. Die Armigkeit ändert sich nicht während der Bewegung
- $K_4 = 1$: Punkt P1 wird mit L2 angefahren. L2 hat die geringere Bewegung von Arm 1 zur Folge

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	+/- 179 Grad
A2:	Positionierungsachse	rotatorisch absolut	+/- 179 Grad

Transformationsschemen

A3:	Positionierungsachse	translatorisch absolut	optional
A4:	Orientierungsachse	rotatorisch	optional

Tab. 4-11: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des 1. Arms	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des 2. Arms	mm bzw. inch (siehe K-0-0054)	>0
L3:	Abstand vom Nullpunkt bis zum Schnittpunkt der Geraden AC	mm bzw. inch (siehe K-0-0054)	optional, siehe (1) ≥0
L4:	Abstand vom Nullpunkt bis zur Spitze C des Dreiecks	mm bzw. inch (siehe K-0-0054)	optional, siehe (1) ≥0
L5:	Abstand vom Nullpunkt zur Geraden AB	mm bzw. inch (siehe K-0-0054)	optional, siehe (1) ≥0

Tab. 4-12: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Faktor	Achskonfiguration	Maßeinheit	Achskopplung
K1:	mechanische Kopplung zwischen Achse 1, 2 und 4	Faktor	optional
K2:	mechanische Kopplung zwischen Motor 4 und 3 (Hub-/Drehspindel)	mm/Grad	optional
K4:	Armigkeitswahl bei PTP-Bewegungen und Punktvorgabe im nicht ACS • 0: Armigkeit wird beibehalten • 1: Armigkeit wird so gewählt, dass die Bewegung des ersten Arms minimiert wird	Faktor	Optional

Tab. 4-13: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.6 2-Achs-Parallelogramm

Zur Verwendung dieser Transformation ist die Transformationsnummer 12 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Transformationsschemen

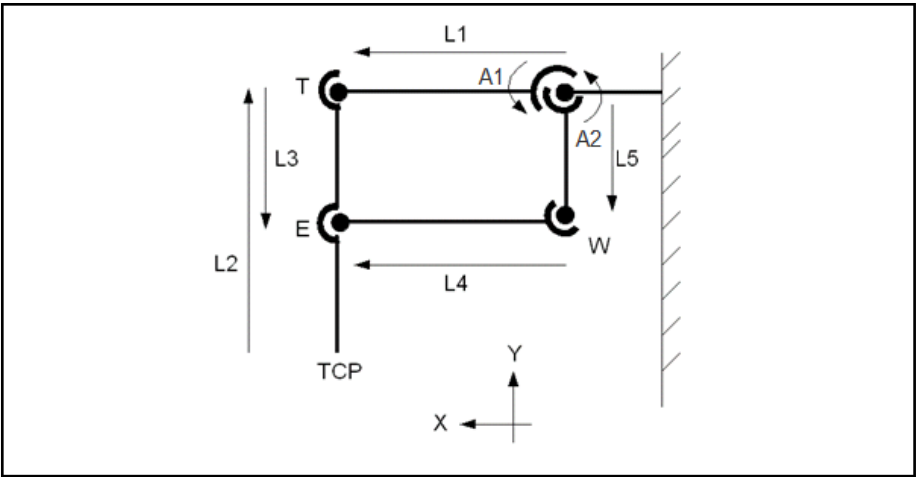


Abb. 4-12: Schematische Ansicht 2-Achs-Parallelogramm

Achs-Drehrichtungen und Nullstellungen

Abb. 4-12 "Schematische Ansicht 2-Achs-Parallelogramm" auf Seite 128 zeigt

- Die schematische Darstellung der Kinematik, bei Nullstellung aller Achsen
- Die Armlängen L_1 , L_2 , L_3 , L_4 , L_5 und die positiven Drehrichtungen der Achsen in Pfeilrichtung
- Der Verfahrbereich der Achsen ist mechanisch so zu begrenzen, dass der Winkel zwischen den beiden Armen mit der Längenbezeichnung L_1 und L_5 nie gleich 180° beträgt. Ansonsten gibt es je nach Mechanik, 2 unterschiedliche mathematische Lösungen, was zu ungewünschten Bewegungen führen kann
- Die Längen L_1 , L_2 , L_3 , L_4 , L_5 sind immer positiv einzugeben und müssen immer ungleich Null sein
- Das zugehörige Basis-Koordinatensystem ist durch die 2 Hauptachsen (x, y) gekennzeichnet
- Der Basiskoordinaten-Nullpunkt liegt im Schnittpunkt der beiden Längenangaben L_1 und L_5

Koordinaten eines Punktes Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	
A2:	Positionierungsachse	rotatorisch absolut	

Tab. 4-14: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des 1. Arms	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des 2. Arms	mm bzw. inch (siehe K-0-0054)	>0

Transformationsschemen

L3:	Länge des 3. Arms	mm bzw. inch (siehe K-0-0054)	>0
L4:	Länge des 4. Arms	mm bzw. inch (siehe K-0-0054)	>0
L5:	Länge des 5. Arms	mm bzw. inch (siehe K-0-0054)	>0

Tab. 4-15: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.7 2-Achs-Parallelogramm invers

Zur Verwendung dieser Transformation ist die Transformationsnummer 13 im Parameter "K-0-0031, Transformationsschema" einzutragen.

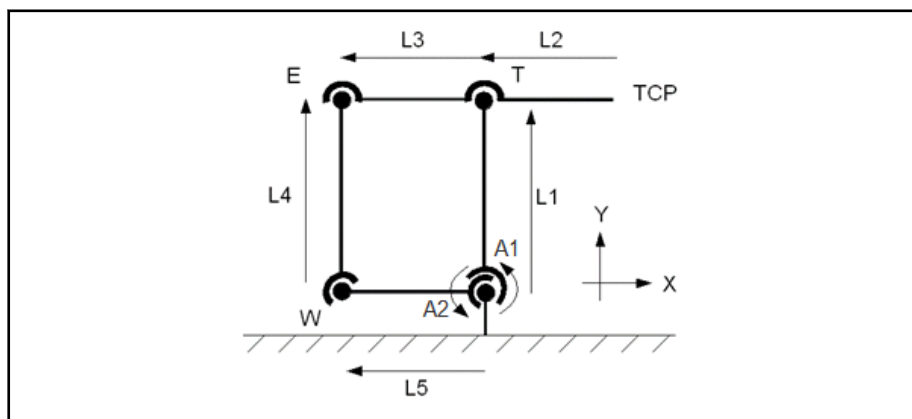


Abb. 4-13: Schematische Ansicht 2-Achs-Parallelogramm

Achs-Drehrichtungen und Nullstellungen

Abb. 4-13 "Schematische Ansicht 2-Achs-Parallelogramm" auf Seite 129 zeigt

- Die schematische Darstellung der Kinematik, bei Nullstellung aller Achsen.
- Die Armlängen L_1 , L_2 , L_3 , L_4 , L_5 und die positiven Drehrichtungen der Achsen in Pfeilrichtung.
- Der Verfahrbereich der Achsen ist mechanisch so zu begrenzen, dass der Winkel zwischen den beiden Armen mit der Längenbezeichnung L_1 und L_5 nie 180° beträgt. In diesem Fall gibt es, zwei unterschiedliche mathematische Lösungen, was zu ungewünschten Bewegungen führen kann.
- Die Längen L_1 , L_2 , L_3 , L_4 , L_5 sind immer positiv und ungleich Null einzugeben.
- Das zugehörige Basiskoordinaten-System ist durch die 2 Hauptachsen (x, y) gekennzeichnet.
- Der Basiskoordinaten-Nullpunkt liegt im Schnittpunkt der beiden Längenangaben L_1 und L_5 .

Koordinaten eines Punktes Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	
A2:	Positionierungsachse	rotatorisch absolut	

Tab. 4-16: K-0-0008 Achskonfigurationsliste, Achszuordnung

Transformationsschemen

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des 1. Arms	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des 2. Arms	mm bzw. inch (siehe K-0-0054)	>0
L3:	Länge des 3. Arms	mm bzw. inch (siehe K-0-0054)	>0
L4:	Länge des 4. Arms	mm bzw. inch (siehe K-0-0054)	>0
L5:	Länge des 5. Arms	mm bzw. inch (siehe K-0-0054)	>0

Tab. 4-17: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.8 2-Achs Presse

Zur Verwendung dieser Transformation ist die Transformationsnummer 14 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

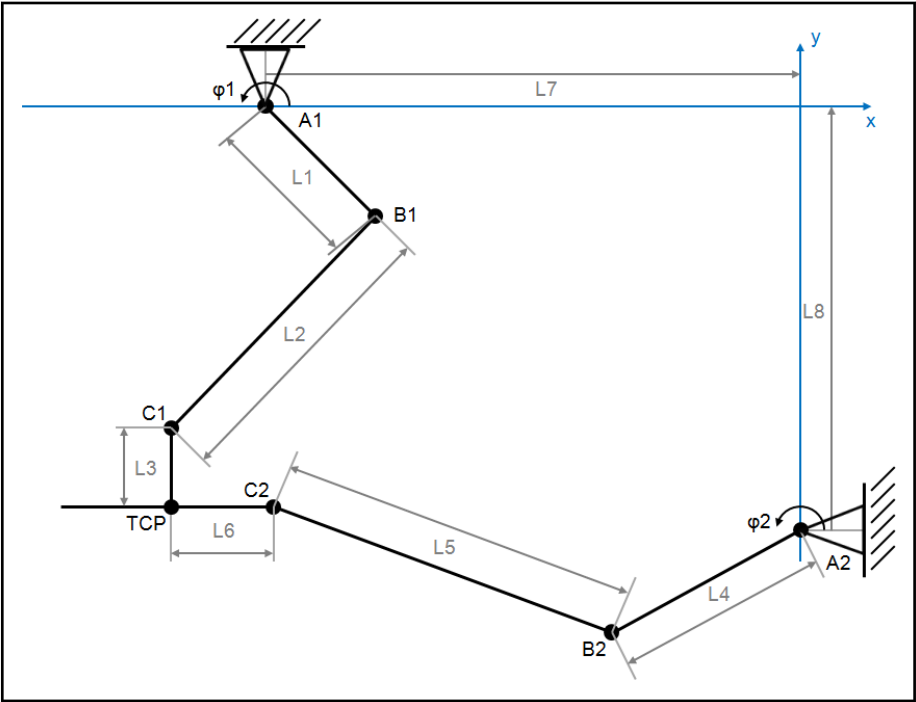


Abb. 4-14: Robotertyp 14: 2-Achs Presse mit Bemaßung

Transformationsschemen

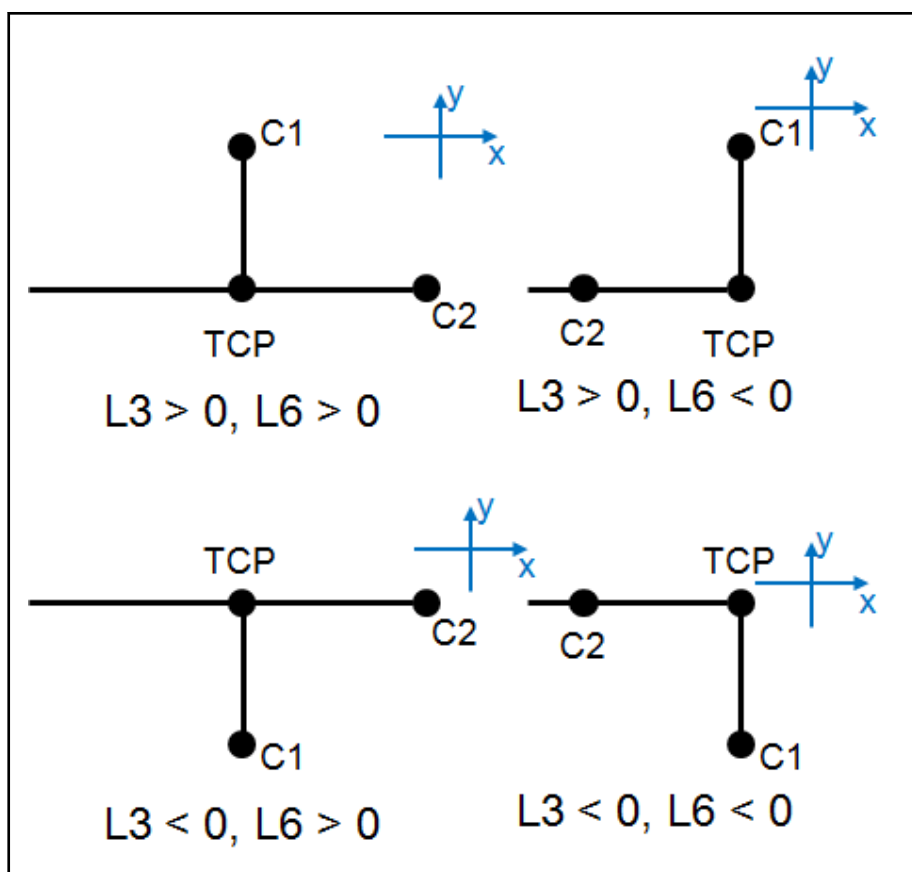


Abb. 4-15: Position von TCP abhängig von Längen $L3$ und $L6$

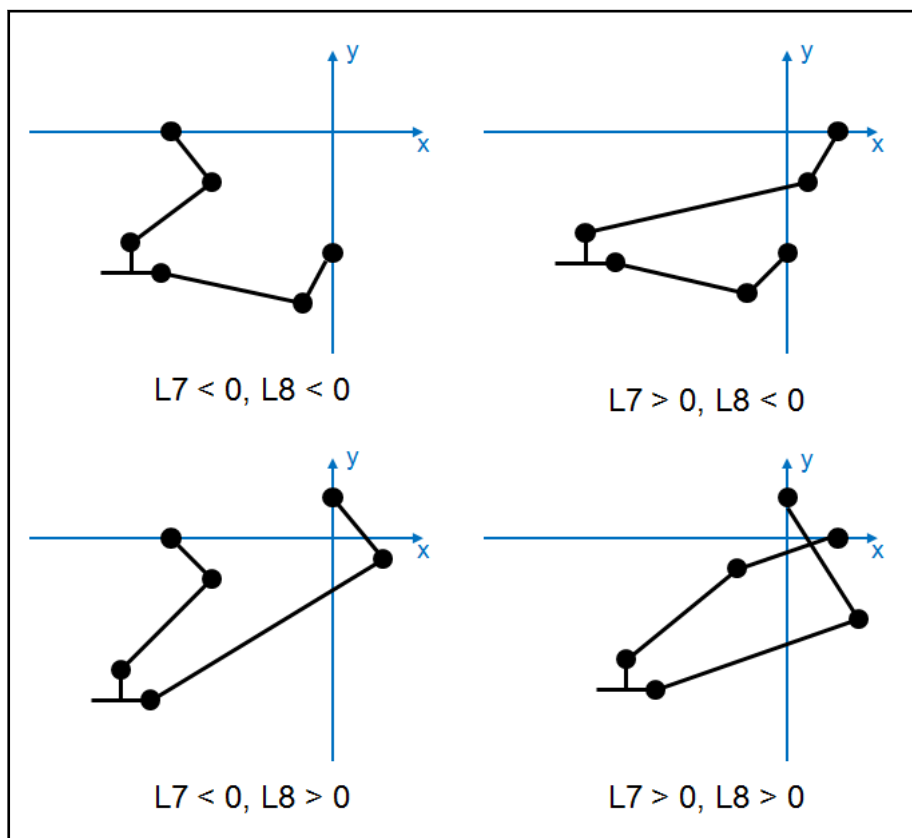


Abb. 4-16: Position der Achsen abhängig von $L7$ und $L8$

Transformationsschemen

Achs-Drehrichtungen und Nullstellungen

- Nullstellung der Achsen A1 und A2 ist die positive x-Richtung. Die Drehrichtung entspricht der mathematisch positiven Richtung
- Die Stablängen L1, L2, L4, L5 sind gemessen von den beidseitigen Aufnahme Punkten. Die Längen L3, L6, L7, L8 dürfen auch negativ sein und sind dann entsprechend als Entfernung in die negative Richtung zu verstehen
- Die Achse A1 liegt auf der x-Achse, die Achse A2 liegt auf der y-Achse

Koordinaten eines Punktes

Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung
A1:	Positionierungsachse	Rotatorisch absolut
A2:	Positionierungsachse	Rotatorisch absolut

Tab. 4-18: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des ersten Stabes des ersten Armes	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des zweiten Stabes des ersten Armes	mm bzw. inch (siehe K-0-0054)	>0
L3:	Entfernung zwischen dem Ende des ersten Armes und dem TCP in y	mm bzw. inch (siehe K-0-0054)	
L4:	Länge des ersten Stabes des zweiten Armes	mm bzw. inch (siehe K-0-0054)	>0
L5:	Länge des zweiten Stabes des zweiten Armes	mm bzw. inch (siehe K-0-0054)	>0
L6:	Entfernung zwischen dem Ende des zweiten Armes und dem TCP in x	mm bzw. inch (siehe K-0-0054)	
L7:	Entfernung zwischen A1 und dem Koordinatenursprung auf der x-Achse	mm bzw. inch (siehe K-0-0054)	
L8:	Entfernung zwischen A2 und dem Koordinatenursprung auf der y-Achse	mm bzw. inch (siehe K-0-0054)	

Tab. 4-19: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.9 Umschlingungskinematik

Zur Verwendung dieser Transformation ist die Transformationsnummer 18 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

Der Durchmesser (d) der Achsen A1 und A2 ist als Länge L1 der Transformationslängen ([K-0-0040](#)) anzugeben, dessen Maßeinheit mm bzw. Inch (siehe [K-0-0054](#)) entspricht. Der Default-Wert des Durchmessers ist 2.

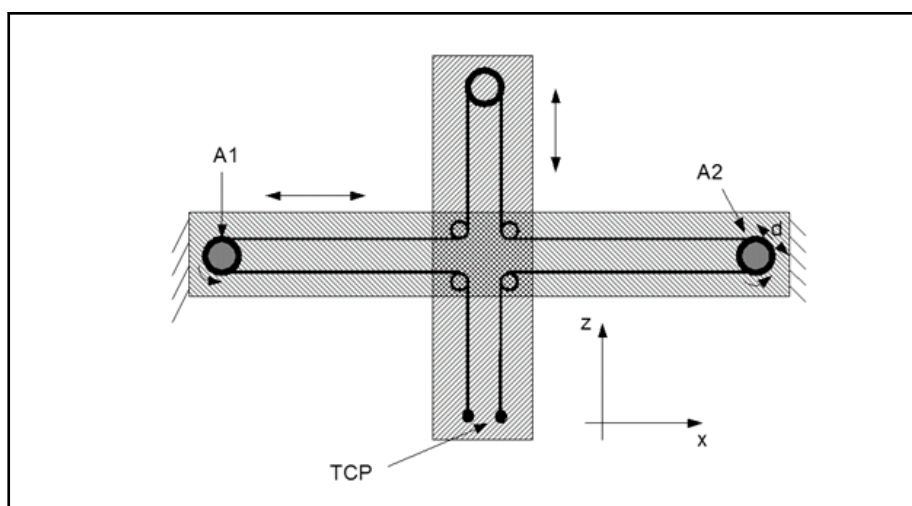


Abb. 4-17: Schematische Darstellung der Umschlingungskinematik

Achs-Drehrichtungen und Nullstellungen

- Obiges Bild zeigt die schematische Darstellung der Kinematik, bei Nullstellung aller Achsen. Für die Anwendung bedeutet dies, dass die Achsen in ihre gewünschte Nullstellung zu fahren sind, dann muss referenziert werden
- Die positiven Drehrichtungen der Achsen sind ebenfalls in den Abbildungen durch die Pfeilrichtung dargestellt
- Die Verfahrbereichsgrenzen der Achsen sind so einzustellen, dass die mechanischen Verfahrbereiche nicht überschritten werden
- Der Durchmesser (d) ist in L_1 (Achslängen) einzutragen (Default-Wert: 2)
- Das zugehörige Basis-Koordinatensystem ist durch die 2 Hauptachsen (x, z) gekennzeichnet

Koordinaten eines Punktes

Punkt(X, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	Rotatorisch absolut	
A2:	Positionierungsachse	Rotatorisch absolut	

Tab. 4-20: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit
L1:	Durchmesser (d) der Achsen A1 und A2	mm bzw. Inch (siehe K-0-0054)

Tab. 4-21: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.10 2-Achs-Parallelogramm V1

Zur Verwendung dieser Transformation ist die Transformationsnummer 19 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Transformationsschemen

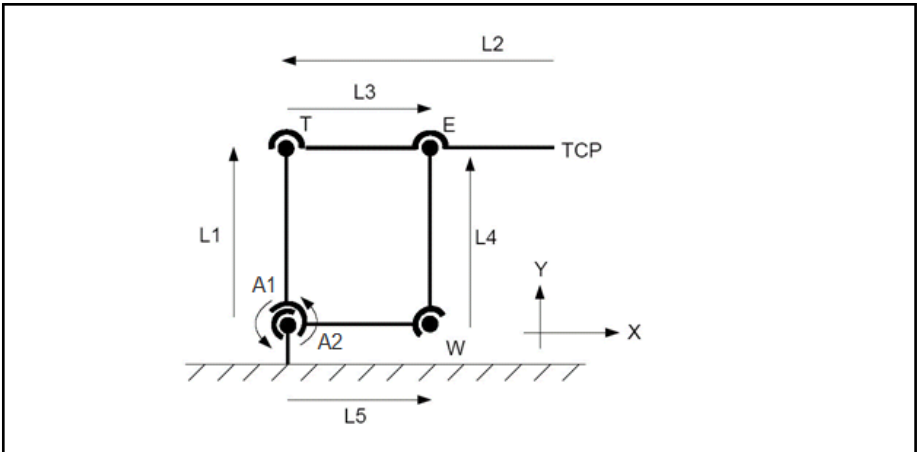


Abb. 4-18: Schematische Ansicht 2-Achs-Parallelogramm V1

Achs-Drehrichtungen und Nullstellungen

Abb. 4-18 "Schematische Ansicht 2-Achs-Parallelogramm V1" auf Seite 134 zeigt

- Die schematische Darstellung der Kinematik, bei Nullstellung aller Achsen.
- Die Armlängen L_1 , L_2 , L_3 , L_4 , L_5 und die positiven Drehrichtungen der Achsen in Pfeilrichtung.
- Der Verfahrbereich der Achsen ist mechanisch so zu begrenzen, dass der Winkel zwischen den beiden Armen mit der Längenbezeichnung L_1 und L_5 nie gleich 180° beträgt. Ansonsten gibt es je nach Mechanik, 2 unterschiedliche mathematische Lösungen, was zu ungewünschten Bewegungen führen kann.
- Die Längen L_1 , L_2 , L_3 , L_4 , L_5 sind immer positiv und ungleich Null einzugeben.
- Das zugehörige Basiskoordinaten-System ist durch die 2 Hauptachsen (x, y) gekennzeichnet.
- Der Basiskoordinaten-Nullpunkt liegt im Schnittpunkt der beiden Längenangaben L_1 und L_5 .

Koordinaten eines Punktes Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	
A2:	Positionierungsachse	rotatorisch absolut	

Tab. 4-22: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des 1. Arms	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des 2. Arms	mm bzw. inch (siehe K-0-0054)	>0

Transformationsschemen

L3:	Länge des 3. Arms	mm bzw. inch (siehe K-0-0054)	>0
L4:	Länge des 4. Arms	mm bzw. inch (siehe K-0-0054)	>0
L5:	Länge des 5. Arms	mm bzw. inch (siehe K-0-0054)	>0

Tab. 4-23: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.11 3-Achs-Delta mit optional 1 Orientierungsachse

Zur Verwendung dieser Transformation ist die Transformationsnummer 20 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

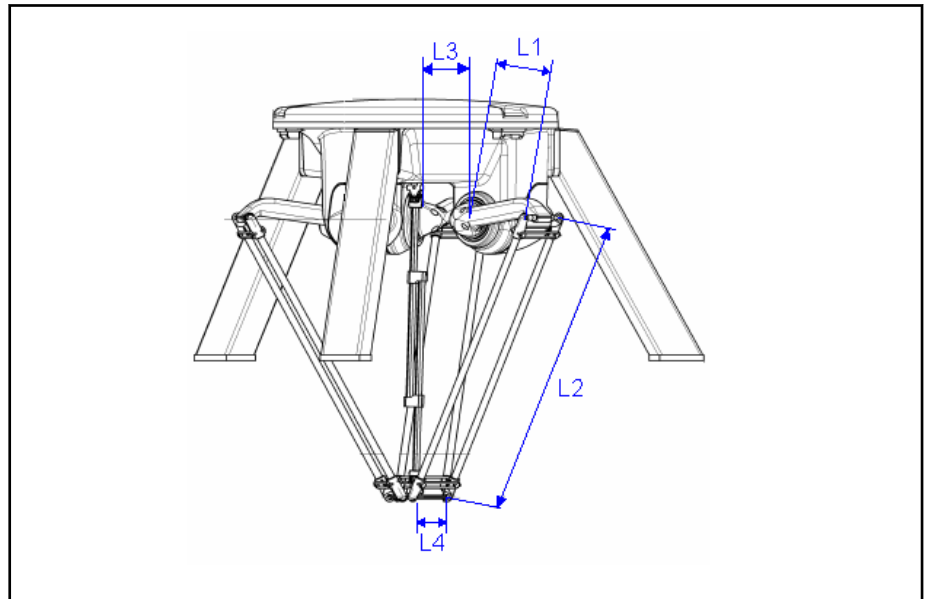


Abb. 4-19: Schematische Ansicht 3-Achs-Delta mit optional 1 Orientierungsachse

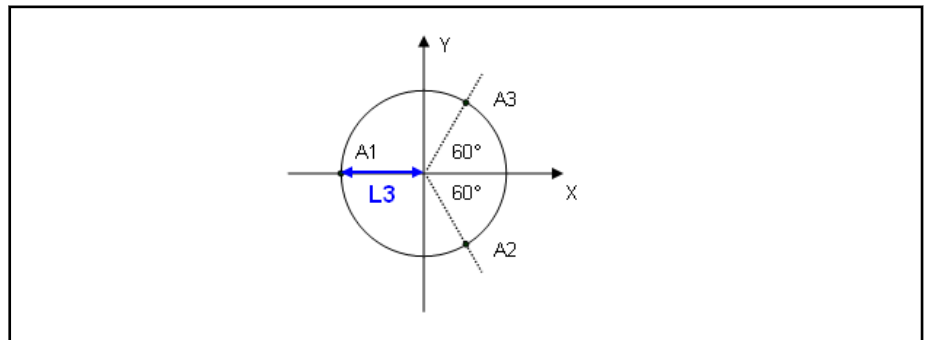


Abb. 4-20: Radius L3 der Motorhalterung und Koordinatensystem (X-Y Ansicht)
 Die drei Achsen A1 bis A3 sind um jeweils 120° versetzt.

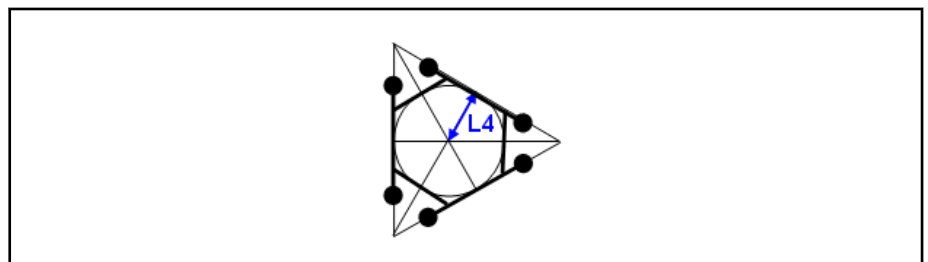
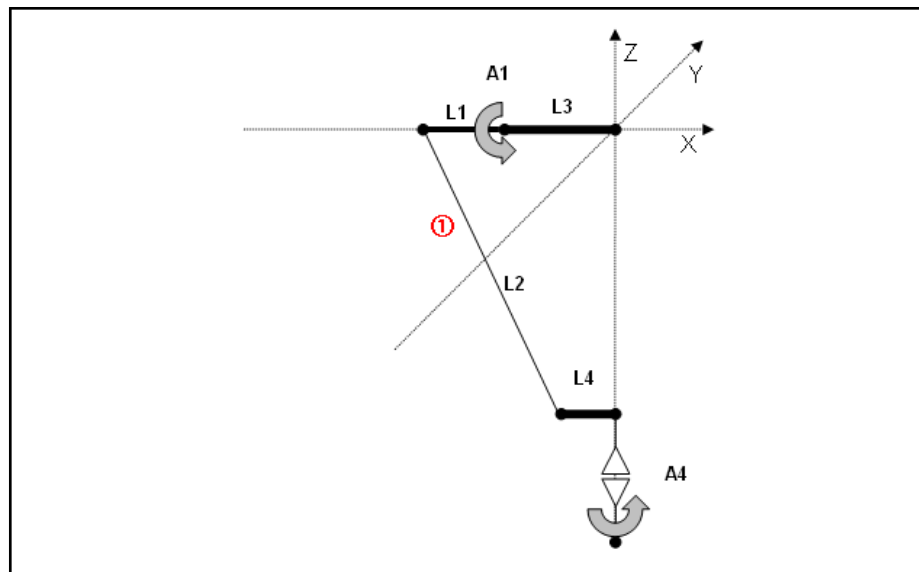


Abb. 4-21: Radius L4 der Werkzeugplatte

Transformationsschemen



① Verbindungsstange

Abb. 4-22: Darstellung der Nullstellung der ersten Achse

Nullstellung der Orientierungsachse A4 ist die positive X-Richtung.

Achs-Drehrichtungen und Nullstellungen

- Abb. 4-19 "Schematische Ansicht 3-Achs-Delta mit optional 1 Orientierungsachse" auf Seite 135 zeigt einen Delta-Roboter in Nullstellung aller Achsen, d.h. nimmt der Roboter die in den Abbildungen dargestellte Stellung ein, so müssen sämtliche Achsen in Maschinenkoordinaten (MCS) Null anzeigen.



Dies ist durch eine entsprechende Einstellung der Referenzpunkt-Istwerte zu gewährleisten.

- In der Nullstellung stehen die Achsen 1, 2 und 3 (Positionierachsen) waagrecht.
- Der Nullpunkt der 4. Achse (Orientierungsachse) ist in positive Richtung der X-Achse.
- Die Armlänge L_1 ist gemessen vom Rotationsmittelpunkt bis zum Mittelpunkt der Aufnahme der Stäbe.
- Die Stablänge L_2 ist gemessen von den beidseitigen Aufnahmepunkten.
- L_3 ist der Radius des durch die 3 Motorspindeln (bzw. Getriebe) gebildeten Umkreises, gemessen an den Rotationsmittelpunkten der Arme (siehe Abb. 4-20 "Radius L_3 der Motorhalterung und Koordinatensystem (X-Y Ansicht)" auf Seite 135).
- L_4 ist der Radius des Innenkreises des gleichseitigen Dreiecks auf der Werkzeugplatte. Das gleichseitige Dreieck wird durch die Aufnahmepunkte der Stäbe definiert. (siehe Abb. 4-21 "Radius L_4 der Werkzeugplatte" auf Seite 135).
- Die Drehrichtung der Positionierachsen A1 bis A3 ist positiv, wenn die Arme in Z-Richtung nach unten bewegt werden. Die Orientierungsachse A4 (Drehung um die Z-Achse) dreht positiv entgegen dem Uhrzeigersinn.
- Der kartesische Nullpunkt des Delta-Roboters liegt im Mittelpunkt des Umkreises, der durch die 3 Motorspindeln definiert ist (siehe auch Abb.

4-20 "Radius L3 der Motorhalterung und Koordinatensystem (X-Y Ansicht)" auf Seite 135).

Koordinaten eines Punktes Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	
A2:	Positionierungsachse	rotatorisch absolut	
A3:	Positionierungsachse	rotatorisch absolut	
A4:	Orientierungsachse	rotatorisch abs./modulo	optional

Tab. 4-24: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achszahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms (Schenkel) am Motor	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des Stabs zur Werkzeugplatte	mm bzw. inch (siehe K-0-0054)	>0
L3:	Radius des durch die 3 Rotationspunkte der Arme definierten Kreises	mm bzw. inch (siehe K-0-0054)	>0
L4:	Radius der Grundplatte	mm bzw. inch (siehe K-0-0054)	>0

Tab. 4-25: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.12 2-Achs-Parallelogramm V2

Zur Verwendung dieser Transformation ist die Transformationsnummer 21 im Parameter "K-0-0031, Transformationsschema" einzutragen.

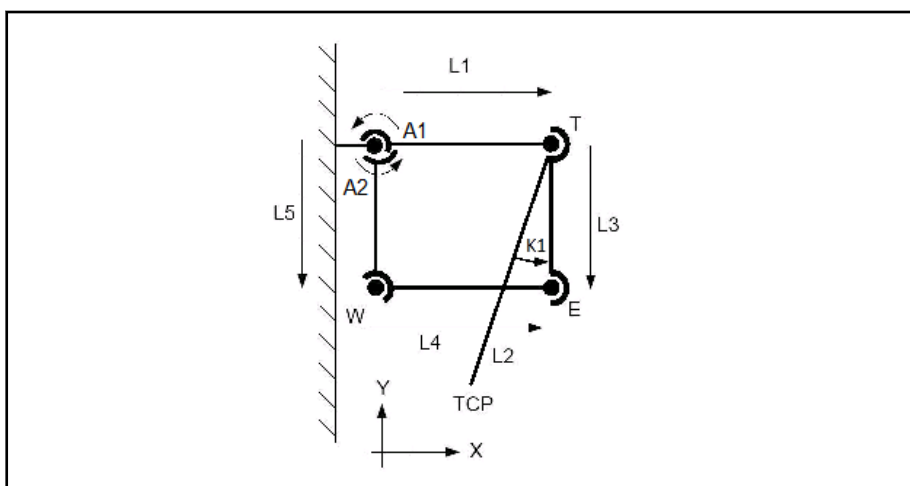


Abb. 4-23: Schematische Ansicht 2-Achs-Parallelogramm V2

Achs-Drehrichtungen und Nullstellungen

Abb. 4-23 "Schematische Ansicht 2-Achs-Parallelogramm V2" auf Seite 137 zeigt

Transformationsschemen

- Die schematische Darstellung der Kinematik, bei Nullstellung aller Achsen
- Die Armlängen L_1 , L_2 , L_3 , L_4 , L_5 und die positiven Drehrichtungen der Achsen in Pfeilrichtung
- Der Verfahrbereich der Achsen ist mechanisch so zu begrenzen, dass der Winkel zwischen den beiden Armen mit der Längenbezeichnung L_1 und L_5 nie 180° ist. In diesem Fall gibt es je nach Mechanik, zwei unterschiedliche mathematische Lösungen, was zu ungewünschten Bewegungen führen kann
- Die Längen L_1 , L_2 , L_3 , L_4 , L_5 , K_1 sind immer positiv und ungleich Null einzugeben
- Das zugehörige Basiskoordinaten-System ist durch die 2 Hauptachsen (x, y) gekennzeichnet
- Der Basiskoordinaten-Nullpunkt liegt im Schnittpunkt der beiden Längenangaben L_1 und L_5

Koordinaten eines Punktes Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	
A2:	Positionierungsachse	rotatorisch absolut	

Tab. 4-26: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des 1. Arms	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des 2. Arms	mm bzw. inch (siehe K-0-0054)	>0
L3:	Länge des 3. Arms	mm bzw. inch (siehe K-0-0054)	>0
L4:	Länge des 4. Arms	mm bzw. inch (siehe K-0-0054)	>0
L5:	Länge des 5. Arms	mm bzw. inch (siehe K-0-0054)	>0

Tab. 4-27: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Faktor	Achskonfiguration	Maßeinheit	Achskopplung
K1:	Winkel zwischen T, E und T, TCP	Grad	>0

Tab. 4-28: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.13 3-Achs-Delta Linear mit optional 1 Orientierungsachse

Zur Verwendung dieser Transformation ist die Transformationsnummer 22 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

Transformationsschemen

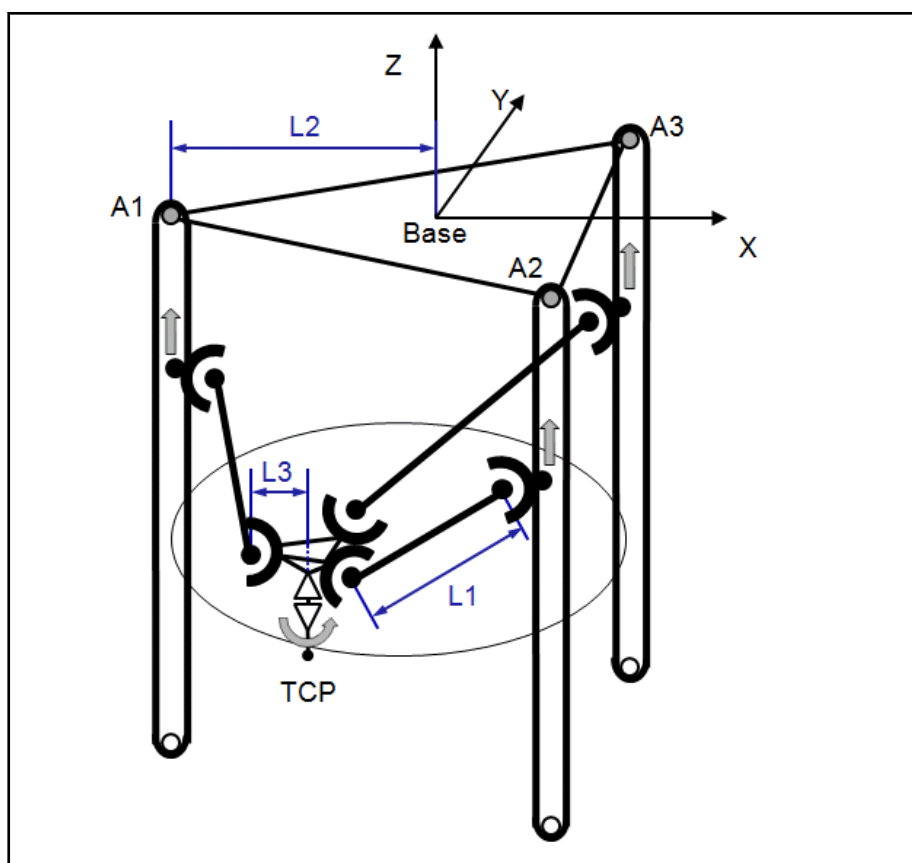


Abb. 4-24: Schematische Ansicht 3-Achs-Delta Linear mit optional 1 Orientierungsachse

Die drei Achsen A1 bis A3 sind um jeweils 120° versetzt.

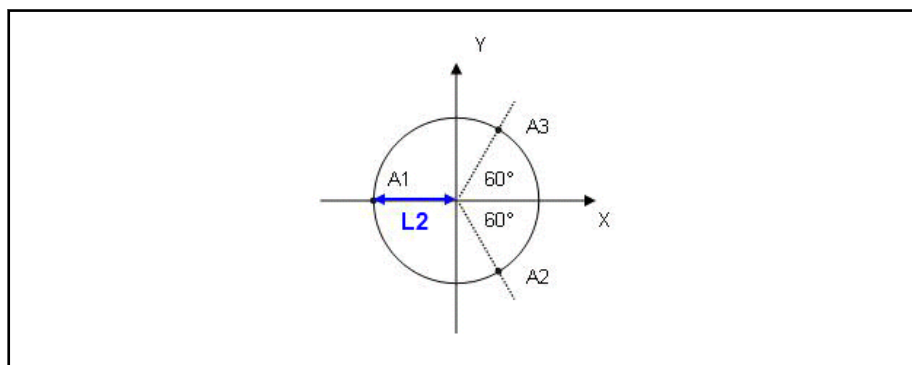


Abb. 4-25: Radius L3 der Motorhalterung und Koordinatensystem (X-Y Ansicht)

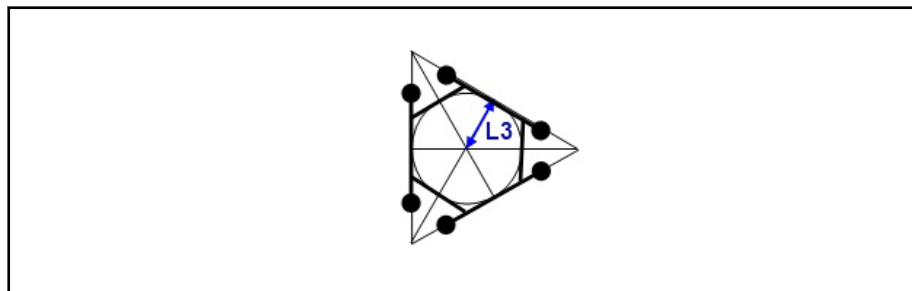


Abb. 4-26: Radius L4 der Werkzeugplatte

Transformationsschemen

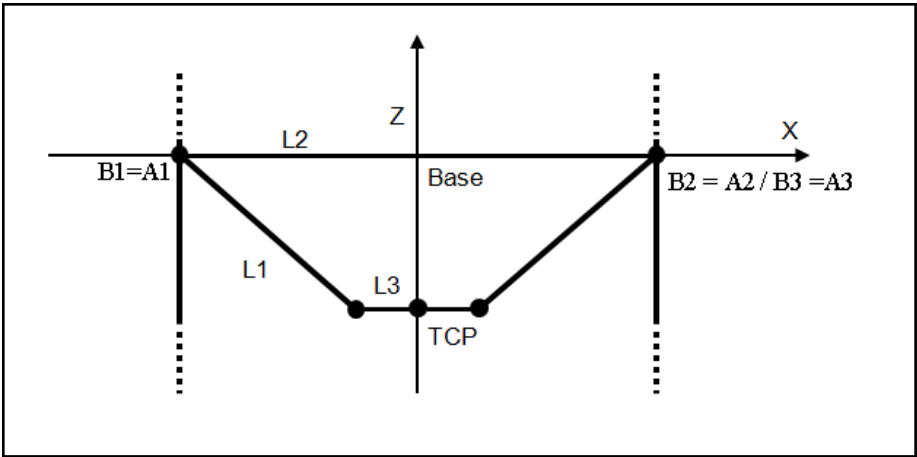


Abb. 4-27: Nullposition der Achsen

Nullstellung der Orientierungsachse A4 ist die positive X-Richtung.

Achs-Drehrichtungen und Nullstellungen

- [Abb. 4-27 "Nullposition der Achsen" auf Seite 140](#) zeigt einen Delta-Linear-Roboter in Nullstellung aller Achsen, d. h. nimmt der Roboter die in den Abbildungen dargestellte Stellung ein, so müssen sämtliche Achsen in Maschinenkoordinaten (MCS) Null anzeigen.



Dies ist durch eine entsprechende Einstellung der Referenzpunkt-Istwerte zu gewährleisten.

- Der Nullpunkt der vierten Achse (Orientierungsachse) ist in positive Richtung der X-Achse
- Die Stablänge L_1 ist gemessen von den beidseitigen Aufnahmepunkten
- L_2 ist der Radius des durch die drei Führungsschienen gebildeten Umkreises, gemessen an den Rotationsmittelpunkten der Arme
- L_3 ist der Radius des Innenkreises des gleichseitigen Dreiecks auf der Werkzeugplatte. Das gleichseitige Dreieck wird durch die Aufnahmepunkte der Stäbe definiert. Der Radius muss größer oder gleich L_2 sein (siehe [Abb. 4-26 "Radius L4 der Werkzeugplatte" auf Seite 139](#))
- Die Ausfahrrichtung der Positionierachsen A1 bis A3 ist positiv, wenn die Arme in Z-Richtung nach oben bewegt werden. Die Orientierungsachse A4 (Drehung um die Z-Achse) dreht positiv entgegen dem Uhrzeigersinn
- Der kartesische Nullpunkt des Delta-Linear-Roboters liegt im Mittelpunkt des Umkreises, der durch die 3 Führungsschienen in Nullposition definiert ist (siehe auch [Abb. 4-25 "Radius L3 der Motorhalterung und Koordinatensystem \(X-Y Ansicht\)" auf Seite 139](#))

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	Translatorisch absolut	
A2:	Positionierungsachse	Translatorisch absolut	

Transformationsschemen

A3:	Positionierungsachse	Translatorisch absolut	
A4:	Orientierungsachse	Rotatorisch absolut/ modulo	Optional

Tab. 4-29: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Stabs zur Werkzeugplatte	mm bzw. inch (siehe K-0-0054)	>0
L2:	Radius des durch die 3 Translationspunkte der Arme definierten Kreises	mm bzw. inch (siehe K-0-0054)	>0
L3:	Radius der Grundplatte	mm bzw. inch (siehe K-0-0054)	>0

Tab. 4-30: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.14 5-Achs-Delta

Zur Verwendung dieser Transformation ist die Transformationsnummer 23 im Parameter "K-0-0031, Transformationsschema" einzutragen.

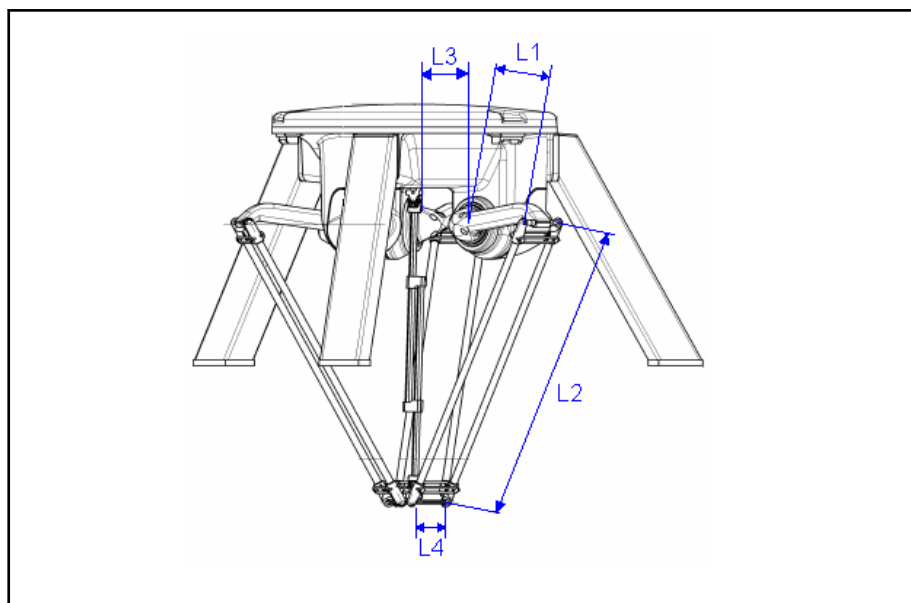


Abb. 4-28: Schematische Ansicht 5-Achs-Delta ohne TCP-Erweiterung

Transformationsschemen

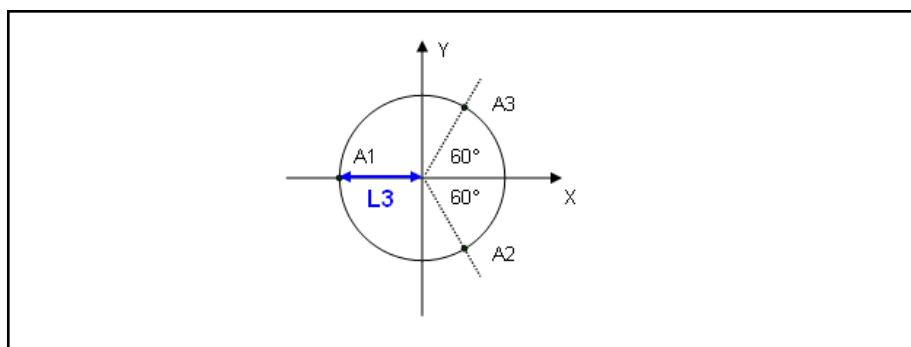


Abb. 4-29: Radius $L3$ der Gelenkhalterung und Koordinatensystem (X-Y Ansicht)

Die drei Gelenke A1 bis A3 sind um jeweils 120° versetzt.

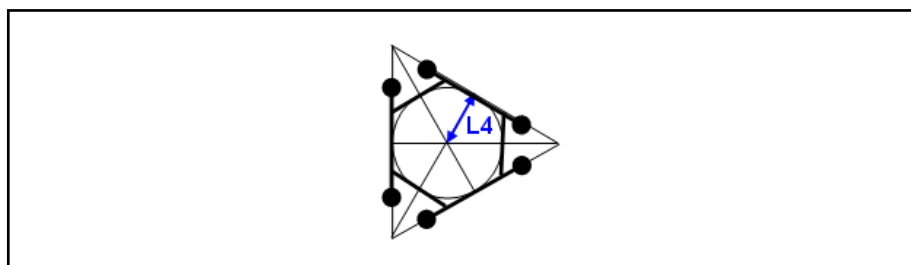
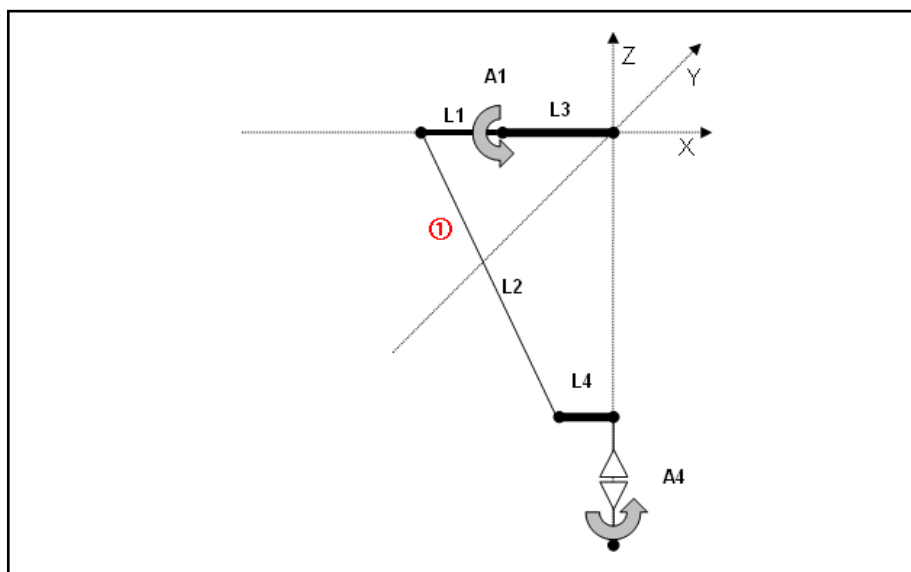


Abb. 4-30: Radius $L4$ der Werkzeugplatte



① Verbindungsstange

Abb. 4-31: Darstellung der Nullstellung der ersten Achse

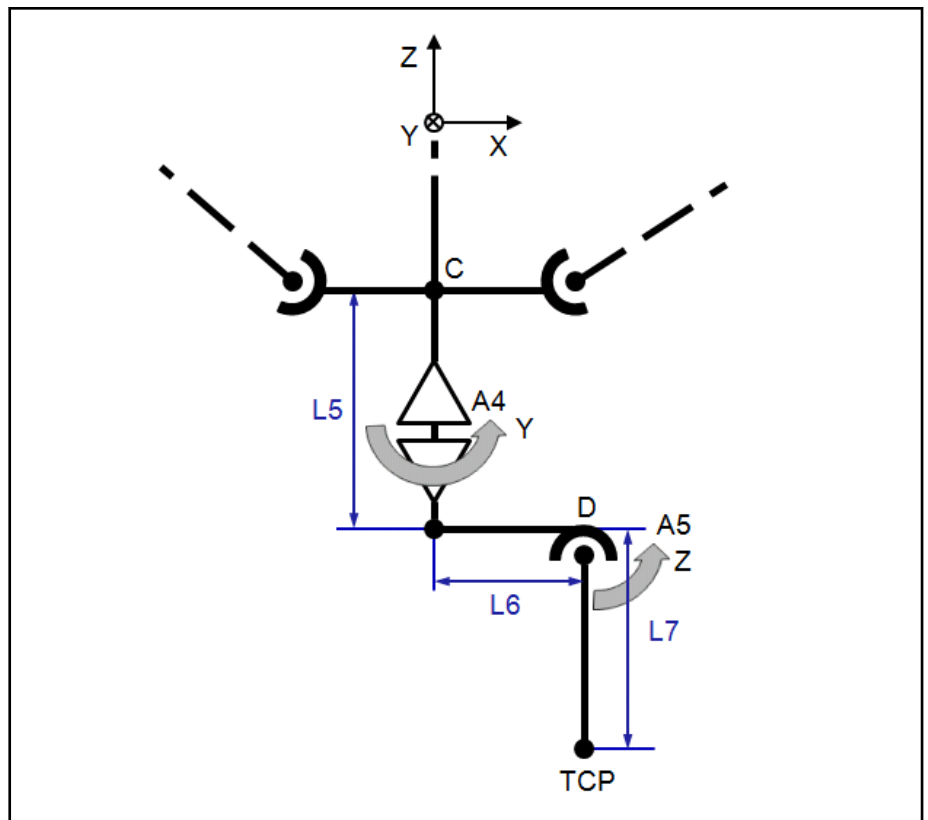


Abb. 4-32: Anordnung der Erweiterung und Nullstellung fünfte Achse

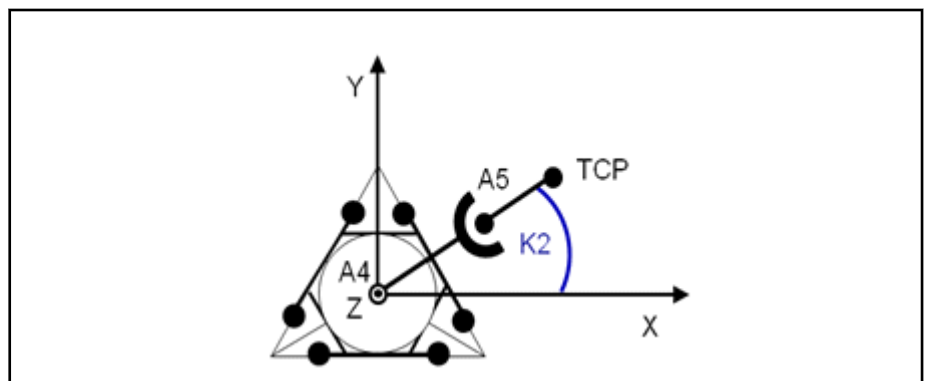


Abb. 4-33: Nullstellung vierte Achse

Achs-Drehrichtungen und Nullstellungen

- Abb. 4-28 "Schematische Ansicht 5-Achs-Delta ohne TCP-Erweiterung" auf Seite 141, Abb. 4-32 "Anordnung der Erweiterung und Nullstellung fünfte Achse" auf Seite 143 und Abb. 4-33 "Nullstellung vierte Achse" auf Seite 143 zeigen einen Delta-Roboter in Nullstellung aller Achsen, d. h. nimmt der Roboter die in den Abbildungen dargestellte Stellung ein, so müssen sämtliche Achsen in Maschinenkoordinaten (MCS) Null anzeigen



Dies ist durch eine entsprechende Einstellung der Referenzpunkt-Istwerte zu gewährleisten.

- In der Nullstellung stehen die Achsen 1, 2 und 3 (Positionierachsen) waagrecht. Die fünfte Achse (Orientierungsachse) zeigt in Nullstellung senkrecht in negative Z-Richtung

Transformationsschemen

- Der Nullpunkt der vierten Achse (Orientierungsachse) ist durch den Faktor K2 in Grad mit mathematisch positiver Drehrichtung bezüglich der XY-Ebene festgelegt
- Die Armlänge L_1 ist gemessen vom Rotationsmittelpunkt bis zum Mittelpunkt der Aufnahme der Stäbe
- Die Stablänge L_2 ist gemessen von den beidseitigen Aufnahmepunkten
- L_3 ist der Radius des durch die 3 Motorspindeln (bzw. Getriebe) gebildeten Umkreises, gemessen an den Rotationsmittelpunkten der Arme (siehe [Abb. 4-29 "Radius L3 der Gelenkhalterung und Koordinatensystem \(X-Y Ansicht\)" auf Seite 142](#))
- L_4 ist der Radius des Innenkreises des gleichseitigen Dreiecks auf der Werkzeugplatte. Das gleichseitige Dreieck wird durch die Aufnahmepunkte der Stäbe definiert (siehe [Abb. 4-30 "Radius L4 der Werkzeugplatte" auf Seite 142](#))
- L_5 ist die senkrechte Länge in Z-Richtung von dem Mittelpunkt des Innenkreises des gleichseitigen Dreiecks auf der Werkzeugplatte zum unteren Aufnahmepunkt der Achse 5. L_6 ist äquivalent die horizontale Länge vom Mittelpunkt zum Aufnahmepunkt [Abb. 4-32 "Anordnung der Erweiterung und Nullstellung fünfte Achse" auf Seite 143](#)
- L_7 ist die Armlänge vom Rotationsmittelpunkt der fünften Achse zum Endpunkt (siehe [Abb. 4-32 "Anordnung der Erweiterung und Nullstellung fünfte Achse" auf Seite 143](#))
- Die Drehrichtung der Positionierachsen A1 bis A3 ist positiv, wenn die Arme in Z-Richtung nach unten bewegt werden. Die Hilfsachse A4 (Drehung um die Z-Achse) dreht positiv entgegen dem Uhrzeigersinn. Die Drehrichtung der Hilfsachse A5 ist positiv, wenn der Arm in Z-Richtung nach oben bewegt wird
- Der kartesische Nullpunkt des Delta-Roboters liegt im Mittelpunkt des Umkreises, der durch die 3 Motorspindeln definiert ist (siehe auch [Abb. 4-29 "Radius L3 der Gelenkhalterung und Koordinatensystem \(X-Y Ansicht\)" auf Seite 142](#))
- Die Transformation berücksichtigt eine mechanische Kopplung der Achsen 4 und 5.
Die Übersetzung der Kopplung wird in K2 angegeben. Dabei ist abtreibendes Element Achse 5 und antreibendes Element Achse 4. K2 ist definiert als Abtrieb durch Antrieb

Koordinaten eines Punktes Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	Rotatorisch absolut	+/- 90 Grad
A2:	Positionierungsachse	Rotatorisch absolut	+/- 90 Grad
A3:	Positionierungsachse	Rotatorisch absolut	+/- 90 Grad
A4:	Orientierungsachse	Rotatorisch absolut	
A5:	Orientierungsachse	Rotatorisch absolut	

Tab. 4-31: K-0-0008 Achskonfigurationsliste, Achszuordnung

Transformationsschemen

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms (Schenkel) am Motor	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des Stabs zur Werkzeugplatte	mm bzw. inch (siehe K-0-0054)	>0
L3:	Radius des durch die 3 Rotationspunkte der Arme definierten Kreises	mm bzw. inch (siehe K-0-0054)	>0
L4:	Radius der Grundplatte	mm bzw. inch (siehe K-0-0054)	>0
L5:	Vertikale Länge der Mitte der Grundplatte zum unteren Aufnahmepunkt	mm bzw. inch (siehe K-0-0054)	>0
L6:	Horizontale Länge der Mitte der Grundplatte zum unteren Aufnahmepunkt	mm bzw. inch (siehe K-0-0054)	>0
L7:	Länge des Arms der fünften Achse	mm bzw. inch (siehe K-0-0054)	>0

Tab. 4-32: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Länge	Achskonfiguration	Maßeinheit	Bemerkung
K1:	Mechanische Kopplung der vierten und fünften Achse	Faktor	>0
K2:	Verschiebung der Nullstellung der vierten Achse	Grad	optional

Tab. 4-33: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.15 Adabot

Zur Verwendung dieser Transformation ist die Transformationsnummer 30 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

Transformationsschemen

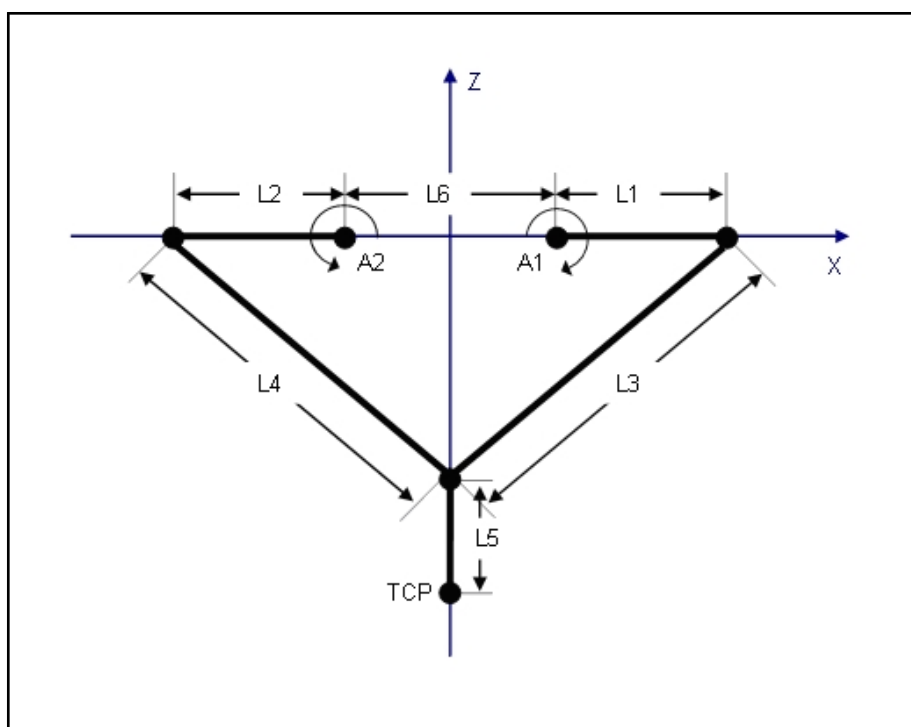


Abb. 4-34: Adabot (X-Z Ansicht, Achsnulldstellung)

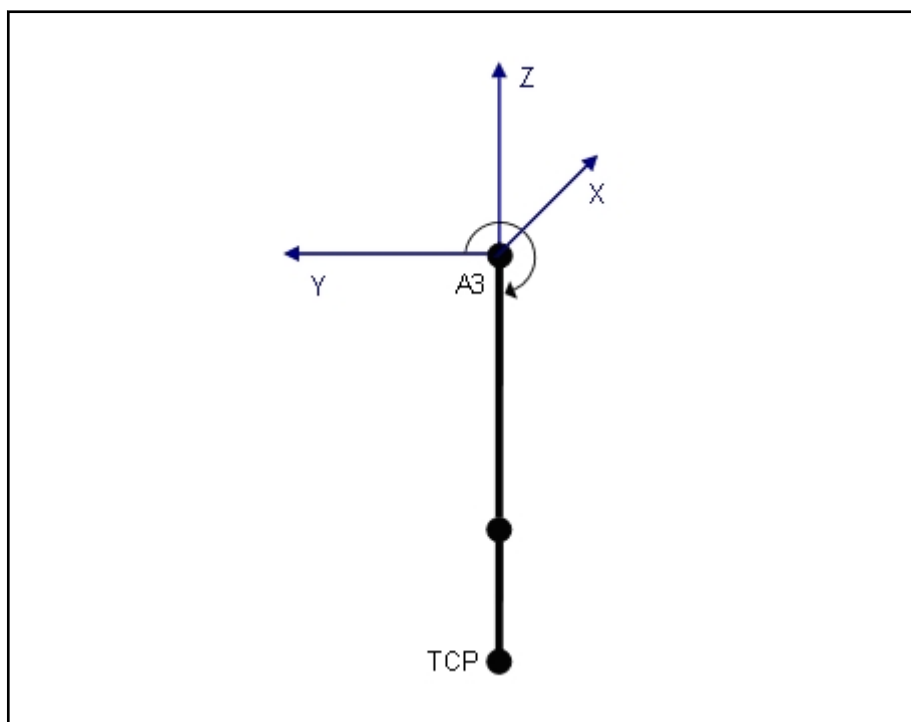


Abb. 4-35: Adabot (Y-Z Ansicht, Achsnulldstellung)

Transformationsschemen

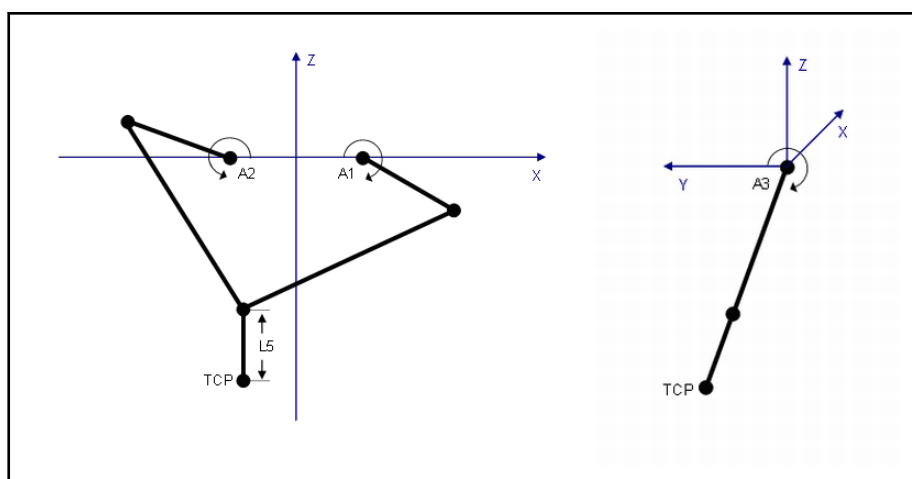


Abb. 4-36: Adabot (Verhalten des Stabs L5)

Achs-Drehrichtungen und Nullstellungen

Abb. 4-34 "Adabot (X-Z Ansicht, Achsnullstellung)" auf Seite 146 und Abb. 4-35 "Adabot (Y-Z Ansicht, Achsnullstellung)" auf Seite 146 zeigen die schematische Darstellung der Adabot-Kinematik, bei Nullstellung aller Achsen.

- Die Längen L_1 , L_2 , L_3 , L_4 , L_5 , L_6 sind ebenfalls in den Abbildungen dargestellt.
- Die positiven Drehrichtungen der Achsen A1, A2, A3 sind ebenfalls in den Abbildungen dargestellt.
- Die Längen L_1 , L_2 , L_3 , L_4 , L_6 sind immer positiv und ungleich Null einzugeben. L_5 ist immer positiv oder Null einzugeben.

Der Stab zum TCP (L_5) wird durch eine Mechanik (nicht in Abb. dargestellt) immer senkrecht zur X-Achse gehalten. Der Stab zum TCP (L_5) liegt immer in der Ebene, die durch die Stäbe L_1 , L_2 , L_3 , L_4 aufgespannt wird (siehe Abb. 4-36 "Adabot (Verhalten des Stabs L5)" auf Seite 147).

Der kartesische Nullpunkt der Adabot-Kinematik liegt in der Mitte der beiden Armdrehpunkte A1 und A2 (siehe Abb. 4-34 "Adabot (X-Z Ansicht, Achsnullstellung)" auf Seite 146).

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	+/- 90 Grad
A2:	Positionierungsachse	rotatorisch absolut	+/- 90 Grad
A3:	Positionierungsachse	rotatorisch absolut	+/- 90 Grad
A4:	Orientierungsachse	rotatorisch	optional

Tab. 4-34: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden

Transformationsschemen

- Bandachsen



Ein Wechsel der Armigkeit der Arme L1,L3 und L2,L4 (z. B. Achse A1 und Achse A2 stehen in [Abb. 4-34 "Adabot \(X-Z Ansicht, Achsnullstellung\)"](#) auf Seite 146 beide bei -86 Grad) wird von der Kinematik nicht zugelassen und als Fehler interpretiert.

TCP Koordinaten außerhalb der Reichweite der eingegebenen Armlängen in MCS, werden von der Kinematik als Fehler interpretiert.

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms (Schenkel) am Motor (rechts)	mm bzw. in (siehe K-0-0054)	>0
L2:	Länge des Arms (Schenkel) am Motor (links)	mm bzw. in (siehe K-0-0054)	>0
L3:	Länge des Stabs an Arm L1	mm bzw. in (siehe K-0-0054)	>0
L4:	Länge des Stabs an Arm L2	mm bzw. in (siehe K-0-0054)	>0
L5:	Stab zu TCP	mm bzw. in (siehe K-0-0054)	>=0
L6:	Abstand der beiden Rotationspunkte der Arme	mm bzw. in (siehe K-0-0054)	>0

Tab. 4-35: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.16 2-Achs-Delta mit asymmetrischem TCP

Zur Verwendung dieser Transformation ist die Transformationsnummer 32 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

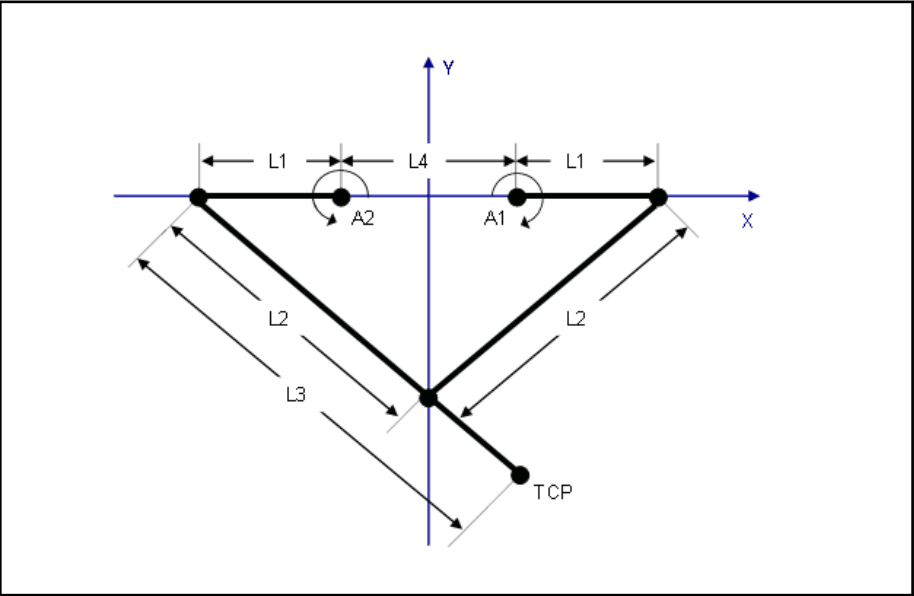


Abb. 4-37: 2-Achs-Delta mit asymmetrischem TCP (X-Y Ansicht, Achsnullstellung)

Koordinaten eines Punktes

Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Transformationsschemen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	+/- 90 Grad
A2:	Positionierungsachse	rotatorisch absolut	+/- 90 Grad
A3:	Orientierungsachse	rotatorisch abs. / modulo	optional

Tab. 4-36: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms (Schenkel) am Motor	mm bzw. in (siehe K-0-0054)	>0
L2:	Länge des kurzen Stabs	mm bzw. in (siehe K-0-0054)	>0
L3:	Länge des langen Stabs zum TCP	mm bzw. in (siehe K-0-0054)	>L2
L4:	Abstand der beiden Rotationspunkte der Arme	mm bzw. in (siehe K-0-0054)	>0

Tab. 4-37: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Achs-Drehrichtungen und Nullstellungen

Abb. 4-37 "2-Achs-Delta mit asymmetrischem TCP (X-Y Ansicht, Achsnullstellung)" auf Seite 148 zeigt die schematische Darstellung der 2-Achs-Delta-Kinematik mit asymmetrischem TCP, bei Nullstellung aller Achsen.

- Die Längen L_1 , L_2 , L_3 , L_4 sind ebenfalls in der Abbildung dargestellt
- Die positiven Drehrichtungen der Achsen A1, A2 sind ebenfalls in der Abbildung dargestellt
- Die Längen L_1 , L_2 , L_3 , L_4 sind immer positiv und ungleich Null einzugeben. Außerdem muss L_3 immer einen größeren Wert enthalten als L_2
- Die Länge $(L_4 / 2 + L_1)$ muss kleiner sein als die Länge L_2
- Der kartesische Nullpunkt liegt in der Mitte der beiden Armdrehpunkte A1 und A2

4.2.17 3-Achs-Twisting Crane

Zur Verwendung dieser Transformation ist die Transformationsnummer 38 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Transformationsschemen

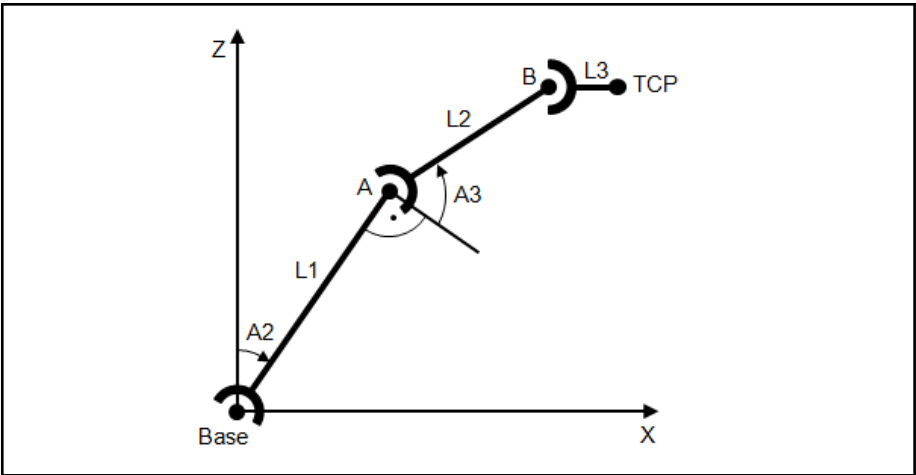


Abb. 4-38: Schematische Ansicht von der Seite 3-Achs-Twisting Crane

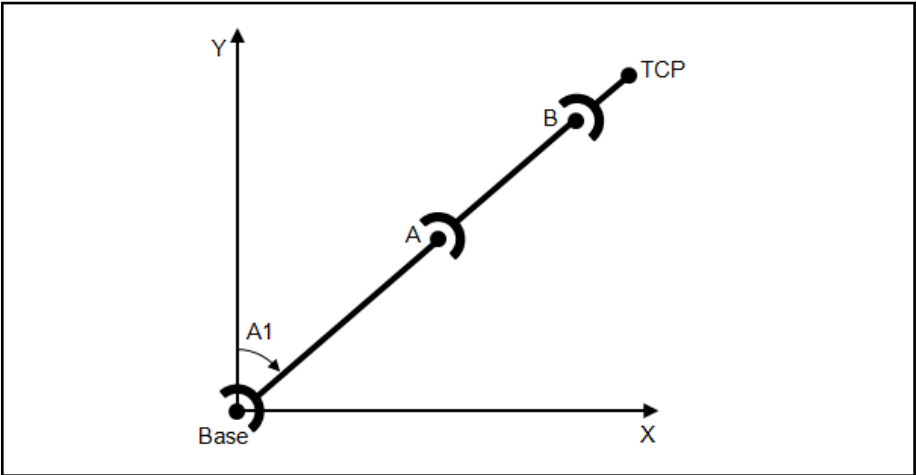


Abb. 4-39: Schematische Ansicht von oben 3-Achs-Twisting Crane

Achs-Drehrichtungen und Nullstellungen

Die schematischen Ansichten zeigen den 3-Achs-Twisting Crane von der Seite und von oben. In den schematischen Ansichten ist folgendes dargestellt:

- Die Drehrichtung und Nullpunkt der Achsen A1, A2 und A3
- Die Armlänge L_1 als Länge zwischen Punkt Base und A
- Die Armlänge L_2 als Länge zwischen Punkte A und B
- Die Armlänge L_3 als Länge zwischen Punkt B und TCP
- Der dritte Arm (B,TCP) wird mechanisch parallel zur X-Y-Ebene gehalten.
- Das zugehörige Basiskoordinaten-System ist durch die 3 Hauptachsen (x, y, z) gekennzeichnet
- Der Basiskoordinaten-Nullpunkt liegt am Montagepunkt von Achse A1 und A2

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	rotatorisch absolut	

Transformationsschemen

A2:	Positionierungsachse	rotatorisch absolut	$-180^\circ < x < 180^\circ$
A3:	Positionierungsachse	rotatorisch absolut	$-90^\circ < x < 90^\circ$

Tab. 4-38: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms Base, A	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des Arms A, B	mm bzw. inch (siehe K-0-0054)	>0
L3:	Länge des Arms B, C	mm bzw. inch (siehe K-0-0054)	>=0

Tab. 4-39: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.18 3-Achs-Flex Stamper

Zur Verwendung dieser Transformation ist die Transformationsnummer 39 im Parameter "K-0-0031, Transformationsschema" einzutragen.

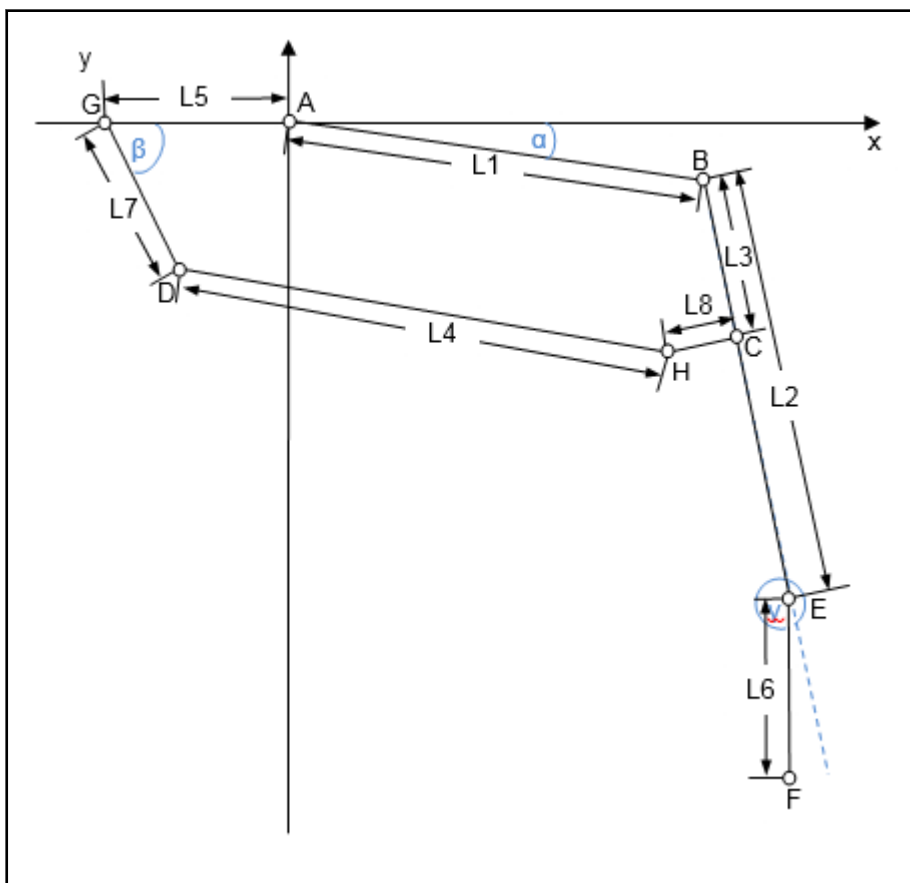


Abb. 4-40: Schematische Ansicht 3-Achs-Flex Stamper

Achs-Drehrichtungen und Nullstellungen

Abb. 4-40 "Schematische Ansicht 3-Achs-Flex Stamper" auf Seite 151 zeigt

- Die schematische Darstellung der Kinematik, bei Stellung Achse 1 = α , Achse 2 = β , Achse 3 = γ
- Nullstellung der Achsen A1 und A2 ist die positive x-Richtung

Transformationsschemen

- Nullstellung der Achse A3 ist die Verlängerung von L2
- Drehsinn der drei Achsen ist im entgegengesetzten Uhrzeigersinn
- Die Armlängen $L_1, L_2, L_3, L_4, L_5, L_6, L_7, L_8$
- Das zugehörige Basiskoordinaten-System ist durch die 2 Hauptachsen (x, y) gekennzeichnet
- Der Basiskoordinaten-Nullpunkt liegt am Montagepunkt von Achse 1

Koordinaten eines Punktes Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	Rotatorisch absolut	$-270^\circ \leq x < 90^\circ$
A2:	Positionierungsachse	Rotatorisch absolut	$-270^\circ \leq x < 90^\circ$
A3:	Orientierungsachse	Rotatorisch absolut	$0^\circ < x < 360^\circ$

Tab. 4-40: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms A, B	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des Arms B, E	mm bzw. inch (siehe K-0-0054)	>0
L3:	Länge des Arms B, C	mm bzw. inch (siehe K-0-0054)	>0 <L2
L4:	Länge des Arms D, H	mm bzw. inch (siehe K-0-0054)	>0
L5:	Positionsunterschied von Achse 1 und Achse 2 (A, G)	mm bzw. inch (siehe K-0-0054)	>=0
L6:	Länge des Arms E, F	mm bzw. inch (siehe K-0-0054)	>=0
L7:	Länge des Arms G, D	mm bzw. inch (siehe K-0-0054)	>0
L8:	Länge des Arms C, H	mm bzw. inch (siehe K-0-0054)	>=0

Tab. 4-41: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Faktor	Achskonfiguration
K1:	0: Winkel von Achse 3 wird automatisch so gewählt, dass Arm E, F parallel zur y-Achse ist. Ist L6 = 0, so wird A3 als optionale Orientierungsachse verwendet 1: Winkel von Achse 3 kann vom Anwender als Orientierung um die z-Achse angegeben werden

Tab. 4-42: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.19 3-Achs-Delta mit Linearachsen mit optional 1 Orientierungsachse

Zur Verwendung dieser Transformation ist die Transformationsnummer 40 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Transformationsschemen

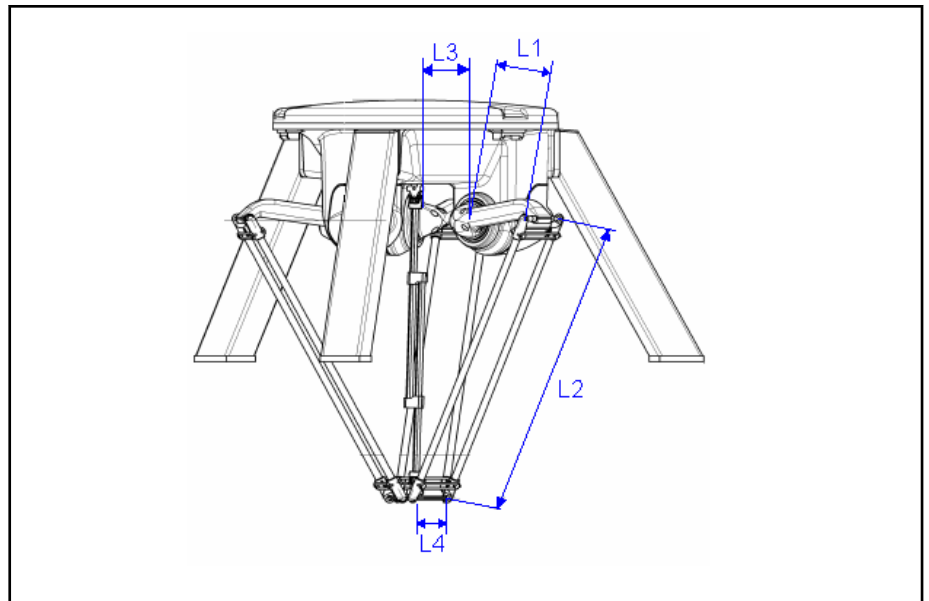


Abb. 4-41: Schematische Ansicht 3-Achs-Delta mit optional 1 Orientierungsachse

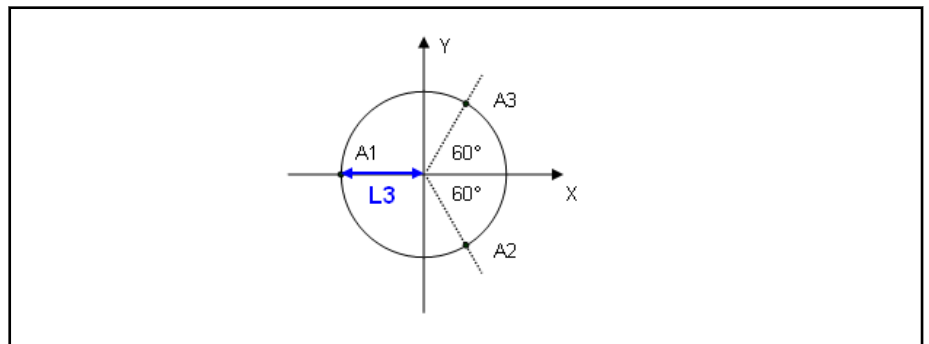


Abb. 4-42: Radius L3 der Gelenkhalterung und Koordinatensystem (X-Y Ansicht)

Die drei Gelenke A1 bis A3 sind um jeweils 120° versetzt.

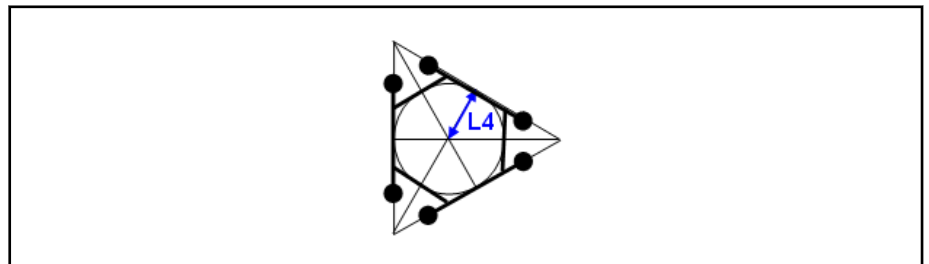
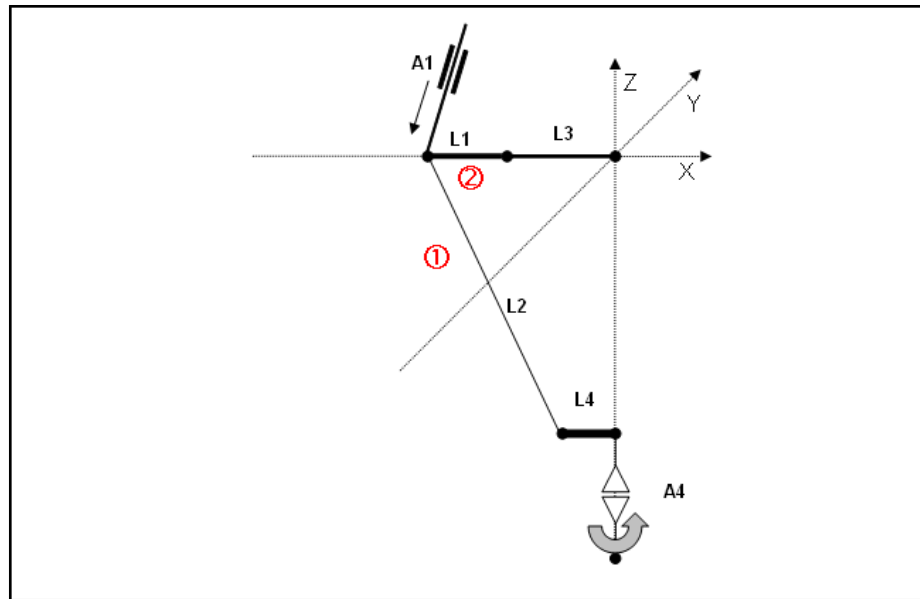


Abb. 4-43: Radius L4 der Werkzeugplatte

Transformationsschemen



① Verbindungsstange

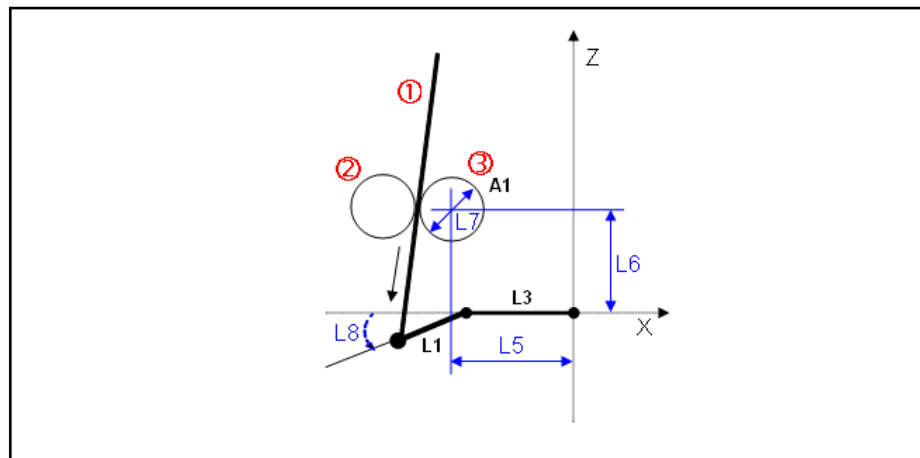
② Arm

Abb. 4-44: Darstellung der Nullstellung

Nullstellung der Orientierungsachse A4 sollte die positive X-Richtung sein.

- [K-0-0040](#) (Transformation, Längen der Mechanik), Transformationslängen, Achskonfiguration,

Achslängen Einträgen L1 bis L8:



① Zahnstange

② Andruckrolle

③ Antriebsrolle

Abb. 4-45: Weitere Einstellungen

Achs-Drehrichtungen und Nullstellungen

Abb. 4-41 "Schematische Ansicht 3-Achs-Delta mit optional 1 Orientierungsachse" auf Seite 153 zeigt einen Delta-Roboter in Nullstellung aller Achsen, d.h. nimmt der Roboter die in den Abbildungen dargestellte Stellung ein, so müssen sämtliche Achsen in Maschinenkoordinaten (MCS) Null anzeigen.



Dies ist durch eine entsprechende Einstellung der Referenzpunkt-Istwerte zu gewährleisten.

- In der Nullstellung stehen die Arme waagrecht

Transformationsschemen

- Der Nullpunkt der 4. Achse (Orientierungsachse) ist in positive Richtung der X-Achse
- Die Armlänge L_1 ist gemessen vom Rotationsmittelpunkt bis zum Mittelpunkt der Aufnahme der Stäbe
- Die Stablänge L_2 ist gemessen von den beidseitigen Aufnahmepunkten
- L_3 ist der Radius des durch die 3 Motorspindeln (bzw. Getriebe) gebildeten Umkreises, gemessen an den Rotationsmittelpunkten der Arme (siehe [Abb. 4-42 "Radius L3 der Gelenkhalterung und Koordinatensystem \(X-Y Ansicht\)" auf Seite 153](#))
- L_4 ist der Radius des Innenkreises des gleichseitigen Dreiecks auf der Werkzeugplatte. Das gleichseitige Dreieck wird durch die Aufnahmepunkte der Stäbe definiert. (siehe [Abb. 4-43 "Radius L4 der Werkzeugplatte" auf Seite 153](#))
- Die Drehrichtung der Positionierachsen A1 bis A3 ist positiv, wenn die Arme in Z-Richtung nach unten bewegt werden. Die Orientierungsachse A4 (Drehung um die Z-Achse) dreht positiv entgegen dem Uhrzeigersinn
- Der kartesische Nullpunkt des Delta-Roboters liegt im Mittelpunkt des Umkreises der durch die 3 Motorspindeln definiert ist (siehe auch [Abb. 4-42 "Radius L3 der Gelenkhalterung und Koordinatensystem \(X-Y Ansicht\)" auf Seite 153](#)).

Koordinaten eines Punktes Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	translatorisch absolut	
A2:	Positionierungsachse	translatorisch absolut	
A3:	Positionierungsachse	translatorisch absolut	
A4:	Orientierungsachse	rotatorisch abs./modulo	optional

Tab. 4-43: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms (Schenkel) am Motor	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des Stabs zur Werkzeugplatte	mm bzw. inch (siehe K-0-0054)	>0
L3:	Radius des durch die 3 Rotationspunkte der Arme definierten Kreises	mm bzw. inch (siehe K-0-0054)	>0
L4:	Radius der Grundplatte	mm bzw. inch (siehe K-0-0054)	>0
L5:	Horizontale Distanz von der Zahnstangen-Antriebsrolle bis zum Ursprung des Koordinatensystems	mm bzw. inch (siehe K-0-0054)	>0

Transformationsschemen

L6:	Vertikaler Abstand der Antriebsrolle zur x-y-Ebene	mm bzw. inch (siehe K-0-0054)	>0
L7:	Durchmesser der Antriebsrolle	mm bzw. inch (siehe K-0-0054)	>0
L8:	Winkelkorrektur	Grad	-45 ... +45

Tab. 4-44: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.20 2-Achs-Delta Slide

Zur Verwendung dieser Transformation ist die Transformationsnummer 45 im Parameter "K-0-0031, Transformationsschema" einzutragen.

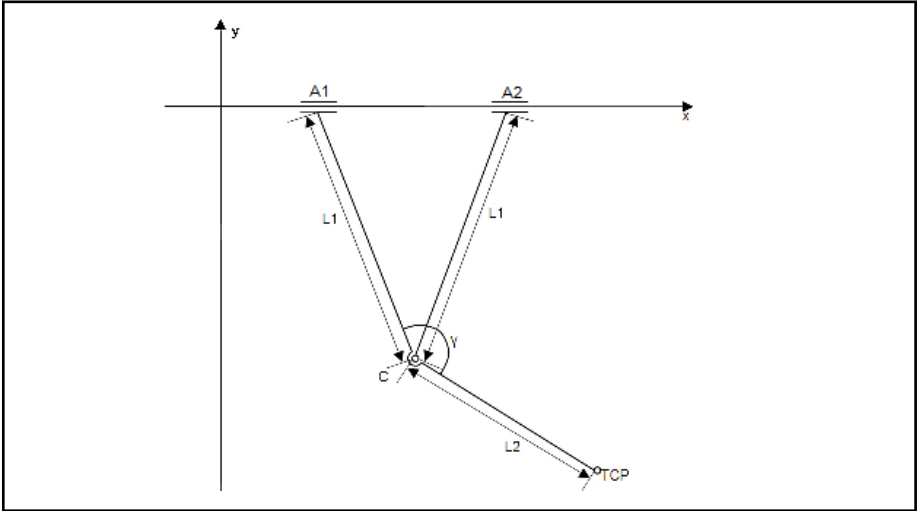


Abb. 4-46: Schematische Ansicht 2-Achs-Delta Slide

Achs-Drehrichtungen und Nullstellungen

- Abb. 4-46 "Schematische Ansicht 2-Achs-Delta Slide" auf Seite 156 zeigt
- Die schematische Darstellung der Kinematik
 - Die Armlängen L_1 , L_2
 - Den Winkel γ am Punkt C
 - Die positiven Richtungen der Achsen in positive X-Richtung
 - Die Länge L_1 ist immer positiv und ungleich NULL einzugeben
 - Die Länge L_2 ist immer positiv einzugeben
 - Der Winkel γ (K_1) ist immer zwischen 0 und 360 Grad einzugeben
 - Das zugehörige Basiskoordinaten-System ist durch die 2 Hauptachsen (x, y) gekennzeichnet

Koordinaten eines Punktes

Punkt(X, Y, H1, ..., H10, B1, ..., B16)
Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung
A1:	Positionierungsachse	translatorisch absolut
A2:	Positionierungsachse	translatorisch absolut

Tab. 4-45: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden

- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge der Arme A, C und B, C	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des Arms C, TCP	mm bzw. inch (siehe K-0-0054)	>=0

Tab. 4-46: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Faktor	Achskonfiguration	Maßeinheit	Achskopplung
K1:	Winkel Gamma	Grad	$0 \leq \gamma < 360$

Tab. 4-47: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.21 3-Achs-Parallelogramm V3

Zur Verwendung dieser Transformation ist die Transformationsnummer 46 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

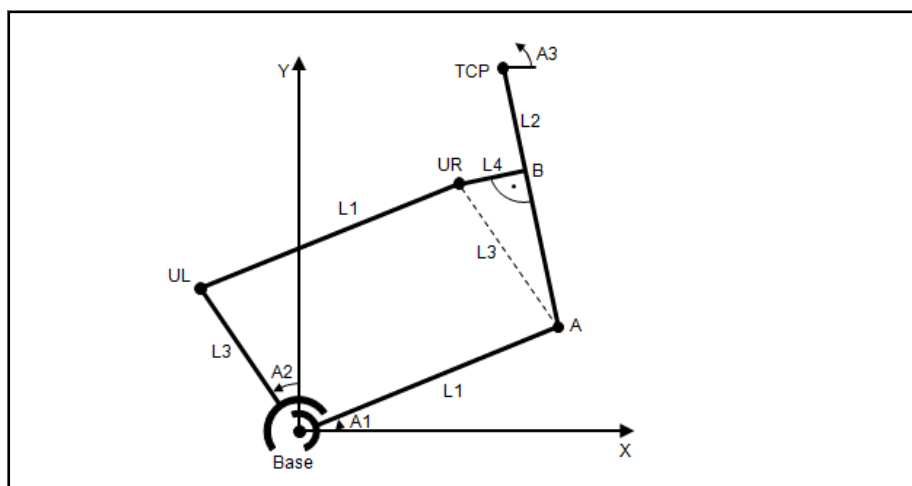


Abb. 4-47: Schematische Ansicht 3-Achs-Parallelogramm V3

Achs-Drehrichtungen und Nullstellungen

In der schematischen Ansicht sind folgende Angaben zu entnehmen:

- Die Armlänge L_1 zwischen Base und A und zwischen UL und UR
- Die Armlänge L_2 zwischen A und TCP
- Die Armlänge L_3 zwischen Base und UL und zwischen A und UR
- Die Armlänge L_4 zwischen B und UR
- die Nullstellung der Achsen A1, A2 und A3 am Pfeilstart
- die positiven Drehrichtungen der Achsen A1, A2 und A3 in Pfeilrichtung
- Die Längen L_1 , L_2 , L_3 sind immer positiv und ungleich Null anzugeben
- Die Länge L_4 ist immer positiv anzugeben
- Die Länge L_4 muss kleiner als die Länge L_3 angegeben werden.
- Das zugehörige Basiskoordinaten-System ist durch die 2 Hauptachsen (x, y) gekennzeichnet
- Der Basiskoordinaten-Nullpunkt liegt im Aufhängpunkt der Achsen A1 und A2

Koordinaten eines Punktes

Punkt(X, Y, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Transformationsschemen

Parametereinstellungen

Achsname	Achstyp	Wichtung
A1:	Positionierungsachse	rotatorisch absolut
A2:	Positionierungsachse	rotatorisch absolut
A3:	Orientierungsachse	rotatorisch absolut

Tab. 4-48: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des 1. Arms	mm bzw. inch (siehe K-0-0054)	>0
L2:	Länge des 2. Arms	mm bzw. inch (siehe K-0-0054)	>0
L3:	Länge des 3. Arms	mm bzw. inch (siehe K-0-0054)	>0
L4:	Länge des 4. Arms	mm bzw. inch (siehe K-0-0054)	>=0

Tab. 4-49: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Faktor	Achskonfiguration
K1:	Mechanische Kopplung zwischen Achse A2 und A3

Tab. 4-50: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.22 6-Achs-Kartesiser

Zur Verwendung dieser Transformation ist die Transformationsnummer 47 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

Transformationsschemen

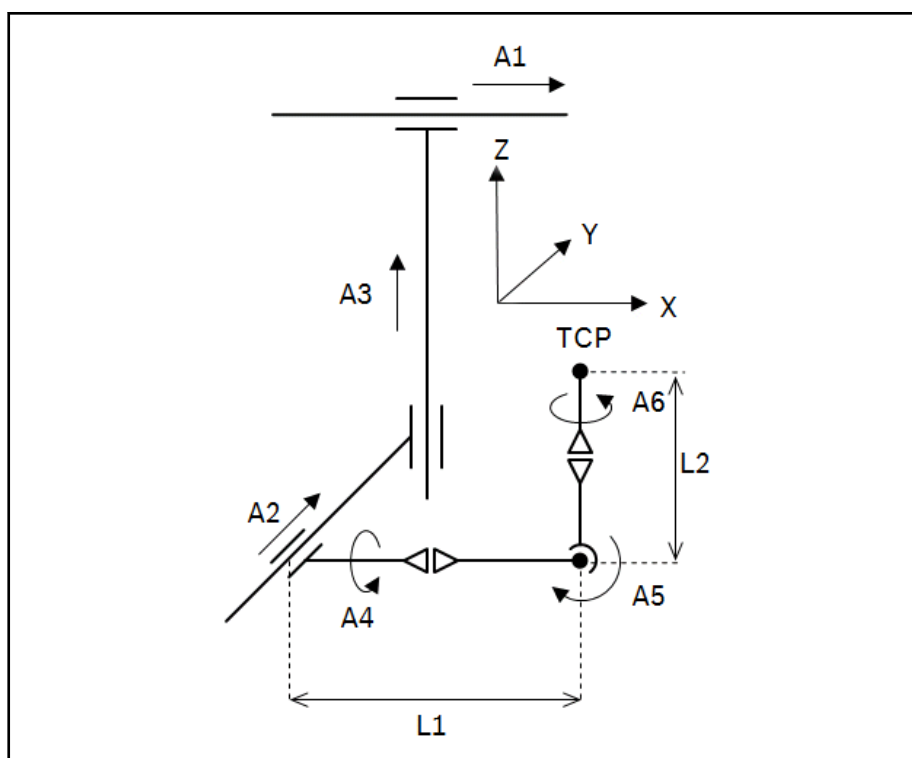


Abb. 4-48: 6-Achs-Karteser (Achsnullstellung)

Achsdrehrichtungen und Nullstellungen

Abb. 4-48 "6-Achs-Karteser (Achsnullstellung)" auf Seite 159 zeigt die schematische Darstellung des 6-Achs-Kartesers bei Nullstellung aller Achsen A4, A5, A6. Die Nullstellungen der Achsen A1, A2 und A3 entsprechen dem Ursprung des Basis-Koordinatensystems.

Die Armlängen L1 und L2 und die positiven Drehrichtungen sind ebenfalls dargestellt.

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	translatorisch absolut	
A2:	Positionierungsachse	translatorisch absolut	
A3:	Positionierungsachse	translatorisch absolut	
A4:	Orientierungsachse	rotatorisch absolut	
A5:	Orientierungsachse	rotatorisch absolut	
A6:	Orientierungsachse	rotatorisch absolut	

Tab. 4-51: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Transformationsschemen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms zwischen A2 und A5	mm bzw. in (siehe K-0-0054)	>=0
L2:	Länge des Arms zwischen A5 und TCP	mm bzw. in (siehe K-0-0054)	>=0

Tab. 4-52: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.23 3-Achs-Delta mit 90 Grad Winkeln

Zur Verwendung dieser Transformation ist die Transformationsnummer 56 im Parameter "[K-0-0031](#), Transformationsschema" einzutragen.

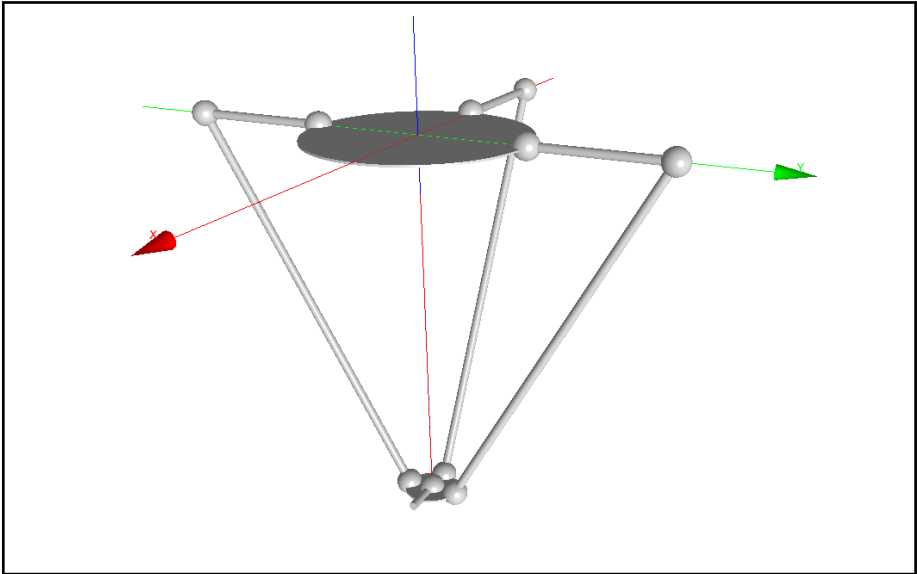


Abb. 4-49: Schematische Ansicht 3-Achs-Delta mit 90 Grad Winkeln

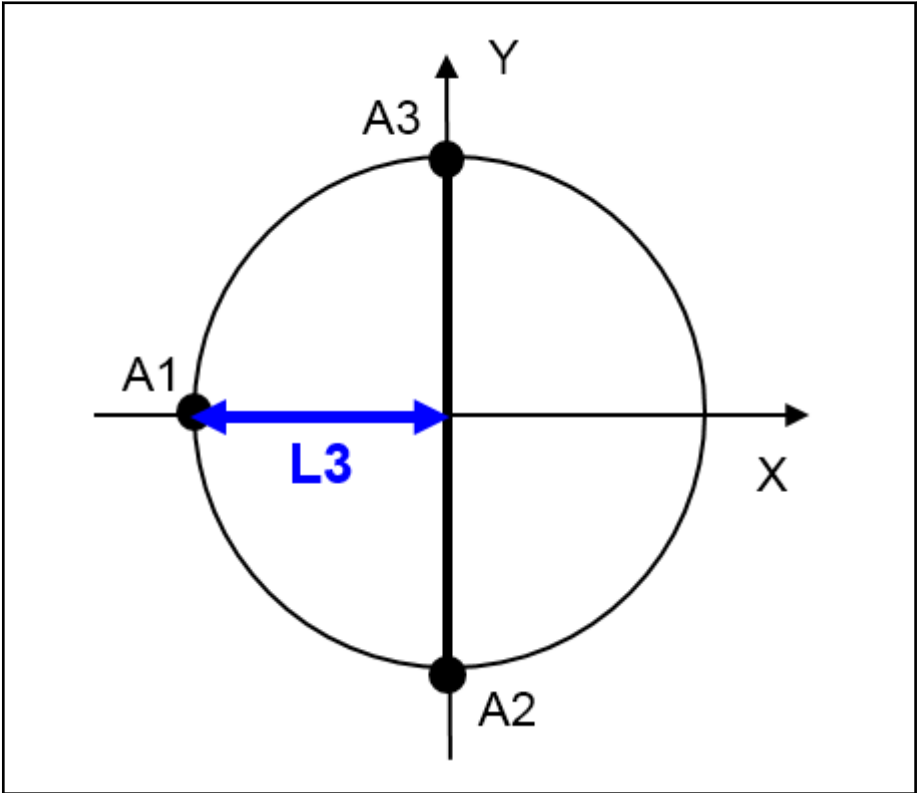


Abb. 4-50: Radius L3 der Motorhalterung und Koordinatensystem (X-Y-Ansicht)

Transformationsschemen

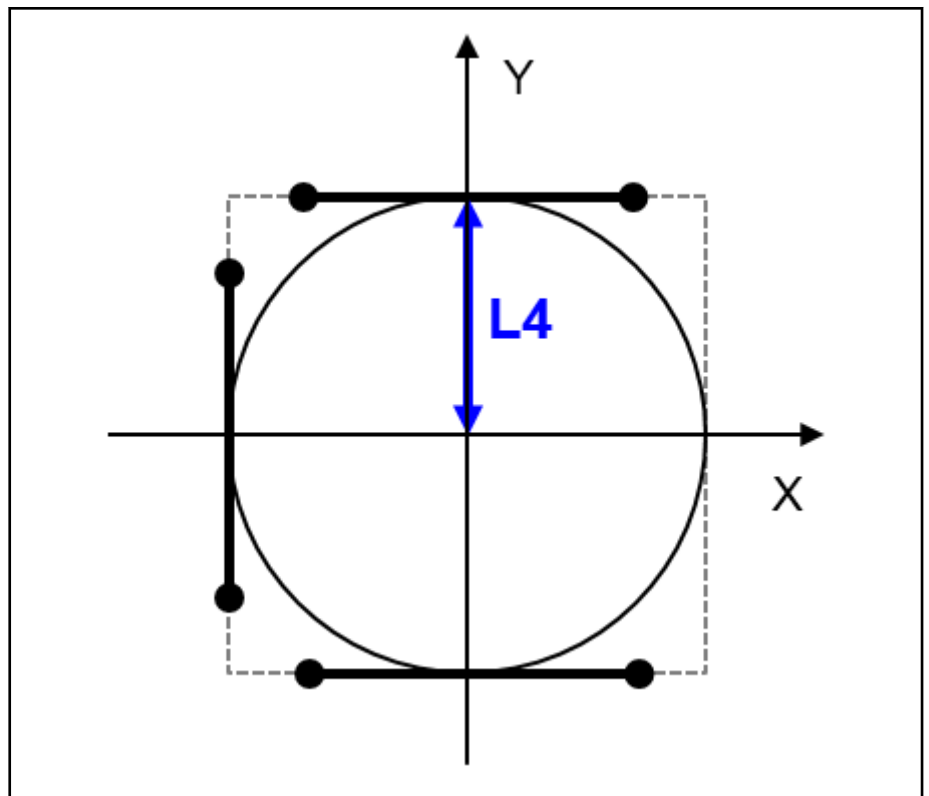
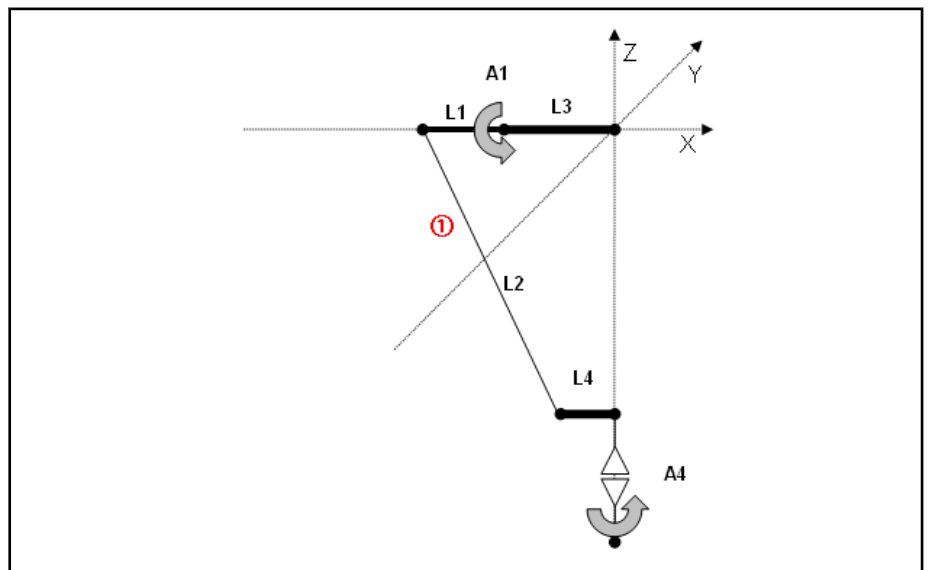


Abb. 4-51: Radius $L4$ der Werkzeugplatte



① Verbindungsstange

Abb. 4-52: Darstellung der Nullstellung der ersten Achse

Nullstellung der Orientierungsachse A4 ist die positive X-Richtung.

Achs-Drehrichtungen und Nullstellungen

- Abb. 4-49 "Schematische Ansicht 3-Achs-Delta mit 90 Grad Winkeln" auf Seite 160 zeigt einen Delta-Roboter in Nullstellung aller Achsen, d. h. nimmt der Roboter die in den Abbildungen dargestellte Stellung ein, so müssen sämtliche Achsen in den Maschinenkoordinaten (MCS) Null anzeigen.

Transformationsschemen



Dies ist durch eine entsprechende Einstellung der Referenzpunkt-Istwerte zu gewährleisten.

- In der Nullstellung stehen die Achsen 1, 2 und 3 (Positionierachsen) waagerecht
- Achse 1 zeigt in Nullstellung in Richtung der positiven Y-Achse
- Achse 2 zeigt in Nullstellung in Richtung der negativen Y-Achse
- Achse 3 zeigt in Nullstellung in Richtung der negativen X-Achse
- Der Nullpunkt der 4. Achse (Orientierungsachse) ist in positiver Richtung der X-Achse
- Die Armlänge L_1 ist gemessen vom Rotationsmittelpunkt bis zum Mittelpunkt der Aufnahme der Stäbe
- Die Stablänge L_2 ist gemessen von den beidseitigen Aufnahmepunkten
- L_3 ist der Radius des durch die 3 Motorspindeln (bzw. Getriebe) gebildeten Umkreises, gemessen an den Rotationsmittelpunkten der Arme (siehe [Abb. 4-50 "Radius L3 der Motorhalterung und Koordinatensystem \(X-Y-Ansicht\)"](#) auf Seite 160)
- L_4 ist der Radius des Innenkreises des Quadrats auf der Werkzeugplatte. Das Quadrat wird durch die Aufnahmepunkte der Stäbe definiert (siehe [Abb. 4-51 "Radius L4 der Werkzeugplatte"](#) auf Seite 161)
- Die Drehrichtung der Positionierachsen A1 bis A3 ist positiv, wenn die Arme in Z-Richtung nach unten bewegt werden. Die Orientierungsachse A4 (Drehung um die Z-Achse) dreht positiv entgegen dem Uhrzeigersinn
- Der kartesische Nullpunkt des Delta-Roboters liegt im Mittelpunkt des Umkreises, der durch die 3 Motorspindeln definiert ist (siehe auch [Abb. 4-52 "Darstellung der Nullstellung der ersten Achse"](#) auf Seite 161)

Koordinaten eines Punktes Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	Rotatorisch absolut	
A2:	Positionierungsachse	Rotatorisch absolut	
A3:	Positionierungsachse	Rotatorisch absolut	
A4:	Orientierungsachse	Rotatorisch absolut/modulo	Optional

Tab. 4-53: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms (Schenkel) am Motor	mm bzw. Inch (siehe K-0-0054)	>0

Transformationsschemen

L2:	Länge des Stabs zur Werkzeugplatte	mm bzw. Inch (siehe K-0-0054)	>0
L3:	Radius des durch die 3 Rotationspunkte der Arme definierten Kreises	mm bzw. Inch (siehe K-0-0054)	>0
L4:	Radius der Grundplatte	mm bzw. Inch (siehe K-0-0054)	>0

Tab. 4-54: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

4.2.24 4-Achs-Speedpicker

Zur Verwendung dieser Transformation ist die Transformationsnummer 61 im Parameter "K-0-0031, Transformationsschema" einzutragen.

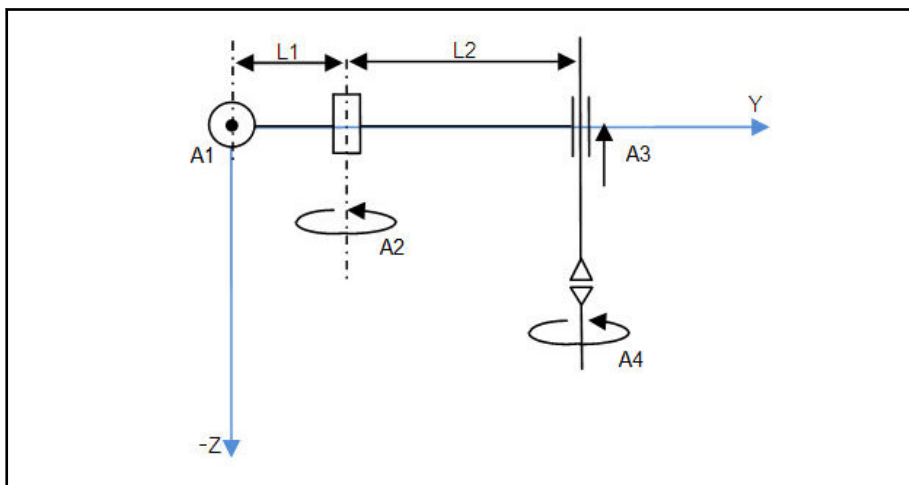


Abb. 4-53: Schematische Ansicht 4-Achs-Speedpicker

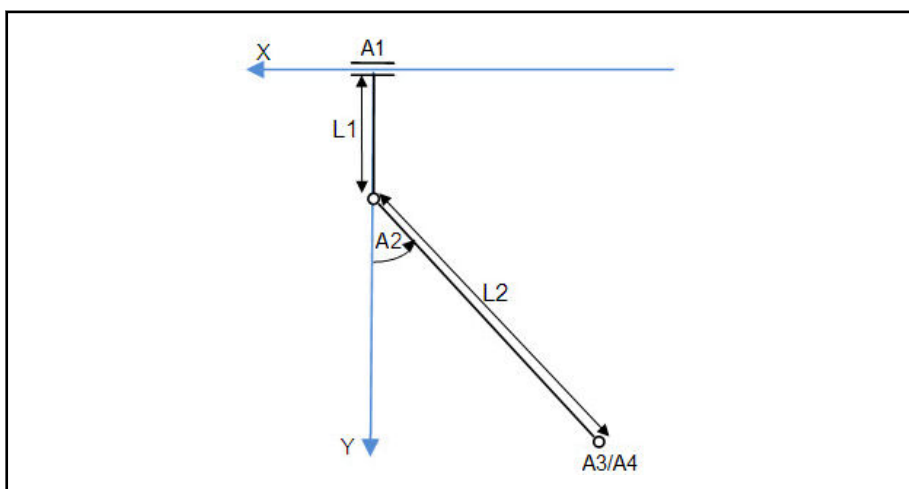


Abb. 4-54: Schematische Ansicht 4-Achs-Speedpicker (von oben)

Achs-Drehrichtungen und Nullstellungen

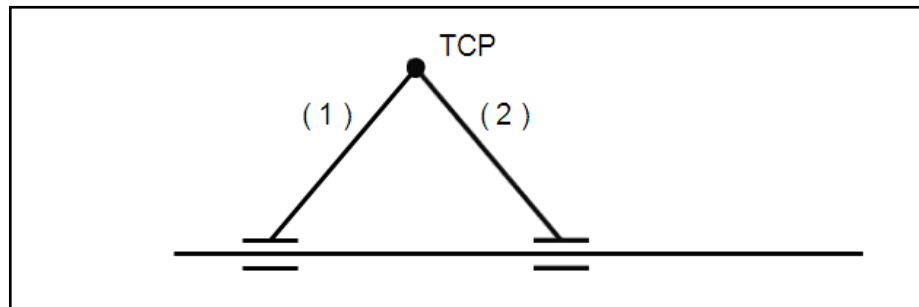
Abb. 4-53 "Schematische Ansicht 4-Achs-Speedpicker" auf Seite 163 und Abb. 4-54 "Schematische Ansicht 4-Achs-Speedpicker (von oben)" auf Seite 163 zeigen

- die schematische Darstellung der Kinematik
- die Armlängen L_1 , L_2
- das zugehörige Basis-Koordinatensystem ist durch die 2 Hauptachsen (x, y) gekennzeichnet

Transformationsschemen

Armigkeitsverhalten bei PTP-Bewegungen

Der Kopplungsfaktor K_3 steuert das Armigkeitsverhalten bei PTP-Bewegungen bei denen der Zielpunkt nicht in Achskoordinaten angegeben wird. Der Speedpicker kann folgende Armigkeiten für den gleichen TCP besitzen:



- (1) Linksarmig
(2) Rechtsarmig

Abb. 4-55: Links-/Rechtsarmigkeit des Speedpickers

Gibt man bei einer PTP-Bewegung den Zielpunkt nicht in Achskoordinaten an, so kann der Zielpunkt links- oder rechtsarmig angefahren werden.

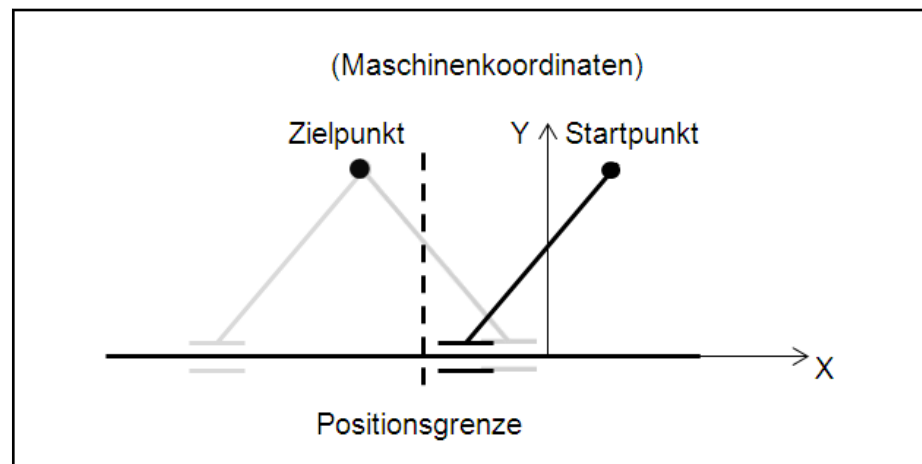


Abb. 4-56: Zielpunkt

Der Wert im Parameter K_3 steuert das Verhalten:

- 0: Die Armigkeit wird beibehalten. In diesem Fall wird der Zielpunkt linksarmig angefahren
- 1: Achse 1 (X-Schlitten) soll sich wenig bewegen. In diesem Fall ist der Positionsabstand von der Achse 1 zwischen Start und Zielpunkt kleiner, wenn der Zielpunkt rechtsarmig angefahren wird. Es wird also diese Lösung verwendet
- 2: Achse 2 (Arm) soll sich wenig bewegen. In diesem Fall ist der Positionsabstand von der Achse 2 zwischen Start und Zielpunkt kleiner, wenn der Zielpunkt linksarmig angefahren wird. Es wird also diese Lösung verwendet
- 3: Es wirkt die Positionsgrenze K_4 zur Entscheidung der Armigkeit. In diesem Fall liegt der Zielpunkt links von der Positionsgrenze, der Zielpunkt wird daher rechtsarmig angefahren

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Transformationsschemen

Achsname	Achstyp	Wichtung
A1:	Positionierungsachse	translatorisch absolut
A2:	Positionierungsachse	rotatorisch absolut
A3:	Positionierungsachse	translatorisch absolut
A4:	Orientierungsachse	rotatorisch absolut

Tab. 4-55: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Länge des Arms zwischen G1 und G2	mm bzw. inch (siehe K-0-0054)	≥ 0
L2:	Länge des Arms zwischen G2 und G3/G4	mm bzw. inch (siehe K-0-0054)	> 0

Tab. 4-56: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Faktor	Achskonfiguration	Maßeinheit
K1:	mechanische Kopplung zwischen Achse 4 und 3 (Hub-/Drehspindel)	mm/Grad
K3:	Armigkeitsverhalten bei PTP-Bewegungen 0: Bei mehrdeutigen Lösungen wird die Lösung verwendet, die die gleiche Armigkeit wie der Startpunkt hat 1: Achse 1 soll sich wenig bewegen: Werden bei PTP-Bewegungen die Zielpositionen nicht in Achskoordinaten angegeben, so wird bei mehrdeutigen Lösungen die Lösung verwendet, die die kleineren Bewegungen der Achse 1 zur Folge hat 2: Achse 2 soll sich wenig bewegen: Werden bei PTP-Bewegungen die Zielpositionen nicht in Achskoordinaten angegeben, so wird bei mehrdeutigen Lösungen die Lösung verwendet, die die kleineren Bewegungen der Achse 2 zur Folge hat 3: Positionsgrenze: Wird bei PTP-Bewegungen der Zielpunkt nicht in Achskoordinaten angegeben, so wird der Zielpunkt linksarmig angefahren, wenn die X-Koordinate im Basis-Koordinatensystem größer oder gleich der Positionsgrenze (K4) ist. Wird bei PTP-Bewegungen der Zielpunkt nicht in Achskoordinaten angegeben, so wird der Zielpunkt rechtsarmig angefahren, wenn die X-Koordinate im Basis-Koordinatensystem kleiner als die Positionsgrenze (K4) ist. Diese Einstellung wird benötigt, wenn Positionen nur über eine Armigkeit erreicht werden können	
K4:	Positionsgrenze (nur aktiv bei K3 = 3)	mm bzw. inch (siehe K-0-0054)

Tab. 4-57: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.25 Stewart-Plattform

Zur Verwendung dieser Transformation ist die Transformationsnummer 66 im Parameter "K-0-0031, Transformationsschema" einzutragen.

Transformationsschemen

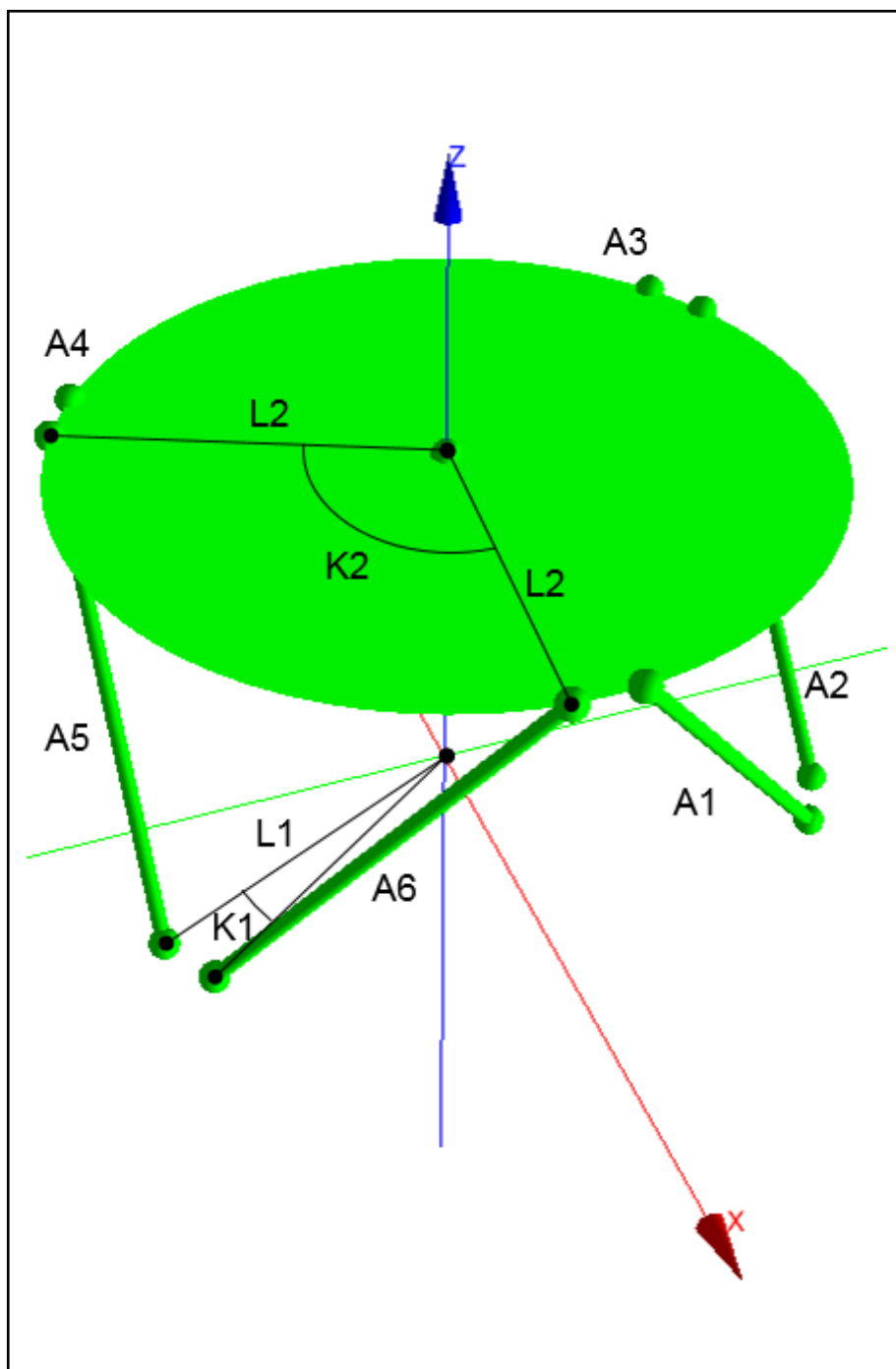


Abb. 4-57: Ansicht Stewart-Plattform

Achslängen und Verfahrrichtung

- Die Länge der Achse ist der Abstand jeder Achse von Montageposition der Basis zu Montageposition der Plattform. Es empfiehlt sich, die Achsen A1 bis A6 komplett einzufahren und die Achsen auf die Länge zwischen den Gelenken an der Basis und der Plattform zu referenzieren.
- Bei positiver Verfahrrichtung wird die Länge der Achse größer.
- Wird als $L1 = 0$ angegeben, können Punkte für die Montagepositionen der Basis direkt in den Längen L3 bis L14 angegeben werden. L1 und K1 werden in diesem Fall ignoriert. Die Montagepositionen der Basis sind dann wie folgt definiert:

Transformationsschemen

- Montageposition A1: X = L3, Y= L4
- Montageposition A2: X = L5, Y= L6
- Montageposition A3: X = L7, Y= L8
- Montageposition A4: X = L9, Y= L10
- Montageposition A5: X = L11, Y= L12
- Montageposition A6: X = L13, Y= L14

Koordinaten eines Punktes Punkt(X, Y, Z, A, B, C, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung
A1	Positionierachse	Translatorisch absolut
A2	Positionierachse	Translatorisch absolut
A3	Positionierachse	Translatorisch absolut
A4	Orientierungsachse	Translatorisch absolut
A5	Orientierungsachse	Translatorisch absolut
A6	Orientierungsachse	Translatorisch absolut

Tab. 4-58: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit
L1:	Radius der Basis	mm oder Inch (siehe K-0-0054)
L2:	Radius der Plattform	mm oder Inch (siehe K-0-0054)
L3:	X-Wert der Montagepositionen der Basis von Achse 1	mm oder Inch (siehe K-0-0054)
L4:	Y-Wert der Montagepositionen der Basis von Achse 1	mm oder Inch (siehe K-0-0054)
L5:	X-Wert der Montagepositionen der Basis von Achse 2	mm oder Inch (siehe K-0-0054)
L6:	Y-Wert der Montagepositionen der Basis von Achse 2	mm oder Inch (siehe K-0-0054)
L7:	X-Wert der Montagepositionen der Basis von Achse 3	mm oder Inch (siehe K-0-0054)
L8:	Y-Wert der Montagepositionen der Basis von Achse 3	mm oder Inch (siehe K-0-0054)
L9:	X-Wert der Montagepositionen der Basis von Achse 4	mm oder Inch (siehe K-0-0054)
L10:	Y-Wert der Montagepositionen der Basis von Achse 4	mm oder Inch (siehe K-0-0054)
L11:	X-Wert der Montagepositionen der Basis von Achse 5	mm oder Inch (siehe K-0-0054)
L12:	Y-Wert der Montagepositionen der Basis von Achse 5	mm oder Inch (siehe K-0-0054)
L13:	X-Wert der Montagepositionen der Basis von Achse 6	mm oder Inch (siehe K-0-0054)
L14:	Y-Wert der Montagepositionen der Basis von Achse 6	mm oder Inch (siehe K-0-0054)

Tab. 4-59: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Transformationsschemen

Trafo-Faktor	Achskonfiguration	Maßeinheit
K1:	Winkel zwischen A1 und A2, A3 und A4, A5 und A6 auf der Basis	Grad
K2:	Winkel zwischen A1 und A2, A3 und A4, A5 und A6 auf der Plattform	Grad
K3:	Genauigkeit der Näherung bei der Vorwärtstransformation, Wenn der Wert 0.0 ist, wird als Defaultwert 0.01 verwendet	mm oder Inch (siehe K-0-0054)
K4:	Anzahl der Iterationen bei der Vorwärtstransformation, nach dem die Vorwärtstransformation fehlerhaft abgebrochen wird. Wenn der Wert 0 ist, wird als Defaultwert 32 verwendet	Anzahl

Tab. 4-60: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

Hinweise zur Vorwärtstransformation

Für die Vorwärtstransformation wird ein Näherungsverfahren nach Newton-Raphson verwendet. Diese Vorwärtstransformation benötigt eine Abbruchbedingung, wie genau die Position im Raum berechnet werden soll. Eine genauere Berechnung der Position bei der Vorwärtstransformation hat eine längere Berechnungsdauer zur Folge.

Weiterhin kann die maximale Anzahl von Iterationen gewählt werden. Wird die maximale Anzahl erreicht, so wird davon ausgegangen, dass die Achswinkel nicht in eine Position im Raum transformiert werden können und der Fehler F22B0021 wird generiert.

Die Transformation von Achs- zu Raumkoordinaten (Vorwärtstransformation) der Stewart-Plattform ist mehrdeutig. Dies bedeutet, dass mehrere Raumpositionen zu einem Satz von Achswinkel existieren. Es ist empfehlenswert den Verfahrbereich im Raum über die Parameter [K-0-0005](#) und [K-0-0006](#) zu begrenzen. Die Daten hierfür werden in der Regel mit der Mechanik mitgeliefert.

Da die Vorwärtstransformation ein iteratives Verfahren ist, benötigt diese Vorwärtstransformation mehr Zeit als bei anderen Transformationen. Diese verlängerten Berechnungszeiten treten bei folgenden Anwendungsfällen auf:

Verlängerte Berechnungszeiten

Zeitpunkt	Kontext
- Steuerung in Phase 4 - Kinematik nicht gruppiert	Motion Kernel Istwerterfassung
- Kinematik gruppiert - Verfahrenweisung mit PTP aktiv	Motion Kernel Sollwertgenerierung
- Kinematik gruppiert - Eine der folgenden Funktionen wird verwendet: - Funktion "Transform Point" - Funktion "Move Point" - Funktion "Get Mechanic"	Aurufende Task

4.2.26 4-Achs-Scara mit Kurbel

Zur Verwendung dieser Transformation ist die Transformationsnummer 73 im Parameter "K-0-0031, Transformationsschema" einzutragen.

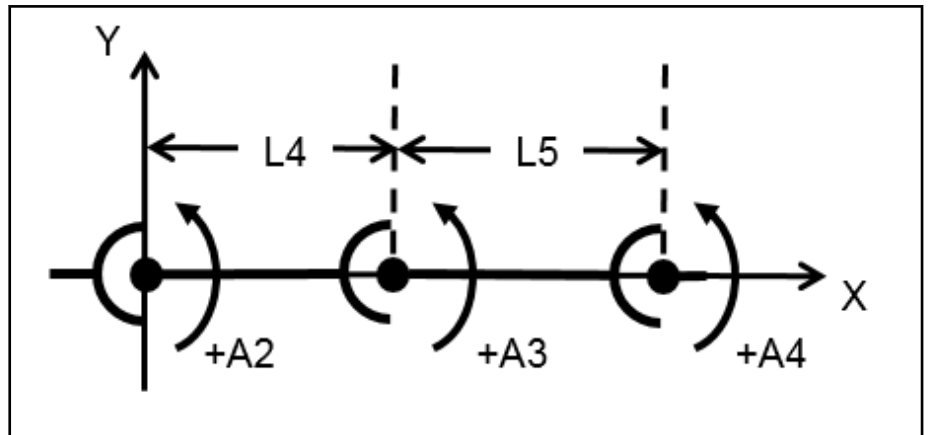


Abb. 4-58: Schematische Ansicht 4-Achs-Scara mit Kurbel (X-Y-Ebene)

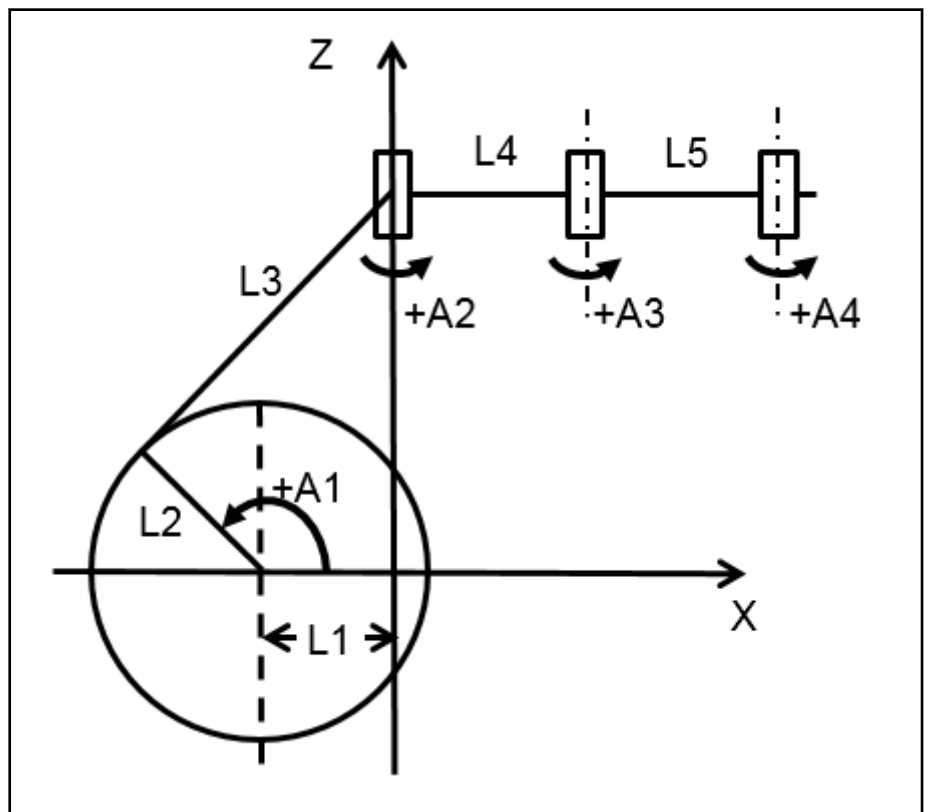


Abb. 4-59: Schematische Ansicht 4-Achs-Scara mit Kurbel (X-Z-Ebene)

Achs-Drehrichtungen und Nullstellungen

Abb. 4-58 "Schematische Ansicht 4-Achs-Scara mit Kurbel (X-Y-Ebene)" auf Seite 169: Zeigt in einer Draufsicht die Nullstellungen der Achsen A₂, A₃ und A₄.

Abb. 4-59 "Schematische Ansicht 4-Achs-Scara mit Kurbel (X-Z-Ebene)" auf Seite 169: Zeigt in einer Seitenansicht die Stellungen aller Achsen.

- Achsen A₁, A₂, A₃ und A₄ zeigen in Nullstellung in Richtung der positiven X-Achse. Der positive Drehsinn ist entgegengesetzt dem Uhrzeigersinn

Transformationsschemen

- Die Armlängen L_4 , L_5 und die positiven Drehrichtungen der Achsen sind ebenfalls in den Abbildungen dargestellt
- Die Längen L_2 , L_3 , L_4 , L_5 sind immer positiv einzugeben
- L_4 , L_5 müssen ungleich Null sein
- Die Länge L_1 definiert den Abstand der Kurbel von der Z-Achse. Ohne Beschränkung der Allgemeinheit wird angenommen, dass sich der Kurbelursprung auf der X-Achse befindet. Positive Werte von L_1 führen zu einer Verschiebung des Kurbelursprungs auf der negativen X-Achse
- Die Länge L_2 gibt die Länge der Kurbel an
- Die Länge L_3 gibt die Länge des Pleuels an
- Die durch die Kurbel hervorgerufene translatorische Bewegung findet entlang der Z-Achse statt
- Es müssen die Beziehungen zwischen $ABS(L_1) < L_2 + L_3$ eingehalten werden
- K_1 gibt die mechanische Kopplung der Achse A_2 mit der Achse A_3 an.
Eine solche Kopplung liegt vor, wenn sich beim Verfahren der Achsen A_2 die Achse A_3 mit bewegt, ohne dass das Messsystem der Achse A_3 diese Bewegung erfasst. Dies ist in der Regel dann der Fall, wenn der Motor der Achse A_3 nicht direkt auf der Achse, sondern auf dem Stativ sitzt. Der Kopplungsfaktor K_1 ist invers zum Kopplungsfaktor K_2 definiert.

*Beispiel:*Kopplungsfaktor K_1

$K_1 = 0$: Eine Bewegung der Achse A_2 hat Auswirkungen auf die Stellung des zweiten Arms im Raum: Der Winkel zwischen dem ersten Arm und dem zweiten Arm verändert sich nicht, wenn Achse A_2 bewegt wird. Die Stellung des zweiten Arms bezüglich der X-Achse verändert sich im gleichen Maße wie Achse A_1 bewegt wird.

$K_1 = 1$: Eine Bewegung der Achse A_2 hat keine Auswirkung auf die Stellung des zweiten Arms im Raum: Der Winkel zwischen dem ersten Arm und dem zweiten Arm verändert sich im gleichen Maße wie Achse A_2 bewegt wird. Die Stellung des zweiten Arms bezüglich der X-Achse bleibt konstant.

- K_2 gibt die mechanische Kopplung der Achsen A_2 und A_3 mit Achs A_4 an.
Eine solche Kopplung liegt vor, wenn sich beim Verfahren der Achsen A_2 und A_3 die Achse A_4 mit bewegt, ohne dass das Messsystem der Achse A_4 diese Bewegung erfasst. Dies ist in der Regel dann der Fall, wenn der Motor der Achse A_4 nicht direkt auf der Achse, sondern auf dem Stativ sitzt.

*Beispiel:*Kopplungsfaktor K_2

$K_2 = 0$: Eine Bewegung der Achse A_2 oder A_3 hat Auswirkungen auf die Orientierung im Raum. Die Bewegung muss von Achse A_4 ausgeglichen werden.

Transformationsschemen

$K_2 = 1$: Eine Bewegung der Achse A_2 oder A_3 hat keine Auswirkungen auf die Orientierung im Raum. Die Bewegung muss nicht von Achse A_4 ausgeglichen werden.



Der Verfahrbereich der Achsen A_2 und A_3 darf maximal ± 179 Grad betragen. Der Verfahrbereich der Achse A_1 und A_4 ist unbeschränkt.

Armigkeitswahl bei PTP-Bewegungen

Das zugehörige Basis-Koordinatensystem (BCS) ist in [Abb. 4-58 "Schematische Ansicht 4-Achs-Scara mit Kurbel \(X-Y-Ebene\)"](#) auf Seite 169 durch die 2 Hauptachsen (X, Y) gekennzeichnet.

Wird eine Bewegung [Punkt zu Punkt](#) auf Seite 42 [Kap. 2.7.4 "Bewegungsart Punkt zu Punkt"](#) auf Seite 42 ausgeführt und der anzufahrende Punkt nicht im ACS angegeben, kann dieser Punkt auf zwei verschiedene Weisen angefahren werden.

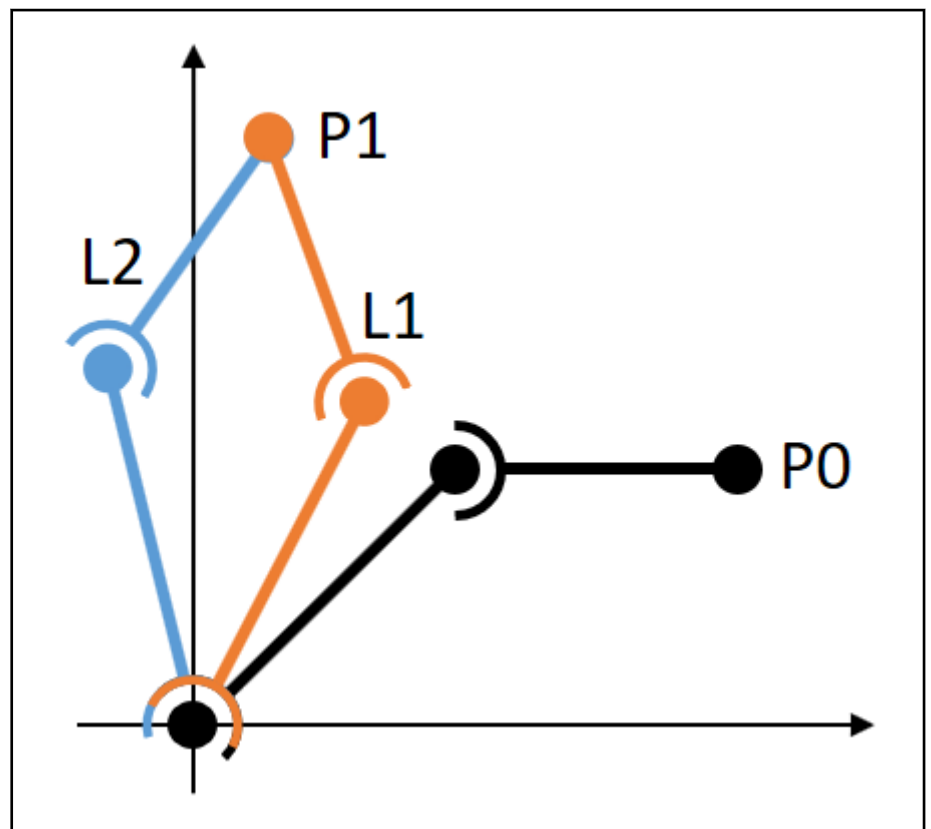


Abb. 4-60: Verschiedene Armigkeiten bei einem Scara Roboter

Von einem Punkt $P0$ kann durch eine Punkt zu Punkt Bewegung der Punkt $P1$, der nicht im ACS angegeben wurde, auf zwei verschiedene Weisen angefahren werden (L1 und L2).

Mittels K_4 kann das Verhalten der Armwahl konfiguriert werden:

- 0: Es wird der Zielpunkt so angefahren, dass die Armigkeit zum Startpunkt nicht geändert wird
- 1: Es wird der Zielpunkt so angefahren, dass die Bewegung des ersten Arms vom Start- zum Zielpunkt minimiert wird. Dabei wird die Armigkeit so gewählt, dass Achse 1 die Grenzen von $\pm 179^\circ$ nicht überschreitet

Transformationsschemen

*Beispiel:*Konfiguration K_4

- $K_4 = 0$: Punkt P1 wird mit L2 angefahren. Die Armigkeit ändert sich nicht während der Bewegung
- $K_4 = 1$: Punkt P1 wird mit L2 angefahren. L2 hat die geringere Bewegung von Arm 1 zur Folge

Koordinaten eines Punktes

Punkt(X, Y, Z, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1:	Positionierungsachse	Rotatorisch absolut	
A2:	Positionierungsachse	Rotatorisch absolut	+/- 179 Grad
A3:	Positionierungsachse	Rotatorisch absolut	+/- 179 Grad
A4:	Orientierungsachse	Rotatorisch	Optional

Tab. 4-61: K-0-0008 Achskonfigurationsliste, Achszuordnung

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit	Achslänge
L1:	Abstand der Kurbel zur X-Achse	mm bzw. Inch (siehe K-0-0054)	$ABS(L_1) < L_2 + L_3$
L2:	Länge der Kurbel	mm bzw. Inch (siehe K-0-0054)	≥ 0
L3:	Länge des Pleuels	mm bzw. Inch (siehe K-0-0054)	≥ 0
L4:	Länge des 1. Arms	mm bzw. Inch (siehe K-0-0054)	> 0
L5:	Länge des 2. Arms	mm bzw. Inch (siehe K-0-0054)	> 0

Tab. 4-62: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Faktor	Achskonfiguration	Maßeinheit	Achskopplung
K1:	Mechanische Kopplung zwischen Achse 2 und 3	Faktor	Optional
K2:	Mechanische Kopplung zwischen Achse 2, 3 und 4	Faktor	Optional
K4:	Armigkeitswahl bei PTP-Bewegungen und Punktvorgabe im nicht ACS • 0: Armigkeit wird beibehalten • 1: Armigkeit wird so gewählt, dass die Bewegung des ersten Arms minimiert wird	Faktor	Optional

Tab. 4-63: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

4.2.27 3-Achs Canfield

Zur Verwendung dieser Transformation ist die Transformationsnummer 81 im Parameter "K-0-0031, Transformationsschema" einzutragen.

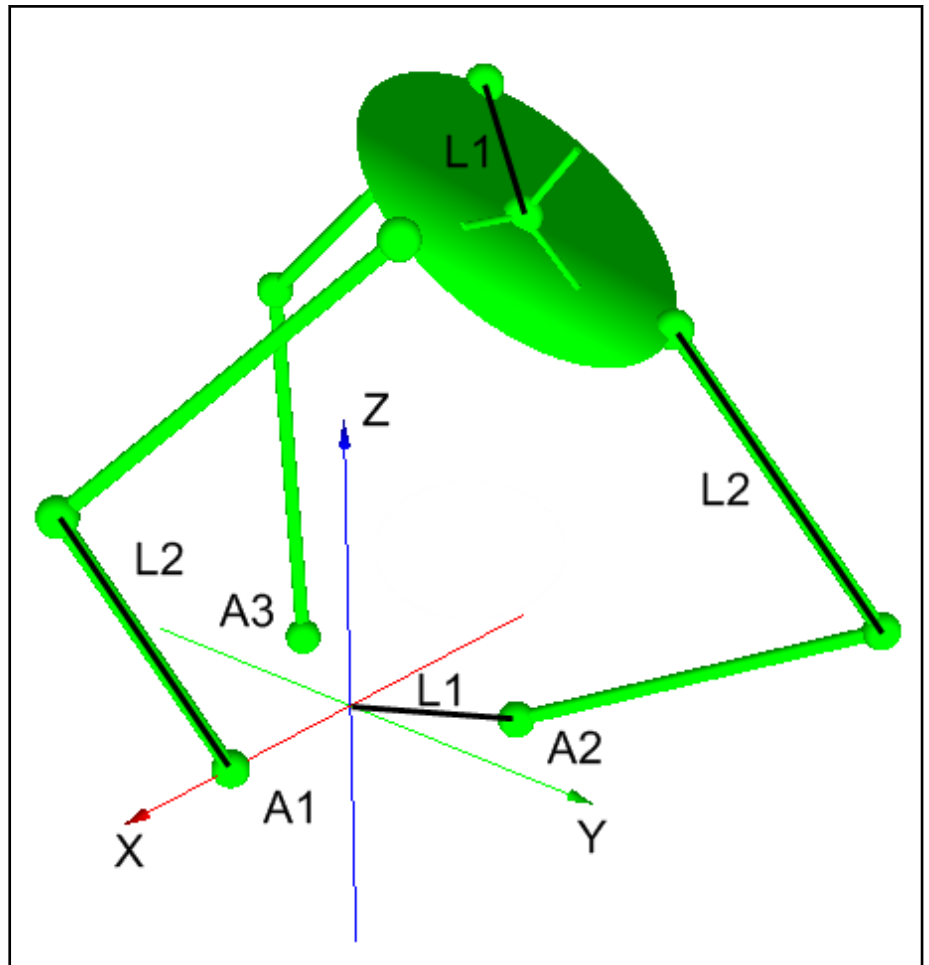


Abb. 4-61: Ansicht 3-Achs Canfield

Achs-Drehrichtungen und Nullstellungen

- Alle Achsen stehen auf 0, wenn sie in der X-Y-Ebene liegen
- Die positive Drehrichtung der Achse ist die mathematisch negative Drehrichtung. In der obigen Abbildung haben alle Achsen negative Positionswerte
- Eine vierte Achse kann in Richtung des Normalen-Vektor der Platte montiert sein

Koordinaten eines Punktes

Punkt(Z, B, C, H1, ..., H10, B1, ..., B16)

Nicht konfigurierte Hilfsachsen und Bandachsen werden nicht dargestellt.

Parametereinstellungen

Achsname	Achstyp	Wichtung	Bemerkung
A1	Positionierachse	Rotatorisch absolut	
A2	Orientierungsachse	Rotatorisch absolut	
A3	Orientierungsachse	Rotatorisch absolut	
A4	Orientierungsachse	Rotatorisch	optional

Tab. 4-64: K-0-0008 Achskonfigurationsliste, Achszuordnung

Transformationsschemen

Alle Orientierungsachsen dieses Transformationsschemas werden als Hilfsachsen interpretiert.

Alle weiteren Achsen, bis zum Maximum der Achsanzahl, können mit folgenden Achsen belegt werden:

- Orientierungsachsen, die als Hilfsachsen interpretiert werden
- Bandachsen

Trafo-Länge	Achskonfiguration	Maßeinheit
L1:	Radius der Platte, Distanz der Achsen zum Zentrum des Basiskoordinatensystems	mm oder Inch (siehe K-0-0054)
L2:	Länge aller Arme	mm oder Inch (siehe K-0-0054)

Tab. 4-65: K-0-0040, Transformationslängen, Achskonfiguration, Achslängen

Trafo-Faktor	Achskonfiguration	Maßeinheit
K1:	Minimale Eintauchtiefe der ersten Schutzzone	mm oder Inch (siehe K-0-0054)
K2:	Maximale Eintauchtiefe der ersten Schutzzone	mm oder Inch (siehe K-0-0054)
K3:	Neigung der ersten Schutzzone	Grad
K4:	Gierwinkel der ersten Schutzzone	Grad
K5:	Winkel der vierten Achse der ersten Schutzzone	Grad
K6:	Minimale Eintauchtiefe der zweiten Schutzzone	mm oder Inch (siehe K-0-0054)
K7:	Maximale Eintauchtiefe der zweiten Schutzzone	mm oder Inch (siehe K-0-0054)
K8:	Neigung der zweiten Schutzzone	Grad
K9:	Gierwinkel der zweiten Schutzzone	Grad
K10:	Winkel der vierten Achse der zweiten Schutzzone	Grad
K11:	Minimale Eintauchtiefe der dritten Schutzzone	mm oder Inch (siehe K-0-0054)
K12:	Maximale Eintauchtiefe der dritten Schutzzone	mm oder Inch (siehe K-0-0054)
K13:	Neigung der dritten Schutzzone	Grad
K14:	Gierwinkel der dritten Schutzzone	Grad
K15:	Winkel der vierten Achse der dritten Schutzzone	Grad

Tab. 4-66: K-0-0041, Transformationsfaktoren, Achskonfiguration, Achskopplung

Koordinaten im MCS

Koordinaten im MCS haben die Form (PD, Neigung, Gierwinkel).

- PD: Eintauchtiefe (Plunge Depth) in mm oder Inch (siehe [K-0-0054](#)) , Distanz des TCP zur Z-Achse des Basiskoordinatensystems (siehe Bild unten)
- Neigung: Rotation in Grad um die y-Achse des Basiskoordinatensystems
- Gierwinkel: Rotation in Grad um die z-Achse des Basiskoordinatensystems

Transformationsschemen

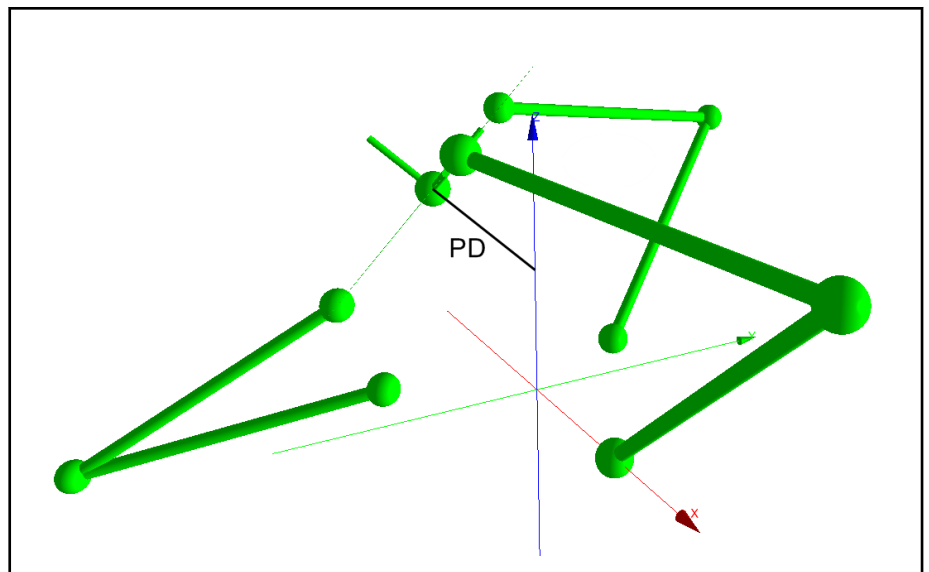


Abb. 4-62: Eintauchtiefe

Empfehlungen

Es wird empfohlen, diesen Kinematiktyp mit Move Direct Absolute und Endpunkten im MCS zu kommandieren.

Wenn dieser Kinematiktyp mit Move Linear Absolute kommandiert wird, wird Neigung und Gierwinkel mittels Eulerwinkeln interpoliert. Dies führt zu kurvenförmigen Pfaden.

Beispiel:

Bewegung von MCS (85.0, 20.0, -179.0) nach MCS (85.0, 70.0, 179.0) mit Move Linear Absolute und Move Direct Absolute

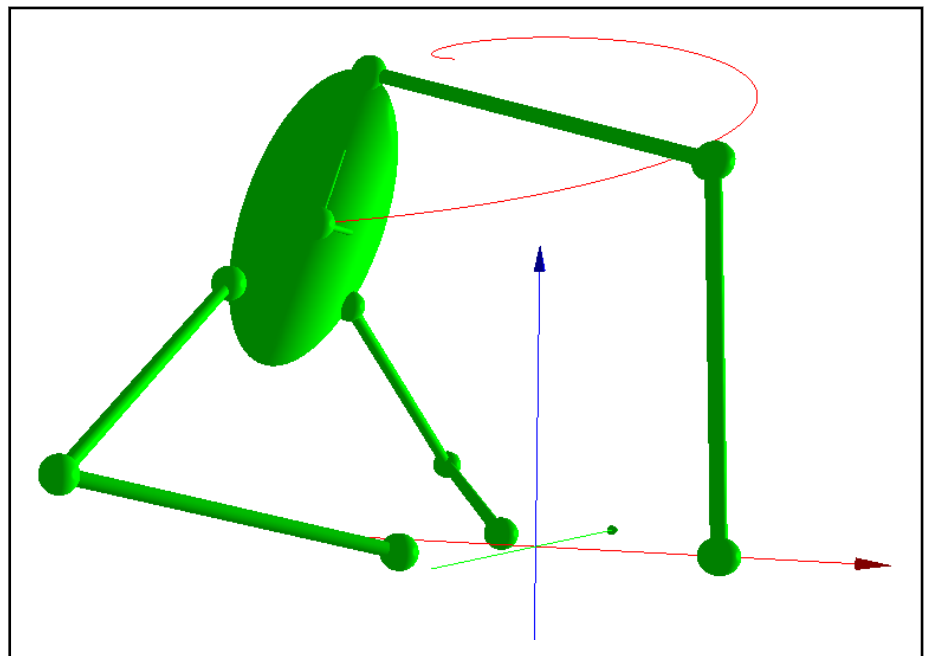


Abb. 4-63: Pfad mit Move Linear Absolute

Transformationsschemen

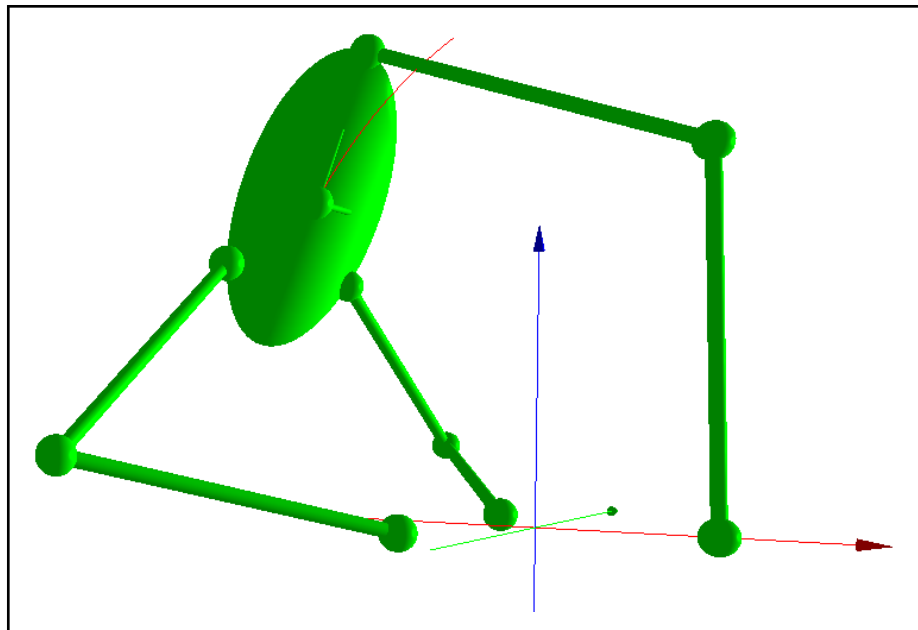


Abb. 4-64: Pfad mit Move Direct Absolute

Schutzzonen der Transformation

Es ist möglich, transformationsspezifische Schutzzonen zu konfigurieren. Ein Punkt muss mindestens in einer Schutzzone liegen, um gültig zu sein. Wenn ein Punkt nicht in mindestens einer Schutzzone liegt, wird der Fehler F22B0023 generiert.

- Schutzzone 1:** Schutzzone 1 wird durch die Kopplungsfaktoren 1 bis 5 definiert.
Wenn $K1 = 0$ wird die Schutzzone 1 nicht verwendet.
Die minimale Position im MCS ist $(K1, -K3, -K4, -K5)$.
Die maximale Position im MCS ist $(K2, K3, K4, K5)$.
- Schutzzone 2:** Schutzzone 2 wird durch die Kopplungsfaktoren 6 bis 10 definiert.
Wenn $K6 = 0$ wird die Schutzzone 2 nicht verwendet.
Die minimale Position im MCS ist $(K6, -K8, -K9, -K10)$.
Die maximale Position im MCS ist $(K7, K8, K9, K10)$.
- Schutzzone 3:** Schutzzone 3 wird durch die Kopplungsfaktoren 6 bis 15 definiert.
Wenn $K11 = 0$ wird die Schutzzone 3 nicht verwendet.
Die Schutzzone 3 existiert 3 mal: Einmal wie konfiguriert, einmal um 120° verdreht um die Z-Achse, einmal um -120° verdreht um die Z-Achse.
Die minimale Position im MCS ist $(K11, -K13, -K14, -K15)$.
Die maximale Position im MCS ist $(K12, K13, K14, K15)$.

4.3 Benutzerdefinierte Transformation

In der Applikation kommt es häufig vor, dass die Transformationen, die das System bereit stellt, nicht zu der eigenen Kinematik passen. Hierfür bietet das System die Möglichkeit des benutzerdefinierten Transformationstyps. Dazu muss als Transformationsschema im Parameter "[K-0-0031](#), Transformationsschema" der Wert 100 eingetragen werden. Bei diesem Typ werden die Umrechnungsvorschriften zwischen Basis- und Achskoordinaten in der SPS implementiert.

Transformationsschemen

Es werden zwei Richtungen der Transformation unterschieden:

- **Vorwärtstransformation:** Die Vorwärtstransformation rechnet Positionen vom ACS ins BCS um.
- **Rückwärtstransformation:** Die Rückwärtstransformation rechnet Positionen vom BCS ins ACS um.

Zu implementierende Methoden

Um eine Transformation zu definieren, muss das Interface MC_KIN_REF, ML_Robot, von einem Funktionsbaustein implementiert werden.

Dieses Interface beinhaltet drei Methoden. Die Methoden brauchen nicht innerhalb der SPS aufgerufen zu werden. Sie werden vom System automatisch aufgerufen.

- **Initialisation:** Methode zum Initialisieren der Transformation. Hier sollte man alle notwendigen Überprüfungen der Randbedingungen und Vorberechnungen durchführen, damit die Transformation fehlerfrei arbeiten kann.

Dazu gehören:

- Überprüfung der notwendigen Achszahl für die Transformation im Parameter "K-0-0008, Achskonfigurationsliste"
- Überprüfungen der richtigen Achswichtung für die Transformation der konfigurierten Achsen im Parameter "K-0-0008, Achskonfigurationsliste"
- Überprüfung der Parameter "K-0-0040, Transformation, Längen der Mechanik" und "K-0-0041, Transformation, Faktoren"
- Vorberechnungen von Werten, die in der Vorwärts- und Rückwärts-Transformation verwendet werden

Der Methode wird der Inhalt der Parameter "K-0-0008, Achskonfigurationsliste", "K-0-0040, Transformation, Längen der Mechanik" und "K-0-0041, Transformation, Faktoren" übergeben. Kann die Initialisierung nicht erfolgreich durchgeführt werden, so wird ein Wert ungleich Null zurückgegeben.

- **Forward:** Methode, um von Koordinaten des Achs-Koordinatensystems in Koordinaten des Basis-Koordinatensystems umzurechnen ([Transformationen auf Seite 28](#)).

Die Methode erhält dazu zwei Referenzen auf die Koordinaten des Achs-Koordinatensystems und die Koordinaten des Basis-Koordinatensystems.

Die Methode muss aus den Koordinaten des Achs-Koordinatensystems die Koordinaten des Basis-Koordinatensystems errechnen.

Kann ein Punkt nicht transformiert werden, muss die Methode einen Wert ungleich Null zurückgeben.

- **Inverse:** Methode, um von Koordinaten des Basis-Koordinatensystems in Koordinaten des Achs-Koordinatensystems umzurechnen ([Transformationen auf Seite 28](#)).

Die Methode erhält dazu zwei Referenzen auf die Koordinaten des Basis-Koordinatensystems und die Koordinaten des Achs-Koordinatensystems.

Die Methode muss aus den Koordinaten des Basis-Koordinatensystems die Koordinaten des Achs-Koordinatensystems errechnen.

Kann ein Punkt nicht transformiert werden, muss die Methode einen Wert ungleich Null zurückgeben.

Fehlerhandling der Methoden

Transformationsschemen

Tritt bei einer der Interfacefunktionen (Methoden) ein Floating Point-Fehler auf, so wird einer der unten stehenden Fehler gemeldet.

Solche Fehler sollten allerdings bereits in der Mathematik der Transformation berücksichtigt werden: So sollten z.B. ein Divisor immer auf Null überprüft werden, bei Gleichheit ein Fehler zurückgegeben und die Division nicht durchgeführt werden.

- F12C0600, Ungültige Operation beim SPS-Funktionsaufruf
- F12C0601, Denormalisierte Zahl beim SPS-Funktionsaufruf
- F12C0602, Division durch Null beim SPS-Funktionsaufruf
- F12C0603, Überlauf beim SPS-Funktionsaufruf
- F12C0604, Unterlauf beim SPS-Funktionsaufruf
- F12C0605, Genauigkeitsfehler beim SPS-Funktionsaufruf

Soll eine Interfacefunktion (Methode) aufgerufen werden, ohne dass diese angemeldet wurde, so wird folgender Fehler gemeldet:

- F12C0498, Transformation in der SPS wurde nicht angemeldet

Kann eine Interfacefunktion nicht erfolgreich ausgeführt werden (Rückgabewert der Funktion ungleich Null, so wird folgender Fehler gemeldet:

- F12C0499, Benutzerdefinierter Transformationsfehler in der SPS

Anmelden einer Transformation

Wurde ein Funktionsbaustein implementiert, der eine Transformation darstellt (der FB implementiert das Interface MC_KIN_REF, ML_Robot), so kann dieser mit dem Funktionsbaustein MC_SetKinTransform, ML_Robot, am System angemeldet werden.

Für die Anmeldung muss eine Instanz des eigenen Transformationsbausteins und die Kinematik, für die die Instanz gelten soll, angegeben werden.

Die Anmeldung der Transformation muss in der Konfigurationsphase der Steuerung (P2) erfolgen.

Ist beim Phasenschalten in den Betriebszustand der Steuerung (BB) in der Kinematik eine benutzerdefinierte Transformation definiert, jedoch keine Transformation angemeldet, so wird der Fehler F12C0498 "Transformation in der SPS wurde nicht angemeldet" gemeldet.

Soll in mehreren Kinematiken die gleiche Transformation verwendet werden, ist es notwendig, jeweils eine eigene Instanzen des selbst erstellten Transformations-Funktionsbausteins für die jeweilige Kinematik zu verwenden.

Anwendungsbeispiel

Ein Anwendungsbeispiel für das Erstellen und die Anwendung von benutzerdefinierter Transformationen findet man unter [Benutzerdefinierte Transformation auf Seite 401](#).

4.4 Erweiterte Benutzerdefinierte Transformation

In der Applikation kommt es häufig vor, dass die Transformationen, die das System bereitstellt, nicht zu der eigenen Kinematik passen. Hierfür bietet das System die Möglichkeit des erweiterten benutzerdefinierten Transformations-typs. Dazu muss als Transformationsschema im Parameter "K-0-0031, Transformationsschema" der Wert 111 eingetragen werden.

Der erweiterte benutzerdefinierte Transformationstyp bietet eine erweiterte Schnittstelle zur Definition einer Transformation.

Diese Schnittstelle bietet folgende Vorteile gegenüber der einfachen benutzerdefinierten Transformation:

Transformationsschemen

- Erhöhte Genauigkeit der Berechnungen durch Verwendung des Datentyps LREAL
- Behandlung von Mehrdeutigkeiten einer Transformation
- Darstellung einer mechanischen Ansicht der Kinematik mit Hilfe von Mechanischen Informationen (siehe [Mechanische Informationen auf Seite 183](#))
- Angabe von Freiheitsgraden einer Kinematik



Es wird empfohlen aus den oben genannten Gründen statt Transformationstyp 100 (Benutzerdefinierte Transformation) den Transformationstyp 111 (Erweiterte Benutzerdefinierte Transformation) zu verwenden.

Der wichtigste Unterschied zwischen einfacher und erweiterter benutzerdefinierter Transformation ist die Darstellung eines Punktes.

Die Koordinaten besitzen feste Positionen für X, Y, Z, A, B, C, H1 ... H10 in genau dieser Reihenfolge.

Fehlende Freiheitsgrade werden bei allen Interface-Methoden einer erweiterten benutzerdefinierten Transformation nicht weggelassen, wie es bei der einfachen benutzerdefinierten Transformation der Fall ist, sondern sind immer vorhanden. Diese Koordinaten können in den Methoden ignoriert werden.

Dazu teilt eine Transformation die unterstützten Freiheitsgrade über die Interfacemethode `GetUsedAxes`, `MC_KIN_REF_EXT`, in `ML_Robot`, dem System mit.

Punkte haben an den angegebenen Freiheitsgraden die Koordinatenwerte des zu transformierenden Punktes. Dies gilt sowohl für ACS- als auch für BCS-Punkte. Im ACS sind die Koordinatenwerte die Achspositionen der Achse. Im BCS haben die Koordinatenwerte folgende Bedeutung:

- X, Y, Z: Kartesische Koordinaten des Punktes
- A, B, C: Verdrehung des Punktes im Raum
 - A: Verdrehung um die X-Achse. Definitionsbereich: -180° - $+180^\circ$
 - B: Verdrehung um die Y-Achse. Definitionsbereich: -90° - $+90^\circ$
 - C: Verdrehung um die Z-Achse. Definitionsbereich: -180° - $+180^\circ$
- H1 ... H10: Hilfsachsen

Bei Transformationen, die nur **einen** Freiheitsgrad in der Orientierung haben, ist es ratsam diesen Freiheitsgrad nicht als den tatsächlichen Freiheitsgrad anzugeben, sondern die Orientierungsachse als Hilfsachse zu betrachten. Bei Angabe **eines** Freiheitsgrad als echte Orientierungsachse entstehen folgende Nachteile:

- Die Werte A, B, C haben einen eingeschränkten Definitionsbereich. Falls die Orientierungsachse einen größeren Definitionsbereich oder Modulo gewichtet ist, so muss dies die Transformation selbst aufwändig umgerechnet werden
- Bei einer Verdrehung der Kinematik im Raum wird auch die Orientierung im Raum transformiert. Dadurch kann es je nach Verdrehung dazu kommen, dass sich Koordinatenwerte der Orientierung vertauschen oder gar nicht mehr auftauchen. Da nicht vorhandene Freiheitsgrade einer Transformation gegenüber dem Anwender weggelassen werden, kann der Anwender dann die Orientierung nicht mehr kommandieren

Transformationsschemen

Definiert man die eine Orientierungsachse als Hilfsachse, so ergeben sich daraus folgende Eigenschaften:

- Für die Achsen gibt es keinen eingeschränkten Definitionsbereich. Sowohl Modulo als absolut gewichtete Achsen können verwendet werden, ohne Einschränkungen im Definitionsbereich in Kauf nehmen zu müssen
- Der Anwender muss die Orientierungsachse immer mit Achswinkeln kommandieren. Die Orientierung wird bei einer Verdrehung der Kinematik nicht mehr mit transformiert. Dies führt beim Anwender zu weniger Verwirrungen beim Kommandieren der Orientierung

Hilfsachsen werden vom System automatisch zwischen ACS und BCS kopiert und müssen so nicht in der Vorwärts- und Rückwärtstransformation kopiert werden. Sie können allerdings verrechnet werden, wenn sie z. B. Einfluss auf die Position des TCP haben.

Beispiel:

Scara mit zwei Positionierachsen und einer Orientierungsachse

Ein Scara-Roboter besitzt 2 Positionierachsen, die X und Y beeinflussen und eine Orientierungsachse um Z (C-Koordinate).

Aus den oben genannten Gründen ist es ungünstig, die Interfacemethode `GetUsedAxes` so zu implementieren, dass sie `ML_KIN_USED_AXES.X_AXIS` OR `ML_KIN_USED_AXES.Y_AXIS` OR `ML_KIN_USED_AXES.Z_AXIS` zurückgibt (Koordinate 1, 2 und 6).

Es ist besser nur `ML_KIN_USED_AXES.X_AXIS` OR `ML_KIN_USED_AXES.Y_AXIS` zurückzugeben (Hilfsachsen sind automatisch immer definiert).

Dadurch sind die Werte der Orientierungsachse nicht an Koordinate 6 (`ML_KIN_USED_AXES.Z_AXIS`) sondern an Koordinate 7 zu übergeben, und die oben genannten Eigenschaften ergeben sich.

ACS- und BCS-Punkte enthalten dann:

- An Koordinate 1 die Koordinate X bzw. Motor 1
- An Koordinate 2 die Koordinate Y bzw. Motor 2
- An Koordinate 7 die Koordinate H1 bzw. Motor 3

Die Transformation muss diese Koordinaten verarbeiten. Alle anderen Koordinaten können ignoriert werden.

Zu implementierende Methoden

Um eine erweiterte Transformation zu definieren, muss das Interface `MC_KIN_REF_EXT`, `ML_Robot`, von einem Funktionsbaustein implementiert werden.

Dieses Interface beinhaltet sechs Methoden. Die Methoden brauchen nicht innerhalb der SPS aufgerufen zu werden. Sie werden vom System automatisch aufgerufen.

- **GetUsedAxes:** Methode zur Definition der Freiheitsgrade der Transformation.

Besitzt eine Transformation den Freiheitsgrad um eine Achse, so muss der entsprechende Wert vom Datentyp `ML_KIN_USED_AXES`, `ML_Robot`, zurückgegeben werden.

Bei mehreren Freiheitsgraden müssen mehrere dieser Werte ODER-verknüpft werden.

Transformationsschemen

- **Initialisation:** Methode zum Initialisieren der Transformation. Hier sollte man alle notwendigen Überprüfungen der Randbedingungen und Vorberechnungen durchführen, damit die Transformation fehlerfrei arbeiten kann.

Dazu gehören:

- Überprüfung der notwendigen Achsanzahl für die Transformation im Parameter "K-0-0008, Achskonfigurationsliste"
- Überprüfungen der richtigen Achswichtung für die Transformation der konfigurierten Achsen im Parameter "K-0-0008, Achskonfigurationsliste"
- Überprüfung der Parameter "K-0-0040, Transformation, Längen der Mechanik" und "K-0-0041, Transformation, Faktoren"
- Vorberechnungen von Werten, die in der Vorwärts- und Rückwärts-Transformation verwendet werden

Der Methode wird der Inhalt der Parameter "K-0-0008, Achskonfigurationsliste", "K-0-0040, Transformation, Längen der Mechanik" und "K-0-0041, Transformation, Faktoren" übergeben.

Kann die Initialisierung nicht erfolgreich durchgeführt werden, muss die Methode einen MLC-Fehlercode zurückgeben.

- **DetermineMemory:** Mit dieser Methode können Mehrdeutigkeiten der Transformation aufgelöst werden. Dazu wird der Methode eine Referenz auf einen ACS Punkt übergeben. Dieser Punkt besitzt die gleiche mehrdeutige Lösung die bei der nächsten Rückwärtstransformation verwendet werden soll. Weiterhin wird der Methode mitgegeben, ob sich die Transformation bei der nächsten Rückwärtstransformation selbst für eine der mehrdeutigen Lösungen entscheiden darf
- **Forward:** Methode, um von Koordinaten des Achs-Koordinatensystems in Koordinaten des Basis-Koordinatensystems umzurechnen ([Transformationen auf Seite 28](#)).

Die Methode erhält dazu zwei Referenzen auf die Koordinaten des Achs-Koordinatensystems und die Koordinaten des Basis-Koordinatensystems.

Die Methode muss aus den Koordinaten des Achs-Koordinatensystems die Koordinaten des Basis-Koordinatensystems errechnen.

Kann ein Punkt nicht transformiert werden, muss die Methode einen MLC-Fehlercode zurückgeben.

- **Inverse:** Methode, um von Koordinaten des Basis-Koordinatensystems in Koordinaten des Achs-Koordinatensystems umzurechnen ([Transformationen auf Seite 28](#)).

Die Methode erhält dazu zwei Referenzen auf die Koordinaten des Basis-Koordinatensystems und die Koordinaten des Achs-Koordinatensystems.

Die Methode muss aus den Koordinaten des Basis-Koordinatensystems die Koordinaten des Achs-Koordinatensystems errechnen.

Kann ein Punkt nicht transformiert werden, muss die Methode einen MLC-Fehlercode zurückgeben.

- **GetMechanic:** Methode, um mechanische Informationen für einen Punkt im ACS von der Transformation zu erhalten.

Die Methode erhält dazu

- Eine Referenz auf einen Punkt im ACS-Koordinatensystem

Transformationsschemen

- Eine Referenz auf eine Struktur, in die die mechanischen Informationen hinterlegt werden sollen
- Die Anzahl der freien Elemente in dieser Struktur
- Eine Referenz auf den Wert, der angibt, wieviele Elemente von dieser Struktur verwendet wurden
- Sowie die Information, ob nur geänderte Werte zurückgegeben werden sollen

Zum Ausfüllen der Struktur steht die Hilfsfunktion ML_AddMechanicData, ML_Robot zur Verfügung.

Fehlerhandling der Methoden

Tritt bei einer der Methoden ein Floating Point-Fehler auf, so wird der Fehler automatisch gemeldet.

Solche Fehler sollten allerdings bereits in der Mathematik der Transformation berücksichtigt werden: So sollten z.B. ein Divisor immer auf Null überprüft, bei Gleichheit ein Fehler zurückgegeben und die Division nicht durchgeführt werden.

- F12C0600: Ungültige Operation beim SPS-Funktionsaufruf
- F12C0601: Denormalisierte Zahl beim SPS-Funktionsaufruf
- F12C0602: Division durch Null beim SPS-Funktionsaufruf
- F12C0603: Überlauf beim SPS-Funktionsaufruf
- F12C0604: Unterlauf beim SPS-Funktionsaufruf
- F12C0605: Genauigkeitsfehler beim SPS-Funktionsaufruf

Anders als zur einfachen benutzerdefinierten Transformation kann bei der erweiterten benutzerdefinierten Transformation als Rückgabewert der Funktionen ein MLC-Fehlercode zurückgegeben werden. Dadurch ist für den Anwender eine einfachere Diagnose möglich. Ein MLC-Fehlercode hat die Form 16#Fxxxxxx. Wurde eine Schnittstellenfunktion fehlerfrei ausgeführt, so muss ein Code der Form 16#0xxxxxx zurückgegeben werden.

Die wichtigsten Fehlercodes für eine Transformation:

- F22B0021: Achswinkel außerhalb Definitionsbereich
- F12C0479: WC-Nullpunkt singulär
- F12C047A: Zu nahe am Stativ
- F22B0022: Punkt kann durch Transformation nicht erreicht werden
- F12C008A: Transformationsparameter passen nicht zum Robotertyp
- F22B0023: In Schutzzone der Transformation
- F22B0024: Achse hat falsche Wichtung für Transformation
- F22B0025: Zu wenig Achsen für Transformation

Anmelden einer Transformation

Wurde ein Funktionsbaustein implementiert, der eine Transformation darstellt (der FB implementiert das Interface MC_KIN_REF_EXT, ML_Robot), so kann dieser FB mit dem Funktionsbaustein MC_SetKinTransformExt, ML_Robot, am System angemeldet werden.

Für die Anmeldung muss eine Instanz des eigenen Transformationsbausteins und die Kinematik angegeben werden, für die die Instanz gelten soll.

Das Anmelden der Transformation muss in der Konfigurationsphase der Steuerung (P2) erfolgen.

Ist beim Phasenschalten in den Betriebszustand der Steuerung (BB) in der Kinematik eine benutzerdefinierte Transformation definiert, jedoch keine

Transformation angemeldet, so wird der Fehler "F12C0498: Transformation in der SPS wurde nicht angemeldet" gemeldet.

Soll in mehreren Kinematiken die gleiche Transformation verwendet werden, ist es notwendig, jeweils eine eigene Instanz des selbst erstellten Transformations-Funktionsbausteins für die jeweilige Kinematik zu verwenden.

Anwendungsbeispiel

Ein Anwendungsbeispiel für das Erstellen und die Anwendung von benutzerdefinierter Transformationen findet man unter [Erweiterte benutzerdefinierte Transformation auf Seite 405](#).

4.5 Mechanische Informationen

Bei der Konfiguration einer Kinematik werden verschiedene mechanische Daten über das System mitgeteilt. Diese sind z.B. Armlängen der Kinematik oder die Lage der Kinematik im Raum.

Diese Daten werden nicht nur für den Betrieb der Anlage benötigt, aus ihnen läßt sich auch eine generisches Visualisierung der Anlage generieren.

Dazu bietet das System die Möglichkeit, sich die mechanischen Informationen in komprimierter Form ausgeben zu lassen.

Die mechanischen Informationen können durch [Get Mechanic auf Seite 38](#) ausgelesen werden.

Folgende Daten werden ausgelesen:

- **ObjektID:** textueller Name des Objekts.
- **Anweisung:** Anweisung, die zur Visualisierung auf das Objekt angewendet werden soll.
- **X, Y, Z:** Daten für die Anweisung.

Folgende Anweisungen können auftreten:

- **Erzeuge Stab:** das Objekt mit der ObjektID ist ein Zylinder mit der Länge 1 und dem Radius 1.
Der Mittelpunkt einer Kreisfläche des Zylinders liegt im Nullpunkt, der Zylinder zeigt in positive X-Richtung. X, Y, Z werden ignoriert.
- **Erzeuge Kugel:** das Objekt mit der ObjektID ist eine Kugel mit der Länge 1 und dem Radius 1.
Der Mittelpunkt der Kugel liegt im Nullpunkt. X, Y, Z werden ignoriert.
- **Ändere Position:** die neue Position des Objekts mit der ObjektID ist (x,y,z).
- **Ändere Größe:** die neue Größe des Objekts mit der ObjektID ist (x,y,z).
- **Ändere Richtung:** die neue Richtung des Objekts mit der ObjektID ist (x,y,z).

[Get Mechanic auf Seite 38](#) kann als Zusatzinformation erhalten, ob alle Daten oder nur veränderte Daten zurückgegeben werden sollen (updateOnly).

Auf diese Weise kann die Menge der zu übertragenden Daten drastisch reduziert werden.

Typischerweise sollten einmalig alle Daten erfragt und danach nur noch geänderten Daten übertragen werden.

Beispiel

Der Aufruf von [Get Mechanic auf Seite 38](#) liefert folgende Daten:

Transformationsschemen

Beispiel mechanische Daten

ObjektID	Anweisung	x	y	z	Bedeutung
TCP	Kugel				Es existiert eine Kugel mit dem Namen "TCP"
S1	Stab				Es existiert ein Stab mit dem Namen "S1"
S2	Stab				Es existiert ein Stab mit dem Namen "S2"
K12	Kugel				Es existiert eine Kugel mit dem Namen "K12"
TCP	Größe	3	3	3	Ändere die Größe von "TCP" auf (3,3,3)
S1	Größe	100	2	2	Ändere die Größe von "S1" auf (100,2,2)
S2	Größe	100	2	2	Ändere die Größe von "S2" auf (100,2,2)
S1	Richtung	70.7	70.7	0	Ändere die Richtung von "S1" auf (70.7,70.7,0)
S2	Richtung	-70.7	-70.7	0	Ändere die Richtung von "S2" auf (-70.7,-70.7,0)
S2	Position	70.7	70.7	0	Ändere die Position von "S2" auf (70.7,70.7,0)
K12	Position	70.7	70.7	0	Ändere die Position von "K12" auf (70.7,70.7,0)
TCP	Position	141.4	0	0	Ändere die Position von "TCP" auf (141.4,3,3)

Tab. 4-67: Beispiel mechanische Daten

Diese Daten können wie folgt visualisiert werden:

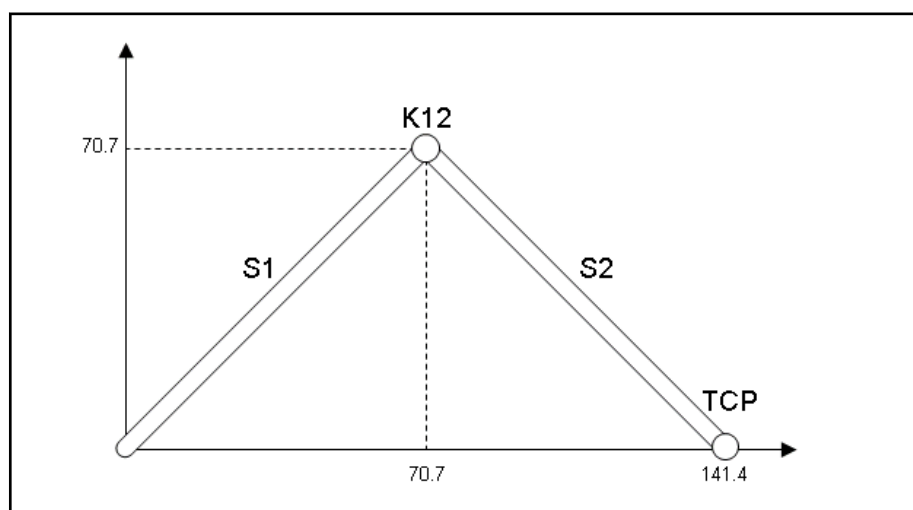


Abb. 4-65: Visualisierte Daten von GetMechanic

Ruft man die Funktion [Get Mechanic auf Seite 38](#) mit "updateOnly" auf, so liefert sie nur Daten von veränderlichen Anweisungen:

Beispiel mechanische Daten

ObjektID	Anweisung	x	y	z	Bedeutung
S1	Richtung	70.7	70.7	0	Ändere die Richtung von "S1" auf (70.7,70.7,0)
S2	Richtung	-70.7	-70.7	0	Ändere die Richtung von "S2" auf (-70.7,-70.7,0)
S2	Position	70.7	70.7	0	Ändere die Position von "S2" auf (70.7,70.7,0)
K12	Position	70.7	70.7	0	Ändere die Position von "K12" auf (70.7,70.7,0)
TCP	Position	141.4	0	0	Ändere die Position von "TCP" auf (141.4,3,3).

Tab. 4-68: Beispiel mechanische Daten

5 ML_Robot.library

5.1 Einführung und Übersicht



Die Bibliothek benötigt das ROB (Robotik) Funktionspaket der Steuerung.

Die ML_Robot-Bibliothek enthält die für die Realisierung der Kinematik-Funktionen des Systems IndraMotion MLC wesentlichen Datentypen, Funktionen und Funktionsbausteine zur Bewegungssteuerung (Coordinated Motion).

Für jeden Steuerungstyp unterscheiden sich einige Teile der ML_Robot-Bibliothek. IndraLogic 2G, Version 14 erkennt jedoch die Hardware und liefert die richtige Bibliothek.



Diese Dokumentation enthält nur die Schnittstellenbeschreibung für ML_Robot-Bibliothek.

Die Beschreibung der Funktionalität hinter der Schnittstelle finden Sie unter [Funktionsbeschreibung auf Seite 23](#).

Die Bibliothek weist folgende Struktur auf:

- Datentypen der ML_Robot Bibliothek, Data Types ([Datentypen, Übersicht auf Seite 303](#)):

Data Types

- **Config:** Datentypen für Funktionsbausteine zur Konfiguration einer Kinematik
- **Info:** Datentypen für Funktionsbausteine zur Informationsgewinnung über eine Kinematik
- **Movement:** Datentypen für Funktionsbausteine zum Bewegen einer Kinematik
- **Teach:** Datentypen für Funktionsbausteine zum Teachen
- Definitionen für Globale Variablen ([Globale Variablen auf Seite 186](#)):

Global Variables List

- Shared Memory (**GVL_Robot**)
- Globale Konstanten (**GVL_RobotConstant**)
- Direktvariablen

- Internal (Do not use!): Interne Definitionen.

- Funktionsbausteine zur freien Verwendung ([Funktionsbausteine, Übersicht auf Seite 190](#)):

Kinematic

- **Config:** Funktionsbausteine zur Konfiguration einer Kinematik
- **Info:** Funktionsbausteine für Informationen über eine Kinematik
- **Movement:** Funktionsbausteine für Bewegungen einer Kinematik
- **Parameter:** Funktionsbausteine für Parameterzugriff auf eine Kinematik
- **TeachIn:** Funktionsbausteine zum Speichern und Wiederverwenden von Positionen einer Kinematik als Punkte, nur zur Verwendung in Robot-Control V2

- **Projektinformationen:** Funktionen zum Zugriff auf Informationen über diese Bibliothek

5.2 Globale Variablen

5.2.1 Einführung und Übersicht



Die Bibliothek ML_Robot benötigt das ROB (Robotik) Funktionspaket der Steuerung

Beim Erzeugen einer neuen Kinematik in IndraWorks werden Geräte- und Antriebsname sowie zugeordnete Antriebe abgefragt.

Der Kinematik wird automatisch eine Kinematiknummer zugeordnet.

Mit Hilfe der automatisch erzeugten Kinematiknummer

`MB_MIN_KIN_NUMBER..MB_MAX_KIN_NUMBER`

(siehe [GVL_RobotConstant](#), [Globale Konstanten auf Seite 187](#))

kann indiziert auf Kinematikdaten zugegriffen werden über

- **Funktionsvariablenzugriff** (siehe [DV_Kin\[\] - Kinematikdaten über Funktionsvariablenzugriff auf Seite 188](#))
- **Shared Memory-Zugriff** (siehe [KinData\[\] - Kinematikdaten im Shared Memory auf Seite 187](#))



Kinematikdaten im Shared Memory werden im Motion-Modus "Betriebsmodus" zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt.

Kinematikdaten über Funktionsvariablenzugriff (alias "Direktvariablen") werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.

Kinematikdaten über Shared Memory und Funktionsvariablen können sich, abhängig von den Taskeigenschaften der POU, über mehrere Zyklen der Motion (Sollwertgenerierung, Istwertbereitstellung) verteilen und sind somit nicht zwingend konsistent zueinander.

Ein Teil der Kinematikdaten stehen sowohl im Shared Memory als auch über Funktionsvariablenzugriff zur Verfügung.

Da Kinematikdaten im Shared Memory unabhängig von der Nutzung zyklisch ausgetauscht werden, sollten weitere redundante Zugriffe über Funktionsvariablen aus Performancegründen vermieden werden.



Der vereinfachte, blockierende Zugriff auf eine Funktionsvariable ist ein impliziter Funktionsaufruf und benötigt mehr Laufzeitressourcen als ein einfacher Speicherzugriff auf eine lokale SPS-Variable. Es wird empfohlen, einen mehrfachen Zugriff auf eine Funktionsvariable zu vermeiden und bei Bedarf das Betriebsdatum auf eine lokale SPS-Variable zu speichern.

Kinematikdaten über Shared Memory und über Funktionsvariablenzugriff können sich, abhängig von den Taskeigenschaften der POU, über mehrere Zyklen der Motion (Sollwertgenerierung, Istwertbereitstellung) verteilen und sind somit nicht zwingend konsistent zueinander.



Eine Funktionsvariable ist kein Bestandteil des Prozessabbildes des SPS-Tasks. Der vereinfachte, blockierende Zugriff auf eine Funktionsvariable ist ein impliziter Funktionsaufruf, welcher zum Aufrufzeitpunkt ausgeführt wird. Ein mehrfacher Zugriff auf eine Funktionsvariable im Laufe eines SPS-Programms (POU) kann zu unterschiedlichen Ergebnissen führen, da durch zwischenzeitliche Taskwechsel das Betriebsdatum modifiziert werden kann.

5.2.2 GVL_RobotConstant, Globale Konstanten

Die Globale Konstantenliste "GVL_RobotConstant" definiert die für die jeweilige Steuerungshardware gültigen Konstanten.

Variablenname	Typ	Beschreibung	MLC			
			L25	L45/ L65/ L75	XM2	VPx
MB_MIN_KIN_NUMBER	INT	Kleinste Kinematiknummer	1			
MB_MAX_KIN_NUMBER	INT	Größte Kinematiknummer	4	16	16	16
MB_NUMBER_OF_KIN	INT	MB_MAX_KIN_NUMBER - MB_MIN_KIN_NUMBER + 1	4	16	16	16
MB_MAX_KIN_AXIS_NUMBER	INT	Maximale Anzahl von Achsen pro Kinematik	16			
MB_MAX_COORDINATES_NUMBER	INT	Maximale Anzahl von Koordinaten in einem kartesischem Punkt	6			
ML_MAX_KIN_ARRAY_DIMENISINS	INT	Maximale Anzahl von Dimensionen für ein Punktefeld	5			
ML_MAX_IDX_ARRAY_USINT	UDINT		255			
KIN_OBJ_2_OBJECT	Array [0..16]] of OBJECTS		[0,1,2,3,4,5,6,7,8,9, 10,11,12,13,14,15,16]			

Tab. 5-1: Globale Konstanten der ML_Robot-Bibliothek

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

5.2.3 KinData[] - Kinematikdaten im Shared Memory

Kurzbeschreibung Kinematikdaten im Shared Memory sind als globales Array "KinData[]" der Datenstruktur "ML_KINEMATICSDATA_SM" deklariert. Die Definition ist Bestandteil der globalen Variablenliste "GVL_Robot" der Bibliothek "ML_Robot".

Definition des globalen Array KinData[]

```
VAR_GLOBAL
...
KinData AT %MBaaa: ARRAY [MB_MIN_KIN_NUMBER..MB_MAX_KIN_NUMBER] OF ML_KINEMATICSDATA_SM;
...
END_VAR
```

ML_Robot.library



Die Adressen "%MBaaa" von KinData entnehmen Sie bitte dem Bibliotheksverwalter der ML_Robot, GVL_Robot Dokumentation.

Siehe auch

- [ML_KINEMATICSDATA_SM auf Seite 312](#)

Funktionsbeschreibung

Der lesende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Lesender Zugriff auf Daten im Shared Memory:

```
MyVar:= KinData[MyKinName.GroupNo].Kin_InPosition;
MyVar:= KinData[Index].Kin_InPosition;
MyVar:= KinData[1].Kin_InPosition;
```

Der schreibende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Schreibende Zugriff auf Daten im Shared Memory:

```
KinData[MyKinName.GroupNo].stJogCmd_q:= MyVar;
KinData[Index].stJogCmd_q:= MyVar;
KinData[1].stJogCmd_q:= MyVar;
```

Der IntelliSense einer Variablen im Shared Memory enthält die Parameteridentnummer und ggf. das Bit, die Zugriffsmöglichkeiten (RD bzw. RW) sowie den Parameternamen und ggf. die Bedeutung des Bits.

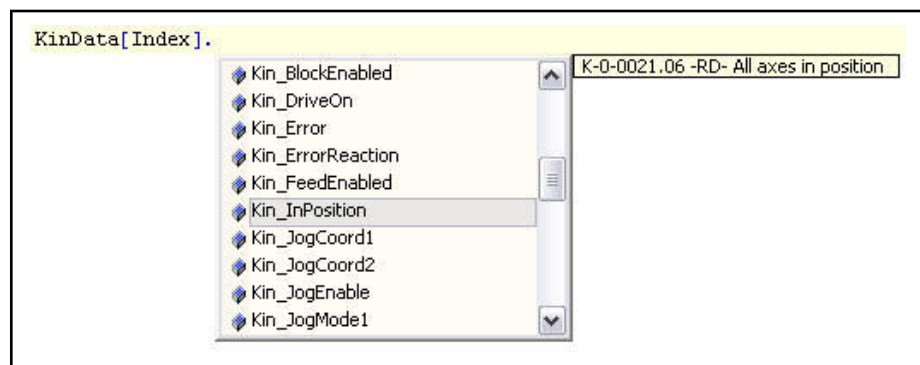


Abb. 5-1: IntelliSense einer Variablen im Shared Memory



Auf Variablen einer Kinematik im Shared Memory kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Benutzerdefinierte Bezeichnung durch Referenzierung:

```
VAR
    MyKinName: REFERENCE TO ML_KINEMATICSDATA_SM:= KinData[MyKinName.GroupNo];
END_VAR

MyVar:= MyKinName.Kin_InPosition;
MyKinName.stJogCmd_q:= MyVar;
```

5.2.4 DV_Kin[] - Kinematikdaten über Funktionsvariablenzugriff

Kinematikdaten über Funktionsvariablenzugriff sind als globales Array "DV_Kin[]" des Funktionsbausteins "ML_DirectVarKin" deklariert.

Die Definition ist Bestandteil der globalen Variablenliste "GVL_Robot" der Bibliothek "ML_Robot".

Der Funktionsbaustein "ML_DirectVarKin" stellt den Funktionsvariablenzugriff als Methoden bereit, die Existenz der Properties GET und SET der jeweiligen Methode definiert die Zugriffsmöglichkeit. Die Ausführung des Schreibzugriffes SET auf eine Funktionsvariable ist u. a. abhängig vom Motion-Modus und erzeugt im negativen Fall keine Fehlermeldung.



Der Funktionsbaustein "ML_DirectVarKin" ist nicht für die direkte Verwendung im Anwenderprogramm vorgesehen.

Auszug aus GVL_Robot

```
VAR_GLOBAL
...
  DV_Kin: ARRAY [MB_MIN_KIN_NUMBER..MB_MAX_KIN_NUMBER] OF ML_DirectVarKin;
...
END_VAR
```

Funktionsbeschreibung

Der lesende Zugriff auf eine Direktvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Lesender Zugriff auf eine Funktionsvariable

```
MyVar:= DV_Kin[MyKinName.GroupNo].K_0_0022;
MyVar:= DV_Kin[Index].K_0_0022;
MyVar:= DV_Kin[1].K_0_0022;
```

Der schreibende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Schreibender Zugriff auf eine Funktionsvariable

```
DV_Kin[MyKinName.GroupNo].K_0_0241:= MyVar;
DV_Kin[Index].K_0_0241:= MyVar;
DV_Kin[1].K_0_0241:= MyVar;
```

Der IntelliSense einer Funktionsvariablen enthält die Parameter-Identnummer, die Zugriffsmöglichkeiten (READ bzw. READ/WRITE) und den Parameternamen.

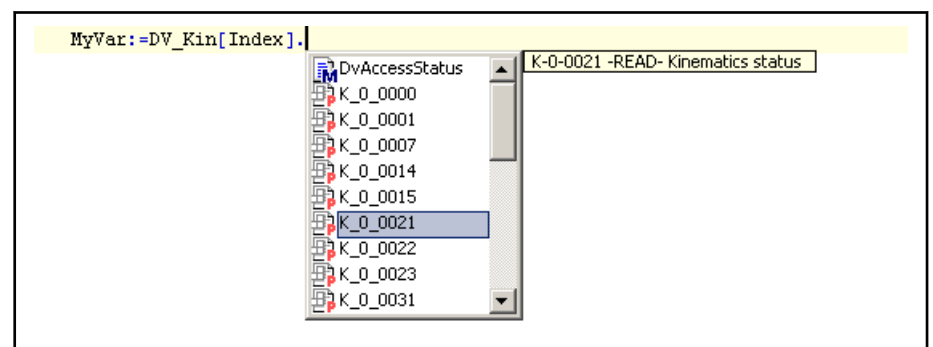


Abb. 5-2: Funktionsvariablenzugriff unter Nutzung der IntelliSense-Funktionalität



Auch wenn der Zugriff auf Funktionsvariablen wie ein Variablenzugriff aussieht, verbirgt sich dahinter ein komplexer Funktionsaufruf, der bedeutend länger als ein Variablenzugriff dauert.

ML_Robot.library



Auf Direktvariablen einer Kinematik kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Zugriff über REFERENCE TO

```
VAR
MyKinName: REFERENCE TO ML_DirectVarKin:= DV_Kin[MyKinName.GroupNo];
END_VAR

MyVar:= MyKinName.K_0_0022;
MyKinName.K_0_0241:= MyVar;
```

Status von Parameterwerten bei SPS-Zugriff über Direktvariablen

Die Funktionsbausteine für Funktionsvariablenzugriff bieten neben den Properties zum Lesen und Schreiben von Betriebsdaten auch die Möglichkeit, den **aktuellen Zugriffstatus** eines Parameters über die Methode **DvAccess-Status** zu ermitteln.

Der Methode wird die entsprechende Parameterkonstante übergeben.

Die Methode gibt den Rückgabewert des letzten Zugriffs (== aktueller Zugriffsstatus) zurück.

Das Lesen des aktuellen Zugriffstatus einer Funktionsvariablen erfolgt mittels folgender Anweisung (ST):

Lesen des aktuellen Zugriffstatus einer Funktionsvariablen

```
K_Status:= DV_Kin[Index].DvAccessStatus(FP_K_0_0015);
```

Vereinbarungsdateien für Globale Konstanten

- [Zugriff auf Daten einer Kinematik auf Seite 282](#)

Status (Auszug, weitere siehe SERCOS_TABLE)

- 16# 0000.0000: Kein Fehler
- 16# F02D1001: Parameter IDN nicht vorhanden
- 16# F02Dxxxx: Sercos Fehlercodes
- 16# F02D7013: Kommando nicht ausführbar (FW: Kommando nicht implementiert)
- 16# F02D8001: Zugriff auf Parameter aktuell nicht möglich (BUSY)

5.3 Funktionsbausteine

5.3.1 Funktionsbausteine, Übersicht



Die Bibliothek benötigt das ROB (Robotik)-Funktionspaket der Steuerung

Funktionsbausteine der ML_Robot:

Anwendungsgebiet (Ordner in der Bibliothek)	
Funktionsbaustein	Beschreibung
Funktionsbausteine zur Konfiguration einer Kinematik (Kinematic / Config)	
MC_GroupReset, Seite 208	Setzt alle anstehenden Fehler einer Kinematik zurück

MC_SetCartesianTransform, Seite 209	Definiert ein kartesisches Produkt-Koordinatensystem
MC_SetKinTransform, Seite 211	Meldet eine benutzerdefinierte Transformation für eine Kinematik an
MC_SetKinTransformExt, Seite 213	Meldet eine Erweiterte Benutzerdefinierte Transformation für eine Kinematik an
MC_UngroupAllAxes, Seite 214	Liest K-0-0008 und entfernt alle Achsen einer Kinematik
ML_AddMechanicData, Seite 215	Fügt einen Eintrag in die ML_KIN_MECHANIC_DATA-Struktur hinzu
ML_GroupAllAxes, Seite 216	Liest K-0-0008 der ausgewählten Kinematik und fügt alle Achsen der Kinematik hinzu
ML_GroupSetSafeZone, Seite 218	Konfiguriert eine Schutzzone einer Kinematik
ML_RobotSetMoveID, Seite 227	Ändert die aktuelle Bewegungsidentifikationsnummer
ML_SetBeltConfiguration, Seite 220	Konfiguriert ein Band
ML_SetBeltDesyncRelConfiguration, Seite 219	Konfiguriert das relative Absynchronisieren eines Bandes
ML_SetBeltErrorReaction, Seite 222	Konfiguriert die Fehlerreaktion eines Band
ML_SetCylindricTransform, Seite 223	Definiert ein zylindrisches Produkt-Koordinatensystem
ML_SetRotaryTableConfiguration, Seite 225	Konfiguriert eine Kreisscheibe
Funktionsbausteine zur Informationsgewinnung über eine Kinematik (Kinematic / Info)	
MC_GroupReadActualPosition, Seite 228	Liest die aktuelle Position der Kinematik
ML_RobotGetBeltConfiguration, Seite 229	Gibt die aktuelle Konfiguration eines Bandes zurück
ML_RobotGetBeltDesyncRelConfiguration, Seite 231	Liest die Konfiguration für das relative Absynchronisieren eines Bandes
ML_RobotGetBeltErrorReaction, Seite 232	Gibt die konfigurierte Bandfehlerreaktion des Roboters zurück
ML_RobotGetCmdInfo, Seite 233	Gibt Informationen zu allen Kommandos im Satzpuffer zurück
ML_RobotGetMechanic, Seite 234	Gibt mechanische Daten der Kinematik zurück
ML_RobotGetMoveID, Seite 236	Gibt die aktuelle Bewegungsidentifikationsnummer zurück
ML_RobotGetPerformance, Seite 237	Gibt Performance-Daten der Kinematik zurück
ML_RobotGetSafeZones, Seite 238	Gibt die Schutzzone einer Kinematik zurück
ML_RobotGetTrafoValues, Seite 239	Gibt die aktuelle Konfiguration eines Produkt- oder des Basis-Koordinatensystems zurück
ML_RobotMovePoint, Seite 241	Gibt die aktuelle Position eines Punktes im Raum zurück
ML_RobotReadCyclicChannel, Seite 242	Liest die aktuelle zyklisch kommandierte Position
ML_RobotSetPerformanceEnable, Seite 244	Ändert den Aktivierungszustand der Performance-Messung
ML_RobotTransformPoint, Seite 245	Transformiert einen Punkt von einem Koordinatensystem in ein anderes Koordinatensystem
ML_SwitchOnPath, Seite 246	Bietet die Möglichkeit, auf einer Bahn zu schalten
Funktionsbausteine zum Bewegen einer Kinematik (Kinematik / Movement)	
MC_GroupContinue, Seite 247	Startet eine durch den Funktionsbaustein MC_GroupInterrupt unterbrochene Bewegung einer Kinematik wieder
MC_GroupInterrupt, Seite 249	Unterbricht die Bewegung einer Kinematik
MC_GroupStop, Seite 250	Stoppt kontrolliert eine Kinematik

ML_Robot.library

MC_MoveCircularAbsolute, Seite 251	Bewegt eine Kinematik zirkular von der zuletzt kommandierten Position des TCP über einen Zwischenpunkt zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem
MC_MoveCircularRelative, Seite 254	Bewegt eine Kinematik zirkular von der zuletzt kommandierten Position des TCP über einen Zwischenpunkt zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem
MC_MoveDirectAbsolute, Seite 257	Bewegt eine Kinematik Punkt-zu-Punkt von der zuletzt kommandierten Position des TCP zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem
MC_MoveDirectRelative, Seite 259	Bewegt eine Kinematik Punkt-zu-Punkt von der zuletzt kommandierten Position des TCP zu einer vorgegebenen relativen Zielposition im angegebenen Koordinatensystem
MC_MoveLinearAbsolute, Seite 261	Bewegt eine Kinematik linear von der zuletzt kommandierten Position des TCP zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem
MC_MoveLinearRelative, Seite 264	Bewegt eine Kinematik linear von der zuletzt kommandierten Position des TCP zu einer vorgegebenen relativen Zielposition im angegebenen Koordinatensystem
ML_BeltDeSynchronisationWithLimits, Seite 266	Synchronisiert mit Bewegungsgrenzwerten von einem Band ab
ML_BeltSynchronisationWithLimits, Seite 267	Synchronisiert mit Bewegungsgrenzwerten auf ein Band auf
ML_GroupJogCont, Seite 269	Bewegt eine Kinematik kontinuierlich in eine Richtung
ML_GroupJogStep, Seite 271	Bewegt eine Kinematik einen Schritt in eine bestimmte Richtung
ML_GroupWait, Seite 273	Kinematik wartet die angegebene Zeit auf der letzten kommandierten Position
ML_RobotMoveJumpAbsolute, Seite 279	Bewegt eine Kinematik mit einer Jump-Bewegung über Start-, Maximal- und Endhöhe von der zuletzt kommandierten Position des TCP zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem
ML_RobotOpenCyclicChannel, Seite 275	Öffnet einen zyklischen Sollwertkanal
ML_RobotOpenDirectCyclicChannel, Seite 276	Öffnet einen zyklischen Sollwertkanal für PTP Bewegungen
ML_RobotWriteCyclicChannel, Seite 277	Schreibt einen zyklischen Sollwert
Funktionsbausteine zum Parameterzugriff einer Kinematik (Kinematik / Parameter)	
MB_GroupReadListParameter, Seite 283	Gibt den Wert eines durch den Anwender vorgegebenen Listen-Parameters zurück
MB_GroupReadParameter, Seite 284	Gibt den Wert eines durch den Anwender vorgegebenen DINT-Parameters zurück
MB_GroupReadRealParameter, Seite 285	Gibt den Wert eines durch den Anwender vorgegebenen REAL-Parameters zurück
MB_GroupReadSercosAttribute, Seite 286	Erlaubt das Lesen der Sercos Attribute eines K-Parameters
MB_GroupReadSercosDataStatus, Seite 288	Gibt den Status eines durch den Anwender vorgegebenen K-Parameters zurück
MB_GroupReadStringParameter, Seite 289	Gibt den Wert eines durch den Anwender vorgegebenen STRING-Parameters zurück

MB_GroupWriteListParameter, Seite 290	Schreibt Speicherinhalt definierter Länge (NoOfByte), der ab der Anfangsadresse (ValueAdr) zur Verfügung gestellt wurde, in einen durch den Anwender vorgegebenen Parameter
MB_GroupWriteParameter, Seite 291	Schreibt einen DINT-Wert in einen durch den Anwender vorgegebenen Parameter
MB_GroupWriteRealParameter, Seite 292	Schreibt einen REAL-Wert in einen durch den Anwender vorgegebenen Parameter
MB_GroupWriteStringParameter, Seite 293	Schreibt einen STRING-Wert in einen durch den Anwender vorgegebenen Parameter
Funktionsbausteine zum Speichern und Wiederverwenden von Positionen einer Kinematik als Punkte (Kinematic / TeachIn)	
ML_GroupDeletePointFile, Seite 295	Löscht eine Punktedatei einer Kinematik
ML_GroupDeletePoints, Seite 296	Löscht geteachte Punkte einer Kinematik
ML_GroupLoadPointFile, Seite 297	Lädt eine Punktedatei einer Kinematik
ML_GroupReadNextPoint, Seite 298	Sucht den nächsten geteachten Punkt und gibt die Daten des nächsten Punkts einer Kinematik aus
ML_GroupReadPoint, Seite 299	Gibt die geteachten Daten eines Punkts einer Kinematik aus
ML_GroupSavePointFile, Seite 301	Speichert eine Punktedatei einer Kinematik
ML_GroupWritePoint, Seite 302	Teacht einen Punkt einer Kinematik

Tab. 5-2: Funktionsbausteine der Bibliothek ML_Robot

5.3.2 Standardschnittstellen an Funktionsbausteinen

Motivation

Die Mehrzahl der Funktionsbausteine hat einen Eingang zur Aktivierung und einen Ausgang, der die fehlerfreie Bearbeitung anzeigt.

Zusätzlich wird oft ein Ausgang benötigt, der den Bearbeitungszeitraum kennzeichnet.

Zum Anzeigen von Fehlern werden weitere Ausgänge definiert.

Eine einheitliche Namensgebung sowie gleiches Verhalten dieser Standardschnittstellen erhöht das Verständnis, verkürzt die Einarbeitung und entlastet den Support.

Bausteintypen

Mit Funktionsbausteinen ist es möglich, komplexe Aufgaben wiederverwendbar zu kapseln und diese Aufgaben über definierte Schnittstellen anzusprechen. Dabei kann die Bearbeitung entweder zustands- oder flankengesteuert erfolgen. Zudem unterscheidet man zwischen Aufträgen, die abschließend behandelt werden können und solchen die einmal angestoßen einen permanent Eingriff erfordern.

Zustandsgesteuert Wiederholt ein Funktionsbaustein nach dem Einschalten seinen Auftrag immer dann, wenn der Funktionsbaustein einen definierten Zustand erreicht hat, arbeitet der Funktionsbaustein "zustandsgesteuert".

Flankengesteuert Erfüllt ein Funktionsbaustein nach dem Einschalten seine Aufgabe genau einmal, so arbeitet der Funktionsbaustein "flankengesteuert".

Abschließende Bearbeitung Kann ein Baustein seinen Auftrag endgültig bearbeiten und seine Arbeit anschließend einstellen, so ist dies eine "abschließende Bearbeitung".

ML_Robot.library

Permanente Bearbeitung Kann ein Baustein seinen Auftrag nie endgültig abschließen, sondern ist dauerhaft im Eingriff, wird dies als "permanente Bearbeitung" definiert.

In Kombination ergeben sich somit vier mögliche Bausteintypen:

Steuerung	Bearbeitung	Beispiel
Zustandsgesteuert (Enable)	Abschließend (Done)	Dauerhaftes Lesen eines Parameters Ist das Lesen abgeschlossen, hat der Baustein seinen Auftrag beendet. Der Baustein wiederholt diesen Auftrag solange, wie der Steuereingang (Enable) gesetzt bleibt, z. B. MB_ReadParameter
Zustandsgesteuert (Enable)	Permanent (In....)	Beendbare Regelung Ist die Regelung aktiv, arbeitet die Regelung solange, bis der Steuereingang (Enable) zurückgesetzt wird und die Regelung damit deaktiviert
Flankengesteuert (Execute)	Abschließend (Done)	Einmaliges Schreiben eines Parameters Ist das Schreiben abgeschlossen, hat der Baustein seinen Auftrag beendet. Es ist eine erneute Flanke am Steuereingang (Execute) notwendig, um ein erneutes Schreiben zu starten, z. B. MB_WriteParameter
Flankengesteuert (Execute)	Permanent (In....)	Nicht beendbare Regelung Ist die Regelung einmal über den Steuereingang (Execute) gestartet, kann die Regelung nicht mehr über die Schnittstellen dieses Bausteines deaktiviert werden, z. B. MC_MoveVelocity

Tab. 5-3: Übersicht Bausteintypen

Bausteintypen und ihre Schnittstellen

Eingänge Um nach außen kenntlich zu machen, ob ein Funktionsbaustein zustands- oder flankengesteuert arbeitet, werden zwei verschiedene Variablennamen für die Eingänge zur Aktivierung des Funktionsbaustein verwendet.

- Enable = zustandsgesteuert
- Execute = flankengesteuert

Ausgänge Um nach außen kenntlich zu machen, ob ein Funktionsbaustein abschließend oder permanent arbeitet, werden verschiedene Variablennamen für die Ausgänge verwendet, welche den Bearbeitungszustand kennzeichnen.



- Done = Abschließende Bearbeitung
- In..... = Permanente Bearbeitung (z.B. InOperation, InSync, InVelocity, InGear, InTorque...)

Der optionale Ausgang Active kann bei abschließend und permanent arbeitenden Bausteinen verwendet werden, hat aber jeweils ein anderes Verhalten!

Alle Ausgänge zur Fehlerkennzeichnung (Error, ErrorID, ErrorIdent) finden sich bei abschließend und permanent arbeitenden Bausteinen und haben bei beiden das gleiche Verhalten.

Standardisierte Bezeichner

Bezeichner	Beschreibung
Eingänge	
Enable	Aktivierungseingang zustandsgesteuerter FBs
Execute	Aktivierungseingang flankengesteuerter FBs. Andere FBs können diesen FB unterbrechen

Bezeichner	Beschreibung
ExecuteLock	Aktivierungseingang flankengesteuerter FBs. Solange dieser Eingang TRUE ist, kann der FB nicht durch andere unterbrochen werden
Ausgänge	
Active	Ausgang, der den Bearbeitungszeitraum kennzeichnet
Done	Bearbeitung erfolgreich beendet und die Datenausgänge sind gültig
In.....	Ausgang signalisiert, dass der Baustein an seiner Aufgabe arbeitet und die Datenausgänge gültig sind
Shutdown (optional)	Ausgang signalisiert, dass sich der Baustein gerade in einen definierten Endzustand bringt (z. B. Achsen stoppt, Ressourcen freigibt, ...). Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird
CommandAborted	Ausgang zeigt an, dass der FB unterbrochen wurde (z. B. von einem anderen FB)
Error	Bearbeitung mit Fehler beendet
ErrorID	Ausgang für grobe Fehlerklassifikation
ErrorIdent	Ausgang für detaillierte Fehlerklassifikation

Tab. 5-4: Übersicht standardisierter Bezeichner

ML_Robot.library

Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung

Steuerung / Bearbeitung	I/O	Variablenname	Beschreibung
Zustandsge- steuert / ab- schließend	I	Enable	Mit einer positiven Flanke von "Enable" werden die Eingangsvariablen erfasst. Neue Eingangswerte werden erst über eine erneute positive Flanke von "Enable" wirksam. "Enable" muss während der gesamten Bausteinbearbeitung TRUE sein! Das Deaktivieren von "Enable" unterbricht die Bearbeitung und setzt die Ausgänge "Done", "Active", "CommandAborted" und "Error" auf FALSE. Falls notwendig wird über "Shutdown" signalisiert, dass weitere Aufrufe notwendig sind, um den Baustein in einen definierten Endzustand zu bringen
	O	Done	Ist "Done" TRUE, hat der Funktionsbaustein seinen Auftrag erfolgreich beendet und ist in einem Endzustand. Datenausgänge sind jetzt gültig. "Active", "Error" und "CommandAborted" sind FALSE! "Done" bleibt für genau einen SPS-Zyklus TRUE. Ist anschließend "Enable" noch TRUE, startet der Baustein seine Bearbeitung erneut
	O	Active	Ist "Active" TRUE, arbeitet der Funktionsbaustein an seiner eigentlichen Aufgabe und ist in einem Zwischenzustand. Eventuelle Vorverarbeitungen werden nicht mit diesem Ausgang gekennzeichnet! "Done", "Error" und "CommandAborted" sind FALSE!
	O	Shutdown (optional)	Ist ein "Shutdown"-Ausgang vorhanden, benötigt der Baustein beim Deaktivieren von "Enable" noch mehrere Zyklen an Rechenzeit, um sich in einen definierten Endzustand zu bringen und verwendete Ressourcen freizugeben. Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird!
	O	CommandAborted	Ist "CommandAborted" TRUE, wurde der Funktionsbaustein unterbrochen und befindet sich in einem Endzustand. "Done", "Active" und "Error" sind FALSE! "CommandAborted" bleibt solange TRUE, bis der Steuereingang "Enable" deaktiviert wird. Ein zusätzlicher Reset-Eingang ist daher nicht notwendig. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "CommandAborted" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Error	Ist "Error" TRUE, wurde der Funktionsbaustein wegen eines Fehlers abgebrochen und befindet sich in einem Endzustand. "Error" bleibt solange TRUE, bis der Steuereingang "Enable" deaktiviert wird. Ein zusätzlicher Reset-Eingang ist daher nicht notwendig. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "Error" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird

Tab. 5-5: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung



Ist es notwendig, dass bestimmte Eingänge nicht nur mit der Flanke an "Enable" übernommen werden, sondern zyklisch während der Bausteinbearbeitung, so ist dies explizit zu dokumentieren und nicht über Variablennamen kenntlich zu machen!



Bei Bausteinen mit "Shutdown" muss genau definiert werden, wie sich der Baustein verhält, wenn "Enable" wieder auf TRUE gesetzt wird, solange "Shutdown" noch gesetzt ist (Retriggern).

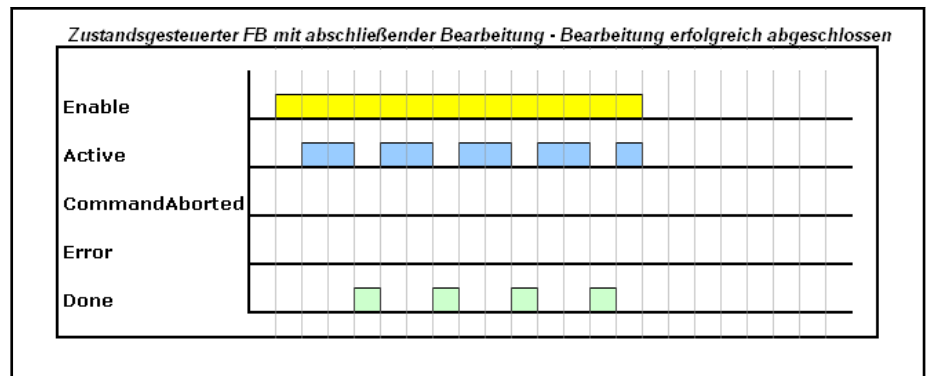


Abb. 5-3: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung erfolgreich abgeschlossen

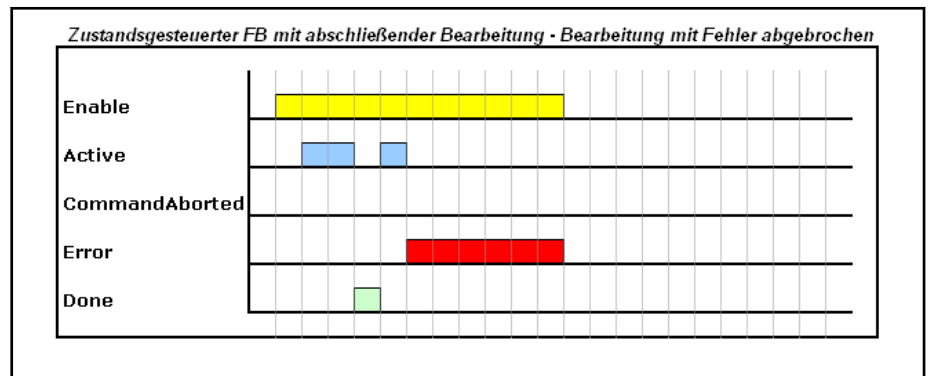


Abb. 5-4: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit Fehler abgebrochen

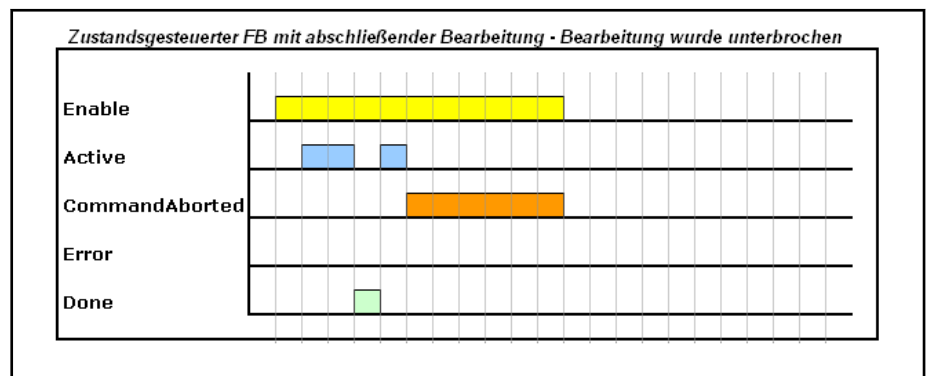


Abb. 5-5: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung unterbrochen

ML_Robot.library

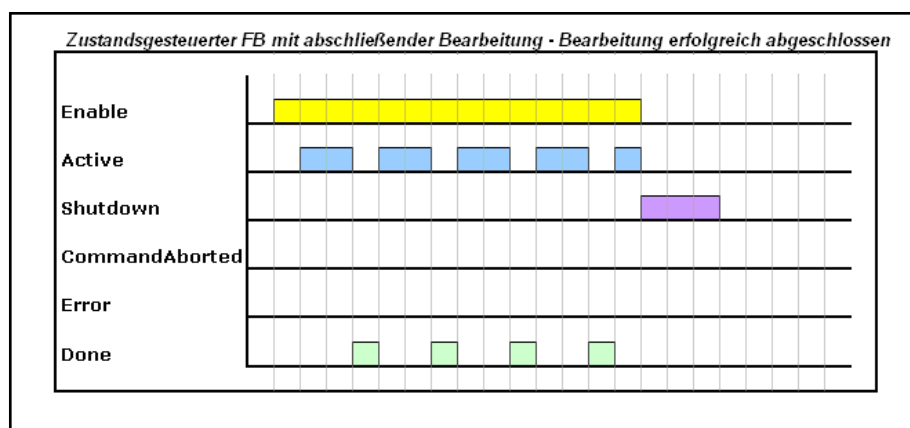


Abb. 5-6: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung erfolgreich abgeschlossen

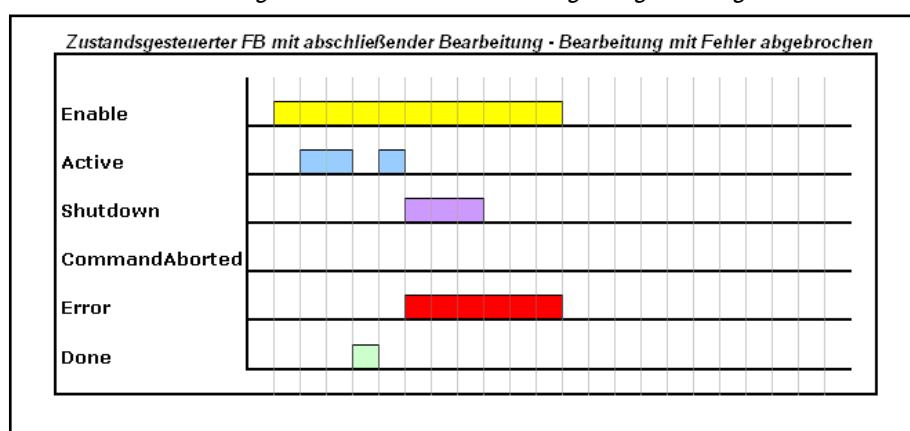


Abb. 5-7: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung mit Fehler abgeschlossen

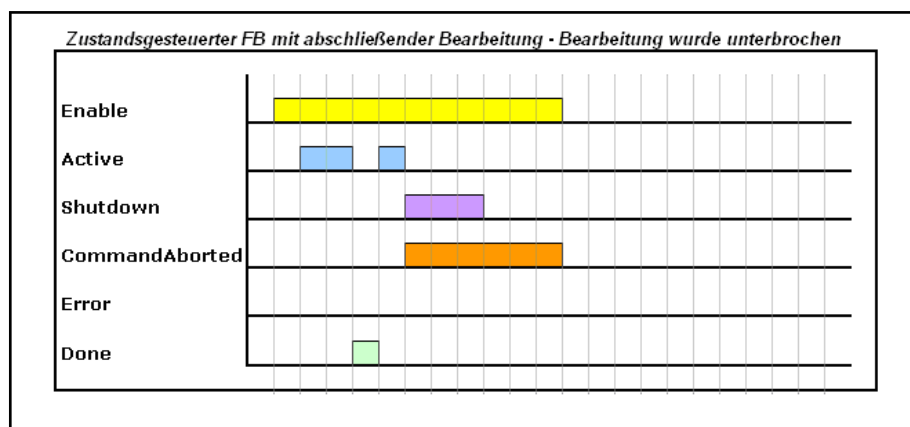


Abb. 5-8: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung wurde unterbrochen

Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung

Steuerung / Bearbeitung	I/O	Name	Beschreibung
Flankenge- steuert / ab- schließend	I	Execute	Mit einer positiven Flanke von "Execute" werden die Eingangsvariablen erfasst. Neue Eingangswerte werden erst über eine erneute positive Flanke von "Execute" wirksam. Der Flankenwechsel an "Execute" reicht aus, um den Baustein zu starten. Zur weiteren Bearbeitung ist der Zustand von "Execute" nicht relevant. Durch einen erneuten Flankenwechsel noch während der Bearbeitung wird der alte Auftrag verworfen, die Eingänge erneut übernommen und mit den neuen Werten der Auftrag fortgesetzt (Retriggerung, Nachtriggern). Ist ein "Shutdown"-Ausgang vorhanden, wird Retriggering möglicherweise nicht unterstützt. Die Bausteindokumentation schafft hierüber Klarheit
	O	Done	Ist "Done" TRUE, hat der Funktionsbaustein seinen Auftrag erfolgreich beendet und ist in einem Endzustand. Datenausgänge sind jetzt gültig. "Active", "Error" und "CommandAborted" sind FALSE! Ist bei Auftragsabschluss der Steuerungseingang "Execute" FALSE, bleibt "Done" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "Done" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden wird dieser Ausgang gleichzeitig mit "Done" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Active	Ist "Active" TRUE, arbeitet der Funktionsbaustein an seiner eigentlichen Aufgabe und ist in einem Zwischenzustand. Eventuelle Vorverarbeitungen werden nicht mit diesem Ausgang gekennzeichnet! "Done", "Error", "Shutdown" und "CommandAborted" sind FALSE!
	O	Shutdown (optional)	Ist ein "Shutdown"-Ausgang vorhanden, benötigt der Baustein nach Beendigung der Aufgabe (signalisiert durch "Done") noch mehrere Zyklen an Rechenzeit, um sich in einen definierten Endzustand zu bringen und verwendete Ressourcen freizugeben. Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird!
	O	CommandAborted	Ist "CommandAborted" TRUE, wurde der Funktionsbaustein unterbrochen und befindet sich in einem Endzustand. "Done", "Active" und "Error" sind FALSE! Ist zum Unterbrechungszeitpunkt der Steuerungseingang "Execute" FALSE, bleibt "CommandAborted" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "CommandAborted" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "CommandAborted" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Error	Ist "Error" TRUE, wurde der Funktionsbaustein wegen eines Fehlers abgebrochen und befindet sich in einem Endzustand. Ist bei Auftreten des Fehlers der Steuerungseingang "Execute" FALSE, bleibt "Error" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "Error" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "Error" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird

Tab. 5-6: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung



Werden bestimmte Eingänge nicht nur mit der 0/1-Flanke an "Execute" übernommen, sondern zyklisch während der Bausteinbearbeitung, so ist dies explizit zu dokumentieren und nicht über Variablennamen kenntlich zu machen!



Bei Bausteinen mit "Shutdown" muss genau definiert werden, wie sich der Baustein verhält, wenn an "Execute" eine erneute Flanke erkannt wird, solange "Shutdown" noch gesetzt ist (Retriggern).

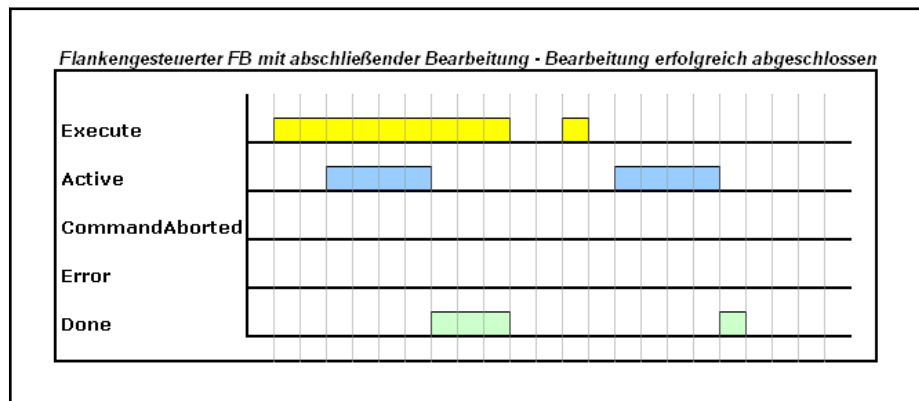


Abb. 5-9: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung erfolgreich abgeschlossen

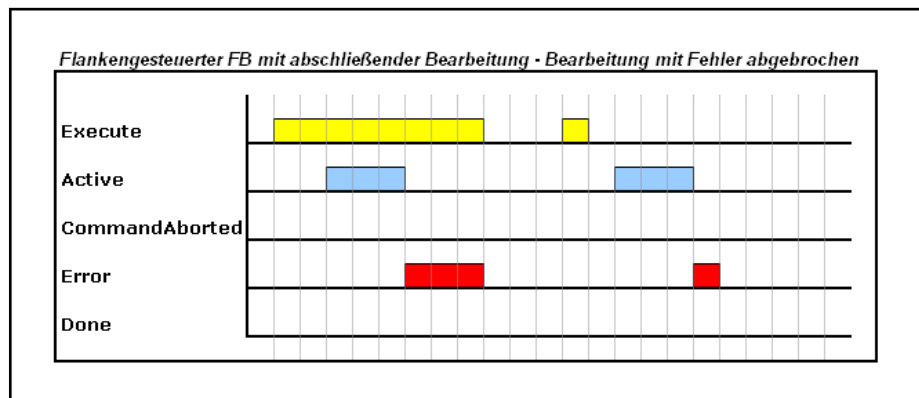


Abb. 5-10: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung mit Fehler abgeschlossen

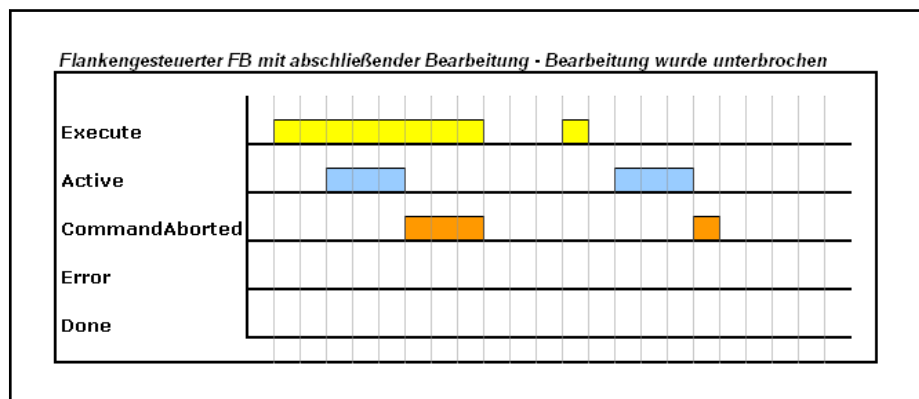


Abb. 5-11: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung unterbrochen

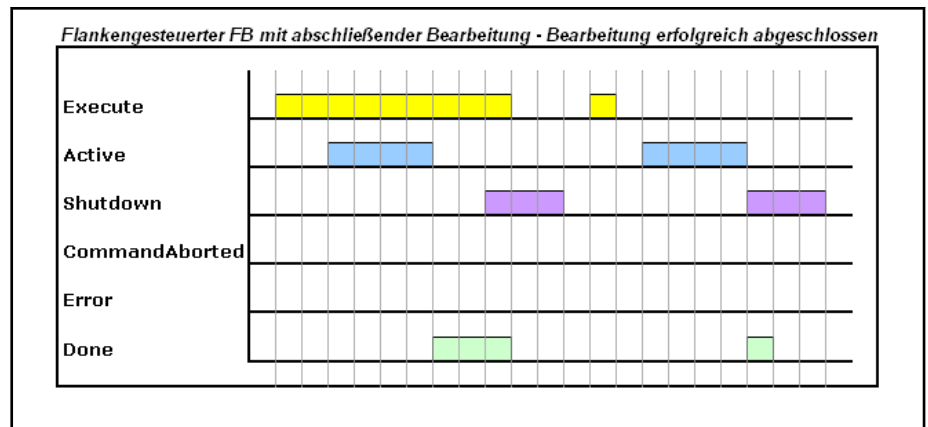


Abb. 5-12: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung erfolgreich abgeschlossen

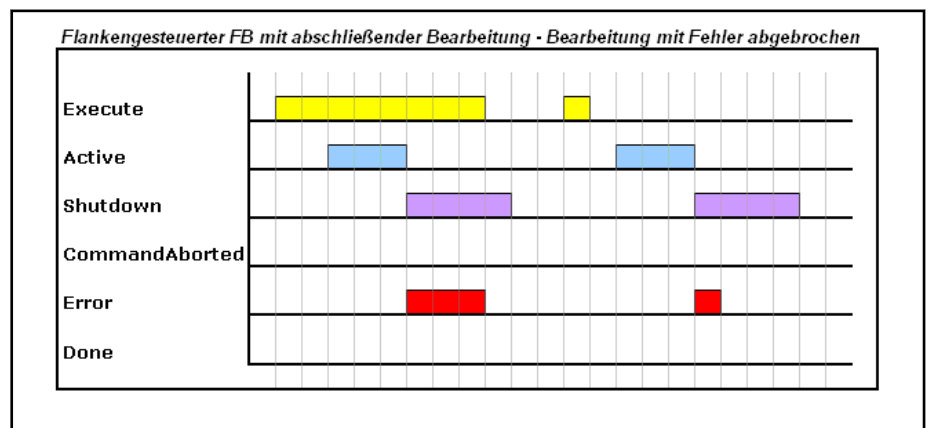


Abb. 5-13: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung mit Fehler abgeschlossen

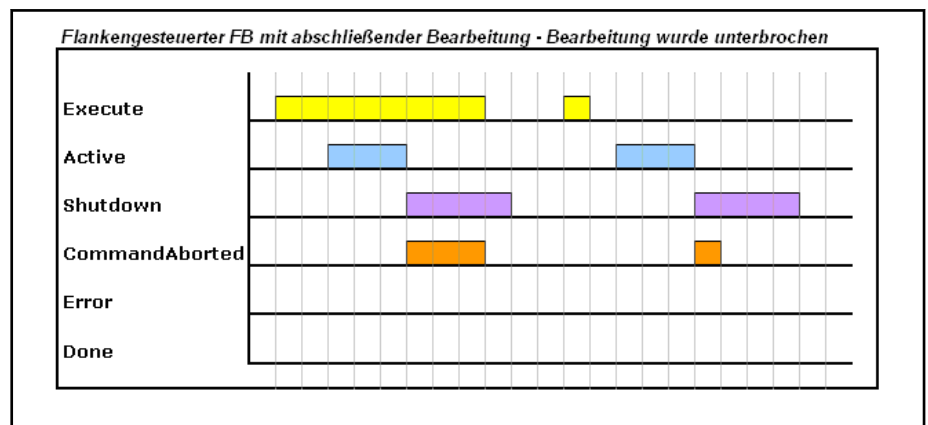


Abb. 5-14: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung unterbrochen

ML_Robot.library

Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung

Steuerung / Bearbeitung	I/O	Variablenname	Beschreibung
Zustandsge- steuert / per- manent	I	Enable	Mit einer positiven Flanke von "Enable" werden die Eingangsvariablen erfasst. Neue Eingangswerte werden erst über eine erneute positive Flanke von "Enable" wirksam. "Enable" muss während der gesamten Bausteinbearbeitung TRUE sein! Das Deaktivieren von "Enable" unterbricht die Bearbeitung und setzt die Ausgänge "In....", "Active", "CommandAborted" und "Error" auf FALSE. Falls notwendig wird über "Shutdown" signalisiert, dass weitere Aufrufe notwendig sind, um den Baustein in einen definierten Endzustand zu bringen
	O	In....	Ist "In...." TRUE, hat der Funktionsbaustein sein Ziel erreicht, befindet sich aber weiter im Eingriff, um das Erreichte zu "bewahren" und befindet sich damit in einem dauerhaften Endzustand. Datenausgänge sind jetzt gültig. Da der FB weiter im Eingriff bleibt, muss auch "Active" TRUE bleiben. "Error" und "CommandAborted" sind FALSE! Solange "Enable" TRUE ist, bleibt "In...." ebenfalls TRUE
	O	Active	Ist Active TRUE, arbeitet der Funktionsbaustein an seiner eigentlichen Aufgabe. Eventuelle Vorverarbeitungen werden nicht mit diesem Ausgang gekennzeichnet! Da der FB permanent im Eingriff bleibt, bleibt "Active" TRUE bis der FB über den Steuereingang "Enable" abgeschaltet wird oder durch "Error" oder "CommandAborted" beendet wird. Solange "Active" TRUE ist, müssen "Error" oder "CommandAborted" FALSE sein!
	O	Shutdown (optional)	Ist ein "Shutdown"-Ausgang vorhanden, benötigt der Baustein beim Deaktivieren von "Enable" noch mehrere Zyklen an Rechenzeit, um sich in einen definierten Endzustand zu bringen und verwendete Ressourcen freizugeben. Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird!
	O	CommandAborted	Ist "CommandAborted" TRUE, wurde der Funktionsbaustein unterbrochen und befindet sich in einem Endzustand. "In....", "Active" und "Error" sind FALSE! "CommandAborted" bleibt solange TRUE, bis der Steuereingang "Enable" deaktiviert wird. Ein zusätzlicher Reset-Eingang ist daher nicht notwendig. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "CommandAborted" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Error	Ist "Error" TRUE, wurde der Funktionsbausteins wegen eines Fehlers abgebrochen und befindet sich in einem Endzustand. "Error" bleibt solange TRUE bis der Steuereingang "Enable" deaktiviert wird. Ein zusätzlicher Reset-Eingang ist daher nicht notwendig. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "Error" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird

Tab. 5-7: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung



Ist es notwendig, dass bestimmte Eingänge nicht nur mit der Flanke an "Enable" übernommen werden, sondern zyklisch während der Bausteinbearbeitung, so ist dies explizit zu dokumentieren und nicht über Variablennamen kenntlich zu machen!



Bei Bausteinen mit "Shutdown" muss genau definiert werden, wie sich der Baustein verhält, wenn "Enable" wieder TRUE gesetzt wird, solange "Shutdown" noch gesetzt ist (Retriggern).

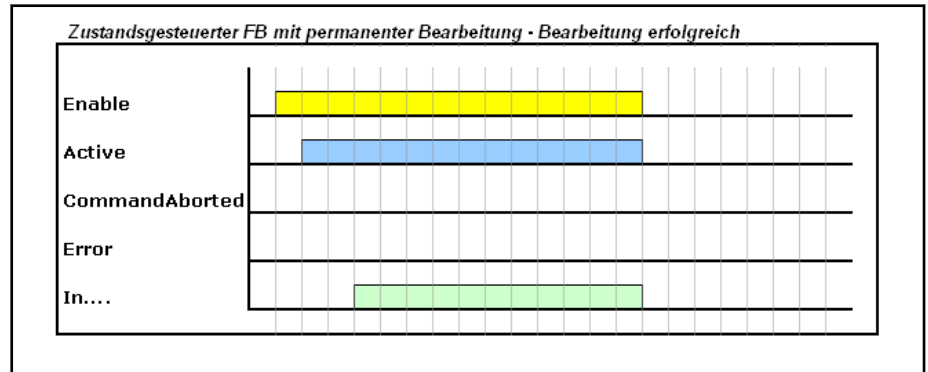


Abb. 5-15: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung erfolgreich

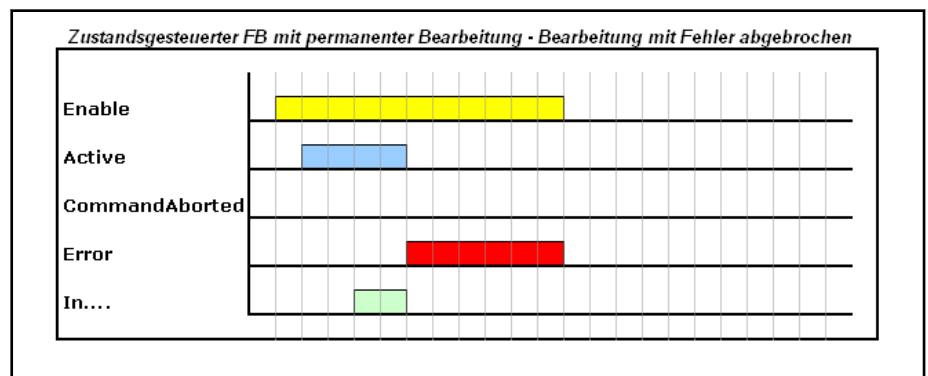


Abb. 5-16: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung mit Fehler abgebrochen

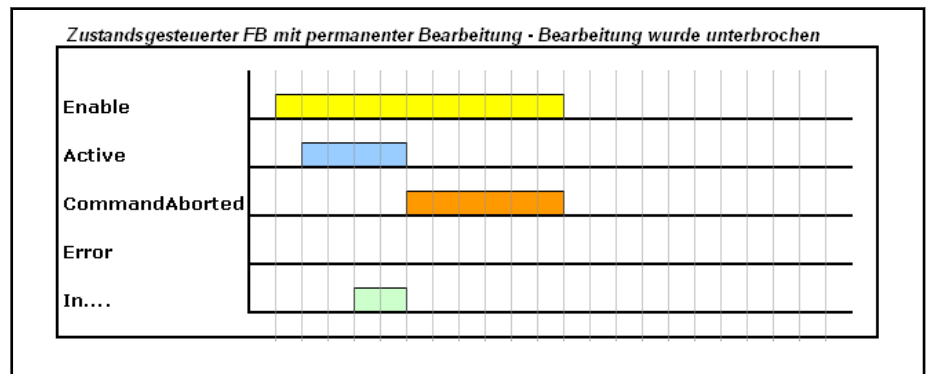


Abb. 5-17: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung unterbrochen

ML_Robot.library

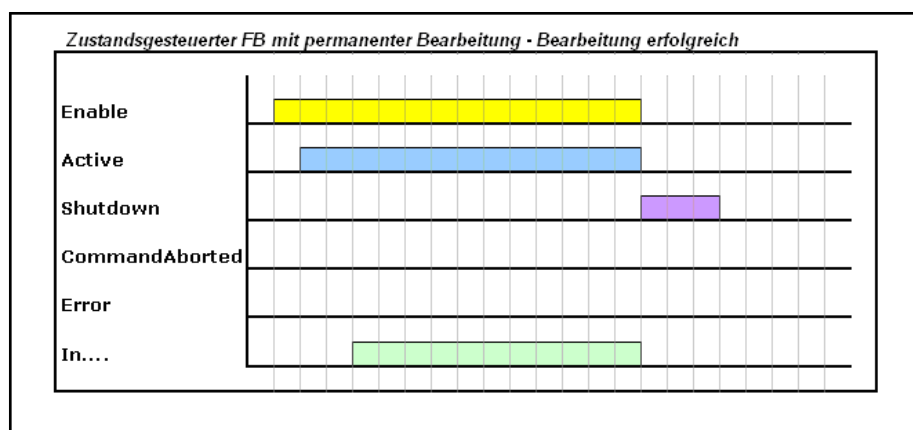


Abb. 5-18: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung erfolgreich

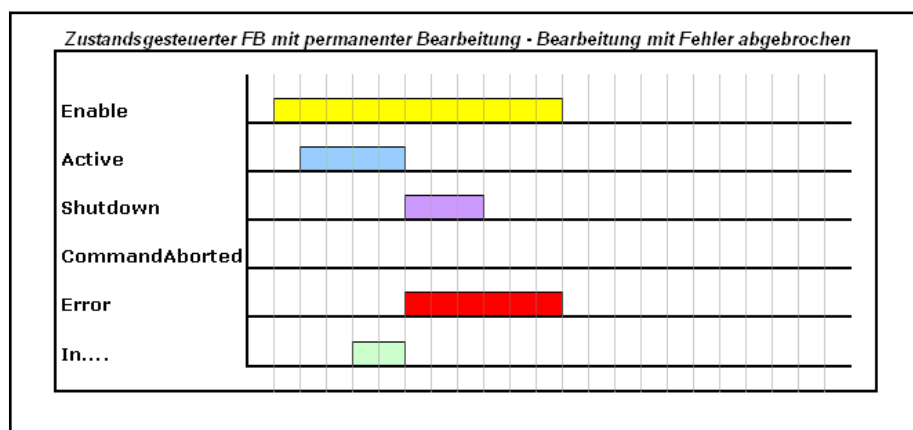


Abb. 5-19: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung mit Fehler abgebrochen

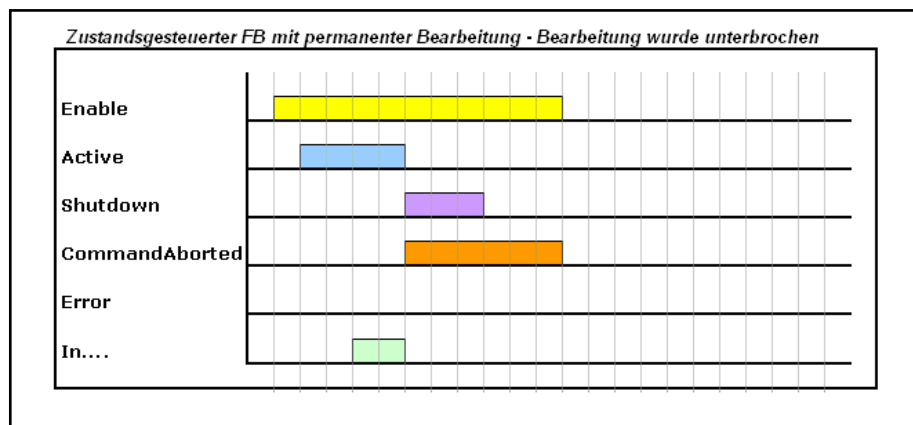


Abb. 5-20: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung unterbrochen

Ein- und Ausgänge flankengesteuerter Funktionsbausteine, permanente Bearbeitung

Steuerung / Bearbeitung	I/O	Variablenname	Beschreibung
Flankenge- steuert / per- manent	i	Execute	Mit einer positiven Flanke von "Execute" werden die Eingangsvariablen erfasst. Neue Eingangswerte werden erst über eine erneute positive Flanke von "Execute" wirksam. Der Flankenwechsel an "Execute" reicht aus, um den Baustein zu starten. Zur weiteren Bearbeitung ist der Zustand von "Execute" nicht relevant. Durch einen erneuten Flankenwechsel noch während der Bearbeitung werden der alte Auftrag verworfen, die Eingänge erneut übernommen und mit den neuen Werten der Auftrag fortgesetzt (Retriggerung, Nachtriggern). Hat der Baustein einen "Shutdown"-Ausgang, ist dieses Retriggern eventuell nicht erlaubt
	O	In....	Ist "In...." TRUE, hat der Funktionsbaustein sein Ziel erreicht, befindet sich aber weiter im Eingriff, um das Erreichte zu "bewahren" und ist damit in einem dauerhaften Endzustand. Datenausgänge sind jetzt gültig. Da der FB weiter im Eingriff bleibt, muss auch "Active" TRUE bleiben. Das gilt auch dann, wenn "In...." wieder abfällt. "Error" und "CommandAborted" sind FALSE! Ist bei Auftragsabschluss der Steuerungseingang "Execute" FALSE, bleibt "In...." für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "In...." solange TRUE, bis "Execute" deaktiviert wird
	O	Active	Ist "Active" TRUE, arbeitet der Funktionsbaustein an seiner eigentlichen Aufgabe. Eventuelle Vorverarbeitungen werden nicht mit diesem Ausgang gekennzeichnet! Da der FB permanent im Eingriff bleibt, bleibt "Active" TRUE bis der Baustein durch "Error" oder "CommandAborted" beendet wird. Solange "Active" TRUE ist, müssen "Error" oder "CommandAborted" sowie „Shutdown“ FALSE sein!
	O	Shutdown (optional)	Ist ein "Shutdown"-Ausgang vorhanden, benötigt der Baustein, sobald der Baustein seine Aufgabe abbricht ("CommandAborted" oder "Error" = TRUE), noch mehrere Zyklen an Rechenzeit, um sich in einen definierten Endzustand zu bringen und verwendete Ressourcen freizugeben. Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird!
	O	CommandAborted	Ist "CommandAborted" TRUE, wurde der Funktionsbaustein unterbrochen und befindet sich in einem Endzustand. "In....", "Active" und "Error" sind FALSE! Ist zum Unterbrechungszeitpunkt der Steuerungseingang "Execute" FALSE, bleibt "CommandAborted" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "CommandAborted" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "CommandAborted" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Error	Ist "Error" TRUE, wurde der Funktionsbaustein wegen eines Fehlers abgebrochen und befindet sich in einem Endzustand. Ist bei Auftreten des Fehlers der Steuerungseingang "Execute" FALSE, bleibt "Error" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "Error" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "Error" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird

Tab. 5-8: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung



Werden bestimmte Eingänge nicht nur mit der 0/1-Flanke an "Execute" übernommen, sondern zyklisch während der Bausteinbearbeitung, so ist dies explizit zu dokumentieren und nicht über Variablennamen kenntlich zu machen!



Bei Bausteinen mit "Shutdown" muss genau definiert werden, wie sich der Baustein verhält, wenn an "Execute" eine erneute Flanke erkannt wird, solange "Shutdown" noch gesetzt ist (Retrigger).

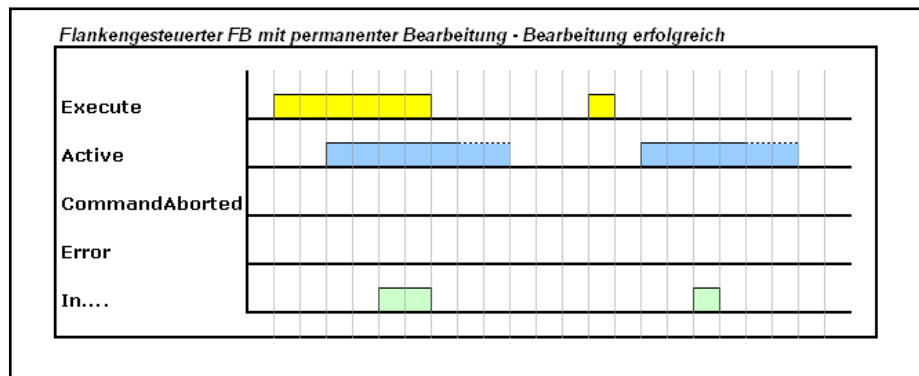


Abb. 5-21: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung erfolgreich

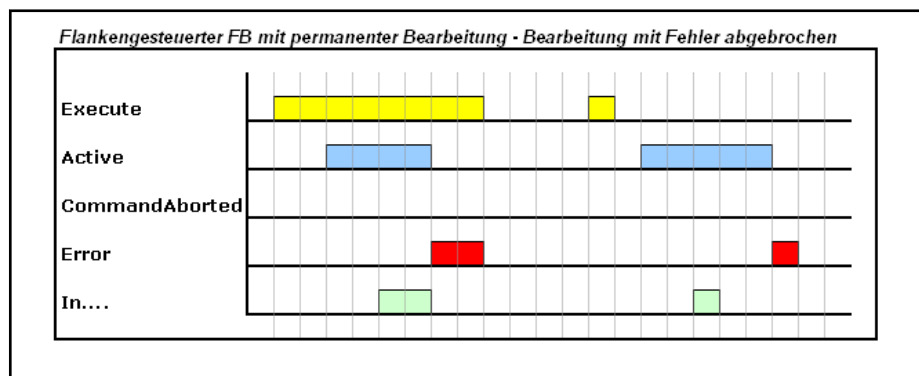


Abb. 5-22: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung mit Fehler abgebrochen

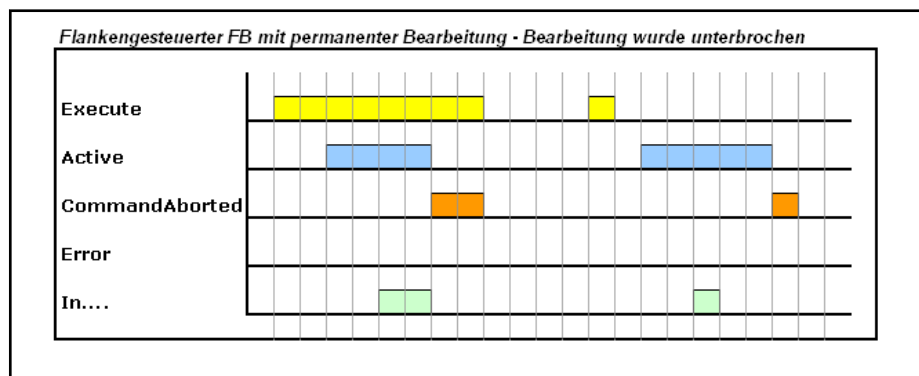


Abb. 5-23: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung unterbrochen

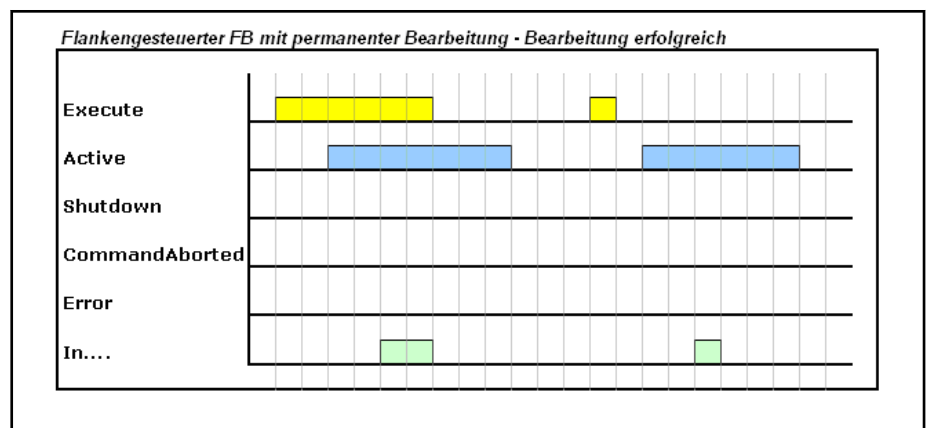


Abb. 5-24: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung erfolgreich

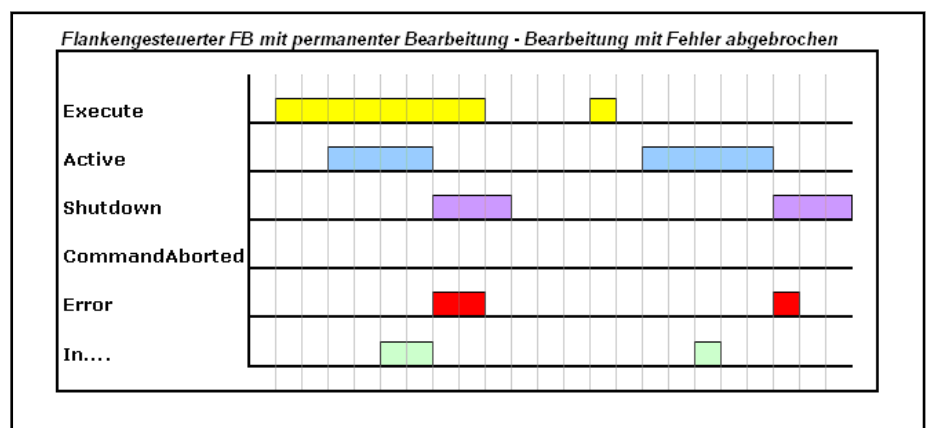


Abb. 5-25: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung mit Fehler abgebrochen

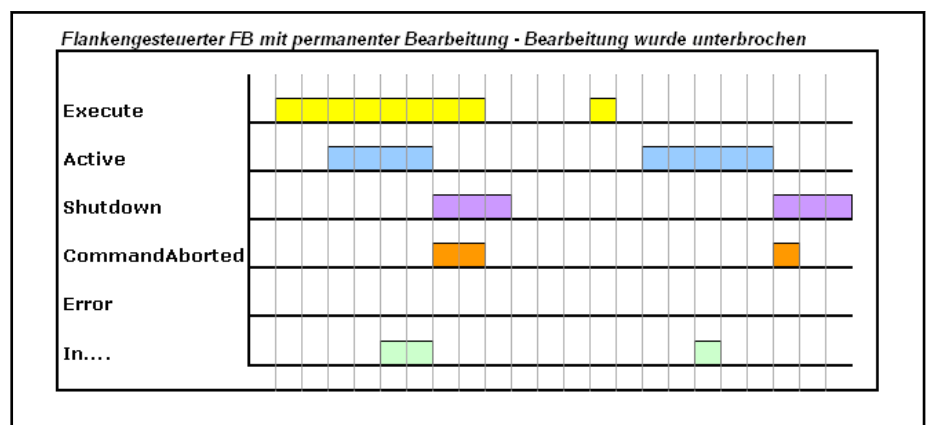


Abb. 5-26: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung unterbrochen

5.3.3 InBuffer-Ausgang an ML_Robot-Funktionsbausteinen

Funktionsbausteine der ML_Robot, die Kommandos repräsentieren, die in den Satz-Puffer gestellt werden, besitzen einen InBuffer-Ausgang.

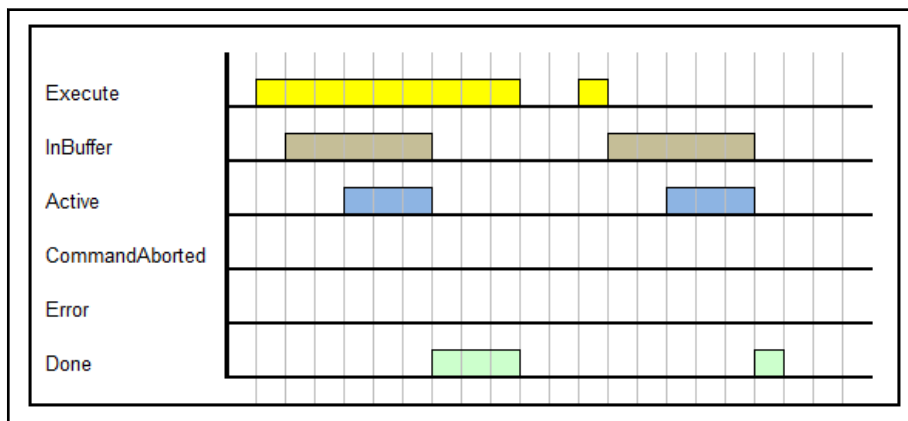
(Siehe [Zustand eines Kommandos](#), Seite 36).

Dieser Ausgang zeigt an, dass sich das Kommando im Satz-Puffer befindet und für mindestens einen Zyklus auf TRUE gesetzt wird.

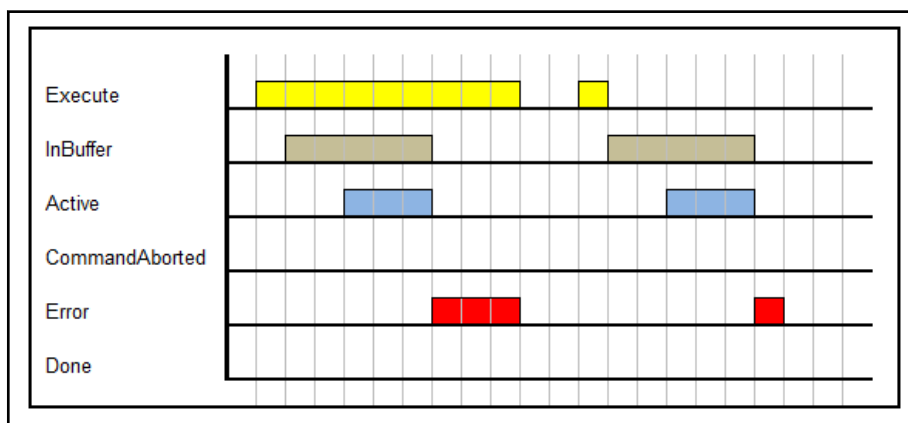
Folgende Diagramme zeigen das Verhalten des Ausgangs in Bezug auf die anderen Ein- und Ausgänge.

ML_Robot.library

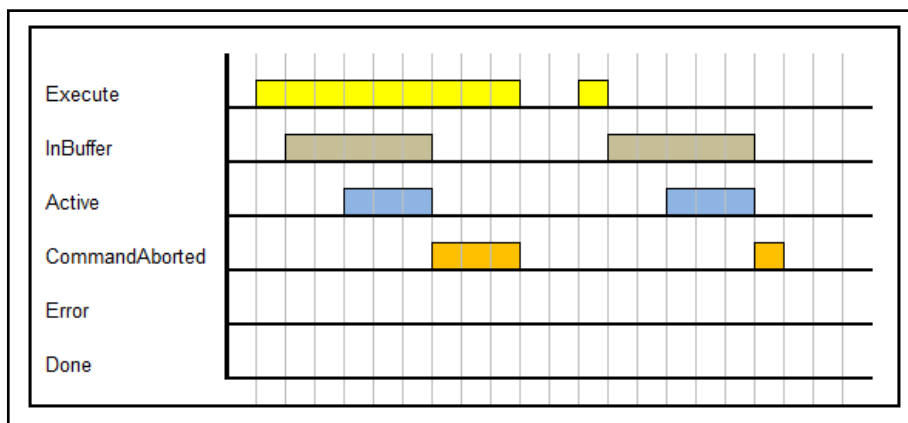
Verhalten des InBuffer-Ausgangs - Bearbeitung erfolgreich

Abb. 5-27: E/A-flankengesteuerter ML_Robot-Kommando-Funktionsbaustein.
Bearbeitung erfolgreich abgeschlossen

Verhalten des InBuffer-Ausgangs - Bearbeitung fehlerhaft

Abb. 5-28: E/A-flankengesteuerter ML_Robot-Kommando-Funktionsbaustein.
Bearbeitung fehlerhaft

Verhalten des InBuffer-Ausgangs - Bearbeitung abgebrochen

Abb. 5-29: E/A-flankengesteuerter ML_Robot-Kommando-Funktionsbaustein.
Bearbeitung abgebrochen

5.3.4 Kinematic/ Config

MC_GroupReset

Kurzbeschreibung

Der Funktionsbaustein "MC_GroupReset" setzt alle anstehenden Fehler einer Kinematik zurück.

Der Funktionsbaustein hat Einfluss auf die Zustandsmaschine der Kinematik.
 (Siehe [Zustandsdiagramm einer Kinematik auf Seite 33](#)).



Falls in der Kinematik dieses Funktionsbausteins kein Fehler vorliegt, geht keine Aktivität vom MC_GroupReset aus.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

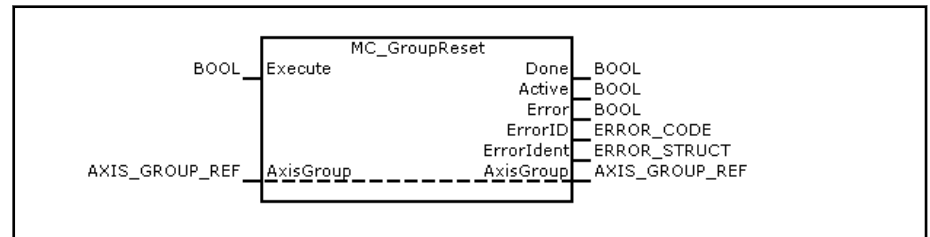


Abb. 5-30: FB MC_GroupReset

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Steuerungsnummer und Objektnummer der Kinematik
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
VAR_OUTPUT	Done	BOOL	Der Kinematik-Reset ist erfolgreich ausgeführt
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detaillinformation über den Fehler

Tab. 5-9: Schnittstelle des FB MC_GroupReset

Signal-Zeit-Diagramm

Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MC_SetCartesianTransform

Kurzbeschreibung

Der Funktionsbaustein "MC_SetCartesianTransform" definiert ein kartesisches Produkt-Koordinatensystem.
 (Siehe [Produkt-Koordinatensystem \(PCS\) auf Seite 27](#)).

ML_Robot.library



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Der Funktionsbaustein kann nur für Produkt-Koordinatensysteme verwendet werden, die kein Band darstellen.

Schnittstellenbeschreibung

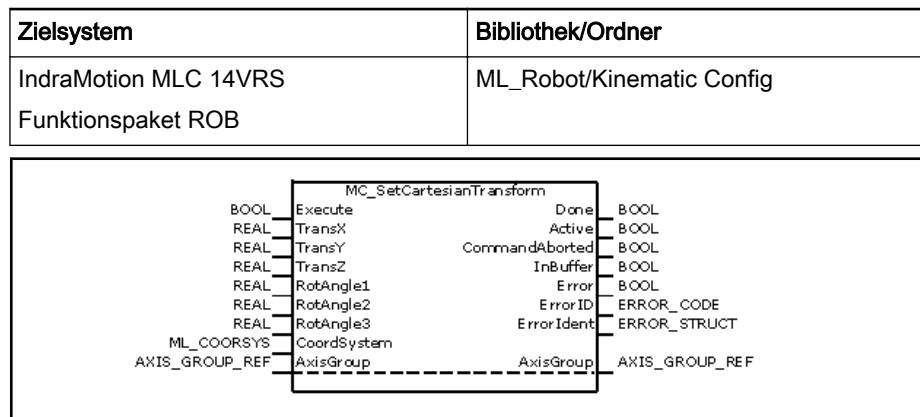


Abb. 5-31: Funktionsbaustein MC_SetCartesianTransform

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	TransX	REAL	X-Position des PCS im MCS
	TransY	REAL	Y-Position des PCS im MCS
	TransZ	REAL	Z-Position des PCS im MCS
	RotAngle1	REAL	Verdrehung des PCS um die x-Achse in MCS
	RotAngle2	REAL	Verdrehung des PCS um die y-Achse in MCS
	RotAngle3	REAL	Verdrehung des PCS um die z-Achse in MCS
	CoordSystem	ML_COORDSYS	Produkt-Koordinatensystem (PCS) für das die Werte gesetzt werden sollen (Siehe ML_COORDSYS auf Seite 317).
VAR_OUTPUT	Done	BOOL	Definition wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-10: Funktionsbaustein MC_SetCartesianTransform

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
TransX	REAL			0.0	steigende Flanke an "Execute"
TransY	REAL			0.0	steigende Flanke an "Execute"
TransZ	REAL			0.0	steigende Flanke an "Execute"
RotAngle1	REAL			0.0	steigende Flanke an "Execute"
RotAngle2	REAL			0.0	steigende Flanke an "Execute"
RotAngle3	REAL			0.0	steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS	CS_PRODU CT_COOR- DINATES_1	CS_PRODU CT_COOR- DINA- TES_16	CS_PRODUCT_COOR DINATES_1	steigende Flanke an "Execute"

Tab. 5-11: Eingangsverhalten MC_SetCartesianTransform

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-12: Fehlernummern des FBs MC_SetCartesianTransform

MC_SetKinTransform

Kurzbeschreibung Der Funktionsbaustein "MC_SetKinTransform" meldet eine benutzerdefinierte Transformation für eine Kinematik an.
 (Siehe [Benutzerdefinierte Transformation auf Seite 176](#)).



Dieser Funktionsbaustein muss schon im Parametriermodus (P2) vor dem Schalten in den Betriebsmodus (P4/BB) ausgeführt werden, sonst ist kein Aktivieren möglich.

ML_Robot.library

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

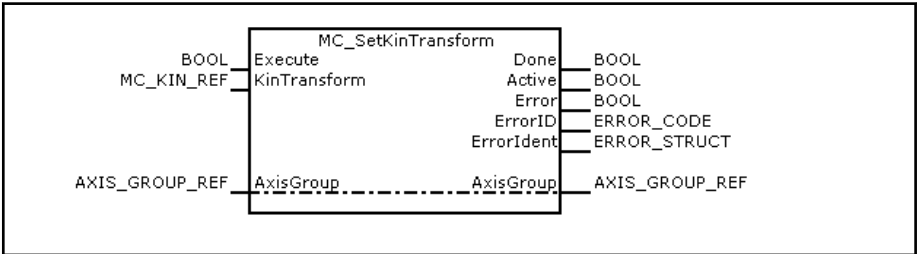


Abb. 5-32: Funktionsbaustein MC_SetKinTransform

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	KinTransform	MC_KIN_REF	Instanz eines Funktionsbausteins, der das Transformationsinterface implementiert. Diese Transformation soll am System angemeldet werden. (Siehe MC_KIN_REF auf Seite 305).
VAR_OUTPUT	Done	BOOL	Anmeldung der Transformation abgeschlossen
	Active	BOOL	Anmeldung der Transformation aktiv
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-13: Funktionsbaustein MC_SetKinTransform

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
KinTransform	MC_KIN_REF auf Seite 305				steigende Flanke an "Execute"

Tab. 5-14: Eingangsverhalten MC_SetKinTransform

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung](#) auf Seite 199

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-15: Fehlernummern des FBs MC_SetKinTransform

MC_SetKinTransformExt

Kurzbeschreibung

Der Funktionsbaustein "MC_SetKinTransformExt" meldet eine erweiterte benutzerdefinierte Transformation für eine Kinematik an.

(Siehe [Erweiterte Benutzerdefinierte Transformation auf Seite 178](#)).



Die Funktion kann nur für die Robot-Control V2 verwendet werden.

Dieser Funktionsbaustein muss schon im Parametriermodus (P2) vor dem Schalten in den Betriebsmodus (P4/BB) ausgeführt werden, sonst ist kein Aktivieren möglich.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

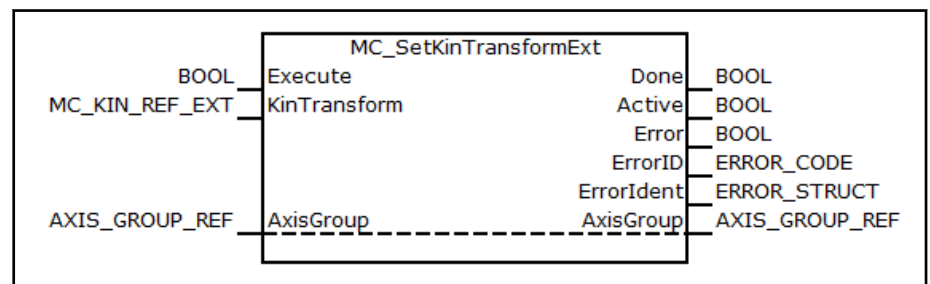


Abb. 5-33: Funktionsbaustein MC_SetKinTransformExt

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	KinTransform	MC_KIN_REF_EXT	Instanz eines Funktionsbausteins, der das Transformationsinterface implementiert. Diese Transformation soll am System angemeldet werden. (Siehe MC_KIN_REF_EXT auf Seite 307).
VAR_OUTPUT	Done	BOOL	Anmeldung der Transformation abgeschlossen
	Active	BOOL	Anmeldung der Transformation aktiv
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-16: Funktionsbaustein MC_SetKinTransformExt

ML_Robot.library

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
KinTransform	MC_KIN_REF_EXT auf Seite 307				Steigende Flanke an "Execute"

Tab. 5-17: Eingangsverhalten MC_SetKinTransformExt

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-18: Fehlernummern des FBs MC_SetKinTransformExt

MC_UngroupAllAxes

Kurzbeschreibung Dieser Coordinated Motion Funktionsbaustein "MC_UngroupAllAxes" entfernt alle Achsen einer Kinematik.

Der Funktionsbaustein hat dadurch direkten Einfluss auf die Zustandsmaschine der Kinematik.

(Siehe [Zustandsdiagramm einer Kinematik auf Seite 33](#)).

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

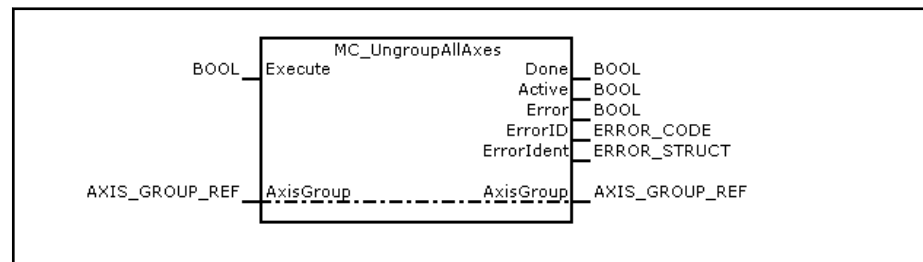


Abb. 5-34: Funktionsbaustein MC_UngroupAllAxes

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
VAR_OUTPUT	Done	BOOL	Bearbeitung abgeschlossen
	Active	BOOL	Bearbeitung läuft
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist.

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Fehlerkennung
	ErrorIdent	ERROR_STRUCT	Fehlerstruktur mit weiterer Unterteilung der Fehler, siehe Fehlercodes F_RELATED_TABLE auf Seite 215.
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-19: Funktionsbaustein MC_UngroupAllAxes

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich

Tab. 5-20: Eingangsverhalten MC_UngroupAllAxes

Fehlercodes F_RELATED_TABLE Der Funktionsbaustein erzeugt in Additional1/Additional2 für F_RELATED_TABLE 16#0170 folgende Fehlermeldungen:

ErrorID	Additional1	Additional2	Beschreibung
ACCESS_ERROR 16#04	16#00000002	16#00	GroupRef liegt außerhalb des gültigen Bereichs
ACCESS_ERROR 16#04	16#00000011	16#00	Sercos Phase 2 ist nicht aktiv

Tab. 5-21: Generierte Fehlernummern des Funktionsbausteins MC_UngroupAllAxes für F_RELATED_TABLE

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
INPUT_RANGE_ERROR 16#06	16#F0290402	16#00	Ungültige Kinematiknummer

Tab. 5-22: Generierte Fehlernummern des Funktionsbausteins MC_UngroupAllAxes für MLC_TABLE

ML_AddMechanicData

Kurzbeschreibung Die Funktion "ML_AddMechanicData" fügt einen Eintrag in die ML_KIN_MECHANIC_DATA-Struktur hinzu.
 (Siehe [Mechanische Informationen auf Seite 183](#)).

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

ML_Robot.library

Schnittstellenbeschreibung

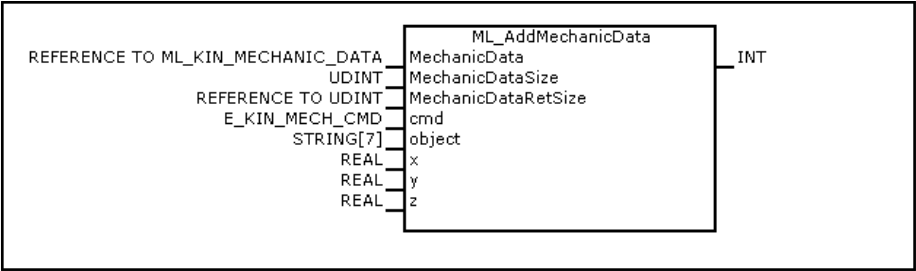


Abb. 5-35: Funktion ML_AddMechanicData

	Name	Typ	Kommentar
VAR_INPUT	MechanicData	REFERENCE TO ML_KIN_MECHANIC_DATA	Auszufüllende Struktur ML_KIN_MECHANIC_DATA auf Seite 314
	MechanicDataSize	UDINT	Anzahl aller Elemente in MechanicData
	MechanicDataRetSize	REFERENCE TO UDINT	Anzahl benutzter Elemente in MechanicData
	cmd	E_KIN_MECH_CMD	Kommando für den Objektname E_KIN_MECH_CMD auf Seite 312
	object	STRING[7]	Objektname
	x	REAL	x-Wert
	y	REAL	y-Wert
	z	REAL	z-Wert
RETURN	ML_AddMechanicData	INT	Rückgabewert der Funktion

Tab. 5-23: Funktion ML_AddMechanicData

ML_GroupAllAxes

Kurzbeschreibung Der Coordinated Motion Funktionsbaustein "ML_GroupAllAxes" fügt alle Achsen einer Kinematik zur Kinematik hinzu.

Der Funktionsbaustein hat dadurch direkten Einfluss auf die Zustandsmaschine der Kinematik

(Siehe [Zustandsdiagramm einer Kinematik auf Seite 33](#)).

Um die Achsen einer Kinematik zu bestimmen, liest der Baustein den Parameter [K-0-0008](#) der ausgewählten Kinematik und fügt alle Positionier- und Orientierungsachsen der Kinematik hinzu.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

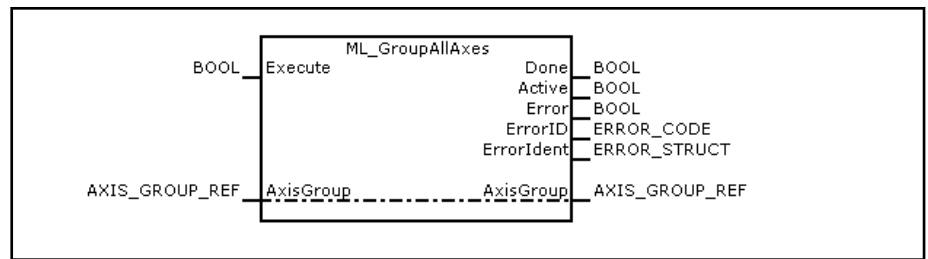


Abb. 5-36: Funktionsbaustein ML_GroupAllAxes

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
VAR_OUTPUT	Done	BOOL	Bearbeitung abgeschlossen
	Active	BOOL	Bearbeitung läuft
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Fehlerkennung
	ErrorIdent	ERROR_STRUCT	Fehlerstruktur mit weiterer Unterteilung der Fehler, siehe Fehlercodes F_Related_Table auf Seite 217.
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-24: Funktionsbaustein ML_GroupAllAxes

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich

Tab. 5-25: Eingangsverhalten ML_GroupAllAxes

Signal-Zeit-Diagramm Siehe Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199

Fehlercodes F_Related_Table Der Funktionsbaustein erzeugt in Additional1/Additional2 für F_RELATED_TABLE 16#0170 folgende Fehlermeldungen:

ErrorID	Additional1	Additional2	Beschreibung
ACCESS_ERROR 16#04	16#00000002	16#00	GroupRef liegt außerhalb des gültigen Bereichs
ACCESS_ERROR 16#04	16#00000011	16#00	Sercos Phase 2 ist nicht aktiv

Tab. 5-26: Generierte Fehlernummern des Funktionsbausteins ML_GroupAllAxes für F_RELATED_TABLE

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
INPUT_RANGE_ERROR 16#06	16#F0290402	16#00	Ungültige Kinematiknummer

Tab. 5-27: Generierte Fehlernummern des Funktionsbausteins ML_GroupAllAxes für MLC_TABLE

ML_Robot.library

ML_GroupSetSafeZone

Kurzbeschreibung

Der Funktionsbaustein "ML_GroupSetSafeZone" konfiguriert eine Schutzzone einer Kinematik.

(Siehe [Verfahrenbereichsgrenzen auf Seite 79](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS	ML_Robot/Kinematic Config
Funktionspaket ROB	

Schnittstellenbeschreibung

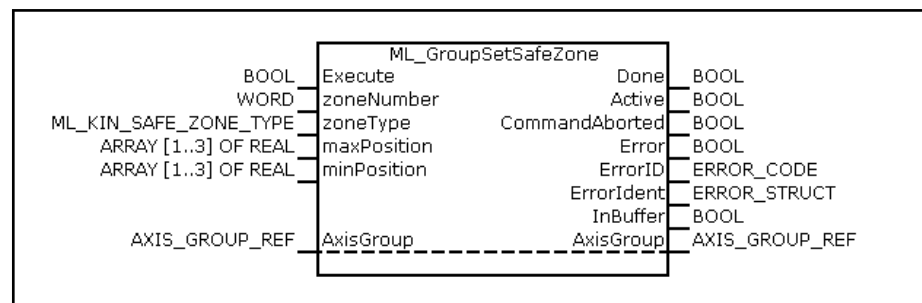


Abb. 5-37: Funktionsbaustein ML_GroupSetSafeZone

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	zoneNumber	WORD	Nummer der Schutzzone
	zoneType	ML_KIN_SAFE_ZONE_TYPE	Typ der Schutzzone (Siehe ML_KIN_SAFE_ZONE_TYPE auf Seite 311)
	maxPosition	ARRAY [1..3] OF REAL	Maximaler Punkt im Maschinen-Koordinatensystem der Schutzzone
	minPosition	ARRAY [1..3] OF REAL	Minimaler Punkt im Maschinen-Koordinatensystem der Schutzzone
VAR_OUTPUT	Done	BOOL	Definition wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-28: Funktionsbaustein ML_GroupSetSafeZone

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
zoneNumber	WORD	0	31	0	Steigende Flanke an "Execute"
zoneType	ML_KIN_SAFE_ZONE_TYPE			ML_KIN_SAFE_ZONE_DEACTIVATE	Steigende Flanke an "Execute"
maxPosition	ARRAY [1..3] OF REAL			[3(0)]	Steigende Flanke an "Execute"
minPosition	ARRAY [1..3] OF REAL			[3(0)]	Steigende Flanke an "Execute"

Tab. 5-29: Eingangsverhalten ML_GroupSetSafeZone

- Signal-Zeit-Diagramm**
- Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199
 - InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_SetBeltDesyncRelConfiguration

Kurzbeschreibung Der Funktionsbaustein ML_SetBeltDesyncRelConfiguration konfiguriert das relative Absynchronisieren eines Bandes.
(Siehe [Bandsynchrone Bewegung auf Seite 67](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

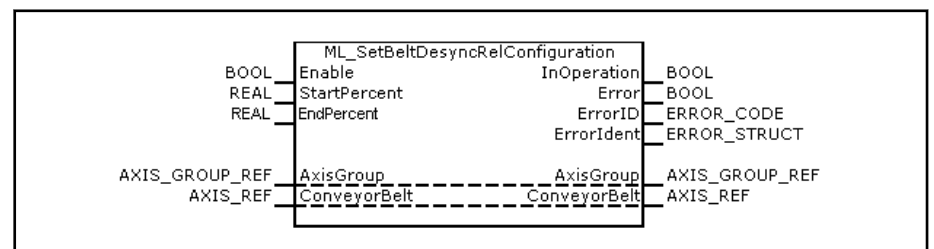


Abb. 5-38: Funktionsbaustein ML_SetBeltDesyncRelConfiguration

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	StartPercent	REAL	Start Prozent zum relativem Absynchronisieren
	EndPercent	REAL	End Prozent zum relativem Absynchronisieren

ML_Robot.library

	Name	Typ	Kommentar
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Achsreferenz der Bandachse

Tab. 5-30: Funktionsbaustein ML_SetBeltDesyncRelConfiguration

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
StartPercent	REAL	0.0	100.0	0.0	Kontinuierlich
EndPercent	REAL	0.0	100.0	50.0	Kontinuierlich

Tab. 5-31: Eingangsverhalten ML_SetBeltDesyncRelConfiguration

Signal-Zeit-Diagramm

Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_SetBeltConfiguration**Kurzbeschreibung**

Der Funktionsbaustein "ML_SetBeltConfiguration" konfiguriert ein Band.
Dieser Baustein muss vor dem bandsynchronen Verfahren einmalig ausgeführt werden.
Das Setzen der Bandparameter darf nur erfolgen, wenn sich die Kinematik weder aufsynchronisiert noch gerade bandsynchron verfährt.
(Siehe [Bandsynchrone Bewegung auf Seite 67](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

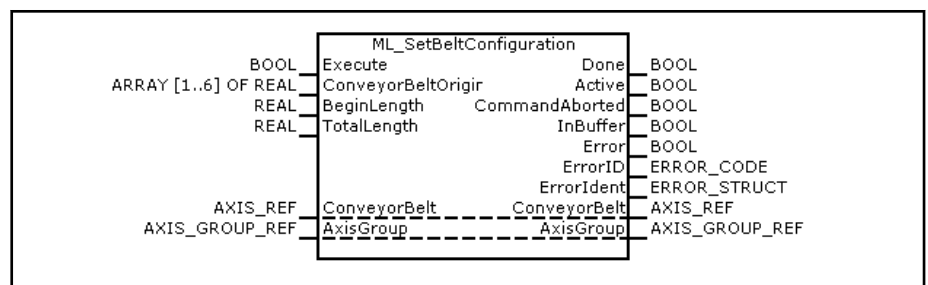


Abb. 5-39: Funktionsbaustein ML_SetBeltConfiguration

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	ConveyorBeltOrigin	ARRAY [1..6] OF REAL	Definiert die Position und Verdrehung des Bandes relativ zum MCS
	BeginLength	REAL	Definiert die Länge des spätesten Starten einer bandsynchronen Bewegung ab der Startposition
	TotalLength	REAL	Definiert die Länge des bandsynchronen Verfahrbereichs ab der Startposition
VAR_OUTPUT	Done	BOOL	Definition wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Achsreferenz der Bandachse

Tab. 5-32: Funktionsbaustein ML_SetBeltConfiguration

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
ConveyorBeltOrigin	ARRAY [1..6] OF REAL	[6(0.0)]		[6(0.0)]	Steigende Flanke an "Execute"
BeginLength	REAL	> 0.0	<Total- Length>	0.0	Steigende Flanke an "Execute"
TotalLength	REAL	> 0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-33: Eingangsverhalten ML_SetBeltConfiguration

Signal-Zeit-Diagramm

- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
- [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

ML_Robot.library

MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#0003	16#F0220201	16#0000	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kine- matik ist falsch

Tab. 5-34: Fehlernummern des FBs ML_SetBeltConfiguration

ML_SetBeltErrorReaction

Kurzbeschreibung

Der Funktionsbaustein "ML_SetBeltErrorReaction" konfiguriert die Fehlerreaktion eines Band.

(Siehe [Bandsynchrone Bewegung auf Seite 67](#)).

Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

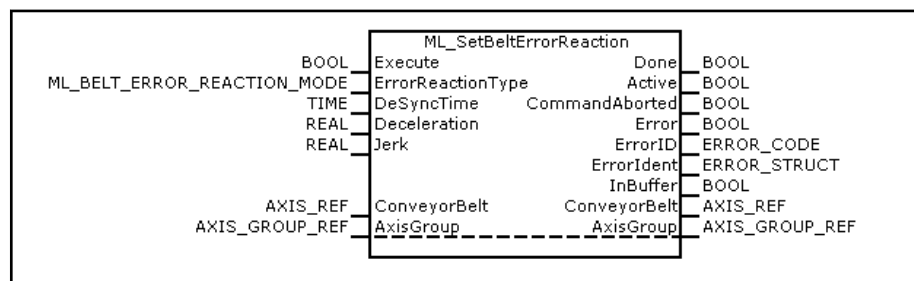


Abb. 5-40: Funktionsbaustein ML_SetBeltErrorReaction

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	ErrorReactionType	ML_BELT_ERROR_REACTION_MODE	Definiert den Typ der Fehlerreaktion (Siehe ML_BELT_ERROR_REACTION_MODE auf Seite 310).
	DeSyncTime	TIME	Zeit zum Absynchronisieren
	Deceleration	REAL	Maximale Bahnverzögerung zum Absynchronisieren
	Jerk	REAL	Maximaler Bahnruck zum Absynchronisieren
VAR_OUTPUT	Done	BOOL	Definition wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen

	Name	Typ	Kommentar
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Achsreferenz der Bandachse

Tab. 5-35: Funktionsbaustein ML_SetBeltErrorReaction

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
ErrorReactionType	ML_BELT_ERROR_REACTION_MODE			ML_BELT_ERROR_NO_SPECIFIC_REACTION	Steigende Flanke an "Execute"
DeSyncTime	TIME	>T#0ms		T#0ms	Steigende Flanke an "Execute"
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-36: Eingangsverhalten ML_SetBeltErrorReaction

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#0003	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-37: Fehlernummern des FBs ML_SetBeltErrorReaction

ML_SetCylindricTransform

Kurzbeschreibung Der Funktionsbaustein "ML_SetCylindricTransform" definiert ein zylindrisches Produkt-Koordinatensystem.
(Siehe [Produkt-Koordinatensystem \(PCS\) auf Seite 27](#)).

ML_Robot.library



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

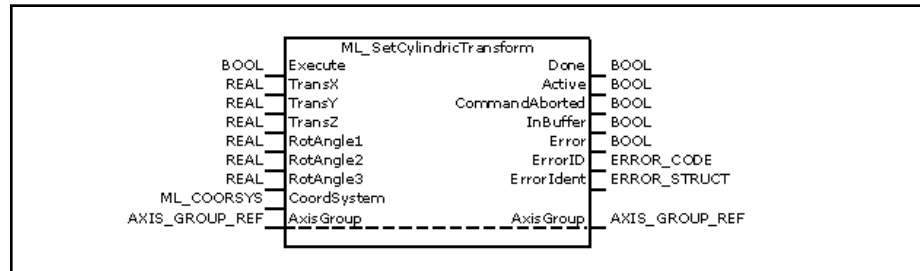


Abb. 5-41: Funktionsbaustein ML_SetCylindricTransform

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	TransX	REAL	X-Position des PCS im MCS
	TransY	REAL	Y-Position des PCS im MCS
	TransZ	REAL	Z-Position des PCS im MCS
	RotAngle1	REAL	Verdrehung des PCS um die x-Achse in MCS
	RotAngle2	REAL	Verdrehung des PCS um die y-Achse in MCS
	RotAngle3	REAL	Verdrehung des PCS um die z-Achse in MCS
	CoordSystem	ML_COORDSYS	Produkt-Koordinatensystem für das die Werte gesetzt werden sollen siehe ML_COORDSYS auf Seite 317
VAR_OUTPUT	Done	BOOL	Definition wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen andern Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-38: Funktionsbaustein ML_SetCylindricTransform

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
TransX	REAL			0.0	steigende Flanke an "Execute"

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
TransY	REAL			0.0	steigende Flanke an "Execute"
TransZ	REAL			0.0	steigende Flanke an "Execute"
RotAngle1	REAL			0.0	steigende Flanke an "Execute"
RotAngle2	REAL			0.0	steigende Flanke an "Execute"
RotAngle3	REAL			0.0	steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS	CS_PRODU CT_COOR- DINATES_1	CS_PRODU CT_COOR- DINA- TES_16	CS_PRODUCT_COOR DINATES_1	steigende Flanke an "Execute"

Tab. 5-39: Eingangsverhalten ML_SetCylindricTransform

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#0003	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-40: Fehlernummern des FBs ML_SetCylindricTransform

ML_SetRotaryTableConfiguration

Kurzbeschreibung Der Funktionsbaustein "ML_SetRotaryTableConfiguration" konfiguriert eine Kreisscheibe.
 (Siehe [Bandsynchrone Bewegung auf Seite 67](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

ML_Robot.library

Schnittstellenbeschreibung

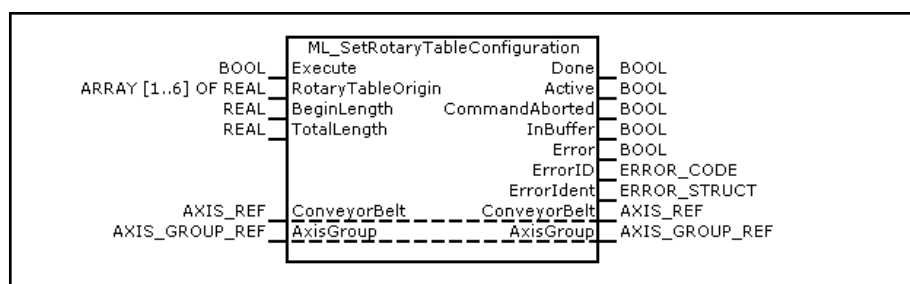


Abb. 5-42: Funktionsbaustein ML_SetRotaryTableConfiguration

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	RotaryTableOrigin	ARRAY [1..6] OF REAL	Definiert die Position und Verdrehung der Kreisscheibe relativ zum MCS
	BeginLength	REAL	Definiert die Länge des spätesten Starten einer bandsynchronen Bewegung ab der Startposition
	TotalLength	REAL	Definiert die Länge des bandsynchronen Verfahrbereichs ab der Startposition
VAR_OUTPUT	Done	BOOL	Definition wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen andern Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Achsreferenz der Bandachse

Tab. 5-41: Funktionsbaustein ML_SetRotaryTableConfiguration

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
ConveyorBeltOrigin	ARRAY [1..6] OF REAL			[6(0)]	Steigende Flanke an "Execute"
BeginLength	REAL	> 0.0		0.0	Steigende Flanke an "Execute"
TotalLength	REAL	> 0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-42: Eingangsverhalten ML_SetRotaryTableConfiguration

Signal-Zeit-Diagramm

- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
- [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#0003	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-43: Fehlernummern des FBs ML_SetRotaryTableConfiguration

ML_RobotSetMoveID

Kurzbeschreibung

Der Funktionsbaustein "ML_RobotSetMoveID" ändert die aktuelle Bewegungsidentifikationsnummer.



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Config

Schnittstellenbeschreibung

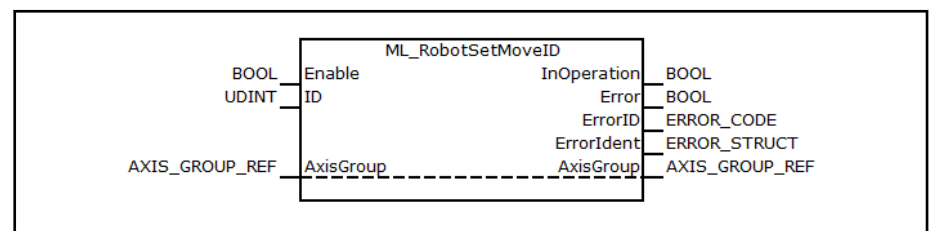


Abb. 5-43: FB ML_RobotSetMoveID

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	ID	UDINT	Neue Bewegungsidentifikationsnummer
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-44: Schnittstelle von FB ML_RobotSetMoveID

ML_Robot.library

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
ID	UDINT			0	Kontinuierlich wenn "Enable" = true

Tab. 5-45: Eingangsverhalten RobotSetMoveID

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-46: Fehlernummern des FBs ML_RobotRobotSetMoveID

5.3.5 Kinematic/ Info

MC_GroupReadActualPosition

Kurzbeschreibung Der Funktionsbaustein "MC_GroupReadActualPosition" liest die aktuelle Po-
sition der Kinematik.

 Eine Änderung der Kinematik am Eingang des Funktionsbausteines setzt einen erneuten 0/1-Übergang an "Enable" voraus.

 Der Funktionsbaustein kann nur für die Robot-Control V2 verwen-
det werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

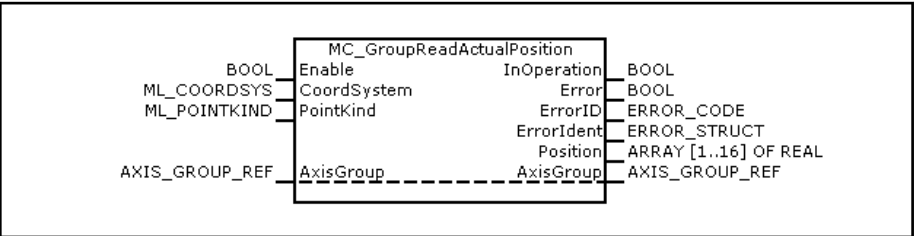


Abb. 5-44: FB MC_GroupReadActualPosition

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Positionswertes, solange "Enable" TRUE ist
	CoordSystem	ML_COORDSYS	Koordinatensystem des Punktes siehe ML_COORDSYS auf Seite 317
	PointKind	ML_POINTKIND	Art des Punktes in Bezug auf den zeitlichen Verlauf: Satzvorbereitung/ Satzabarbeitung siehe ML_POINTKIND auf Seite 315
VAR_OUTPUT	InOperation	BOOL	Gültiger Wert ist in seiner Task verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Position	ARRAY [1..16] OF REAL	Aktuelle Position der Kinematik im Raum
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-47: Schnittstelle von FB MC_GroupReadActualPosition

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
CoordSystem	ML_COORDSYS	CS_PRODU CT_COOR DINATES_1	CS_PRODU CT_COOR DINA- TES_16	CS_PRODUCT_COOR DINATES_1	Kontinuierlich wenn "Enable" = true
PointKind	ML_POINTKIND	POIN- TKIND_PRE- PARE	POIN- TKIND_EXE CUTE	POINTKIND_PREPA- RE	Kontinuierlich wenn "Enable" = true

Tab. 5-48: Eingangsverhalten GroupReadActualPosition

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_RobotGetBeltConfiguration

Kurzbeschreibung Der Funktionsbaustein "ML_RobotGetBeltConfiguration" gibt die aktuelle Konfiguration eines Bandes zurück.
 (Siehe [Bandsynchrone Bewegung auf Seite 67](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

ML_Robot.library

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

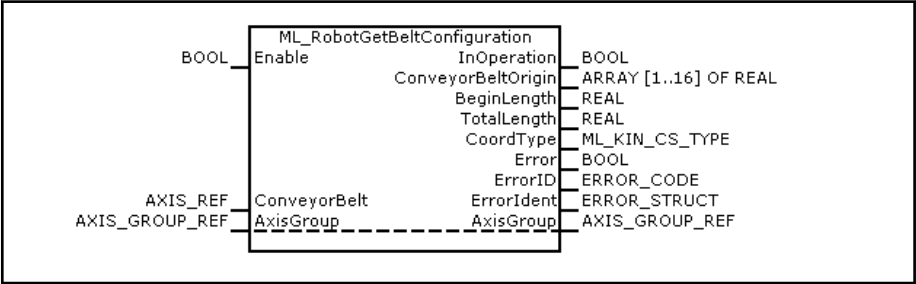


Abb. 5-45: Funktionsbaustein ML_RobotGetBeltConfiguration

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	ConveyorBeltOrigin	ARRAY [1..16] OF REAL	Definiert die Position und Verdrehung des Bandes relativ zum MCS
	BeginLength	REAL	Definiert die Länge des spätesten Starten einer bandsynchronen Bewegung ab der Startposition
	TotalLength	REAL	Definiert die Länge des bandsynchronen Verfahrbereichs ab der Startposition
	CoordType	ML_KIN_CS_TYPE	Art des Koordinatensystems (Siehe ML_KIN_CS_TYPE auf Seite 314)
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Achsreferenz des Bandes

Tab. 5-49: Funktionsbaustein ML_RobotGetBeltConfiguration

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich

Tab. 5-50: Eingangsverhalten ML_RobotGetBeltConfiguration

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-51: Fehlernummern des FBs ML_RobotGetBeltConfiguration

ML_RobotGetBeltDesyncRelConfiguration

Kurzbeschreibung Der Funktionsbaustein "ML_RobotGetBeltDesyncRelConfiguration" liest die Konfiguration für das relative Absynchronisieren eines Bandes.

(Siehe [Bandsynchrone Bewegung auf Seite 67](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

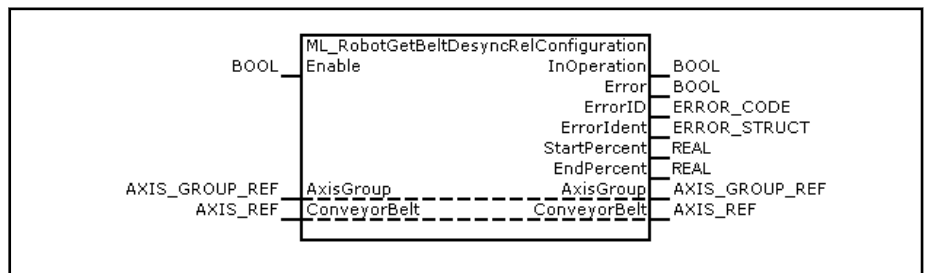


Abb. 5-46: Funktionsbaustein ML_RobotGetBeltDesyncRelConfiguration

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	StartPercent	REAL	Start Prozent zum relativen Absynchronisieren
	EndPercent	REAL	End Prozent zum relativen Absynchronisieren
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Achsreferenz der Bandachse

Tab. 5-52: Funktionsbaustein ML_RobotGetBeltDesyncRelConfiguration

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich

Tab. 5-53: Eingangsverhalten ML_RobotGetBeltDesyncRelConfiguration

ML_Robot.library

- Signal-Zeit-Diagramm
- Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine](#), permanente Bearbeitung auf Seite 202
- Fehlerbehandlung
- Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_RobotGetBeltErrorReaction

Kurzbeschreibung

Der Funktionsbaustein "ML_RobotGetBeltErrorReaction" gibt die konfigurierte Bandfehlerreaktion des Roboters zurück.
(Siehe [Mechanische Informationen auf Seite 183](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

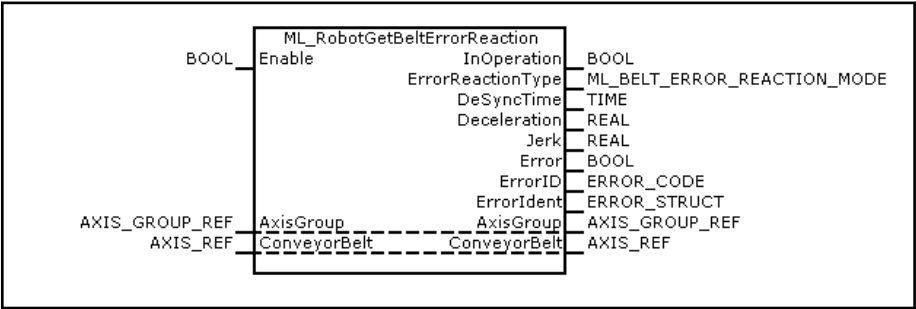


Abb. 5-47: Funktionsbaustein ML_RobotGetBeltErrorReaction

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	ErrorReactionType	ML_BELT_ERROR_REACTION_MODE	Konfigurierte Fehlerreaktion
	DeSyncTime	TIME	Konfigurierte Zeit zum Absynchronisieren
	Deceleration	REAL	Konfigurierte maximale Bahnverzögerung
	Jerk	REAL	Konfigurierter maximaler Bahnruck
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Referenz auf das Band

Tab. 5-54: Funktionsbaustein ML_RobotGetBeltErrorReaction

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich

Tab. 5-55: Eingangsverhalten ML_RobotGetBeltErrorReaction

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-56: Fehlernummern des FBs ML_RobotGetBeltErrorReaction

ML_RobotGetCmdInfo

Kurzbeschreibung Der Funktionsbaustein "ML_RobotGetCmdInfo" gibt Informationen zu allen Kommandos im Satzpuffer zurück.



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

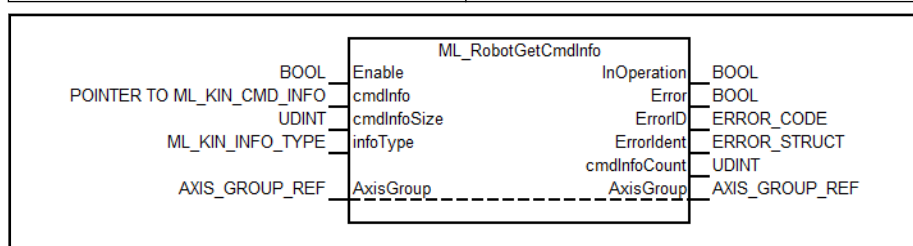


Abb. 5-48: Funktionsbaustein ML_RobotGetCmdInfo

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	cmdInfo	POINTER TO ML_KIN_CMD_INFO	Zeiger auf das zu füllende Array der Informationen zu Kommandos im Satzpuffer
	cmdInfoSize	UDINT	Anzahl der verfügbaren Elemente im Array der Informationen
	infoType	ML_KIN_INFO_TYPE	Art der Information, die ausgegeben werden soll

ML_Robot.library

	Name	Typ	Kommentar
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	cmdInfoCount	UDINT	Anzahl gelesener Daten im Array der Informationen
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-57: Funktionsbaustein ML_RobotGetCmdInfo

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
cmdInfo	POINTER TO ML_KIN_MOVE_IN- FO			NULL	Kontinuierlich wenn "Enable" = true
cmdInfoSize	UDINT			0	Kontinuierlich wenn "Enable" = true
infoType	ML_KIN_INFO_TY- PE			KIN_INFO_TY- PE_PROGRESS	Kontinuierlich wenn "Enable" = true

Tab. 5-58: Eingangsverhalten ML_RobotGetCmdInfo

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-59: Fehlernummern des FBs ML_RobotGetCmdInfo

ML_RobotGetMechanic

Kurzbeschreibung Der Funktionsbaustein "ML_RobotGetMechanic" gibt mechanische Daten der Kinematik zurück.
(Siehe [Mechanische Informationen auf Seite 183](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Schnittstellenbeschreibung

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

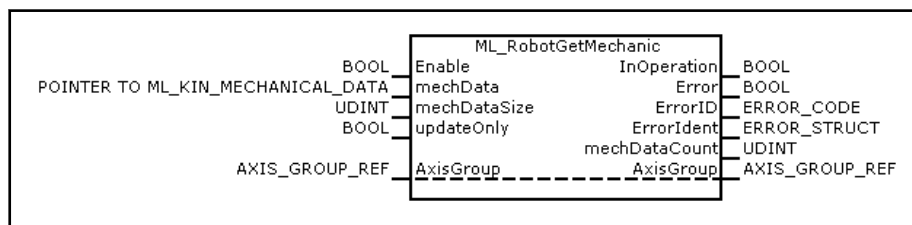


Abb. 5-49: Funktionsbaustein ML_RobotGetMechanic

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	mechData	POINTER TO ML_KIN_MECHANICAL_DATA	Zeiger auf das zu füllende Array mechanischer Daten
	mechDataSize	UDINT	Anzahl der verfügbaren Elemente im Array der mechanischen Daten
	updateOnly	BOOL	Soll ein Update oder die kompletten mechanischen Daten angefragt werden
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailed information about the error
	mechDataCount	UDINT	Anzahl gelesener Daten im Array mechanischer Daten
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-60: Funktionsbaustein ML_RobotGetMechanic

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
mechData	POINTER TO ML_KIN_MECHANICAL_DATA			NULL	Kontinuierlich wenn "Enable" = true
mechDataSize	UDINT			0	Kontinuierlich wenn "Enable" = true
updateOnly	BOOL			FALSE	Kontinuierlich wenn "Enable" = true

Tab. 5-61: Eingangsverhalten ML_RobotGetMechanic

Signal-Zeit-Diagramm

Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine](#), permanente Bearbeitung auf Seite 202

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

ML_Robot.library

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-62: Fehlernummern des FBs ML_RobotGetMechanic

ML_RobotGetMoveID

Kurzbeschreibung

Der Funktionsbaustein "ML_RobotGetMoveID" gibt die aktuelle Bewegungsidentifikationsnummer zurück.



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

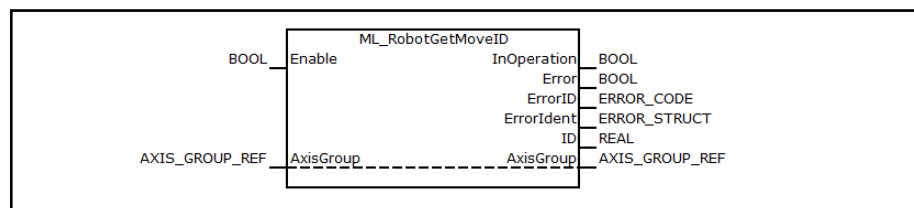


Abb. 5-50: FB ML_RobotGetMoveID

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	ID	REAL	Aktuelle Bewegungsidentifikationsnummer
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-63: Schnittstelle von FB ML_RobotGetMoveID

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich

Tab. 5-64: Eingangsverhalten ML_RobotGetMoveID

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-65: Fehlernummern des FBs ML_RobotGetMoveID

ML_RobotGetPerformance

Kurzbeschreibung

Der Funktionsbaustein "ML_RobotGetPerformance" gibt Performance-Daten der Kinematik zurück.

(Siehe [Performancemessung auf Seite 85](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

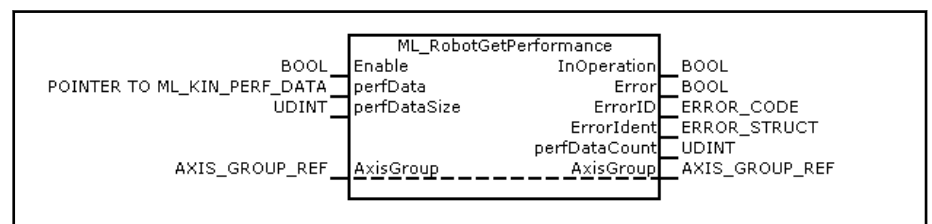


Abb. 5-51: Funktionsbaustein ML_RobotGetPerformance

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	perfData	POINTER TO ML_KIN_PERF_DATA	Zeiger auf das zu füllende Array der Performance-Daten
	perfDataSize	UDINT	Anzahl der verfügbaren Elemente im Array der Performance-Daten
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	perfDataCount	UDINT	Anzahl gelesener Daten im Array der Performance-Daten
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-66: Funktionsbaustein ML_RobotGetPerformance

ML_Robot.library

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
perfData	POINTER TO ML_KIN_PERF_DATA			NULL	Kontinuierlich wenn "Enable" = true
perfDataSize	UDINT			0	Kontinuierlich wenn "Enable" = true

Tab. 5-67: Eingangsverhalten ML_RobotGetPerformance

Signal-Zeit-Diagramm

Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine](#), permanente Bearbeitung auf Seite 202

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-68: Fehlernummern des FBs ML_RobotGetPerformance

ML_RobotGetSafeZones

Kurzbeschreibung

Der Funktionsbaustein "ML_RobotGetSafeZones" gibt die Schutzzone einer Kinematik zurück.



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

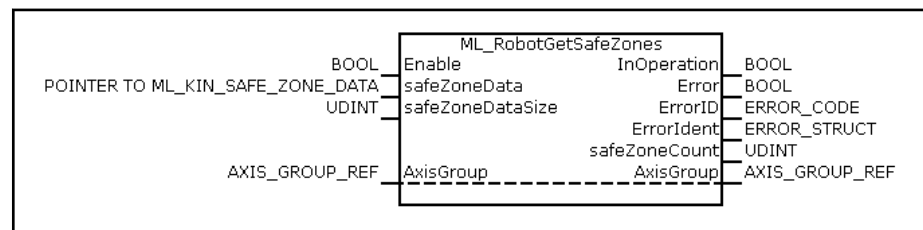


Abb. 5-52: FB ML_RobotGetSafeZones

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	safeZoneData	POINTER TO ML_KIN_SAFE_ZONE_DATA	Zeiger auf das zu füllende Array der Schutzzonen-Daten Siehe ML_KIN_SAFE_ZONE_DATA auf Seite 315
	safeZoneDataSize	UDINT	Anzahl der verfügbaren Elemente im Array der Schutzzonen-Daten
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	safeZoneCount	UDINT	Anzahl gelesener Daten im Array der Schutzzonen-Daten
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-69: Schnittstelle von ML_RobotGetSafeZones

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
safeZoneData	POINTER TO ML_KIN_SAFE_ZONE_DATA			NULL	Kontinuierlich wenn "Enable" = true
safeZoneDataSize	UDINT			0	Kontinuierlich wenn "Enable" = true

Tab. 5-70: Eingangsverhalten ML_RobotGetSafeZones

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-71: Fehlernummern des FBs ML_RobotGetSafeZones

ML_RobotGetTrafoValues

Kurzbeschreibung Der Funktionsbaustein "ML_RobotGetTrafoValues" gibt die aktuelle Konfiguration eines Produkt- oder des Basis-Koordinatensystems zurück.
 (Siehe [Koordinatensysteme auf Seite 24](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

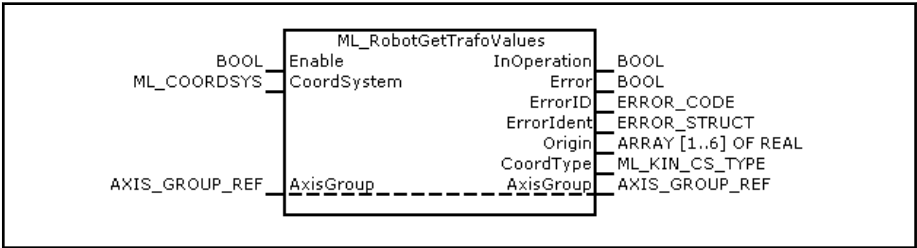


Abb. 5-53: Funktionsbaustein ML_RobotGetTrafoValues

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	CoordSystem	ML_COORDSYS	Koordinatensystem des Punktes (nur Produkt- oder Basis-Koordinatensystem zulässig!)
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Origin	ARRAY [1..6] OF REAL	Position und Orientierung des Koordinatensystems
	CoordType	ML_KIN_CS_TYPE	Art des Koordinatensystems
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-72: Funktionsbaustein ML_RobotGetTrafoValues

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Kontinuierlich wenn "Enable" = true

Tab. 5-73: Eingangsverhalten ML_RobotGetTrafoValues

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-74: Fehlernummern des FBs ML_RobotGetTrafoValues

ML_RobotMovePoint

Kurzbeschreibung

Der Funktionsbaustein "ML_RobotMovePoint" verschiebt einen Punkt an die aktuelle Position im Raum. Üblicherweise wird die Funktion für Punkte aufgerufen, die sich auf einem Band befinden. Enthält der Punkt einen oder mehrere Bandwerte, so wird der Punkt durch diese Funktion an den aktuellen Bandwert verschoben. Der Punkt auf dem Band hat nach dem Aufruf die tatsächliche momentane Position im Raum mit dem aktuellen Bandwert im angegebenen Koordinatensystem.

Hat der Punkt keinen Bandwert, wird der Punkt durch die Funktion nicht modifiziert.

(Siehe [Bandsynchrone Bewegung auf Seite 67](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

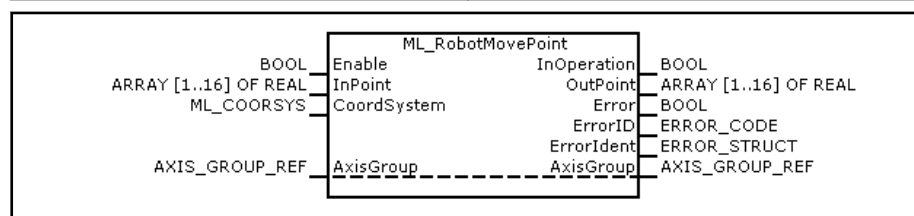


Abb. 5-54: Funktionsbaustein ML_RobotMovePoint

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	InPoint	ARRAY [1..16] OF REAL	Punkt im Raum, von dem die aktuelle Position erfragt werden soll
	CoordSystem	ML_COORDSYS	Koordinatensystem des Punktes
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

ML_Robot.library

	Name	Typ	Kommentar
	OutPoint	ARRAY [1..16] OF REAL	Aktuelle Position des Punktes im angegebenen Koordinatensystem
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-75: Funktionsbaustein ML_RobotMovePoint

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
InPoint	ARRAY [1..16] OF REAL			[16(0)]	Kontinuierlich wenn "Enable" = true
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Kontinuierlich wenn "Enable" = true

Tab. 5-76: Eingangsverhalten ML_RobotMovePoint

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine](#), permanente Bearbeitung auf Seite 202

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
			48
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-77: Fehlernummern des FBs ML_RobotMovePoint

ML_RobotReadCyclicChannel

Kurzbeschreibung Der Funktionsbaustein "ML_RobotReadCyclicChannel" liest die aktuelle zyklisch kommandierte Position.
(Siehe [Zyklische Sollwertvorgabe auf Seite 83](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

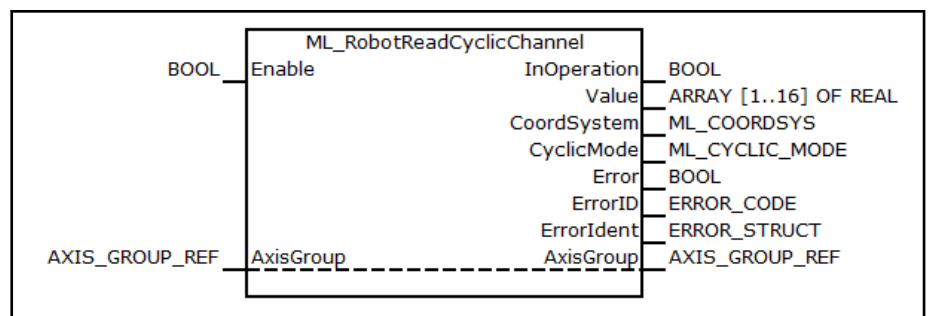


Abb. 5-55: Funktionsbaustein ML_RobotReadCyclicChannel

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Value	ARRAY [1..16] OF REAL	Letzte zyklisch kommandierte Sollwerte, abhängig von der Art der zyklischen Sollwertvorgabe (CyclicMode)
	CoordSystem	ML_COORDSYS	Koordinatensystem der letzten zyklisch kommandierten Position ML_COORDSYS auf Seite 317
	CyclicMode	ML_CYCLIC_MODE	Art der zyklischen Sollwertvorgabe ML_CYCLIC_MODE auf Seite 318
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-78: Funktionsbaustein ML_RobotReadCyclicChannel

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich

Tab. 5-79: Eingangsverhalten ML_RobotReadCyclicChannel

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-80: Fehlernummern des FBs ML_RobotReadCyclicChannel

ML_Robot.library

ML_RobotSetPerformanceEnable

Kurzbeschreibung Der Funktionsbaustein "ML_RobotSetPerformanceEnable" ändert den Aktivierungszustand der Performance-Messung.
(Siehe [Performancemessung auf Seite 85](#)).

 Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS	ML_Robot/Kinematic Info
Funktionspaket ROB	

Schnittstellenbeschreibung

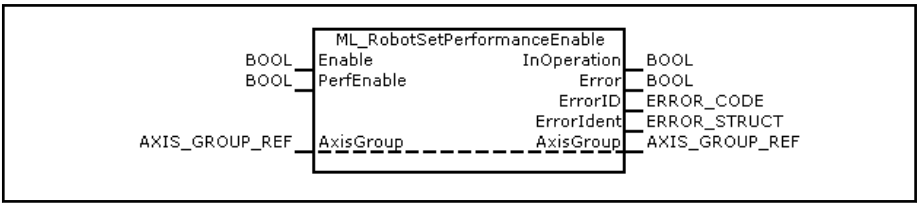


Abb. 5-56: Funktionsbaustein ML_RobotSetPerformanceEnable

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	PerfEnable	BOOL	Neuer Aktivierungszustand der Performance-Messung
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-81: Funktionsbaustein ML_RobotSetPerformanceEnable

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
PerfEnable	BOOL			FALSE	Kontinuierlich wenn "Enable" = true

Tab. 5-82: Eingangsverhalten ML_RobotSetPerformanceEnable

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-83: Fehlernummern des FBs ML_RobotSetPerformanceEnable

ML_RobotTransformPoint

Kurzbeschreibung

Der Funktionsbaustein "ML_RobotTransformPoint" transformiert einen Punkt von einem Koordinatensystem in ein anderes Koordinatensystem.

(Siehe [Koordinatensysteme auf Seite 24](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

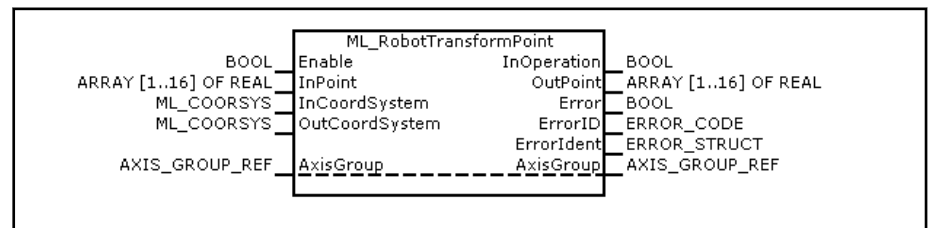


Abb. 5-57: Funktionsbaustein ML_RobotTransformPoint

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	InPoint	ARRAY [1..16] OF REAL	Punkt im Raum vom dem die aktuelle Position erfragt werden soll
	InCoordSystem	ML_COORDSYS	Koordinatensystem des Punktes ML_COORDSYS auf Seite 317
	OutCoordSystem	ML_COORDSYS	Koordinatensystem, in das der Punkt transformiert werden soll
VAR_OUTPUT	InOperation	BOOL	Definition wurde abgeschlossen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	OutPoint	ARRAY [1..16] OF REAL	Aktuelle Position des Punktes im Raum
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-84: Funktionsbaustein ML_RobotTransformPoint

ML_Robot.library

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
InPoint	ARRAY [1..16] OF REAL			[16(0)]	Kontinuierlich wenn "Enable" = true
InCoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Kontinuierlich wenn "Enable" = true
OutCoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Kontinuierlich wenn "Enable" = true

Tab. 5-85: Eingangsverhalten ML_RobotTransformPoint

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-86: Fehlernummern des FBs ML_RobotTransformPoint

ML_SwitchOnPath

Kurzbeschreibung Der Funktionsbaustein "ML_SwitchOnPath" bietet die Möglichkeit, auf einer Bahn zu schalten.

Man kann während einer Verfahrbewegung positionsgenau und mit zeitlichem Vorhalt SPS-Ausgänge ansteuern.

(Siehe [Bandsynchrone Bewegung auf Seite 67](#)).

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Info

Schnittstellenbeschreibung

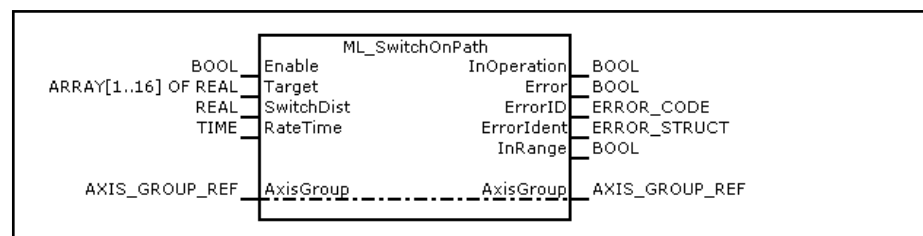


Abb. 5-58: Funktionsbaustein ML_SwitchOnPath

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Target	ARRAY[1..16] OF REAL	Zielposition, an der geschaltet werden soll (Raumposition)
	SwitchDist	REAL	Distanz/Radius von der Zielposition zur Schaltposition
	RateTime	TIME	Vorhaltezeit
VAR_OUTPUT	InOperation	BOOL	Bearbeitung wird ausgeführt
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InRange	BOOL	Kinematik im Schaltbereich
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-87: Funktionsbaustein ML_SwitchOnPath

Name	Typ	Default-Wert	Übernahme
Enable	BOOL	FALSE	Kontinuierlich
Target	ARRAY[1..16] OF REAL	16(0.0)	Kontinuierlich
SwitchDist	REAL	16(0.0)	Kontinuierlich
RateTime	TIME	0.0	Kontinuierlich

Tab. 5-88: Eingangsverhalten ML_SwitchOnPath

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch

Tab. 5-89: Fehlernummern des FBs ML_SwitchOnPath

5.3.6 Kinematic/ Movement

MC_GroupContinue

Kurzbeschreibung Der Funktionsbaustein "MC_GroupContinue" startet eine durch den Funktionsbaustein [MC_GroupInterrupt auf Seite 249](#) unterbrochene Bewegung einer Kinematik wieder.

(Siehe [Wiederaufnahme von Bewegungen auf Seite 66](#))



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

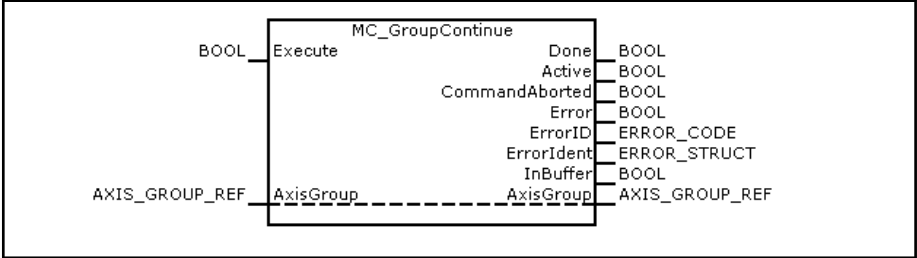


Abb. 5-59: Funktionsbaustein MC_GroupContinue

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Startet mit steigender Flanke den Funktionsbaustein
VAR_OUTPUT	Done	BOOL	Alle unterbrochenen Bewegungen sind wieder gestartet
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satzpuffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-90: Funktionsbaustein MC_GroupContinue

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich

Tab. 5-91: Eingangsverhalten MC_GroupContinue

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MC_GroupInterrupt

Kurzbeschreibung Der Funktionsbaustein "MC_GroupInterrupt" unterbricht die Bewegung einer Kinematik.

(Siehe [Unterbrechen von Bewegungen auf Seite 65](#)).

Die Bewegung kann mit dem Funktionsbaustein [MC_GroupContinue auf Seite 247](#) wieder aufgenommen werden.



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

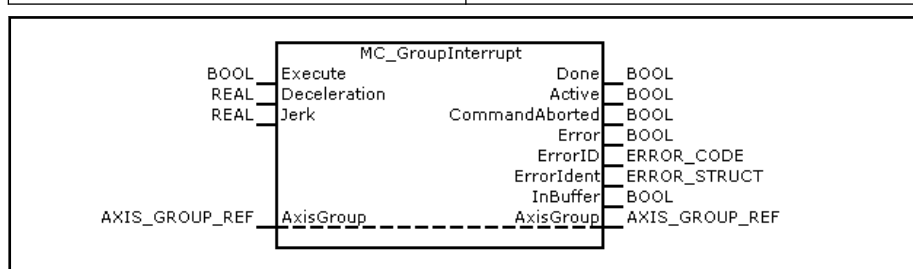


Abb. 5-60: Funktionsbaustein MC_GroupInterrupt

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Startet mit steigender Flanke den Funktionsbaustein
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
VAR_OUTPUT	Done	BOOL	Geschwindigkeit 0 ist erreicht
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satzpuffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-92: Funktionsbaustein MC_GroupInterrupt

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-93: Eingangsverhalten MC_GroupInterrupt

ML_Robot.library

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MC_GroupStop

Kurzbeschreibung Der Funktionsbaustein "MC_GroupStop" stoppt kontrolliert eine Kinematik.
(Siehe [Stoppen von Bewegungen auf Seite 64](#)).

Weitere Kommandos können erst abgesetzt werden, wenn am Funktionsbaustein der "Done"-Ausgang gesetzt ist und am "Execute"-Eingang eine fallende Flanke angelegt wird.



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

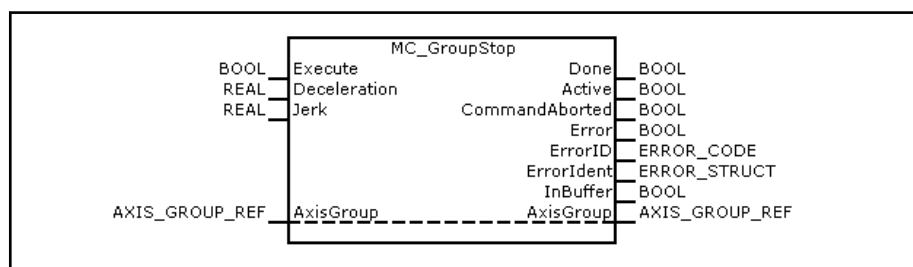


Abb. 5-61: Funktionsbaustein MC_GroupStop

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Startet mit steigender Flanke den Funktionsbaustein
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
VAR_OUTPUT	Done	BOOL	Geschwindigkeit 0 ist erreicht
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

	Name	Typ	Kommentar
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satzpuffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-94: Funktionsbaustein MC_GroupStop

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-95: Eingangsverhalten MC_GroupStop

Signal-Zeit-Diagramm

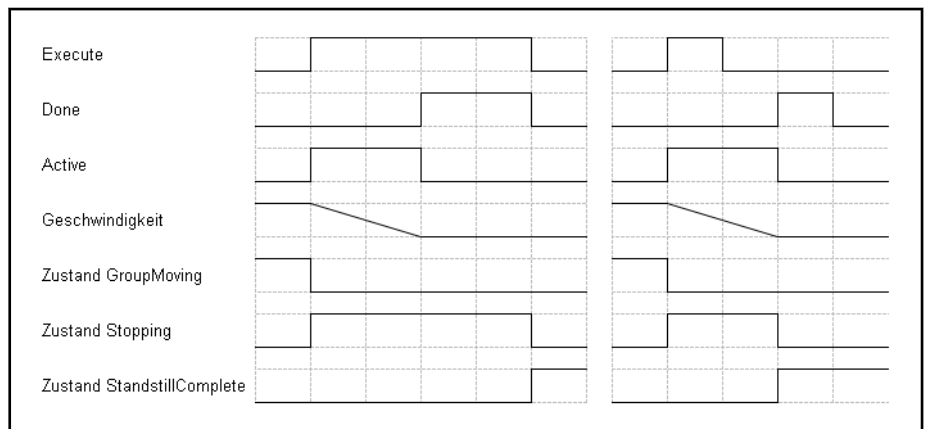


Abb. 5-62: Zeitdiagramm des MC_GroupStop

- [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MC_MoveCircularAbsolute

Kurzbeschreibung

Der Funktionsbaustein "MC_MoveCircularAbsolute" bewegt eine Kinematik zirkular von der zuletzt kommandierten Position des TCP über einen Zwischenpunkt zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

ML_Robot.library

Schnittstellenbeschreibung

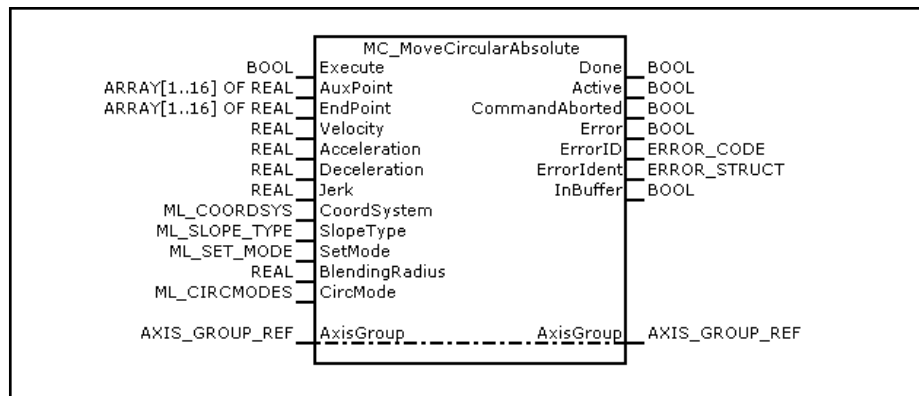


Abb. 5-63: Funktionsbaustein MC_MoveCircularAbsolute

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	AuxPoint	ARRAY[1..16] OF REAL	Zwischenposition der Bewegung
	EndPoint	ARRAY[1..16] OF REAL	Zielposition der Bewegung
	Velocity	REAL	Maximale Bahngeschwindigkeit
	Acceleration	REAL	Maximale Bahnbeschleunigung
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS auf Seite 317
	SlopeType	ML_SLOPE_TYPE	Slope-Art der Bewegung ML_SLOPE_TYPE auf Seite 319
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE auf Seite 319
	BlendingRadius	REAL	Überschleifradius Robot-Control V1: Überschleifradius in Prozent von A-0-0230 der beteiligten Achsen Robot-Control V2: R>0 --> VIA; R=0 --> TO
	CircMode	ML_CIRCMODES	Art der Kreisbewegung ML_CIRCMODES auf Seite 317
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

	Name	Typ	Kommentar
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-96: Funktionsbaustein MC_MoveCircularAbsolute

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
AuxPoint	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
EndPoint	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Execute"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke an "Execute"
SlopeType	ML_SLOPE_TYPE			ST_BLOCK_SLOPE	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0		0.0	Steigende Flanke an "Execute"
CircMode	ML_CIRCMODES			CM_BORDER	Steigende Flanke an "Execute"

Tab. 5-97: Eingangsverhalten MC_MoveCircularAbsolute

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_Robot.library

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#0000	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#0000	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#0000	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes auf Seite 216 zur Kinematik gruppiert worden

Tab. 5-98: Fehlernummern des FBs MC_MoveCircularAbsolute

MC_MoveCircularRelative

Kurzbeschreibung

Der Funktionsbaustein "MC_MoveCircularRelative" bewegt eine Kinematik zirkular von der zuletzt kommandierten Position des TCP über einen Zwischenpunkt zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem.

Der Zwischen- und der Endpunkt wird dabei relativ zum Startpunkt angegeben.



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

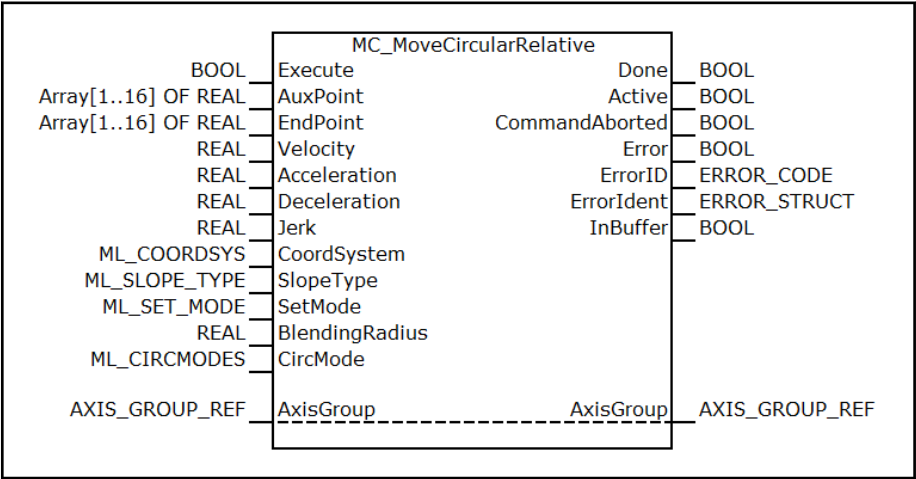


Abb. 5-64: Funktionsbaustein MC_MoveCircularRelative

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	AuxPoint	ARRAY[1..16] OF REAL	Zwischenposition der Bewegung, relative Angabe zum Startpunkt
	EndPoint	ARRAY[1..16] OF REAL	Zielposition der Bewegung, relative Angabe zum Startpunkt
	Velocity	REAL	Maximale Bahngeschwindigkeit
	Acceleration	REAL	Maximale Bahnbeschleunigung
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS auf Seite 317
	SlopeType	ML_SLOPE_TYPE	Slope-Art der Bewegung ML_SLOPE_TYPE auf Seite 319
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE auf Seite 319
	BlendingRadius	REAL	Überschleifradius
	CircMode	ML_CIRCMODES	Art der Kreisbewegung ML_CIRCMODES auf Seite 317
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen andern Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-99: Funktionsbaustein MC_MoveCircularRelative

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
AuxPoint	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
EndPoint	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Execute"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"

ML_Robot.library

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke an "Execute"
SlopeType	ML_SLOPE_TYPE			ST_BLOCK_SLOPE	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0		0.0	Steigende Flanke an "Execute"
CircMode	ML_CIRCMODES			CM_BORDER	Steigende Flanke an "Execute"

Tab. 5-100: Eingangsverhalten MC_MoveCircularRelative

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#0000	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#0000	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#0000	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes auf Seite 216 zur Kinematik gruppiert worden

Tab. 5-101: Fehlernummern des FBs MC_MoveCircularRelative

MC_MoveDirectAbsolute

Kurzbeschreibung

Der Funktionsbaustein "MC_MoveDirectAbsolute" bewegt eine Kinematik Punkt-zu-Punkt von der zuletzt kommandierten Position des TCP zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

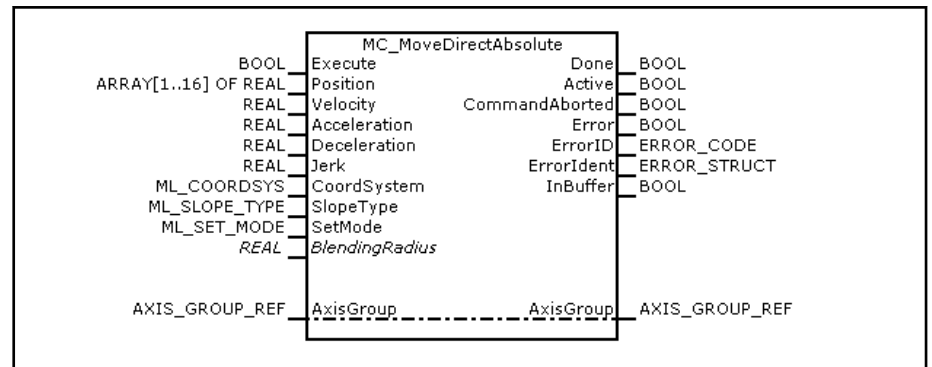


Abb. 5-65: Funktionsbaustein MC_MoveDirectAbsolute

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Position	ARRAY[1..16] OF REAL	Zielposition der Bewegung
	Velocity	REAL	Prozentsatz der max. Achsgeschwindigkeit (Achsparemeter A-0-0032 und A-0-0033)
	Acceleration	REAL	Prozentsatz der max. Achsbeschleunigung (Achsparemeter A-0-0034)
	Deceleration	REAL	Prozentsatz der max. Achsverzögerung (Achsparemeter A-0-0034)
	Jerk	REAL	Prozentsatz des max. Achsrucks (Achsparemeter A-0-0036)
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS auf Seite 317
	SlopeType	ML_SLOPE_TYPE	Slope-Art der Bewegung ML_SLOPE_TYPE auf Seite 319
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE auf Seite 319
	BlendingRadius	REAL	Robot-Control V1: Überschleifradius in Prozent von A-0-0230 der beteiligten Achsen Robot-Control V2: R>0 --> VIA; R=0 --> TO
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen

ML_Robot.library

	Name	Typ	Kommentar
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-102: Funktionsbaustein MC_MoveDirectAbsolute

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Position	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Execute"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	Real	>=0.0		100.0	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke an "Execute"
SlopeType	ML_SLOPE_TYPE			ST_BLOCK_SLOPE	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0	1000.0	0.0	Steigende Flanke an "Execute"

Tab. 5-103: Eingangsverhalten MC_MoveDirectAbsolute

Signal-Zeit-Diagramm

- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
- [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#00	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#00	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes zur Kinematik gruppiert worden

Tab. 5-104: Fehlernummern des FBs MC_MoveDirectAbsolute

MC_MoveDirectRelative

Kurzbeschreibung

Der Funktionsbaustein "MC_MoveDirectRelative" bewegt eine Kinematik Punkt-zu-Punkt von der zuletzt kommandierten Position des TCP zu einer vorgegebenen relativen Zielposition im angegebenen Koordinatensystem.



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

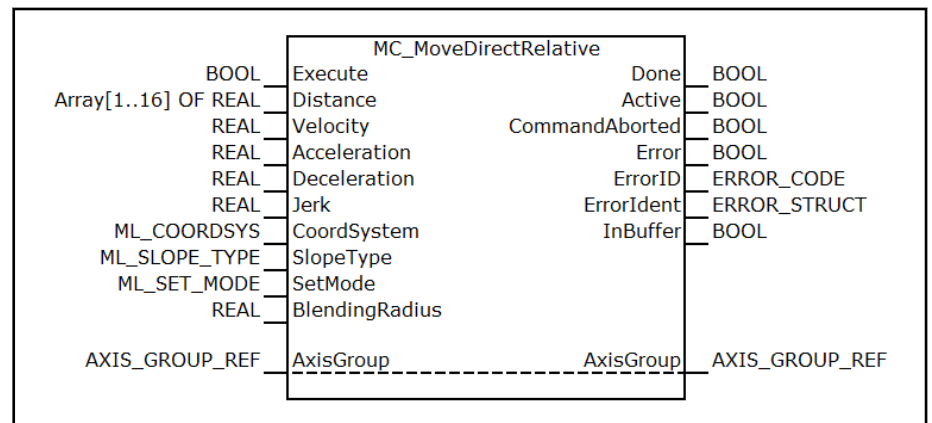


Abb. 5-66: Funktionsbaustein MC_MoveDirectRelative

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Distance	ARRAY[1..16] OF REAL	Zielposition der Bewegung relativ zum Startpunkt der Bewegung
	Velocity	REAL	Prozentsatz der max. Achsgeschwindigkeit (Achsparemeter A-0-0032 und A-0-0033)

ML_Robot.library

	Name	Typ	Kommentar
	Acceleration	REAL	Prozentsatz der max. Achsbeschleunigung (Achsparemeter A-0-0034)
	Deceleration	REAL	Prozentsatz der max. Achsverzögerung (Achsparemeter A-0-0034)
	Jerk	REAL	Prozentsatz des max. Achsrucks (Achsparemeter A-0-0036)
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS auf Seite 317
	SlopeType	ML_SLOPE_TYPE	Slope-Art der Bewegung ML_SLOPE_TYPE auf Seite 319
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE auf Seite 319
	BlendingRadius	REAL	Robot-Control V2: R>0 --> VIA; R=0 --> TO
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen andern Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-105: Funktionsbaustein MC_MoveDirectRelative

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Position	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Execute"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	Real	>=0.0		100.0	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke an "Execute"
SlopeType	ML_SLOPE_TYPE			ST_BLOCK_SLOPE	Steigende Flanke an "Execute"

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0	1000.0	0.0	Steigende Flanke an "Execute"

Tab. 5-106: Eingangsverhalten MC_MoveDirectRelative

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG***V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#00	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#00	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes zur Kinematik gruppiert worden

Tab. 5-107: Fehlernummern des FBs MC_MoveDirectRelative

MC_MoveLinearAbsolute

Kurzbeschreibung Der Coordinated Motion Funktionsbaustein "MC_MoveLinearAbsolute" bewegt eine Kinematik linear von der zuletzt kommandierten Position des TCP zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

ML_Robot.library

Schnittstellenbeschreibung

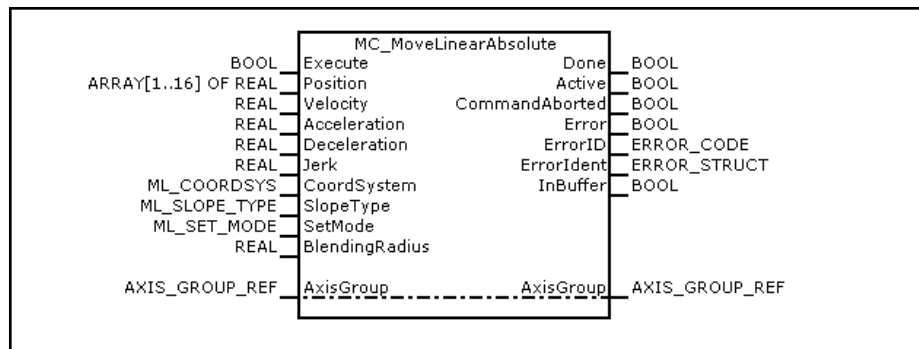


Abb. 5-67: Funktionsbaustein MC_MoveLinearAbsolute

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Position	ARRAY[1..16] OF REAL	Absolute Zielposition der Bewegung
	Velocity	REAL	Maximale Bahngeschwindigkeit
	Acceleration	REAL	Maximale Bahnbeschleunigung
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS auf Seite 317
	SlopeType	ML_SLOPE_TYPE	Slope-Art der Bewegung ML_SLOPE_TYPE auf Seite 319
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE auf Seite 319
	BlendingRadius	REAL	Überschleifradius R>0 --> VIA; R=0 --> TO
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen andern Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-108: Funktionsbaustein MC_MoveLinearAbsolute

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Position	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Execute"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke an "Execute"
SlopeType	ML_SLOPE_TYPE			ST_BLOCK_SLOPE	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-109: Eingangsverhalten MC_MoveLinearAbsolute

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_Robot.library

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#00	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#00	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes zur Kinematik gruppiert worden

Tab. 5-110: Fehlernummern des FBs MC_MoveLinearAbsolute

MC_MoveLinearRelative

Kurzbeschreibung

Der Funktionsbaustein "MC_MoveLinearRelative" bewegt eine Kinematik linear von der zuletzt kommandierten Position des TCP zu einer vorgegebenen relativen Zielposition im angegebenen Koordinatensystem.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

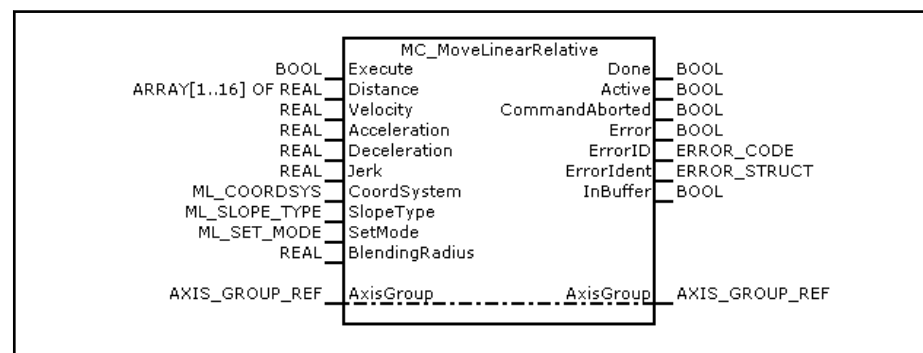


Abb. 5-68: Funktionsbaustein MC_MoveLinearRelative

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Distance	ARRAY[1..16] OF REAL	Relative Entfernung der Bewegung
	Velocity	REAL	Maximale Bahngeschwindigkeit
	Acceleration	REAL	Maximale Bahnbeschleunigung
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS auf Seite 317

	Name	Typ	Kommentar
	SlopeType	ML_SLOPE_TYPE	Slope-Art der Bewegung ML_SLOPE_TYPE auf Seite 319
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE auf Seite 319
	BlendingRadius	REAL	Überschleifradius Robot-Control V1: Überschleifradius in Prozent von A-0-0230 der beteiligten Achsen Robot-Control V2: R>0 --> VIA; R=0 --> TO
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen andern Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satzbuffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-111: Funktionsbaustein MC_MoveLinearRelative

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Position	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Execute"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke an "Execute"
SlopeType	ML_SLOPE_TYPE			ST_BLOCK_SLOPE	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-112: Eingangsverhalten MC_MoveLinearRelative

ML_Robot.library

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#0000	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#0000	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#0000	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes zur Kinematik gruppiert worden

Tab. 5-113: Fehlernummern des FBs MC_MoveLinearRelative

ML_BeltDeSynchronisationWithLimits

Kurzbeschreibung Der Funktionsbaustein "ML_BeltDeSynchronisationWithLimits" synchronisiert mit Bewegungsgrenzwerten von einem Band ab.
(Siehe [Bandsynchrone Bewegung auf Seite 67](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

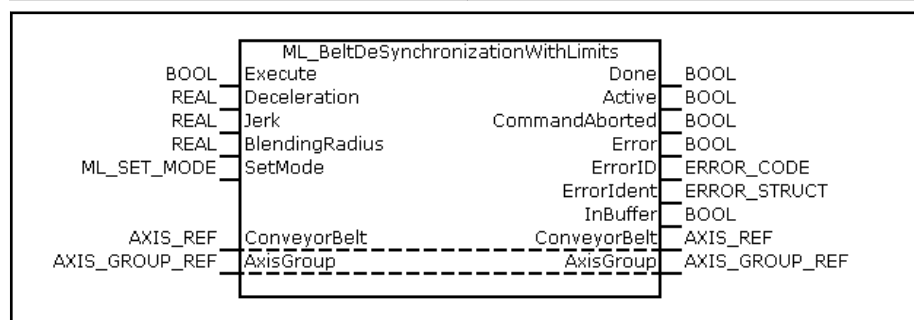


Abb. 5-69: Funktionsbaustein ML_BeltDeSynchronisationWithLimits

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
	BlendingRadius	REAL	Überschleifradius
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE, Seite 319
VAR_OUTPUT	Done	BOOL	Anmeldung der Transformation abgeschlossen
	Active	BOOL	Anmeldung der Transformation aktiv
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Achsreferenz der Bandachse

Tab. 5-114: Funktionsbaustein ML_BeltDeSynchronisationWithLimits

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0		0.0	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"

Tab. 5-115: Eingangsverhalten ML_BeltDeSynchronisationWithLimits

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_BeltSynchronisationWithLimits

Kurzbeschreibung Der Funktionsbaustein "ML_BeltSynchronisationWithLimits" synchronisiert mit Bewegungsgrenzwerten auf ein Band auf.
 (Siehe [Bandsynchrone Bewegung auf Seite 67](#)).

ML_Robot.library



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

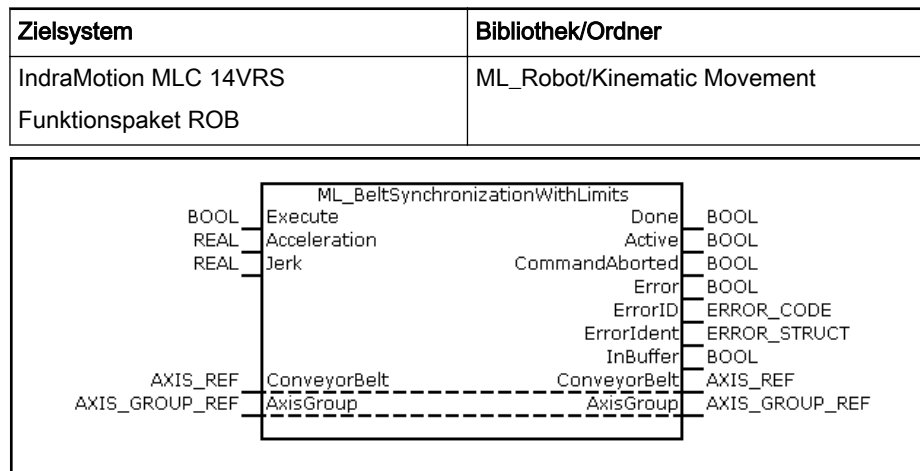
Schnittstellenbeschreibung

Abb. 5-70: Funktionsbaustein ML_BeltSynchronisationWithLimits

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Acceleration	REAL	Maximale Bahnbeschleunigung
	Jerk	REAL	Maximaler Bahnruck
VAR_OUTPUT	Done	BOOL	Anmeldung der Transformation abgeschlossen
	Active	BOOL	Anmeldung der Transformation aktiv
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt.
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik
	ConveyorBelt	AXIS_REF	Achsreferenz der Bandachse

Tab. 5-116: Funktionsbaustein ML_BeltSynchronisationWithLimits

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-117: Eingangsverhalten ML_BeltSynchronisationWithLimits

Signal-Zeit-Diagramm

- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
- [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_GroupJogCont

Kurzbeschreibung

Der Coordinated Motion Funktionsbaustein ML_GroupJogCont bewegt eine Kinematik kontinuierlich in eine Richtung.

Die Bewegung stoppt mit einer fallenden Flanke am "Enable"-Eingang.

Nähere Informationen siehe: [Handbetrieb auf Seite 49](#)



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

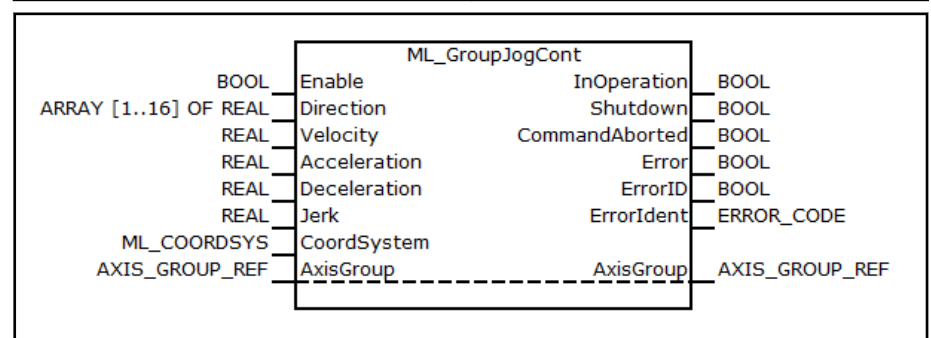


Abb. 5-71: Funktionsbaustein ML_GroupJogCont

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Steigende Flanke: Start der Bewegung Fallende Flanke: Ende der Bewegung (Startet das Abbremsen)
	Direction	ARRAY[1..16] OF REAL	Richtungsvektor der Bewegung
	Velocity	REAL	Maximale Bahngeschwindigkeit CoordSystem = CS_AXES_COORDINATES: Angabe in Prozent CoordSystem ≠ CS_AXES_COORDINATES: Angabe in mm/s bzw. in/s
	Acceleration	REAL	Maximale Bahnbeschleunigung CoordSystem = CS_AXES_COORDINATES: Angabe in Prozent CoordSystem ≠ CS_AXES_COORDINATES: Angabe in mm/s² bzw. in/s²

ML_Robot.library

	Name	Typ	Kommentar
	Deceleration	REAL	Maximale Bahnverzögerung CoordSystem = CS_AXES_COORDINATES: Angabe in Prozent CoordSystem ≠ CS_AXES_COORDINATES: Angabe in mm/s ² bzw. in/s ²
	Jerk	REAL	Maximaler Bahnruck CoordSystem = CS_AXES_COORDINATES: Angabe in Prozent CoordSystem ≠ CS_AXES_COORDINATES: Angabe in mm/s ³ bzw. in/s ³
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS, Seite 317
VAR_OUTPUT	InOperation	BOOL	Bewegung wird ausgeführt
	Shutdown	BOOL	Bewegung wird angehalten
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-118: Funktionsbaustein ML_GroupJogCont

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
Direction	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Enable"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Enable"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Enable"
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Enable"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Enable"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke an "Enable"

Tab. 5-119: Eingangsverhalten ML_GroupJogCont

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#00	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#00	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes zur Kinematik gruppiert worden

Tab. 5-120: Fehlernummern des FBs ML_GroupJogCont

ML_GroupJogStep

Kurzbeschreibung Der Coordinated Motion-Funktionsbaustein "ML_GroupJogStep" bewegt eine Kinematik einen Schritt in eine bestimmte Richtung.

Nähere Informationen siehe: [Handbetrieb auf Seite 49](#)



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

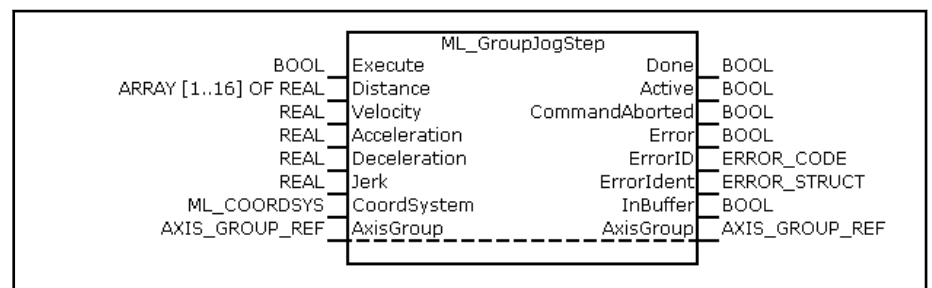


Abb. 5-72: Funktionsbaustein ML_GroupJogStep

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Distance	ARRAY[1..16] OF REAL	Relative Zielposition der Bewegung

ML_Robot.library

	Name	Typ	Kommentar
	Velocity	REAL	Maximale Bahngeschwindigkeit CoordSystem = CS_AXES_COORDINATES: Angabe in Prozent CoordSystem ≠ CS_AXES_COORDINATES: Angabe in mm/s bzw. in/s
	Acceleration	REAL	Maximale Bahnbeschleunigung CoordSystem = CS_AXES_COORDINATES: Angabe in Prozent CoordSystem ≠ CS_AXES_COORDINATES: Angabe in mm/s ² bzw. in/s ²
	Deceleration	REAL	Maximale Bahnverzögerung CoordSystem = CS_AXES_COORDINATES: Angabe in Prozent CoordSystem ≠ CS_AXES_COORDINATES: Angabe in mm/s ² bzw. in/s ²
	Jerk	REAL	Maximaler Bahnruck CoordSystem = CS_AXES_COORDINATES: Angabe in Prozent CoordSystem ≠ CS_AXES_COORDINATES: Angabe in mm/s ³ bzw. in/s ³
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS, Seite 317
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen andern Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-121: Funktionsbaustein ML_GroupJogStep

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Distance	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Execute"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COOR-DINATES	Steigende Flanke an "Execute"

Tab. 5-122: Eingangsverhalten ML_GroupJogStep

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#00	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#00	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes zur Kinematik gruppiert worden

Tab. 5-123: Fehlernummern des FBs ML_GroupJogStep

ML_GroupWait

Kurzbeschreibung Durch den Coordinated Motion-Funktionsbaustein "ML_GroupWait" wartet eine Kinematik die angegebene Zeit auf der letzten kommandierten Position.
 (Siehe [Warten auf Seite 67](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

ML_Robot.library

Schnittstellenbeschreibung

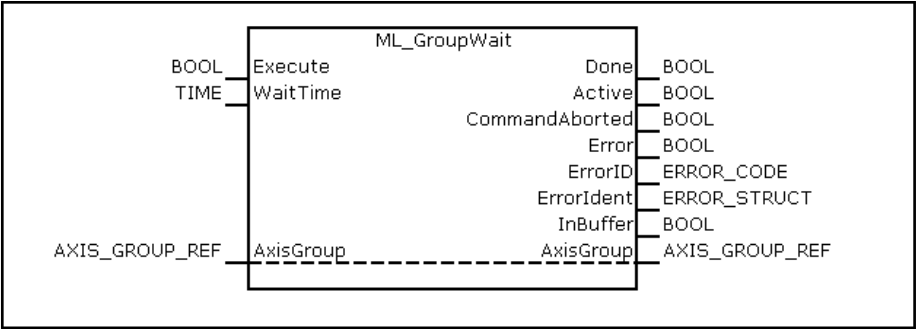


Abb. 5-73: Funktionsbaustein ML_GroupWait

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins.
	WaitTime	TIME	Zeit, welche der Roboter auf der letzten kommandierten Position wartet
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt.
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-124: Funktionsbaustein ML_GroupWait

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
WaitTime	TIME	0.0		0.0	Steigende Flanke an "Execute"

Tab. 5-125: Eingangsverhalten ML_GroupWait

- Signal-Zeit-Diagramm
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_RobotOpenCyclicChannel

Kurzbeschreibung Der Funktionsbaustein "ML_RobotOpenCyclicChannel" öffnet einen zyklischen Sollwertkanal.

(Siehe [Zyklische Sollwertvorgabe auf Seite 83](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

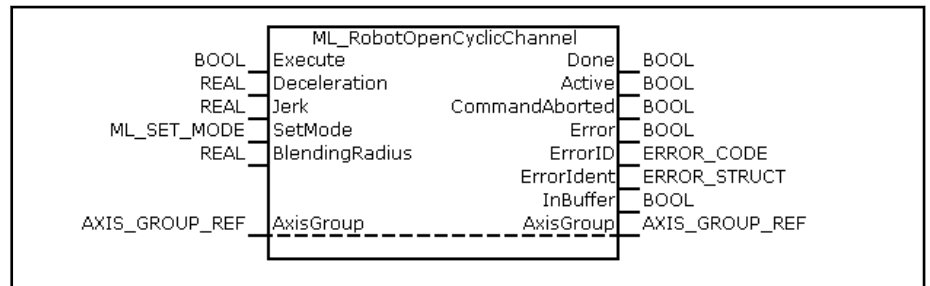


Abb. 5-74: Funktionsbaustein ML_RobotOpenCyclicChannel

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
	BlendingRadius	REAL	Überschleifradius
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE, Seite 319
VAR_OUTPUT	Done	BOOL	"Öffnen eines zyklischen Sollwertkanals" abgeschlossen
	Active	BOOL	"Öffnen eines zyklischen Sollwertkanals" aktiv
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-126: Funktionsbaustein ML_RobotOpenCyclicChannel

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"

ML_Robot.library

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
BlendingRadius	REAL	0.0		0.0	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"

Tab. 5-127: Eingangsverhalten ML_RobotOpenCyclicChannel

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_RobotOpenDirectCyclicChannel

Kurzbeschreibung Der Funktionsbaustein "ML_RobotOpenDirectCyclicChannel" öffnet einen zyklischen Sollwertkanal für PTP Bewegungen.
(Siehe [Zyklische Sollwertvorgabe auf Seite 83](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

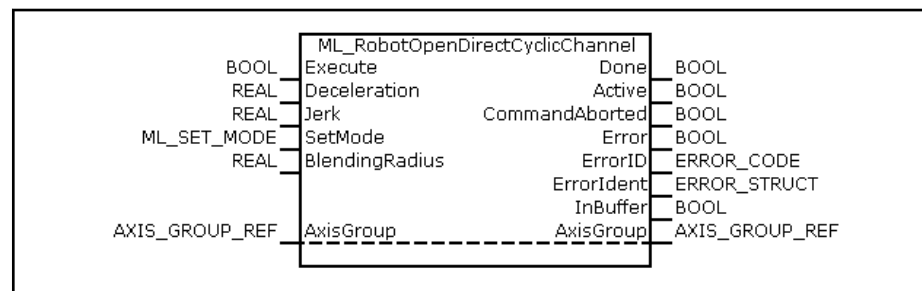


Abb. 5-75: Funktionsbaustein ML_RobotOpenDirectCyclicChannel

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Deceleration	REAL	Maximale Bahnverzögerung in Prozent
	Jerk	REAL	Maximaler Bahnruck in Prozent
	BlendingRadius	REAL	Überschleifradius
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE, Seite 319

	Name	Typ	Kommentar
VAR_OUTPUT	Done	BOOL	"Öffnen eines zyklischen Sollwertkanals" abgeschlossen
	Active	BOOL	"Öffnen eines zyklischen Sollwertkanals" aktiv
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-128: Funktionsbaustein ML_RobotOpenDirectCyclicChannel

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0		0.0	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"

Tab. 5-129: Eingangsverhalten ML_RobotOpenDirectCyclicChannel

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_RobotWriteCyclicChannel

Kurzbeschreibung Der Funktionsbaustein "ML_RobotWriteCyclicChannel" schreibt einen zyklischen Sollwert.
 (Siehe [Zyklische Sollwertvorgabe auf Seite 83](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.



Der Funktionsbaustein **muss zwingend** im Motionzyklus aufgerufen werden, da es sonst zu unerwarteten Fehlermeldungen oder ruckenden Achsen kommen kann!

ML_Robot.library

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

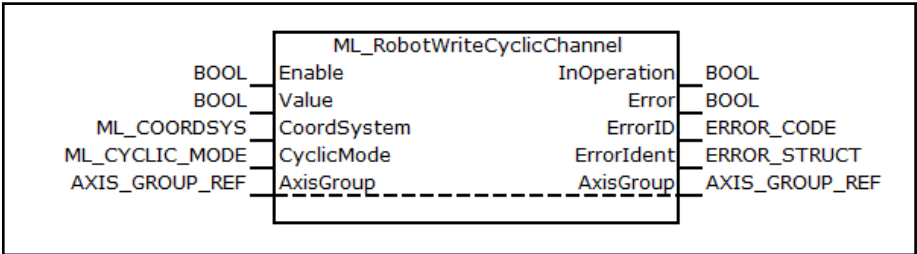


Abb. 5-76: Funktionsbaustein ML_RobotWriteCyclicChannel

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Value	ARRAY[1..16] OF REAL	Zyklischer Sollwert, abhängig von der Art der zyklischen Sollwertvorgabe (CyclicMode)
	CoordSystem	ML_COORDSYS	Koordinatensystem der zyklischen Sollposition ML_COORDSYS, Seite 317
	CyclicMode	ML_CYCLIC_MODE	Art der zyklischen Sollwertvorgabe ML_CYCLIC_MODE, Seite 318
VAR_OUTPUT	InOperation	BOOL	"Schreiben eines zyklischen Sollwerts" abgeschlossen
	Active	BOOL	"Schreiben eines zyklischen Sollwerts" aktiv
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-130: Funktionsbaustein ML_RobotWriteCyclicChannel

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Value	ARRAY[1..16] OF REAL			[16(0.0)]	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke an "Execute"
CyclicMode	ML_CYCLIC_MODE			CYCLIC_MODE_POS_ABS	Steigende Flanke an "Execute"

Tab. 5-131: Eingangsverhalten ML_RobotWriteCyclicChannel

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_RobotMoveJumpAbsolute

Kurzbeschreibung

Der Funktionsbaustein "ML_RobotMoveJumpAbsolute" bewegt eine Kinematik mit einer Jump-Bewegung über Start-, Maximal- und Endhöhe von der zuletzt kommandierten Position des TCP zu einer vorgegebenen absoluten Zielposition im angegebenen Koordinatensystem.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Movement

Schnittstellenbeschreibung

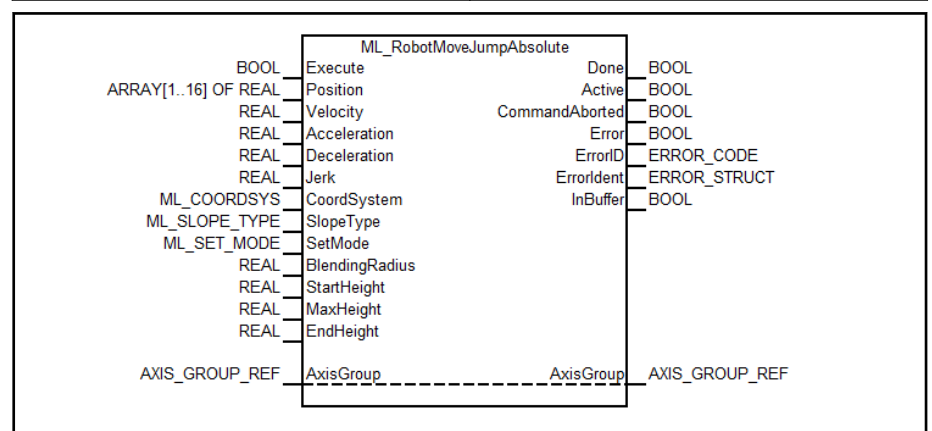


Abb. 5-77: Funktionsbaustein ML_RobotMoveJumpAbsolute

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	Position	ARRAY[1..16] OF REAL	Absolute Zielposition der Bewegung
	Velocity	REAL	Maximale Bahngeschwindigkeit
	Acceleration	REAL	Maximale Bahnbeschleunigung
	Deceleration	REAL	Maximale Bahnverzögerung
	Jerk	REAL	Maximaler Bahnruck
	CoordSystem	ML_COORDSYS	Koordinatensystem der Zielposition der Bewegung ML_COORDSYS auf Seite 317
	SlopeType	ML_SLOPE_TYPE	Slope-Art der Bewegung ML_SLOPE_TYPE auf Seite 319
	SetMode	ML_SET_MODE	Art des Überschleifens (VIA, TO) ML_SET_MODE auf Seite 319
	BlendingRadius	REAL	Überschleifradius R>0 --> VIA; R=0 --> TO
	StartHeight	REAL	Minimale Höhe zu Beginn der horizontalen Bewegung
	MaxHeight	REAL	Maximale Höhe der Jump-Bewegung

ML_Robot.library

	Name	Typ	Kommentar
	EndHeight	REAL	Minimale Höhe am Ende der horizontalen Bewegung
VAR_OUTPUT	Done	BOOL	Bewegung abgeschlossen
	Active	BOOL	Bewegung wird ausgeführt
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-132: Funktionsbaustein ML_RobotMoveJumpAbsolute

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
Position	ARRAY[1..16] OF REAL			16(0.0)	Steigende Flanke an "Execute"
Velocity	REAL	1.0		0.0	Steigende Flanke an "Execute"
Acceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Deceleration	REAL	1.0		0.0	Steigende Flanke an "Execute"
Jerk	REAL	>=0.0		0.0	Steigende Flanke an "Execute"
CoordSystem	ML_COORDSYS			CS_MACHINE_COOR- DINATES	Steigende Flanke an "Execute"
SlopeType	ML_SLOPE_TYPE			ST_BLOCK_SLOPE	Steigende Flanke an "Execute"
SetMode	ML_SET_MODE			SM_TO	Steigende Flanke an "Execute"
BlendingRadius	REAL	0.0		0.0	Steigende Flanke an "Execute"
StartHeight	REAL			0.0	Steigende Flanke an "Execute"
MaxHeight	REAL			0.0	Steigende Flanke an "Execute"
EndHeight	REAL			0.0	Steigende Flanke an "Execute"

Tab. 5-133: Eingangsverhalten ML_RobotMoveJumpAbsolute

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

- [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR 16#03	16#F0220201	16#00	Ungültige Steuerungsnummer Ursache: Steuerungsnummer der zu kommandierenden Kinematik ist falsch
RESOURCE_ERROR 16#03	16#F0220202	16#00	Ungültige Achsnummer Ursache: Achsnummer der zu kommandierenden Kinematik ist falsch
STATE_MACHINE_ERROR 16#05	16#F0290405	16#00	Kommando in 'Standstill Incomplete' zurückgewiesen Ursache: Achsen sind zuvor nicht mit dem Funktionsbaustein ML_GroupAllAxes zur Kinematik gruppiert worden
OTHER_ERROR 16#FE	16#F12C0F70	16#00	Spline Parameterlänge ungültig Ursache: Die Parameterlänge des Splines konnte nicht bestimmt werden
OTHER_ERROR 16#FE	16#F12C0F72	16#00	Längen für Splinedefinition ungültig Ursache: Eine Länge für die Splinedefinition ist nahe 0
CALCULATION_ERROR 16#07	16#F2299582	16#00	Fehler während Splineberechnung Ursache: Das Geschwindigkeitsfenster für die Berechnung wurde überschritten. Der Fehler tritt typischerweise dann auf, wenn beim Kommando MoveJump eine sehr geringe Höhe im Verhältnis zum Abstand zwischen Start und Zielpunkt kommandiert wurde

ML_Robot.library

ErrorID	Additional1	Additional2	Beschreibung
CALCULATION_ERROR 16#07	16#F2299583	16#00	Geschwindigkeitsfehlerüberschreitung bei Splineberechnung Ursache: Das Geschwindigkeitsfenster für die Berechnung wurde überschritten. Der Fehler tritt typischerweise dann auf, wenn beim Kommando MoveJump eine sehr geringe Höhe im Verhältnis zum Abstand zwischen Start und Zielpunkt kommandiert wurde. Siehe Bewegungsart Jump auf Seite 42
INPUT_RANGE_ERROR 16#06	16#F2299584	16#00	Ungültige Höhen bei Move Jump Ursache: Es wurde eine oder mehrere ungültige Höhen angegeben. Siehe Bewegungsart Jump auf Seite 42

Tab. 5-134: Fehlernummern des FBs ML_RobotMoveJumpAbsolute

5.3.7 Kinematic/ Parameter

Zugriff auf Daten einer Kinematik

Aufbau und Zugriff auf Parameter aus dem SPS-Programm

Die Daten einer Kinematik sind über Parameterzugriff lesend (MB_GroupRead***) und teilweise auch schreibend (MB_GroupWrite***) erreichbar:

- K-Parameter (kinematics parameters), die sich auf eine Kinematik beziehen.

Alle IndraMotion-Parameter werden nach dem gleichen Bildungsgesetz aufgebaut:

Bsp.: **K-0-0001**, mit "K" als Parametertyp-Kennzeichner gefolgt von der Gruppennummer "0" und der vierstelligen Parameternummer "0001".

Abgeleitet von der Parameternummer kann im SPS-Programm auf diese Parameter zugegriffen werden.

Bei Kenntnis der Nummer des Parameters:

Programm:

```
VAR_GLOBAL CONSTANT
  FP_K_0_0000: MB_IDN:= 184683593728 ; // Dummy K Parameter
  FP_K_0_0001: MB_IDN:= 184683593729 ; // Logical kinematics number
  FP_K_0_0002: MB_IDN:= 184683593730 ; // Kinematics name
  FP_K_0_0003: MB_IDN:= 184683593731 ; // Maximum path speed
  //...
  FP_K_0_2301: MB_IDN:= 184683596029 ; // PLC integer input from RCL
  FP_K_0_2302: MB_IDN:= 184683596030 ; // PLC real input from RCL
END_VAR
```

Die Nummer kann alternativ zur Konstanten selbst an den Eingang "ParameterNumber" des jeweiligen Funktionsbausteines angelegt werden.

Offsets der Parameter

Parametergruppe	dezimal	hexadezimal
S-0-0000	0	16#0
P-0-0000	32768	16#8000
A-0-0000	4294967296	16#1 0000 0000
C-0-0000	8589934592	16#2 0000 0000
K-0-0000	184683593728	16#2B 0000 0000

Parametergruppe	dezimal	hexadezimal
M-0-0000	193273528320	16#2D 0000 0000
N-0-0000	197568495616	16#2E 0000 0000
O-0-0000	201863462912	16#2F 0000 0000

Tab. 5-135: Parameter-Offsets

Zugriff auf K-Parameter:

- Analog zu den Read/Write-Parameter-Bausteinen für A- und C-Parameter gibt es Group-Read/Write-Parameter-Bausteine für K-Parameter.

Der Adressbezeichner wird durch IndraWorks MLC automatisch generiert (Kinematikname/Kinematiknummer), Typ AXIS_GROUP_REF, RIL_CommonTypes.library.

Bsp.: MB_GroupReadParameter

Referenz des Bausteines

```
Kin_001:AXIS_GROUP_REF:=(CntrlNo:=LOCAL_CNTRL,GroupNo:=KINEMATIC_1); //Kinematik Kin_001, Nr. 1
```

MB_GroupReadListParameter

Kurzbeschreibung

Der Funktionsbaustein "MB_GroupReadListParameter" gibt den Wert eines durch den Anwender vorgegebenen Parameters zurück.

Dem Baustein wird ein Pointer übergeben, der einen in **Anwenderverantwortung** festgelegten Speicherraum von maximal 64KByte adressiert. Ferner wird die gewünschte Nutzdatenlänge übergeben (maximal lesbare Anzahl an Bytes).

Als Rückgabewert erhält man die Anzahl der tatsächlich gelesenen Bytes.



Der Datentyp der Elemente des zu lesenden Parameters und der Datentyp der Elemente des ARRAYS müssen übereinstimmen.



Nur für K-Parameter!

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

Schnittstellenbeschreibung

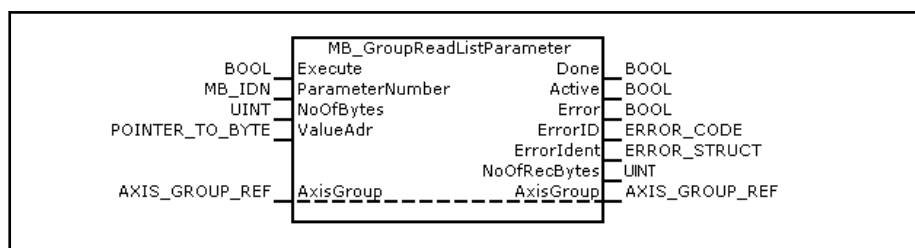


Abb. 5-78: FB MB_GroupReadListParameter

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Execute	BOOL	Lesen des Parameters, getriggert durch die positive Flanke von "Execute"

ML_Robot.library

	Name	Typ	Kommentar
	Parameter-Number	MB_IDN	Nummer des gewünschten K-Parameters (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
	NoOfBytes	UINT	Maximal zu lesende Datenmenge in Byte
	ValueAdr	POINTER TO BYTE	Zeigt auf Anfangsadresse, ab der die gelesenen Daten (maximal NoOfBytes) abgelegt werden sollen
VAR_OUTPUT	Done	BOOL	Gültige Daten sind verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	NoOf RecBytes	UINT	Anzahl der tatsächlich gelesenen Bytes

Tab. 5-136: Schnittstelle von FB MB_GroupReadListParameter

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
- MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

Zusätzlich können auftreten:

- F00D1205, Daten eines Singleparameters angefordert
- F90D1204, Kein Speicher

MB_GroupReadParameter

Kurzbeschreibung Der Funktionsbaustein "MB_GroupReadParameter" gibt den Wert eines durch den Anwender vorgegebenen Parameters zurück.

Der Rückgabewert wird als DINT-Zahl dargestellt.



Eine Änderung, z. B. der Parameternummer oder der Kinematik am Eingang des Funktionsbausteines setzt einen erneuten 0/1-Übergang an "Enable" voraus.



Nur für K-Parameter!

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

Schnittstellenbeschreibung

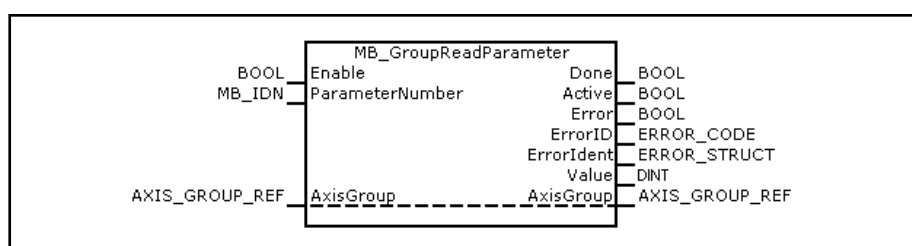


Abb. 5-79: FB MB_GroupReadParameter

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters, (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
VAR_OUTPUT	Done	BOOL	Gültiger Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Value	DINT	4-Byte-Wert, als DINT dargestellt

Tab. 5-137: Schnittstelle von FB MB_GroupReadParameter

Signal-Zeit-Diagramm

Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 196](#)

Fehlerbehandlung

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
- MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MB_GroupReadRealParameter

Kurzbeschreibung

Der Funktionsbaustein "MB_GroupReadRealParameter" gibt den Wert eines durch den Anwender vorgegebenen Parameters zurück.

Der Rückgabewert wird als REAL-Zahl dargestellt.



Eine Änderung, z. B. der Parameternummer oder der Kinematik am Eingang des Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Enable" voraus.



Nur für K-Parameter!

ML_Robot.library

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

Schnittstellenbeschreibung

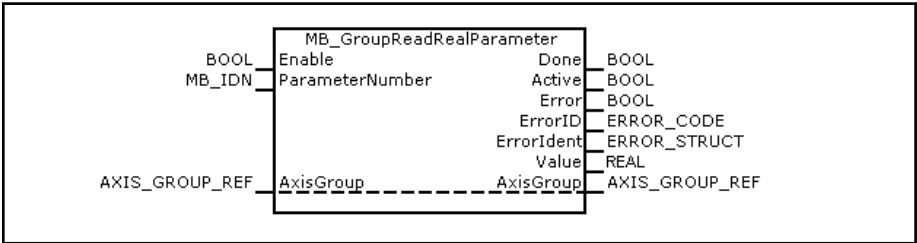


Abb. 5-80: FB MB_GroupReadRealParameter

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters, (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
VAR_OUTPUT	Done	BOOL	Gültiger Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Value	REAL	4-Byte-Wert als REAL-Zahl dargestellt

Tab. 5-138: Schnittstelle von FB MB_GroupReadRealParameter

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 196](#)

- Fehlerbehandlung** Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
 - MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MB_GroupReadSercosAttribute

Kurzbeschreibung Der Funktionsbaustein "MB_GroupReadSercosAttribute" erlaubt das Lesen der Sercos Attribute eines K-Parameters.
Der Rückgabewert wird als DWORD dargestellt.



Eine Änderung, z. B. der Parameternummer oder der Kinematik am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.



Nur für K-Parameter!

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

Schnittstellenbeschreibung

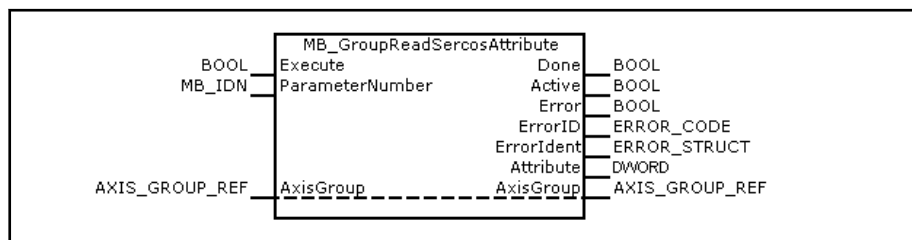


Abb. 5-81: FB MB_GroupReadSercosAttribute

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Execute	BOOL	Lesen des Parameters, getriggert durch die positive Flanke von "Execute"
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
VAR_OUTPUT	Done	BOOL	Gültiger Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Value	DWORD	4-Byte-Wert, enthält Sercos Attribute des K-Parameters

Tab. 5-139: Schnittstelle von FB MB_GroupReadSercosAttribute

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
- MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_Robot.library

Zusätzlich können auftreten:

- F00D1230, Fehler ReadSERCOSAttribute, kein Speicher

MB_GroupReadSercosDataStatus

Kurzbeschreibung Der Funktionsbaustein "MB_GroupReadSercosDataStatus" gibt den Status eines durch den Anwender vorgegebenen K-Parameters zurück.
Der Rückgabewert wird als ein ENUM-Wert (INT-Zahl) dargestellt.



Eine Änderung, z. B. der Parameternummer oder der Kinematik am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.



Nur für K-Parameter!

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

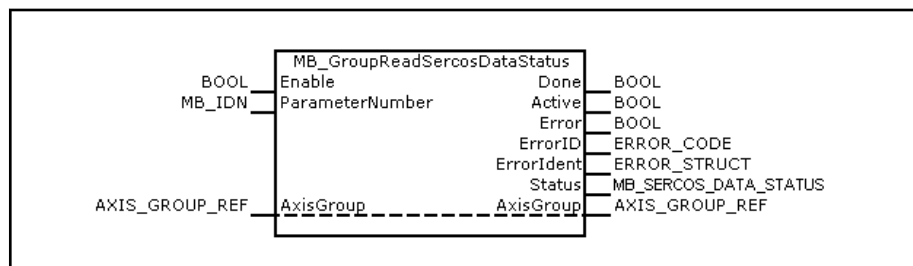
Schnittstellenbeschreibung

Abb. 5-82: FB MB_GroupReadSercosDataStatus

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters, (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
VAR_OUTPUT	Done	BOOL	Gültiger Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Status	MB_SERCOS_DATA_STATUS	Status des gewählten Parameters (ML_Base)

Tab. 5-140: Schnittstelle von FB MB_GroupReadSercosDataStatus

Kurzbeschreibung Mit Hilfe dieses Funktionsbausteines und eines [MB_GroupWriteParameter auf Seite 291](#), ist es möglich, Antriebskommandos auszulösen und in ihrer Abarbeitung zu verfolgen:

- 0 - Kommando nicht gesetzt und nicht freigegeben

- 1 - Kommando gesetzt und freigegeben
- 3 - Kommando korrekt abgearbeitet
- 5 - Kommandoabarbeitung unterbrochen
- 7 - Kommandoabarbeitung läuft
- 15 - Fehler, Kommandoabarbeitung unmöglich
- 256 - Wert des Parameters außerhalb seines Min.- / Max.-Bereiches

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 196](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
- MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MB_GroupReadStringParameter

Kurzbeschreibung Der Funktionsbaustein "MB_GroupReadStringParameter" gibt den Wert eines durch den Anwender vorgegebenen Parameters zurück.

Der Rückgabewert wird als Zeichenkette (STRING) entsprechend seiner tatsächlichen Länge mit maximal 255 Zeichen dargestellt.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

Schnittstellenbeschreibung

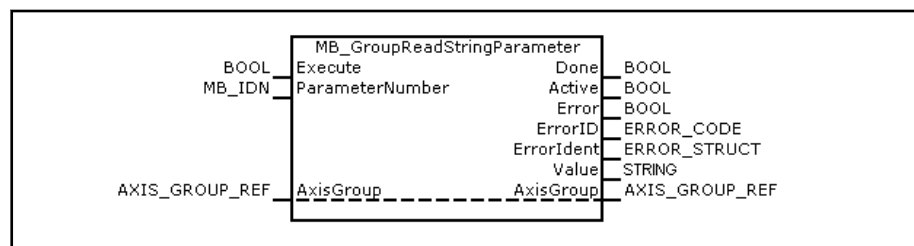


Abb. 5-83: FB MB_GroupReadStringParameter

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Execute	BOOL	Startet das Lesen des Textes vom Parameter mit steigender Flanke
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
VAR_OUTPUT	Done	BOOL	Gültiger Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft

ML_Robot.library

	Name	Typ	Kommentar
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Value	STRING	Zeichenkette mit maximal 255 Zeichen

Tab. 5-141: Schnittstelle von FB MB_GroupReadStringParameter

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
- MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

Zusätzlich können auftreten:

- F00D1218, Kein Speicher
- F00D1219, Daten eines Singleparameters angefordert
- F00D1220, Nicht unterstützte Auswahl des sercos Elementes
- F00D1221, Zeitüberschreitung bei Parameterzugriff

MB_GroupWriteListParameter

Kurzbeschreibung Der Funktionsbaustein "MB_GroupWriteListParameter" schreibt Speicherinhalt definierter Länge (NoOfByte), der ab der Anfangsadresse (ValueAdr) zur Verfügung gestellt wurde, in einen durch den Anwender vorgegebenen Parameter.



Die vorgegebene Anzahl an Bytes muss genau mit der Datenmenge, die übertragen werden soll, übereinstimmen!

Ist die Anzahl zu groß, wird Datenmüll übertragen. Ist die Anzahl zu klein, fehlen Daten im Zielparameter!



Der Datentyp der Elemente des zu schreibenden Parameters und der Datentyp der Elemente des ARRAYs müssen übereinstimmen.



Nur für K-Parameter!

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

Schnittstellenbeschreibung

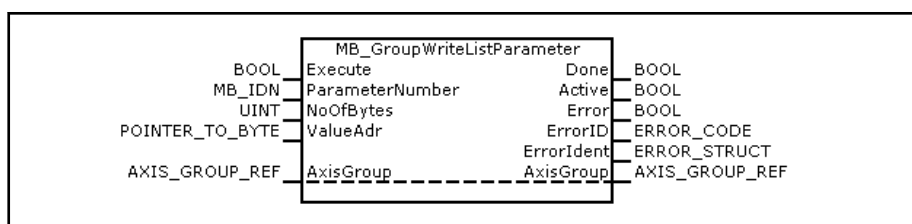


Abb. 5-84: FB MB_GroupWriteListParameter

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
	NoOfBytes	UINT	Anzahl der Bytes, die geschrieben werden sollen
	ValueAdr	POINTER TO BYTE	Pointer auf Anfangsadresse des Speichers in dem die zu schreibenden Daten liegen
VAR_OUTPUT	Done	BOOL	Wert ist erfolgreich in die Steuerung/den Antrieb geschrieben
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 5-142: Schnittstelle von FB MB_GroupWriteListParameter

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
- MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

Zusätzlich können auftreten:

- F00D1207, Daten größer als 64KByte
- F00D1210, Daten eines Singleparameters angefordert
- F00D1211, Fehler ReadSERCOSDataStatus, kein Speicher
- F90D1209, Kein Speicher

MB_GroupWriteParameter

Kurzbeschreibung

Der Funktionsbaustein "MB_GroupWriteParameter" schreibt einen Wert in einen durch den Anwender vorgegebenen Parameter.



Nur für K-Parameter!

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

Schnittstellenbeschreibung

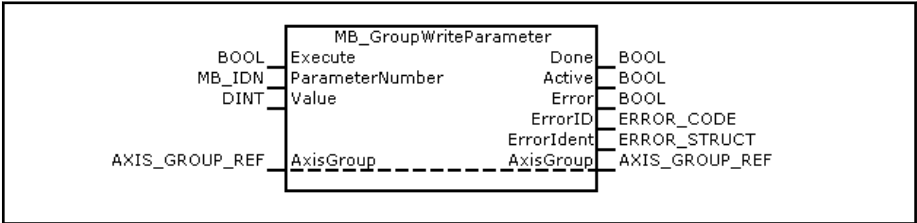


Abb. 5-85: FB MB_GroupWriteParameter

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke an- steht, in den Parameter
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters, (siehe Zugriff auf Da- ten einer Kinematik auf Seite 282)
	Value	DINT	Wert, der in den Parameter geschrieben werden soll
VAR_OUTPUT	Done	BOOL	Wert erfolgreich in die Steuerung / den Antrieb geschrieben
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 5-143: Schnittstelle von FB MB_GroupWriteParameter

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschlie-
ßende Bearbeitung auf Seite 199](#)

- Fehlerbehandlung** Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
 - MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steue-
rung

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MB_GroupWriteRealParameter

Kurzbeschreibung Der Funktionsbaustein "MB_GroupWriteRealParameter" schreibt einen Wert
in einen durch den Anwender vorgegebenen Parameter.



Nur für K-Parameter!

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic Parameter

Schnittstellenbeschreibung

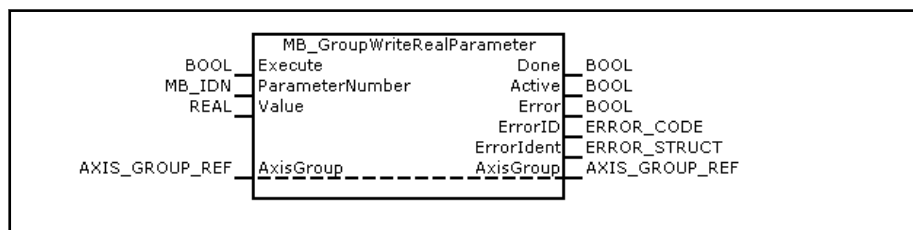


Abb. 5-86: FB MB_GroupWriteRealParameter

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
	Value	REAL	Wert, der in den Parameter geschrieben werden soll
VAR_OUTPUT	Done	BOOL	Wert erfolgreich in die Steuerung / den Antrieb geschrieben
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 5-144: Schnittstelle von FB MB_GroupWriteRealParameter

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
- MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

MB_GroupWriteStringParameter

Kurzbeschreibung Der Funktionsbaustein "MB_GroupWriteStringParameter" schreibt eine Zeichenkette mit max. 255 Zeichen in einen durch den Anwender vorgegebenen Parameter.



Nur für K-Parameter!

ML_Robot.library

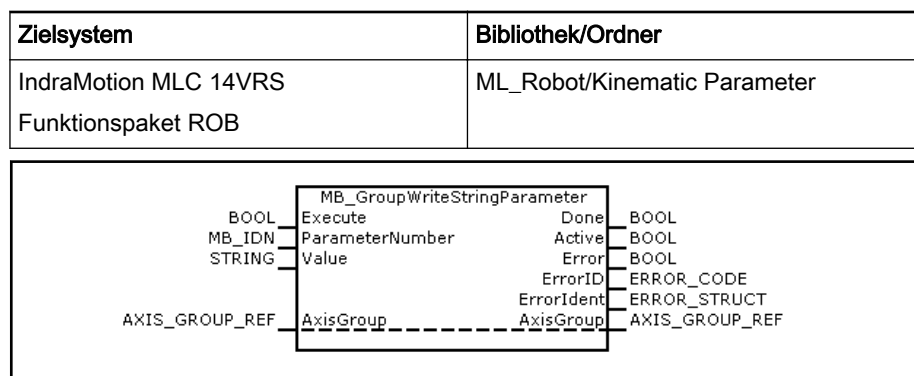
Schnittstellenbeschreibung

Abb. 5-87: FB MB_GroupWriteStringParameter

	Name	Typ	Kommentar
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Enthält Informationen zur konkreten Kinematik
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter- Number	MB_IDN	Nummer des gewünschten K-Parameters (siehe Zugriff auf Daten einer Kinematik auf Seite 282)
	Value	STRING	Wert, der in den Parameter geschrieben werden soll
VAR_OUTPUT	Done	BOOL	Wert erfolgreich in die Steuerung/den Antrieb geschrieben
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 5-145: Schnittstelle von FB MB_GroupWriteStringParameter

Signal-Zeit-DiagrammSiehe [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)**Fehlerbehandlung**

Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:

- SERCOS_TABLE, 16#10 - Fehler beim Zugriff auf den Parameter (z.B. Parameter wird nicht unterstützt)
- MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Kinematik/ die Steuerung

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

Zusätzlich können auftreten:

- F00D1222, Datengröße überschritten, Daten größer als 255 Byte
- F00D1224, Kein Speicher
- F00D1225, Daten eines Singleparameters angefordert
- F00D1227, Fehler beim Datum schreiben

5.3.8 Kinematic/ TeachIn

ML_GroupDeletePointFile

Kurzbeschreibung

Der Funktionsbaustein "ML_GroupDeletePointFile" löscht eine Punktedatei einer Kinematik.

(Siehe [Teach-in auf Seite 50](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic TeachIn

Schnittstellenbeschreibung

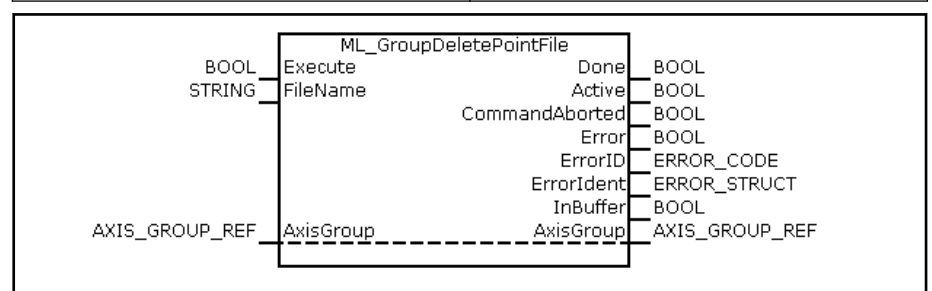


Abb. 5-88: Funktionsbaustein ML_GroupDeletePointFile

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	FileName	STRING(20)	Name der zu löschenden Datei
VAR_OUTPUT	Done	BOOL	Datei löschen wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-146: Funktionsbaustein ML_GroupDeletePointFile

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
FileName	STRING(20)			"	Steigende Flanke an "Execute"

Tab. 5-147: Eingangsverhalten ML_GroupDeletePointFile

Signal-Zeit-Diagramm

- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)

ML_Robot.library

- [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_GroupDeletePoints

Kurzbeschreibung Der Funktionsbaustein "ML_GroupDeletePoints" löscht geteachte Punkte einer Kinematik.
(Siehe [Teach-in auf Seite 50](#)).

 Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic TeachIn

Schnittstellenbeschreibung

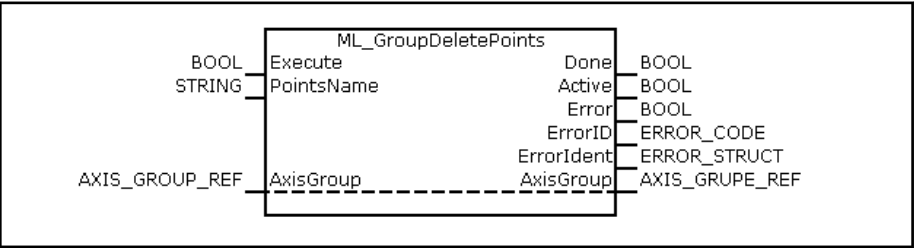


Abb. 5-89: Funktionsbaustein ML_GroupDeletePoints

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	PointsName	STRING(20)	Punkt(e), die gelöscht werden sollen Name kann Wildcard "*" enthalten
VAR_OUTPUT	Done	BOOL	Löschen wurde abgeschlossen
	Active	BOOL	Löschen ist aktiv
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-148: Funktionsbaustein ML_GroupDeletePoints

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
PointsName	STRING(20)			"	Steigende Flanke von "Execute"

Tab. 5-149: Eingangsverhalten ML_GroupDeletePoints

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_GroupLoadPointFile

Kurzbeschreibung Der Funktionsbaustein "ML_GroupLoadPointFile" lädt eine Punktedatei einer Kinematik.

(Siehe [Teach-in auf Seite 50](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic TeachIn

Schnittstellenbeschreibung

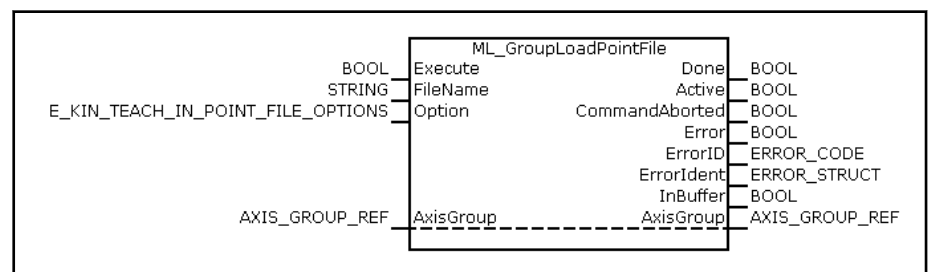


Abb. 5-90: Funktionsbaustein ML_GroupLoadPointFile

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	FileName	STRING(20)	Name der zu ladenden Datei
	Option	E_KIN_TEACH_IN_POINT_FILE_OPTIONS	Optionen zum Laden der Datei siehe E_KIN_TEACH_IN_POINT_FILE_OPTIONS auf Seite 315
VAR_OUTPUT	Done	BOOL	Definition wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist

ML_Robot.library

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-150: Funktionsbaustein ML_GroupLoadPointFile

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
FileName	String(20)			"	Steigende Flanke an "Execute"
Option	E_KIN_TEACH_IN_POINT_FILE_OPTIONS			E_GROUP_TEACH_IN_NEW	Steigende Flanke an "Execute"

Tab. 5-151: Eingangsverhalten ML_GroupLoadPointFile

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_GroupReadNextPoint

Kurzbeschreibung

Der Funktionsbaustein "ML_GroupReadNextPoint" sucht den nächsten getachten Punkt und gibt die Daten des nächsten Punkts einer Kinematik aus. (Siehe [Teach-in auf Seite 50](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic TeachIn

Schnittstellenbeschreibung

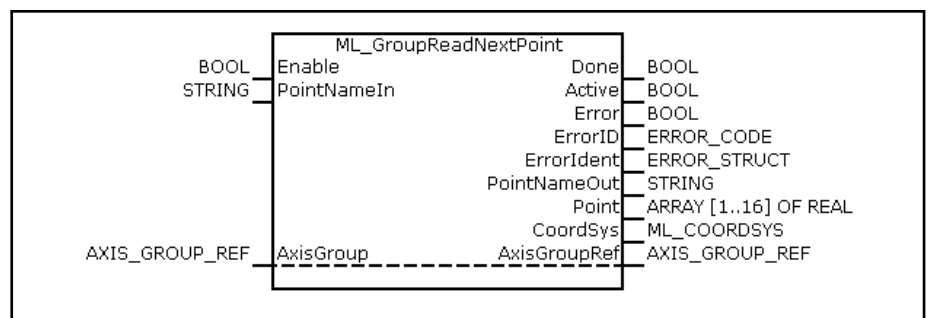


Abb. 5-91: Funktionsbaustein ML_GroupReadNextPoint

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	PointNameIn	STRING(20)	Name des Punktes
VAR_OUTPUT	Done	BOOL	Lesen des Punktes wurde abgeschlossen
	Active	BOOL	Lesen des Punktes aktiv
	PointNameOut	STRING(19)	Name des nächsten Punkts
	CoordSys	ML_COORDSYS	Koordinatensystem des nächsten Punkts (Siehe ML_COORDSYS auf Seite 317).
	Point	ARRAY [1..16] OF REAL	Koordinaten des nächsten Punkts
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detaillinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-152: Funktionsbaustein ML_GroupReadNextPoint

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
PointNameIn	STRING(20)			"	Kontinuierlich

Tab. 5-153: Eingangsverhalten ML_GroupReadNextPoint

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung](#) auf Seite 202

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_GroupReadPoint

Kurzbeschreibung Der Funktionsbaustein "ML_GroupReadPoint" gibt die geteachten Daten eines Punktes einer Kinematik aus.
(Siehe [Teach-in](#) auf Seite 50).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic TeachIn

Schnittstellenbeschreibung

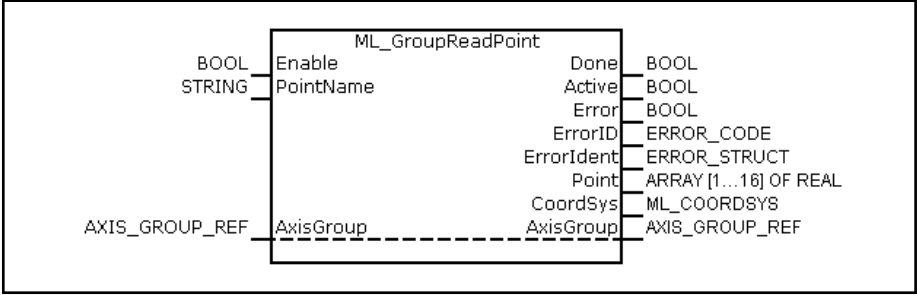


Abb. 5-92: Funktionsbaustein ML_GroupReadPoint

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	PointName	STRING(20)	Name des Punktes
VAR_OUTPUT	Done	BOOL	Lesen des Punktes wurde abgeschlossen
	Active	BOOL	Lesen des Punktes aktiv
	CoordSys	ML_COORDSYS	Koordinatensystem des Punktes (Siehe ML_COORDSYS auf Seite 317)
	Point	ARRAY [1..16] OF REAL	Koordinaten des Punkts
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-154: Funktionsbaustein ML_GroupReadPoint

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	Kontinuierlich
PointName	STRING(20)			"	Kontinuierlich

Tab. 5-155: Eingangsverhalten ML_GroupReadPoint

- Signal-Zeit-Diagramm
- Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung](#) auf Seite 202
- Fehlerbehandlung
- Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_GroupSavePointFile

Kurzbeschreibung

Der Funktionsbaustein "ML_GroupSavePointFile" speichert eine Punktedatei einer Kinematik.

(Siehe [Teach-in auf Seite 50](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic TeachIn

Schnittstellenbeschreibung

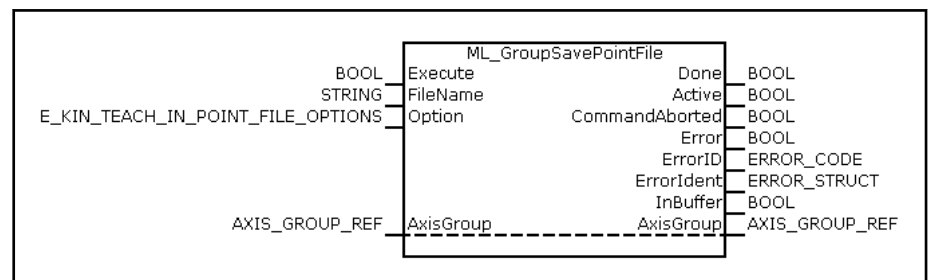


Abb. 5-93: Funktionsbaustein ML_GroupSavePointFile

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	FileName	STRING(20)	Name der zu speichernden Datei
	Option	E_KIN_TEACH_IN_POINT_FILE_OPTIONS	Optionen zum Speichern der Datei siehe E_KIN_TEACH_IN_POINT_FILE_OPTIONS auf Seite 315
VAR_OUTPUT	Done	BOOL	Datei speichern wurde abgeschlossen
	Active	BOOL	Funktionsbaustein ist aktiv
	CommandAborted	BOOL	Kommando wurde durch einen anderen Funktionsbaustein abgebrochen
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	InBuffer	BOOL	Kommando wurde erfolgreich in den Satz-Puffer gestellt
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-156: Funktionsbaustein ML_GroupSavePointFile

ML_Robot.library

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
FileName	STRING(20)			"	Steigende Flanke an "Execute"
Option	E_KIN_TEACH_IN_POINT_FILE_OPTIONS			E_GROUP_TEACH_IN_NEW	Steigende Flanke an "Execute"

Tab. 5-157: Eingangsverhalten ML_GroupSavePointFile

- Signal-Zeit-Diagramm**
- [Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung auf Seite 199](#)
 - [InBuffer-Ausgang an ML_Robot-Funktionsbausteinen auf Seite 207](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

ML_GroupWritePoint

Kurzbeschreibung Der Funktionsbaustein "ML_GroupWritePoint" teucht einen Punkt einer Kinematik.
(Siehe [Teach-in auf Seite 50](#)).



Der Funktionsbaustein kann nur für die Robot-Control V2 verwendet werden.

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_Robot/Kinematic TeachIn

Schnittstellenbeschreibung

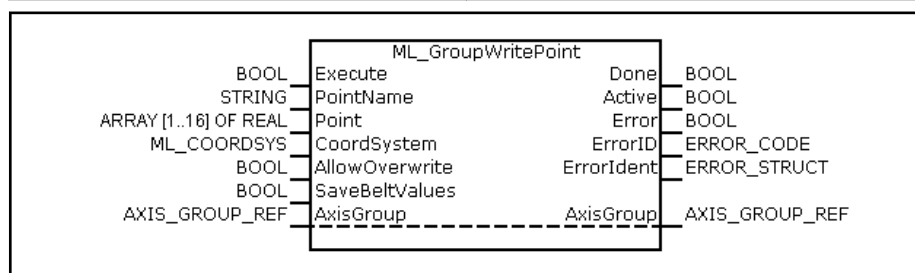


Abb. 5-94: Funktionsbaustein ML_GroupWritePoint

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins
	PointName	STRING(20)	Name des Punkts
	Point	ARRAY [1..16] OF REAL	Koordinaten des Punkts
	CoordSys	ML_COORDSYS	Koordinatensystem des Punkts (Siehe ML_COORDSYS auf Seite 317).

	Name	Typ	Kommentar
	AllowOverwrite	BOOL	Erlaubt das Überschreiben eines bereits existierenden Punkts
	SaveBeltValues	BOOL	Speichert die Bandwerte des Punkts
VAR_OUTPUT	Done	BOOL	Definition wurde abgeschlossen
	Active	BOOL	Definition wird durchgeführt
	Error	BOOL	Zeigt an, dass ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Kinematikreferenz der kommandierten Kinematik

Tab. 5-158: Funktionsbaustein ML_GroupWritePoint

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	Kontinuierlich
PointName	STRING(20)			"	Steigende Flanke von "Execute"
Point	ARRAY [1..16] OF REAL			16(0.0)	Steigende Flanke von "Execute"
CoordSys	ML_COORDSYS			CS_MACHINE_COORDINATES	Steigende Flanke von "Execute" Kontinuierlich
AllowOverwrite	BOOL			FALSE	Steigende Flanke von "Execute"
SaveBeltValues	BOOL			FALSE	Steigende Flanke von "Execute"

Tab. 5-159: Eingangsverhalten ML_GroupWritePoint

Signal-Zeit-Diagramm Siehe [Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung auf Seite 202](#)

Fehlerbehandlung Der Funktionsbaustein erzeugt folgende Arten von Fehlermeldungen:
 MLC_TABLE, 16#30 - Fehler beim Zugriff auf die Achse / die Steuerung.

Detaillierte Beschreibungen sind enthalten in:
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen
 DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

5.4 Datentypen

5.4.1 Datentypen, Übersicht

Folgende Datentypen unterstützen die Funktionsbausteine der Bibliothek "ML_Robot".



Die Bibliothek ML_Robot benötigt das ROB (Robotik) Funktionspaket der Steuerung

ML_Robot.library

Datentypen der ML_Robot

Anwendungsgebiet (Ordner in der Bibliothek)	
Datentypen	Beschreibung
Datentypen für Funktionsbausteine zur Konfiguration einer Kinematik (Data Types / Config)	
MC_KIN_REF auf Seite 305	Definiert die Methoden einer Transformation MC_KIN_REF, Initialisation auf Seite 305 MC_KIN_REF, Forward auf Seite 306 MC_KIN_REF, Inverse auf Seite 306 Fehlerhandling bei Interfacefunktionen auf Seite 306
MC_KIN_REF_EXT auf Seite 307	Definiert die Methoden einer erweiterten Transformation MC_KIN_REF_EXT, GetUsedAxes auf Seite 309 MC_KIN_REF_EXT, DetermineMemory auf Seite 307 MC_KIN_REF_EXT, Initialisation auf Seite 309 MC_KIN_REF_EXT, Forward auf Seite 308 MC_KIN_REF_EXT, Inverse auf Seite 309 MC_KIN_REF_EXT, GetMechanic auf Seite 308 Fehlerhandling bei Interfacefunktionen auf Seite 309
ML_BELT_ERROR_REACTION_MODE auf Seite 310	Definiert, welche Fehlerreaktion bei Bandsynchronität erfolgen soll
ML_KIN_AXES_CONFIG_DATA auf Seite 310	In der Struktur ist die Achskonfiguration einer Kinematik gespeichert
ML_KIN_AXIS_CONFIG auf Seite 311	Konfiguration einer Achse in einer Kinematik
ML_KIN_SAFE_ZONE_TYPE auf Seite 311	Definiert, wie eine Schutzzone konfiguriert ist
ML_KIN_USED_AXES auf Seite 311	Definiert die Freiheitsgrade einer Kinematik
Datentypen für Funktionsbausteine zur Informationsgewinnung über eine Kinematik (Data Types / Info)	
E_KIN_MECH_CMD auf Seite 312	Definiert auf welche Weise ein mechanisches Teilobjekt einer Kinematik erzeugt bzw. modifiziert werden soll
ML_KINEMATICS_DATA_SM auf Seite 312	Enthält die Informationen der korrespondierenden Kinematik, auf die per Shared Memory zugegriffen werden kann
ML_KIN_CMD_INFO auf Seite 313	Definiert den Aufbau einer Information eines Kommandos im Satzpuffer
ML_KIN_CS_TYPE auf Seite 314	Definiert die Art der Transformation
ML_KIN_INFO_TYPE auf Seite 314	Definiert, welche Art von Information eines Kommandos im Satzpuffer bereitgestellt werden soll
ML_KIN_MECHANIC_DATA auf Seite 314	Definiert die mechanische Struktur eines Roboters
ML_KIN_PERF_DATA auf Seite 315	Definiert die Daten einer Performancemessung
ML_POINTKIND auf Seite 315	Definiert die Art des Punktes in Bezug auf den zeitlichen Verlauf
Datentypen für Funktionsbausteine zum Bewegen einer Kinematik (Data Types / Movement)	
ML_ARR_COORDS auf Seite 317	Koordinaten einer Kinematik-Position in Maschinen-, Welt- oder Werkstück-Koordinaten
ML_CIRCmodes auf Seite 317	Definiert einen zu fahrenden Kreisabschnitt (Interpretation des Eingangs "AuxPoint")

ML_COORDSYS auf Seite 317	Definiert in welchem Koordinatensystem Punkte angegeben werden
ML_CYCLIC_MODE auf Seite 318	Definiert auf welche Art die zyklische Sollwertvorgabe erfolgen soll
ML_SET_MODE auf Seite 319	Definiert die Art des Überschleifens, d. h. ob die Bewegung am Endpunkt endet oder überschiffen wird
ML_SLOPE_TYPE auf Seite 319	Definiert die Slope-Art der Bewegung
Datentypen für Funktionsbausteine zum Speichern und Wiederverwenden von Positionen einer Kinematik als Punkte (Data Types / TeachIn)	
E_KIN_TEACH_IN_POINT_FILE_OPTIONS auf Seite 315	Enthält die Optionen zum Laden und Speichern von Punktedateien

Tab. 5-160: Datentypen der Bibliothek ML_Robot

5.4.2 Data Types/ Config

MC_KIN_REF

MC_KIN_REF, Übersicht

Das Interface MC_KIN_REF wird benötigt, um Methoden einer Transformation zu definieren.

Ein Funktionsbaustein muss dieses Interface implementieren, um eine Transformation mit dem Funktionsbaustein [MC_SetKinTransform auf Seite 211](#) anzumelden

Methode	Rückgabewert	Argumente	Beschreibung
Initialisation	INT	AxesConfigData TransformationLengths TransformationFactors	Initialisierungsfunktion einer Transformation
Forward	INT	OriginalCoordinates JointCoordinates	Vorwärtstransformation
Inverse	INT	OriginalCoordinates JointCoordinates	Rückwärtstransformation

Tab. 5-161: Interface MC_KIN_REF

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF, Initialisation

Diese Methode wird vom System aufgerufen, wenn eine Transformation initialisiert werden soll. Das Initialisieren findet beim Phasenschalten von P2 nach BB statt.

In dieser Funktion können Vorberechnungen und Überprüfungen der Angaben vorgenommen werden.

Argument	Typ	Beschreibung
AxesConfigData	REFERENCE TO ARRAY [1..16] OF ML_KIN_AXES_CONFIG_DATA	K-0-0008 Kinematikkonfiguration ML_KIN_AXES_CONFIG_DATA auf Seite 310
TransformationLengths	REFERENCE TO ARRAY [1..16] OF REAL	K-0-0040 Transformation, Längen der Kinematik
TransformationFactors	REFERENCE TO ARRAY [1..16] OF REAL	K-0-0041 Transformation, Faktoren der Kinematik

Tab. 5-162: Argumente der Interfacefunktion Initialisation

ML_Robot.library

Kann die Initialisierung nicht erfolgreich abgeschlossen werden, muss die Funktion einen Wert ungleich Null zurückgeben.

Mögliche Floating Point-Fehler (wie z. B. Division durch Null) werden vom System abgefangen als Fehler ausgegeben (siehe [Fehlerhandling bei Interfacefunktionen auf Seite 306](#)).

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF, Forward

Diese Methode wird vom System aufgerufen, wenn vom Achs-Koordinatensystem (ACS) ins Basis-Koordinatensystem (BCS) umgerechnet werden soll. Die Methode wird innerhalb des Betriebsmodus zyklisch aufgerufen.

Argument	Typ	Beschreibung
OriginalCoordinates	REFERENCE TO ARRAY [1..16] OF REAL	Basis-Koordinatensystem
JointCoordinates	REFERENCE TO ARRAY [1..16] OF REAL	Achs-Koordinatensystem

Tab. 5-163: Argumente der Interfacefunktion Forward

Kann ein Punkt nicht transformiert werden, so muss die Funktion einen Wert ungleich Null zurückgeben.

Mögliche Floating Point-Fehler (wie z. B. Division durch Null) werden vom System abgefangen als Fehler ausgegeben (siehe [Fehlerhandling bei Interfacefunktionen auf Seite 306](#)).

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF, Inverse

Diese Methode wird vom System aufgerufen, wenn vom Basis-Koordinatensystem (BCS) ins Achs-Koordinatensystem (ACS) umgerechnet werden soll. Die Methode wird innerhalb des Betriebsmodus zyklisch aufgerufen.

Argument	Typ	Beschreibung
OriginalCoordinates	REFERENCE TO ARRAY [1..16] OF REAL	Basis-Koordinatensystem
JointCoordinates	REFERENCE TO ARRAY [1..16] OF REAL	Achs-Koordinatensystem

Tab. 5-164: Argumente der Interfacefunktion Inverse

Kann ein Punkt nicht transformiert werden, so muss die Funktion einen Wert ungleich Null zurückgeben.

Mögliche Floating Point-Fehler (wie z. B. Division durch Null) werden vom System abgefangen als Fehler ausgegeben (siehe [Fehlerhandling bei Interfacefunktionen auf Seite 306](#)).

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

Fehlerhandling bei Interfacefunktionen

Tritt bei einer der Interfacefunktionen (Methoden) ein Floating Point-Fehler auf, so wird einer der unten stehenden Fehler gemeldet.

Solche Fehler sollten allerdings bereits in der Mathematik der Transformation berücksichtigt werden: So sollten z.B. ein Divisor immer auf Null überprüft werden, bei Gleichheit ein Fehler zurückgegeben und die Division nicht durchgeführt werden.

- F12C0600, Ungültige Operation beim SPS-Funktionsaufruf
- F12C0601, Denormalisierte Zahl beim SPS-Funktionsaufruf
- F12C0602, Division durch Null beim SPS-Funktionsaufruf

- F12C0603, Überlauf beim SPS-Funktionsaufruf
- F12C0604, Unterlauf beim SPS-Funktionsaufruf
- F12C0605, Genauigkeitsfehler beim SPS-Funktionsaufruf

Soll eine Interfacefunktion (Methode) aufgerufen werden, ohne dass diese angemeldet wurde, so wird folgender Fehler gemeldet:

- F12C0498, Transformation in der SPS wurde nicht angemeldet

Kann eine Interfacefunktion nicht erfolgreich ausgeführt werden (Rückgabewert der Funktion ungleich Null, so wird folgender Fehler gemeldet:

- F12C0499, Benutzerdefinierter Transformationsfehler in der SPS

MC_KIN_REF_EXT

MC_KIN_REF_EXT, Übersicht

Das Interface MC_KIN_REF_EXT wird benötigt, um Methoden einer erweiterten Transformation zu definieren.

Ein Funktionsbaustein muss dieses Interface implementieren, um eine erweiterte Transformation mit nachstehendem Funktionsbaustein anmelden zu können:

[MC_SetKinTransformExt auf Seite 213](#)

Methode	Rückgabewert	Argumente	Beschreibung
DetermineMemory	UDINT	ACS TrafoChoise	Methode zum Auflösen von Mehrdeutigkeiten
Forward	UDINT	BCS ACS	Vorwärtstransformation
GetMechanic	UDINT	ACS MechanicData MechanicDataSize MechanicDataRetSize UpdateOnly	Methode für mechanische Informationen einer Transformation
GetUsedAxes	UINT		Freiheitsgrade einer Transformation
Initialisation	UDINT	AxesConfigData TransformationLengths TransformationFactors	Initialisierungsfunktion einer Transformation
Inverse	UDINT	BCS ACS	Rückwärtstransformation

Tab. 5-165: Interface MC_KIN_REF

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF_EXT, DetermineMemory

Diese Methode wird vom System aufgerufen, um der Transformation zu ermöglichen Mehrdeutigkeiten bei der nächsten Rückwärtstransformation aufzulösen.

ML_Robot.library

Argument	Typ	Beschreibung
ACS	REFERENCE TO ARRAY [1..16] OF LREAL	Punkt im Achs-Koordinatensystem
TrafoChoise	BOOL	Transformation darf bei der nächsten Rückwärtstransformation selbst entscheiden, welche der mehrdeutigen Lösungen verwendet werden soll

Tab. 5-166: Argumente der Interfacefunktion DetermineMemory

Kann die Methode nicht erfolgreich abgeschlossen werden, muss ein MLC-Fehlercode zurückgegeben werden:

(siehe [Fehlerhandling bei Interfacefunktionen auf Seite 309](#)).

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF_EXT, Forward

Diese Methode wird vom System aufgerufen, wenn vom Achs-Koordinatensystem (ACS) ins Basis-Koordinatensystem (BCS) umgerechnet werden soll. Die Methode wird innerhalb des Betriebsmodus zyklisch aufgerufen.

Argument	Typ	Beschreibung
BCS	REFERENCE TO ARRAY [1..16] OF LREAL	Punkt im Basis-Koordinatensystem
ACS	REFERENCE TO ARRAY [1..16] OF LREAL	Punkt im Achs-Koordinatensystem

Tab. 5-167: Argumente der Interfacefunktion Forward

Kann ein Punkt nicht transformiert werden, muss ein MLC-Fehlercode zurückgegeben werden:

(siehe [Fehlerhandling bei Interfacefunktionen auf Seite 309](#)).

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF_EXT, GetMechanic

Diese Methode wird vom System aufgerufen, wenn von einem Punkt im Achs-Koordinatensystem (ACS) die dazugehörigen mechanischen Informationen erfragt werden sollen. Um die mechanischen Informationen auszufüllen steht die Hilfsfunktion [ML_AddMechanicData auf Seite 215](#) zur Verfügung.

Argument	Typ	Beschreibung
ACS	REFERENCE TO ARRAY [1..16] OF LREAL	Punkt im Achs-Koordinatensystem
MechanicData	REFERENCE TO ML_KIN_MECHANIC_DATA	ML_KIN_MECHANIC_DATA auf Seite 314 , Auszufüllende Mechanische Daten
MechanicDataSize	UDINT	Anzahl Elemente in MechanicData
MechanicDataRetSize	REFERENCE TO UDINT	Anzahl ausgefüllter Elemente in MechanicData
UpdateOnly	BOOL	Es sollen nur änderbare Einträge zurückgegeben werden

Tab. 5-168: Argumente der Interfacefunktion Inverse

Kann die Methode nicht erfolgreich abgeschlossen werden, muss ein MLC-Fehlercode zurückgegeben werden:

(siehe [Fehlerhandling bei Interfacefunktionen auf Seite 309](#)).

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF_EXT, GetUsedAxes

Diese Methode wird vom System aufgerufen, um von der Transformation die möglichen Freiheitsgrade zu erfragen. Es müssen ODER-verknüpfte Werte vom Typ ML_KIN_USED_AXES zurückgegeben werden, wenn die Transformation einen der Freiheitsgrade unterstützt.

(siehe [ML_KIN_USED_AXES](#) auf Seite 311)

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF_EXT, Initialisation

Diese Methode wird vom System aufgerufen, wenn eine Transformation initialisiert werden soll. Das Initialisieren findet beim Phasenschalten von P2 nach BB statt.

In dieser Funktion können Vorberechnungen und Überprüfungen der Angaben vorgenommen werden.

Argument	Typ	Beschreibung
AxesConfigData	REFERENCE TO ARRAY [1..16] OF ML_KIN_AXES_CONFIG_DATA	K-0-0008 Kinematikkonfiguration ML_KIN_AXES_CONFIG_DATA auf Seite 310
TransformationLengths	REFERENCE TO ARRAY [1..16] OF LREAL	K-0-0040 Transformation, Längen der Kinematik
TransformationFactors	REFERENCE TO ARRAY [1..16] OF LREAL	K-0-0041 Transformation, Faktoren der Kinematik

Tab. 5-169: Argumente der Interfacefunktion Initialisation

Kann die Initialisierung nicht erfolgreich abgeschlossen werden, muss ein MLC-Fehlercode zurückgegeben werden:

(siehe [Fehlerhandling bei Interfacefunktionen](#) auf Seite 309).

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

MC_KIN_REF_EXT, Inverse

Diese Methode wird vom System aufgerufen, wenn vom Basis-Koordinatensystem (BCS) ins Achs-Koordinatensystem (ACS) umgerechnet werden soll. Die Methode wird innerhalb des Betriebsmodus zyklisch aufgerufen.

Argument	Typ	Beschreibung
OriginalCoordinates	REFERENCE TO ARRAY [1..16] OF LREAL	Punkt im Basis-Koordinatensystem
JointCoordinates	REFERENCE TO ARRAY [1..16] OF LREAL	Punkt im Achs-Koordinatensystem

Tab. 5-170: Argumente der Interfacefunktion Inverse

Kann ein Punkt nicht transformiert werden, muss ein MLC-Fehlercode zurückgegeben werden:

(siehe [Fehlerhandling bei Interfacefunktionen](#) auf Seite 309).

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

Fehlerhandling bei Interfacefunktionen

Tritt bei einer der Methoden ein Floating Point-Fehler auf, so wird der Fehler automatisch gemeldet.

Solche Fehler sollten allerdings bereits in der Mathematik der Transformation berücksichtigt werden: So sollten z.B. ein Divisor immer auf Null überprüft, bei Gleichheit ein Fehler zurückgegeben und die Division nicht durchgeführt werden.

ML_Robot.library

- F12C0600: Ungültige Operation beim SPS-Funktionsaufruf
- F12C0601: Denormalisierte Zahl beim SPS-Funktionsaufruf
- F12C0602: Division durch Null beim SPS-Funktionsaufruf
- F12C0603: Überlauf beim SPS-Funktionsaufruf
- F12C0604: Unterlauf beim SPS-Funktionsaufruf
- F12C0605: Genauigkeitsfehler beim SPS-Funktionsaufruf

Anders als zur einfachen benutzerdefinierten Transformation kann bei der erweiterten benutzerdefinierten Transformation als Rückgabewert der Funktionen ein MLC-Fehlercode zurückgegeben werden. Dadurch ist für den Anwender eine einfachere Diagnose möglich. Ein MLC-Fehlercode hat die Form 16#Fxxxxxxx. Wurde eine Schnittstellenfunktion fehlerfrei ausgeführt, so muss ein Code der Form 16#0xxxxxxx zurückgegeben werden.

Die wichtigsten Fehlercodes für eine Transformation:

- F22B0021: Achswinkel außerhalb Definitionsbereich
- F12C0479: WC-Nullpunkt singulär
- F12C047A: Zu nahe am Stativ
- F22B0022: Punkt kann durch Transformation nicht erreicht werden
- F12C008A: Transformationsparameter passen nicht zum Robotertyp
- F22B0023: In Schutzzone der Transformation
- F22B0024: Achse hat falsche Wichtung für Transformation
- F22B0025: Zu wenig Achsen für Transformation

ML_BELT_ERROR_REACTION_MODE

Kurzbeschreibung Der Aufzählungstyp "ML_BELT_ERROR_REACTION_MODE" definiert, welche Fehlerreaktion bei Bandsynchronität erfolgen soll.

Der Aufzählungstyp wird von nachstehendem Funktionsbaustein verwendet:

[ML_SetBeltErrorReaction auf Seite 222](#)

Element	Wert	Beschreibung
ML_BELT_ERROR_NO_SPECIFIC_REACTION	0	Keine Fehlerreaktion, Roboter fährt weiter bandsynchron mit
ML_BELT_ERROR_DESYNC_OVER_TIME	1	Absynchronisieren über Zeit
ML_BELT_ERROR_DESYNC_WITH_LIMITS	2	Absynchronisieren über Grenzwerte

Tab. 5-171: Elemente des Aufzählungstyps ML_BELT_ERROR_REACTION_MODE

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_AXES_CONFIG_DATA

In der Struktur "ML_KIN_AXES_CONFIG_DATA" ist die Achskonfiguration einer Kinematik gespeichert.

Die Struktur wird von nachstehenden Interfaces verwendet:

- [MC_KIN_REF auf Seite 305](#)
- [MC_KIN_REF_EXT auf Seite 307](#)

Strukturelement	Typ	Beschreibung
Configuration	ML_KIN_AXIS_CONFIG	Achskonfiguration Siehe ML_KIN_AXIS_CONFIG auf Seite 311
ControlNumber	WORD	Steuerungsnummer der Achse
AxisNumber	WORD	Achsnummer der Achse

Tab. 5-172: Datenstruktur ML_KIN_AXES_CONFIG_DATA

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_AXIS_CONFIG

Kurzbeschreibung Der Aufzählungstyp "ML_KIN_AXIS_CONFIG" wird von nachstehender Struktur für die Konfiguration einer Achse in einer Kinematik verwendet:

[ML_KIN_AXES_CONFIG_DATA](#) auf Seite 310

Element	Wert	Beschreibung
ML_KIN_AXIS_TYPE_NONE	16#00	Achse wird in der Kinematik nicht verwendet
ML_KIN_AXIS_TYPE_POSITION	16#08	Achse wird als Positionierachse der Kinematik verwendet
ML_KIN_AXIS_TYPE_ORIENTATION	16#10	Achse wird als Orientierungsachse der Kinematik verwendet
ML_KIN_AXIS_TYPE_BELT	16#20	Achse wird als Bandachse der Kinematik verwendet

Tab. 5-173: Definition der Elemente des Aufzählungstyps ML_KIN_AXIS_CONFIG

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_SAFE_ZONE_TYPE

Typdefinitionen Der Aufzählungstyp "ML_KIN_SAFE_ZONE_TYPE" wird von den Funktionsbausteinen [ML_GroupSetSafeZone](#) auf Seite 218 und [ML_RobotGetSafeZones](#) auf Seite 238 verwendet.

Er definiert, wie eine Schutzzone konfiguriert ist:

Element	Wert	Beschreibung
ML_KIN_SAFE_ZONE_DEACTIVATE	0	Deaktiviert
ML_KIN_SAFE_ZONE_INSIDE_OK	1	Software-Endschalter
ML_KIN_SAFE_ZONE_OUTSIDE_OK	2	Schutzzone

Tab. 5-174: Elemente des Aufzählungstyps ML_KIN_SAFE_ZONE_TYPE

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_USED_AXES

Typdefinitionen Der Aufzählungstyp "ML_KIN_USED_AXES" definiert die Freiheitsgrade einer Kinematik.

Der Aufzählungstyp wird für die Definition einer Kinematik verwendet:

[Erweiterte Benutzerdefinierte Transformation](#) auf Seite 178

Element	Wert	Beschreibung
X_AXIS	16#01	Freiheitsgrad X-Achse
Y_AXIS	16#02	Freiheitsgrad Y-Achse
Z_AXIS	16#04	Freiheitsgrad Z-Achse

ML_Robot.library

Element	Wert	Beschreibung
A_AXIS	16#08	Freiheitsgrad Rotation um X-Achse
B_AXIS	16#10	Freiheitsgrad Rotation um Y-Achse
C_AXIS	16#20	Freiheitsgrad Rotation um Z-Achse

Tab. 5-175: Definition der Elemente des Aufzählungstyps ML_KIN_USED_AXES

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

5.4.3 Data Types/ Info

E_KIN_MECH_CMD

Kurzbeschreibung Der Aufzählungstyp "E_KIN_MECH_CMD" definiert, auf welche Weise ein mechanisches Teilobjekt einer Kinematik erzeugt bzw. modifiziert werden soll.

Der Aufzählungstyp wird von nachstehender Struktur verwendet:

[ML_KIN_MECHANIC_DATA auf Seite 314](#)

Element	Wert	Beschreibung
KIN_MECH_CMD_CREATE_ROD	0	- Erzeuge einen Stab - Der Stab zeigt vom Nullpunkt in positive X Richtung - Der Stab hat die Länge 1 und den Radius 1
KIN_MECH_CMD_CREATE_SPHERE	1	- Erzeuge eine Kugel - Die Kugel liegt im Nullpunkt - Die Kugel hat einen Radius von 1
KIN_MECH_CMD_POSITION	2	Ändere die Position eines Objekts auf (x,y,z)
KIN_MECH_CMD_DIRECTION	3	Ändere die Richtung eines Objekts auf (x,y,z)
KIN_MECH_CMD_SIZE	4	Ändere die Größe eines Objekts auf (x,y,z)

Tab. 5-176: Elemente des Aufzählungstyps E_KIN_MECH_CMD

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KINEMATICSDATA_SM

Kurzbeschreibung Eine Instanz der Struktur "ML_KINEMATICSDATA_SM" enthält die Informationen der korrespondierenden Kinematik, auf die per **Shared Memory** zugegriffen werden kann.

Folgende Strukturelemente in ML_KINEMATICSDATA_SM werden vom jeweiligen Kinematiktyp unterstützt:

Strukturelement	Beschreibung	Zugriff
uiCoordSum_i	K-0-0007 , Achsanzahl der Kinematik	lesend
uiUpdateCount_i	Zähler für Update (ungerade: Beginn des Updates, gerade: Update abgeschlossen)	lesend
arPositionJointCoord_i	Generierte Sollwerte für die zugeordneten Achsen	lesend
arPositionOriginalCoord_i	K-0-0100 , Aktuelle Position, Kartesische Maschinenkoordinaten	lesend
arPositionWorkpieceCoord_i	K-0-0105 , Aktuelle Position, Kartesische Werkstückkoordinaten (Robot-Control V1)	lesend
K-0-0021 Kinematikstatus		

Strukturelement	Beschreibung	Zugriff
Kin_StandStill	K-0-0021.05 Alle Achsen der Kinematik stehen	lesend
Kin_InPosition	K-0-0021.06 Programmierte Endposition ist erreicht	lesend
Kin_BeltSynchronous	K-0-0021.07 Bandsynchron	lesend
Kin_Warning	K-0-0021.08 Warnung	lesend
Kin_Error	K-0-0021.09 Fehler	lesend
Kin_NotHomed	K-0-0021.10 Homing erforderlich, nicht alle Achsen sind referenziert	lesend
Kin_Blending	K-0-0021.11 Überschleifen aktiv	lesend
Kin_DrivesOn	K-0-0021.14 Alle Achsen betriebsbereit (AF)	lesend
Kin_ErrorReaction	K-0-0021.17 Fehlerreaktion aktiviert	lesend
Kin_FullyBeltSynchronous	K-0-0021.20 Voll Bandsynchron	lesend
Kin_WaitBeltStartCondition	K-0-0021.22 Warten auf bandsynchrone Startbedingung	lesend
Kis_SplineCalcActive	K-0-0021.28 Spline für Überschleifbewegung wird berechnet	lesend
Kis_SplineValid	K-0-0021.29 Spline ist gültig	lesend
K-0-0022 Erweiterter Kinematikstatus		
Kis_Errorstop	K-0-0022.00 Kinematik ist in "Errorstop"	lesend
Kis_Stopping	K-0-0022.01 Kinematik ist in "Stopping"	lesend
Kis_Incomplete	K-0-0022.03 Nicht alle Achsen sind der Kinematik zugeordnet (incomplete)	lesend
Kis_StandStill	K-0-0022.07 Kinematik ist in "StandStill"	lesend
Kis_ManualMotion	K-0-0022.10 Kinematik ist im Tipp-Zustand KinData[].stJogCom_q.enJogCmdVel ≠ KIN_JOG_STOP	lesend
Kis_GroupMoving	K-0-0022.12 Kinematik ist in "GroupMoving"	lesend

Tab. 5-177: Strukturelemente in ML_KINEMATICS_DATA_SM

Siehe auch:

- [KinData\[\]](#) - Kinematikdaten im Shared Memory auf Seite 187

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_CMD_INFO

Kurzbeschreibung In der Struktur "ML_KIN_CMD_INFO" ist der Aufbau einer Information eines Kommandos im Satzpuffer definiert.

Die Struktur wird verwendet von:

- FB [ML_RobotGetCmdInfo](#) auf Seite 233

Strukturelement	Typ	Beschreibung
infol	UDINT	ID des Kommandos
infoType	ML_KIN_INFO_TYPE	Art der Information des Kommandos Siehe ML_KIN_INFO_TYPE auf Seite 314
infoValue	REAL	Wert der Informationen des Kommandos

Tab. 5-178: Datenstruktur ML_KIN_CMD_INFO

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_Robot.library

ML_KIN_CS_TYPE

Kurzbeschreibung Der Aufzählungstyp "ML_KIN_CS_TYPE" definiert die Art der Transformation.

Element	Wert	Beschreibung
CS_TYPE_CARTESIAN	0	Kartesische Transformation
CS_TYPE_CYLINDRIC	1	Zylindrische Transformation

Tab. 5-179: Elemente des Aufzählungstyps ML_KIN_CS_TYPE

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_INFO_TYPE

Kurzbeschreibung Der Aufzählungstyp "ML_KIN_INFO_TYPE" definiert, welche Art von Information eines Kommandos im Satzpuffer bereitgestellt werden soll.

Der Aufzählungstyp wird von nachstehender Struktur verwendet:

[ML_KIN_CMD_INFO auf Seite 313](#)

Element	Wert	Beschreibung
KIN_INFO_TYPE_ALL	0	Lese alle Informationen zu allen Kommandos, die aktuell im Satzpuffer vorhanden sind
KIN_INFO_TYPE_PROGRESS	1	Zeige den Fortschritt der Kommandos an
KIN_INFO_TYPE_TOTAL_DISTANCE	2	Zeige die totale Distanz der Kommandos an
KIN_INFO_TYPE_UID	3	Zeige eine eindeutige Identifikationsnummer der Kommandos an
KIN_INFO_TYPE_STATE	4	Zeige den Zustand der Kommandos an

Tab. 5-180: Elemente des Aufzählungstyps ML_KIN_INFO_TYPE

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_MECHANIC_DATA

Kurzbeschreibung In der Struktur "ML_KIN_MECHANIC_DATA" ist die mechanische Struktur eines Roboters definiert.

Die Struktur wird verwendet von:

- FB [ML_RobotGetMechanic auf Seite 234](#)
- FN [ML_AddMechanicData auf Seite 215](#)

Strukturelement	Typ	Beschreibung
cmd	E_KIN_MECH_CMD	Kommando auf ein mechanisches Teilobjekt Siehe E_KIN_MECH_CMD auf Seite 312
object	STRING[7]	Identifiziert das mechanische Teilobjekt
x	REAL	Daten für das Kommando auf das mechanische Teilobjekt
y	REAL	Daten für das Kommando auf das mechanische Teilobjekt
z	REAL	Daten für das Kommando auf das mechanische Teilobjekt

Tab. 5-181: Datenstruktur ML_KIN_MECHANIC_DATA

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_PERF_DATA

Kurzbeschreibung In der Struktur "ML_KIN_PERF_DATA" sind die Daten einer Performance-messung definiert.

Die Struktur wird von nachstehendem Funktionsbaustein verwendet:

[ML_RobotGetPerformance auf Seite 237](#)

Strukturelement	Typ	Beschreibung
name	STRING[15]	Name der Performancemessung
count	UDINT	Anzahl der Messungen
fMin	REAL	Minimale Dauer in µs
fMax	REAL	Maximal Dauer in µs
fAvg	REAL	Durchschnittliche Dauer in µs

Tab. 5-182: Datenstruktur ML_KIN_PERF_DATA

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_KIN_SAFE_ZONE_DATA

Kurzbeschreibung In der Struktur "ML_KIN_SAFE_ZONE_DATA" sind die Daten einer Schutz-zone definiert.

Die Struktur wird von nachstehendem Funktionsbaustein verwendet:

[ML_RobotGetSafeZones auf Seite 238](#).

Strukturelement	Typ	Beschreibung
zoneNumber	UINT	Anzahl der Schutzzonen
zoneType	ML_KIN_SAFE_ZONE_TYPE	Typ der Schutzzone (Siehe ML_KIN_SAFE_ZONE_TYPE auf Seite 311)
maxPos	ARRAY [1..3] OF REAL	Maximale Position
minPos	ARRAY [1..3] OF REAL	Minimale Position

Tab. 5-183: Datenstruktur ML_KIN_SAFE_ZONE_DATA

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_POINTKIND

Kurzbeschreibung (Siehe [Robot-Control, Kommandos und Funktionen auf Seite 35](#)).

Element	Wert	Beschreibung
POINTKIND_PREPARE	0	Punkt bezieht sich auf die Satzvorbereitung
POINTKIND_EXECUTE	1	Punkt bezieht sich auf die Satzabarbeitung

Tab. 5-184: Definition der Elemente des Aufzählungstyps ML_POINTKIND

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

5.4.4 Data Types/ TeachIn

E_KIN_TEACH_IN_POINT_FILE_OPTIONS

Kurzbeschreibung Der Aufzählungstyp "E_KIN_TEACH_IN_POINT_FILE_OPTIONS" enthält die zur Auswahl stehenden Optionen zum Laden und Speichern von Punktedateien.

Der Aufzählungstyp wird durch folgende Funktionsbausteine verwendet:

ML_Robot.library

- [ML_GroupLoadPointFile auf Seite 297](#)
- [ML_GroupSavePointFile auf Seite 301](#)

Typdefinition

```

TYPE E_KIN_TEACH_IN_POINT_FILE_OPTIONS :
(
  E_GROUP_TEACH_IN_NEW           := 0, (*new points*)
  E_GROUP_TEACH_IN_APPEND       := 1, (*append points*)
  E_GROUP_TEACH_IN_APPEND_OVERWRITE := 2, (*append points and override existing points*)
);
END_TYPE

```

Beim Laden der Punktedatei wirken die Optionen wie folgt:

- **Neu**
Der Inhalt der Kinematik wird gelöscht.
Die Punkte der Punktedatei werden in die Kinematik eingetragen.
- **Hinzufügen**
Der Inhalt der Kinematik bleibt erhalten.
Punkte, die nicht in der Kinematik enthalten sind, werden aus der Punktedatei der Kinematik hinzugefügt.
- **Hinzufügen und Überschreiben**
Punkte, die in der Kinematik und in der vorhandenen Punktedatei enthalten sind, werden mit den Werten aus der Punktedatei überschrieben.
Der Inhalt der anderen Punkte in der Kinematik bleibt erhalten.
Die anderen Punkte der vorhandenen Punktedatei werden der Kinematik hinzugefügt.

Beim Speichern der Punktedatei wirken die Optionen wie folgt:

- Ist die angegebene Punktedatei noch nicht vorhanden, wird sie neu angelegt
- Ist die angegebene Punktedatei schon vorhanden, gibt es folgende Optionen:
 - **Neu**
Der Inhalt der vorhandenen Punktedatei wird gelöscht.
Die Punkte der Kinematik werden in die Punktedatei eingetragen.
 - **Hinzufügen**
Der Inhalt der vorhandenen Punktedatei bleibt erhalten.
Punkte, die nicht in der vorhandenen Punktedatei enthalten sind, werden aus der Kinematik der Punktedatei hinzugefügt.
 - **Hinzufügen und Überschreiben**
Punkte, die in der vorhandenen Punktedatei und in der Kinematik enthalten sind, werden mit den Werten aus der Kinematik überschrieben.
Der Inhalt der anderen Punkte in der vorhandenen Punktedatei bleibt erhalten.
Die anderen Punkte der Kinematik werden der Punktedatei hinzugefügt.

5.4.5 Data Types/ Movement

ML_ARR_COORDS

Kurzbeschreibung Das Feld dient zur Abspeicherung der Koordinaten einer Kinematik-Position in Maschinen-, Welt- oder Werkstück-Koordinaten.

Typdefinition

```
TYPE (* 1..max. Zahl der Koordinaten der Kinematik *)
    ML_ARR_COORDS:ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL;
END_TYPE
```

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_CIRCMODES

Typdefinitionen Der Aufzählungstyp "ML_CIRCMODES" definiert die Interpretation des Eingangs "AuxPoint".

Der Aufzählungstyp wird für die Definition des zu fahrenden Kreisabschnittes von nachstehendem Funktionsbaustein verwendet:

[MC_MoveCircularAbsolute auf Seite 251](#)

Element	Beschreibung
CM_BOARDER	Der Eingang "AuxPoint" definiert einen zu durchfahrenden Punkt auf dem Kreisabschnitt

Tab. 5-185: Elemente des Aufzählungstyps ML_CIRCMODES

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_COORDSYS

Kurzbeschreibung Der Aufzählungstyp "ML_COORDSYS" definiert in welchem Koordinatensystem Punkte angegeben werden.

Der Aufzählungstyp wird von nachstehenden Funktionsbausteinen der Bibliothek "ML_Robot" für die Definition des Koordinatensystems verwendet:

- [MC_GroupReadActualPosition auf Seite 228](#)
- [MC_MoveCircularAbsolute auf Seite 251](#)
- [MC_MoveCircularRelative auf Seite 254](#)
- [MC_MoveDirectAbsolute auf Seite 257](#)
- [MC_MoveDirectRelative auf Seite 259](#)
- [MC_MoveLinearAbsolute auf Seite 261](#)
- [MC_MoveLinearRelative auf Seite 264](#)
- [MC_SetCartesianTransform auf Seite 209](#)
- [ML_GroupJogCont auf Seite 269](#)
- [ML_GroupJogStep auf Seite 271](#)
- [ML_GroupReadNextPoint auf Seite 298](#)
- [ML_GroupReadPoint auf Seite 299](#)
- [ML_GroupWritePoint auf Seite 302](#)
- [ML_RobotGetTrafoValues auf Seite 239](#)
- [ML_RobotMovePoint auf Seite 241](#)
- [ML_RobotReadCyclicChannel auf Seite 242](#)

ML_Robot.library

- [ML_RobotTransformPoint auf Seite 245](#)
- [ML_RobotWriteCyclicChannel auf Seite 277](#)

Element	Wert	Beschreibung
CS_MACHINE_COORDINATES	0	Maschinen-Koordinatensystem
CS_AXES_COORDINATES	1	Achs-Koordinatensystem
CS_BASE_COORDINATES	2	Basis-Koordinatensystem
CS_GRIPPER_COORDINATES	4	Greifer-Koordinatensystem
CS_PRODUCT_COORDINATES_1	16#0003	Produkt-Koordinatensystem 1
CS_PRODUCT_COORDINATES_2	16#0103	Produkt-Koordinatensystem 2
CS_PRODUCT_COORDINATES_3	16#0203	Produkt-Koordinatensystem 3
CS_PRODUCT_COORDINATES_4	16#0303	Produkt-Koordinatensystem 4
CS_PRODUCT_COORDINATES_5	16#0403	Produkt-Koordinatensystem 5
CS_PRODUCT_COORDINATES_6	16#0503	Produkt-Koordinatensystem 6
CS_PRODUCT_COORDINATES_7	16#0603	Produkt-Koordinatensystem 7
CS_PRODUCT_COORDINATES_8	16#0703	Produkt-Koordinatensystem 8
CS_PRODUCT_COORDINATES_9	16#0803	Produkt-Koordinatensystem 9
CS_PRODUCT_COORDINATES_10	16#0903	Produkt-Koordinatensystem 10
CS_PRODUCT_COORDINATES_11	16#0A03	Produkt-Koordinatensystem 11
CS_PRODUCT_COORDINATES_12	16#0B03	Produkt-Koordinatensystem 12
CS_PRODUCT_COORDINATES_13	16#0C03	Produkt-Koordinatensystem 13
CS_PRODUCT_COORDINATES_14	16#0D03	Produkt-Koordinatensystem 14
CS_PRODUCT_COORDINATES_15	16#0E03	Produkt-Koordinatensystem 15
CS_PRODUCT_COORDINATES_16	16#0F03	Produkt-Koordinatensystem 16

Tab. 5-186: Elemente des Aufzählungstyps ML_COORDSYS

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_CYCLIC_MODE

Kurzbeschreibung Der Aufzählungstyp "ML_CYCLIC_MODE" definiert auf welche Art die zyklische Sollwertvorgabe erfolgen soll.

Der Aufzählungstyp wird verwendet vom Funktionsbaustein

[ML_RobotWriteCyclicChannel auf Seite 277](#)

Element	Wert	Beschreibung
CYCLIC_MODE_POS_ABS	0	Zyklische Sollwertvorgabe erfolgt als absolute Position
CYCLIC_MODE_POS_REL	1	Zyklische Sollwertvorgabe erfolgt als relatives Delta zur letzten kommandierten Position in jedem Takt
CYCLIC_MODE_VEL	2	Zyklische Sollwertvorgabe erfolgt als Geschwindigkeit

Tab. 5-187: Definition der Elemente des Aufzählungstyps ML_CYCLIC_MODE

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_SET_MODE

Kurzbeschreibung Der Aufzählungstyp "ML_SET_MODE" definiert die Art des Überschliffens, d. h. ob die Bewegung am Endpunkt endet oder überschliffen wird.

Der Aufzählungstyp wird von folgenden Funktionsbausteinen verwendet:

- [MC_MoveLinearAbsolute auf Seite 261](#)
- [MC_MoveLinearRelative auf Seite 264](#)
- [MC_MoveDirectAbsolute auf Seite 257](#)
- [MC_MoveCircularAbsolute auf Seite 251](#)

Element	Wert	Beschreibung
SM_TO	0	Bewegung endet am Endpunkt mit Geschwindigkeit 0
SM_VIA	1	Bewegung kann überschliffen werden

Tab. 5-188: Definition der Elemente des Aufzählungstyps ML_SET_MODE

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

ML_SLOPE_TYPE

Kurzbeschreibung Der Aufzählungstyp "ML_SLOPE_TYPE" definiert die Slope-Art der Bewegung.

Der Aufzählungstyp wird von folgenden Funktionsbausteinen verwendet:

- [MC_MoveLinearAbsolute auf Seite 261](#)
- [MC_MoveLinearRelative auf Seite 264](#)
- [MC_MoveDirectAbsolute auf Seite 257](#)
- [MC_MoveDirectRelative auf Seite 259](#)
- [MC_MoveCircularAbsolute auf Seite 251](#)
- [MC_MoveCircularRelative auf Seite 254](#)

Element	Wert	Beschreibung
ST_BLOCK_SLOPE	0	Räumliche Abweichung im Überschleifbereich ist geschwindigkeitsabhängig
ST_PROGRAM_SLOPE	1	Räumliche Abweichung im Überschleifbereich ist nicht geschwindigkeitsabhängig
ST_CONTINUOUS_SLOPE	2	Räumliche Abweichung im Überschleifbereich ist nicht geschwindigkeitsabhängig, die Stetigkeit der Bahnbeschleunigung wird beibehalten

Tab. 5-189: Elemente des Aufzählungstyps ML_SLOPE_TYPE

Bibliothek ML_Robot für IndraMotion MLC 14VRS, Funktionspaket ROB

6 Kinematik-Interface

6.1 Einführung

Das **Kinematik-Interface** bündelt und erweitert Robot-Control-Funktionalitäten. Es stellt das Gegenstück zum **Achs-Interface** dar und dient der einfachen und komfortablen Bedienung einer Kinematik. So ist es möglich in kürzester Zeit gut strukturierte Applikationsprogramme für Robot-Control-Applikationen zu entwickeln.

Systemvoraussetzung *Für das Kinematik-Interface ist folgende Hard- und Software notwendig:*

- Software: IndraMotion MLC 14VRS
- Firmware: IndraMotion ML* 14VRS
- *CML-Steuerung:*
 - IndraControl L25 oder
 - IndraControl L45 oder
 - IndraControl L65 oder
 - IndraControl L75
- *XM-Steuerung:*
 - IndraControl XM2
- Template: Achs-Interface Beispielprojekt oder GAT

Integration Kinematik-Interface Die Integration des Kinematik-Interface ist davon abhängig, ob ein bestehendes oder neues Projekt als Grundlage dient.

Bei bestehenden Projekten ist das Achs-Interface Beispielprojekt oder eine GAT-Variante Voraussetzung.

Es kann wie folgt vorgegangen werden:

1. Starten Sie den GAT-Wizard über das Symbol .
2. Wählen Sie den Knoten "Plugins"
3. Wählen Sie die Funktion "Plugin hinzufügen"
4. Wählen Sie das Plug-in "KinInterface" und bestätigen den Import mit **Übernehmen**

Bei neuen Projekten kann das Kinematik-Interface direkt beim Anlegen einer Steuerung mit ausgewählt werden.

Folgende Schritte sind durchzuführen:

1. Wählen Sie im zweiten Schritt "Konfiguration" beim Anlegen einer Steuerung als SPS Applikation "GAT Wizard" aus.
2. Wählen Sie im folgenden Schritt "SPS Applikation" das Plug-in "Kinematik-Interface" aus.
3. Schließen Sie das Anlegen der Steuerung ab. Das Kinematik-Interface wird zusammen mit dem ausgewählten Template automatisch hinzugefügt.

6.2 Funktionsbeschreibung

6.2.1 Überblick

Das Kinematik-Interface wird als Plug-in zum IndraMotion Programmier-Template Achs-Interface zur Verfügung gestellt. Der strukturelle Aufbau und die

Kinematik-Interface

Funktionsweise ähnelt daher der des Achs-Interfaces. Was beim Achs-Interface die Achse ist, ist beim Kinematik-Interface die Kinematik.

Die folgende Abbildung zeigt die Datenstruktur des Kinematik-Interfaces:

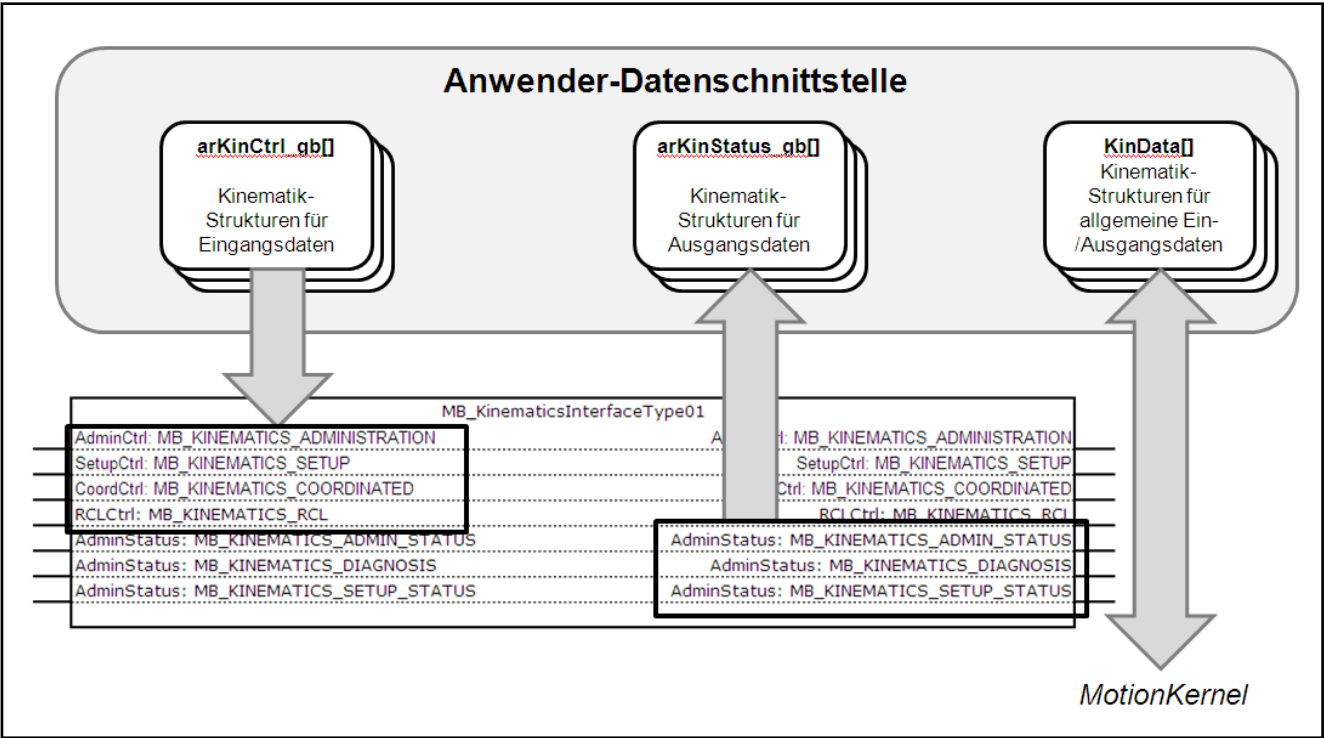


Abb. 6-1: Kinematik-Interface - Datenstruktur des Interfaces

Die folgende Tabelle zeigt das Anwender-Interface zur Kinematik-Interface Datenstruktur:

Anwender-Interface	Typ	Beschreibung
arKinCtrl[]_gb	MB_KINEMATICS_CONTROL_TYPE01 (Anwender-Erweiterung: TE_KINEMATICS_CONTROL_TYPE01) MB_KINEMATICS_CONTROL_TYPE11 (Anwender-Erweiterung: TE_KINEMATICS_CONTROL_TYPE11)	Steuerungsstruktur inklusive Sollwerte und Variablen zum Aktivieren der Betriebsarten
arKinStatus[]_gb	MB_KINEMATICS_STATUS_TYPE01 (Anwender-Erweiterung: TE_KINEMATICS_CONTROL_TYPE01)	Statusstruktur inklusive Diagnose-Info und Quitungen für die Betriebsarten
KinData[]		Status- und Steuerungsstruktur inklusive Istwerte und Status-Bits (wird durch die Motion-Control in der Firmware bearbeitet)

Tab. 6-1: Kinematik-Interface - Datenstruktur des Anwender-Interfaces

Wie beim Programmier-Template "Achs-Interface Beispielprojekt" liefert auch das Kinematik-Interface ein äquivalentes Beispielprojekt, welches beim Einbinden des Plug-in automatisch erstellt wird. Dieses Beispielprojekt ist unter dem Reiter **Logic** ▶ **Application** ▶ **AxisInterface** ▶ **Plugins** ▶ **KinInterface** zu finden.

Der Offset von 10 weist auf die performanceoptimierte ereignisgesteuerte Arbeitsweise hin. Damit entspricht der TYPE11 im Funktionsumfang dem TYPE01, arbeitet jedoch ereignisgesteuert.

Änderungen von MB_KinInterfaceType11 gegenüber MB_KinInterfaceType01

Das ereignisgesteuerte Kinematik-Interface MB_KinInterfaceType11 ist, bis auf die nachfolgenden Punkte, funktionsgleich zu dem zyklischen Kinematik-Interface MB_KinInterfaceType01:

1. Unterelemente der Steuerungsstruktur arKinCtrl[]_gb enthalten Interfaces und Properties anstelle von Strukturen. Durch Nutzung der Interfaces und Properties wird Laufzeit gespart, da intern bei Zuweisung auf die Steuerungsstruktur direkt Code ausgeführt wird. Damit sind direkte Aktionen bei der Zuweisung möglich und ein zyklisches Scannen der Daten entfällt.
2. Beim Setzen der Betriebsartenvorgabe über Bits (arKinCtrl_gb[].Admin._ OpMode.b) werden alle anderen Betriebsartenbits automatisch gelöscht.
3. Der Ausgang CmdDone (arKinStatus_gb[].Admin.CmdDone) wird bei Betriebsarten- und Sollwertänderung sofort zurückgesetzt. Dadurch kann der Ausgang CmdDone direkt nach der Betriebsarten- oder Sollwertvorgabe in der nächsten Codezeile abgefragt werden.
4. Die Laufzeit für den zyklischen Aufruf des Kinematik-Interface-Funktionsbausteins ist deutlich geringer.

6.2.2 Programm zum Kinematik-Interface

*Das Programm **TE_KinInterfaceType01Prog** besitzt folgende Aufgaben:*

- **Initialisierung des Kinematik-Interfaces:**
Bei Erreichen des Parametriermodus (Phase 2) wird für alle konfigurierten Kinematiken mit Hilfe des Funktionsbausteins **TE_KinematicsInitAllKinematicsType01** bzw. **TE_KinematicsInitAllKinematicsType11** das Kinematik-Interface initialisiert. Bei Erreichen des Betriebsmodus (Phase 4) wird die Initialisierung erneut ausgeführt, um Änderungen in der Parametrierung zu erkennen
- **Zyklischer Aufruf des Kinematik-Interfaces:**
Nach erfolgreicher Initialisierung (signalisiert durch "TE_KinInterfaceType01Prog.InitDone") wird das Kinematik-Interface zyklisch aufgerufen. Dazu wird der Funktionsbaustein **TE_KinematicsInterface** aufgerufen, welcher vom Funktionsbaustein "MB_KinInterfaceType01" bzw. "MB_KinInterfaceType11" abgeleitet ist und somit Anwendererweiterungen ermöglicht.
- **Anwendererweiterungen des Kinematik-Interfaces:**
Der Funktionsbaustein "TE_KinematicsInterface" gibt dem Anwender die Möglichkeit den Strukturen des Kinematik-Interfaces weitere Elemente hinzuzufügen, um somit seine Anwendung optimal zu erweitern. Nähere Hinweise siehe [Kinematik-Interface Anwendererweiterung auf Seite 350](#).

6.2.3 Visualisierungen zum Kinematik-Interface

Überblick

Zum Kinematik-Interface werden Visualisierungsmasken mitgeliefert, um ein vorgefertigtes und einfaches Interface zum Einstellen und Ansteuern der Kinematiken zur Verfügung zu stellen.

Kinematik-Interface

Folgende Visualisierungen sind enthalten:

Visualisierung	Beschreibung
SystemOverview	
KinSystem_Overview_All	Gesamtüberblick über alle definierten Kinematiken, einschließlich einfacher Diagnosen, Status und Koordinaten
KinematicsModes	
Kinematics_Overview	Zeigt aktuelle Betriebsart und aktuellen Status des Kinematik-Interfaces, zusammen mit Navigationsschaltflächen zu allen weiteren Visualisierungen
KinHoming_mode	Kommandierung und Überwachen der Referenzierbetriebsart
KinPosition_mode	Kommandierung und Überwachen der Positionierbetriebsarten
Utilities	
ConveyorBelt	Konfiguration eines oder mehrerer Bänder
Setup	Kommandierung und Überwachen des Tipp-Betriebs
TeachIn	Schreiben, Lesen und Löschen eines Punktes bzw. Lesen eines nächsten Punktes Speichern, Laden und Löschen von Punktedateien
ProductCoordSystem	Konfiguration eines oder mehrerer Produktkoordinatensysteme

Tab. 6-2: Kinematik-Interface Visualisierungen

Die folgenden globalen Variablen werden zum Steuern und für den Zugriff auf Systeminformationen innerhalb der Visualisierungen benutzt:

- arKinCtrl_gb[]
- arKinStatus_gb[]
- KinData[]
- VisuKinematicsNo

Systemübersicht Visualisierung

Die Visualisierung **KinSystem_Overview_All** erlaubt es dem Anwender jede, im Projekt konfigurierte Kinematik schnell anzukoppeln. Die angezeigte Standard-Kinematik besitzt keine Referenz. Wie diese Referenz gesetzt werden muss, wird im Folgenden beschrieben. Zusätzliche Kinematiken können im Offline-Betrieb zur Anzeige hinzugefügt werden.

Die Visualisierung liefert einen Gesamt-Systemstatus und ermöglicht das Löschen von Fehlern. Einzelne Kinematik-Bedienelemente liefern den Kinematiknamen und die Kinematiknummer. Einige Kinematik-Bedienelemente liefern Diagnosen sowie die aktuelle Position in Weltkoordinaten zusammen mit den Betriebsarten-Statusanzeigen.

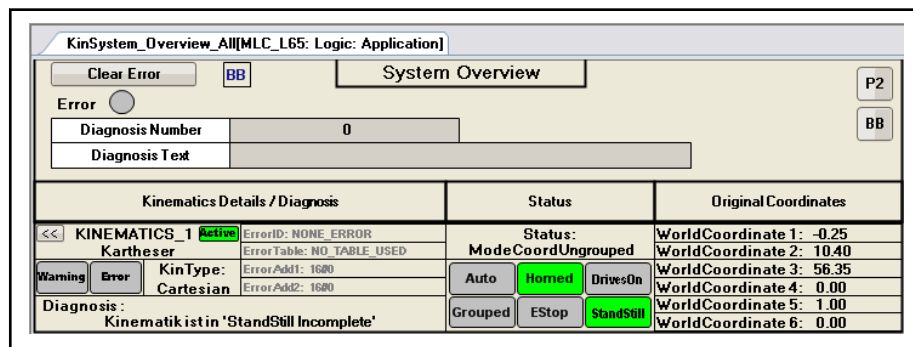


Abb. 6-2: KinSystem_Overview_All Anzeige

Clear Error Mit dieser Schaltfläche wird ein "Clear Error" über das IMC-Interface ausgelöst.



Es werden keine Kinematik- oder Achsfehler gelöscht. Diese müssen separat in den jeweiligen Kinematik- bzw. Achsübersichten gelöscht werden.

P2 Mit dieser Schaltfläche kann in den Parametriemodus P2 umgeschaltet werden.

BB Mit dieser Schaltfläche kann in den Betriebsmodus BB umgeschaltet werden.

Hinzufügen einer Kinematik zur Systemübersicht

Das Hinzufügen einer Kinematik zur Systemübersichtsanzeige erfolgt wie das Hinzufügen einer neuen Visualisierung durch Anwählen des entsprechenden Elements und der Festlegung der Kinematikreferenz. Nachfolgend aufgeführt sind zwei Vorgehensweisen, wie eine neue Kinematik der Systemübersicht hinzugefügt werden kann:

Möglichkeit 1: Erstellen einer Kinematik-Visualisierung mittels Frame

1. Öffnen Sie das entsprechende IndraWorks-Projekt und wählen Sie den Reiter **Logic** ► **Application** ► **AxisInterface** ► **Plugins** ► **KinInterface** ► **Visualizations** ► **SystemOverview** an.
2. Mit IndraLogic im Offline-Betrieb Doppelklick auf die Visualisierung "KinSystem_Overview_All".
3. Wählen Sie im Fenster **Werkzeuge** die Schaltfläche **Frame**  an und erzeugen Sie einen Grundriss unterhalb der letzten Kinematik in entsprechender Größe.

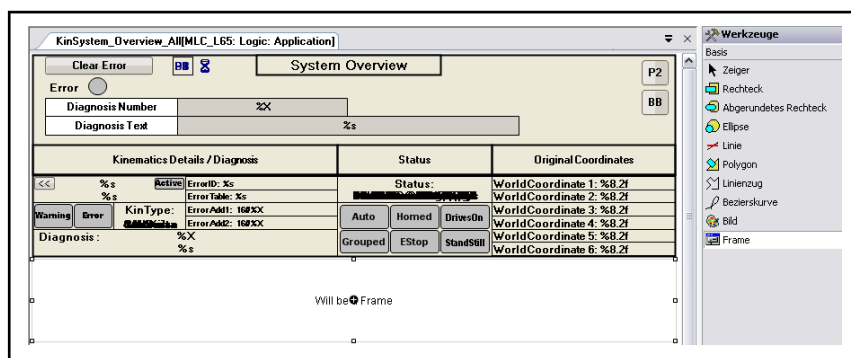


Abb. 6-3: Hinzufügen einer Kinematik zur Systemübersicht

4. Mit rechter Maustaste auf den Frame klicken und "Frameauswahl" aktivieren. Wählen Sie das **KinSystem_Overview**-Element aus dem Visualisierungsauswahlfenster aus.

Eine komplett neue Systemübersicht für die Kinematik erscheint.

5. Öffnen Sie das Eigenschaften-Fenster für den neu eingefügten Frame und tragen Sie unter **Referenzierte Visualisierungen** ► **Referenzierte Visualisierungen** ► **KinSystem_Overview** ► **m_Input_KinRef** den Namen der Kinematik ein. In gleicher Art und Weise kann so auch die Referenz der ersten Kinematik angepasst werden.



Der Name der Kinematik ist unter anderem in der globalen Variablenliste "MlcVarGlobal" zu finden.

Die Systemübersicht steht nun für die angelegte Kinematik zur Verfügung.

Kinematik-Interface

Möglichkeit 2: Kopieren einer bestehenden Kinematikvisualisierung

1. Öffnen Sie das entsprechende IndraWorks-Projekt und wählen Sie den Reiter **Logic** ▶ **Application** ▶ **AxisInterface** ▶ **Plugins** ▶ **KinInterface** ▶ **Visualizations** ▶ **SystemOverview** an.
2. Mit IndraLogic im Offline-Betrieb Doppelklick auf die Visualisierung "Kin-System_Overview_All".
3. Kopieren Sie die bereits vorhandene Kinematik-Visualisierung und fügen Sie sie beliebig oft ein.
4. Öffnen Sie das Eigenschaften-Fenster für die neu eingefügten Kinematikvisualisierungen und tragen Sie unter **Referenzierte Visualisierungen** ▶ **Referenzierte Visualisierungen** ▶ **KinSystem_Overview** ▶ **m_Input_KinRef** den Namen der Kinematik ein. In gleicher Art und Weise kann so auch die Referenz der ersten Kinematik angepasst werden.



Der Name der Kinematik ist unter anderem in der globalen Variablenliste "MlcVarGlobal" zu finden.

Die Systemübersicht steht nun für die angegebene Kinematik zur Verfügung.



Eine der oben genannten Möglichkeiten muss für alle, zusätzlich zum Projekt hinzugefügte Kinematiken wiederholt werden.

Systemübersicht Navigation

Aus der Kinematik-Systemübersicht ist es über  möglich, zur Visualisierung der Einzelkinematik "Kinematics_Overview" zu gelangen. Diese Schaltfläche befindet sich links in der Spalte "Kinematics Details/Diagnosis". Aus der Einzelkinematikanzeige sind alle weiteren Visualisierungen erreichbar.

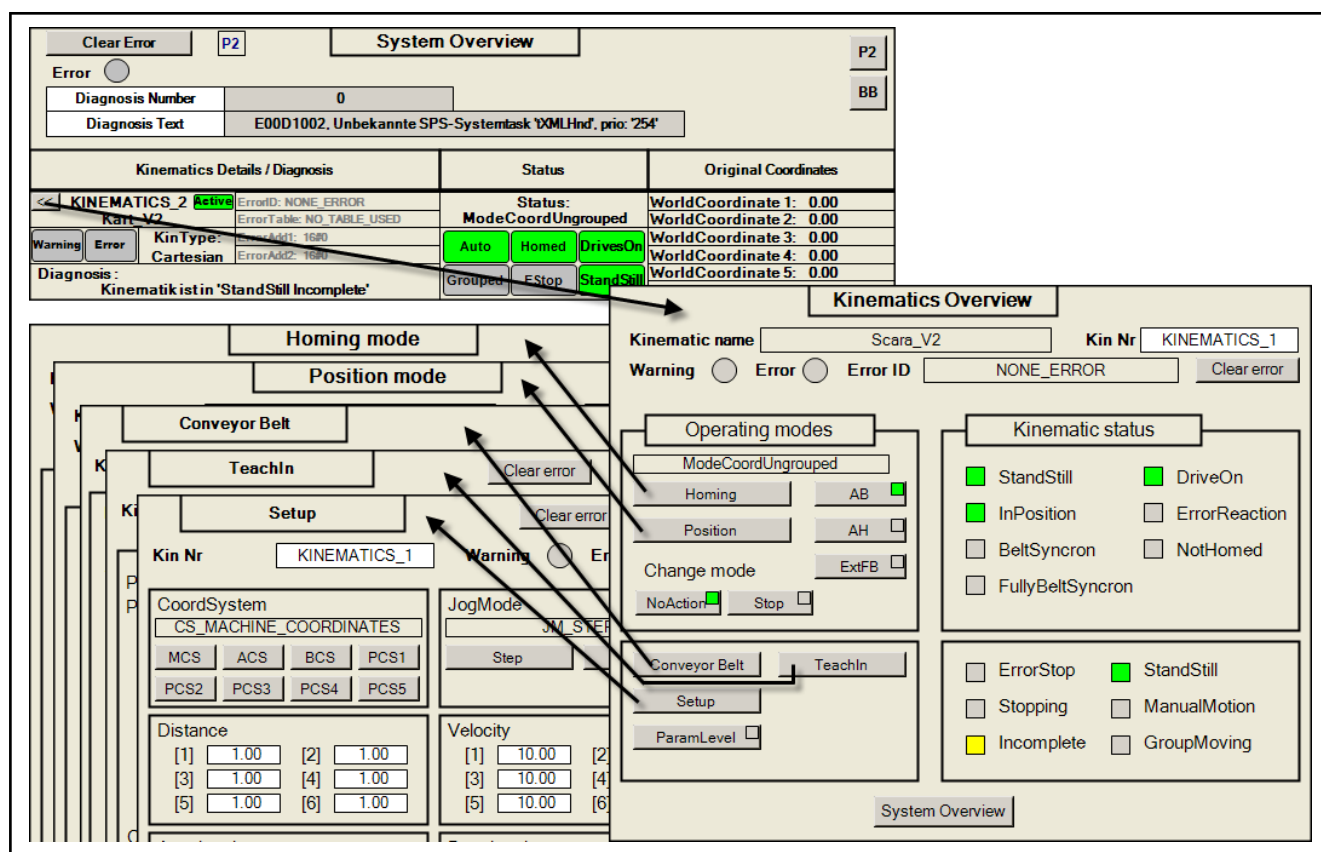


Abb. 6-4: Systemübersicht Navigation



Jede Visualisierung bietet die Möglichkeit durch Eingabe einer Kinematiknummer im Feld **Kin Nr** zwischen verschiedenen Kinematiken hin und her zu wechseln.

Kinematikübersicht Visualisierung

Die Visualisierung **Kinematics_Overview** stellt übersichtlich die wichtigsten Funktionen des Kinematik-Interfaces und Diagnosen der KinData-Schnittstelle dar.

Kinematik-Interface

Kinematics Overview

Kinematic name: Kin Nr:

Warning ☐ Error ☐ Error ID:

Operating modes

☒

☐

Change mode ☐

☒ NoAction ☐ Stop

Kinematic status

☒ StandStill ☒ DriveOn

☒ InPosition ☐ ErrorReaction

☐ BeltSynchron ☐ NotHomed

☐ FullyBeltSynchron

☐

Abb. 6-5: Kinematikübersicht Visualisierung

Über die Schaltflächen **Homing** und **Position** kann die Visualisierung für die entsprechende Betriebsart umgeschaltet werden, in welcher spezielle Werte für die jeweilige Betriebsart eingegeben werden können. Die Schaltflächen **AB**, **AH** und **ExtFB** erlauben eine direkte Umschaltung in die entsprechende Betriebsart. Mithilfe der Schaltflächen **NoAction** und **Stop** kann die Reaktion nach einem Betriebsartenwechsel definiert werden.

Über die Schaltflächen **Conveyor Belt**, **Setup** und **TeachIn** öffnen sich die entsprechenden unter "Utilities" abgelegten Visualisierungen zum Konfigurieren eines Bandes, Vertippen der Kinematik sowie teachen von Punkten. Ist der Tipp-Betrieb aktiviert, so wird dies über die Meldung "SetupMode ACTIVE" signalisiert.

Der rechte Teil der Visualisierung stellt die Diagnosedaten der KinData-Schnittstelle dar und zeigt den aktuellen PLCOpen-Zustand der Kinematik an.

Referenzierübersicht (Homing) Visualisierung

Die Visualisierung **KinHoming_mode** dient zur Referenzierung von Kinematikachsen.

Homing mode

Kinematic name

Kin Nr

Warning ☐ Error ☐

Error ID

Homing sequence

1. Axis No	1
2. Axis No	2
3. Axis No	3
4. Axis No	4
5. Axis No	5
6. Axis No	6
7. Axis No	0
8. Axis No	0
9. Axis No	0
10. Axis No	0
11. Axis No	0
12. Axis No	0
13. Axis No	0
14. Axis No	0
15. Axis No	0
16. Axis No	0

Actual values

☐	1. Axis homed
☐	2. Axis homed
☐	3. Axis homed
☐	4. Axis homed
☐	5. Axis homed
☐	6. Axis homed
☐	7. Axis homed
☐	8. Axis homed
☐	9. Axis homed
☐	10. Axis homed
☐	11. Axis homed
☐	12. Axis homed
☐	13. Axis homed
☐	14. Axis homed
☐	15. Axis homed
☐	16. Axis homed
☐	Kinematics homed

Abb. 6-6: Referenzierübersicht Visualisierung

Die Reihenfolge der zu referenzierenden Achsen kann beliebig gewählt werden. Standardmäßig sind alle, der Kinematik zugeordneten Achsen entsprechend der Konfigurationsreihenfolge eingetragen. Als Abbruchbedingung für die Homing-Sequenz steht die Achsnummer 0.

Im rechten Teil der Visualisierung ist der Referenzstatus jeder angegebenen Achse sowie der Kinematik dargestellt. Ist die Achse bzw. die Kinematik in Referenz, so färbt sich das Anzeigeelement grün.



Das Anzeigeelement für "Kinematics homed" wird erst grün, wenn die Homing-Sequenz fehlerfrei abgearbeitet wurde, auch wenn alle Achsen bereits referenziert sind.

Positionierübersicht (Position) Visualisierung

Die Visualisierung **KinPosition_mode** dient dem Positionieren der Kinematik.

Kinematik-Interface

Position mode			
Kinematic name		Scara_V2	
Kin Nr		KINEMATICS_1	
Warning	☐	Error	☐
Error ID	NONE_ERROR		Clear error
Command values		Current values	
Point [1] 0.00 [2] 0.00 [3] 0.00 [4] 0.00 [5] 0.00 [6] 0.00 [7] 0.00 [8] 0.00 [9] 0.00 [10] 0.00 [11] 0.00 [12] 0.00 [13] 0.00 [14] 0.00 [15] 0.00 [16] 0.00 Velocity 10.00 Acceleration 10.00 Deceleration 10.00 Jerk 0.00 BlendRadius 0.00 AuxPoint [1] 0.00 [2] 0.00 [3] 0.00 [4] 0.00 [5] 0.00 [6] 0.00 [7] 0.00 [8] 0.00 [9] 0.00 [10] 0.00 [11] 0.00 [12] 0.00 [13] 0.00 [14] 0.00 [15] 0.00 [16] 0.00		OrigCoord 1 0.00 OrigCoord 2 0.00 OrigCoord 3 0.00 OrigCoord 4 0.00 OrigCoord 5 0.00 OrigCoord 6 0.00 OrigCoord 7 0.00 OrigCoord 8 0.00 OrigCoord 9 0.00 OrigCoord 10 0.00 OrigCoord 11 0.00 OrigCoord 12 0.00 OrigCoord 13 0.00 OrigCoord 14 0.00 OrigCoord 15 0.00 OrigCoord 16 0.00 Velocity 0.00 Acceleration 0.00 Jerk 0.00	
CoordSystem MCS ACS BCS PCS1 PCS2 PCS3 PCS4 PCS5 PCS6 PCS7 SlopeType BS CS SetMode TO VIA		Current coordination system CS_MACHINE_COORDINATES Current slope type ST_BLOCK_SLOPE Current set mode SM_TO	
☐ Direct Abs ☐ Linear Abs ☐ Circular Abs ☐ Direct Rel ☐ Linear Rel ☐ Circular Rel ☐ Stop ☐ Interrupt ☐ Continue ☒ EnableCyclicScanning Retrigger OpMode		☐ InBuffer ☐ Stopping ☐ CmdDone ☐ ErrorStop ☒ InPosition ☐ GroupMoving ☐ BeltSynchron ☐ FullyBeltSynchron Current operating mode ModeCoordUngrouped	
Kin Overview		System Overview	

Abb. 6-7: Positionierübersicht Visualisierung

Eingegeben bzw. ausgewählt werden kann:

- Position
- Geschwindigkeit
- Beschleunigung
- Verzögerung
- Ruck
- Überschleifradius
- Zwischenposition
- Koordinatensystem
- Slope-Art der Bewegung
- Art des Überschleifens

Über die Schaltflächen **DirectAbs**, **LinearAbs**, etc. kann die entsprechende Positionierbetriebsart gewählt werden. Nähere Details sind der Betriebsartenbeschreibung zu entnehmen.

Im rechten Teil der Visualisierung wird die aktuelle Position in Weltkoordinaten, die parametrierten Randbedingungen, wichtige Statuselemente des Kinematik-Interfaces und der KinData-Schnittstelle sowie die aktuelle Betriebsart angezeigt.

Bandkonfiguration (ConveyorBelt) Visualisierung

Die Visualisierung **ConveyorBelt** dient der Konfiguration aller, der Kinematik zugehörigen Bänder.

Abb. 6-8: Bandkonfiguration Visualisierung

Durch Angabe der Bandkonfiguration und der Bandfehlerreaktion kann das entsprechende Band parametriert werden. Entsprechend der Anzahl der Bänder ist die Bandkonfigurationsübersicht zu vervielfachen. Dies erfolgt in gleicher Art und Weise, wie das Hinzufügen einer Kinematik zur Systemübersicht.

Siehe [Hinzufügen einer Kinematik zur Systemübersicht auf Seite 325](#).

Als referenzierte Visualisierung ist hierbei eine fortlaufende Bandnummer anzugeben. Mithilfe dieser Bandnummer wird intern auf die Kinematik-Interface-Struktur `arKinCtrl_gb[].Admin.ConveyorBelt` und `arKinStatus_gb[].Admin.ConveyorBelt` zugegriffen.



Die Konfiguration des Bandes erfolgt intern mit den Funktionsbausteinen `ML_SetBeltConfiguration` sowie `ML_SetBeltErrorReaction`. Das Auslesen der aktuellen Werte geschieht über die Funktionsbausteine `ML_RobotGetBeltConfiguration` und `ML_RobotGetBeltErrorReaction`. Nähere Informationen hierzu sind der entsprechenden Dokumentation zu entnehmen.

Kinematik-Interface

Tipp-/Einrichtbetrieb (Setup) Visualisierung

Die Visualisierung **Setup** bietet die Möglichkeit, die Kinematik zu vertippen.

Setup Clear error

Kin Nr KINEMATICS_1 **Warning** ☐ **Error** ☐

CoordSystem
CS_MACHINE_COORDINATES
MCS ACS BCS PCS1
PCS2 PCS3 PCS4 PCS5

JogMode
JM_STEP
Step Cont

Distance
[1] 1.00 [2] 1.00
[3] 1.00 [4] 1.00
[5] 1.00 [6] 1.00

Velocity
[1] 10.00 [2] 10.00
[3] 10.00 [4] 10.00
[5] 10.00 [6] 10.00

Acceleration
[1] 10.00 [2] 10.00
[3] 10.00 [4] 10.00
[5] 10.00 [6] 10.00

Deceleration
[1] 10.00 [2] 10.00
[3] 10.00 [4] 10.00
[5] 10.00 [6] 10.00

Jerk
[1] 10.00 [2] 10.00
[3] 10.00 [4] 10.00
[5] 10.00 [6] 10.00

1+ 2+ 3+ 4+ 5+ 6+
1- 2- 3- 4- 5- 6-
Enable ☐ EnableAck ☐

Kin Overview System Overview TeachIn

Abb. 6-9: Tipp-/Einrichtbetrieb Visualisierung

Im Abschnitt "CoordSystem" wird das Koordinatensystem definiert, in welchem vertippt werden soll. In gleicher Art und Weise wird unter "JogMode" festgelegt, ob ein Tipp-Befehl eine inkrementelle ("Step") oder kontinuierliche ("Cont") Verfahrensbewegung auslöst.

In den weiteren Abschnitten können für die ersten 6 Achsen der Kinematik die Distanz, Geschwindigkeit, Beschleunigung, Verzögerung und der Ruck festgelegt werden. Die hier eingegebenen Werte werden in Abhängigkeit des gewählten Koordinatensystems als absolute (MCS, BCS, PCS) bzw. prozentuale (ACS) Werte interpretiert. Im Falle der prozentualen Interpretation beziehen sich die Werte auf die entsprechenden Grenzwerte der Achse.

Mit den Schaltflächen 1+ bis 6- ist es möglich, die ersten 6 Achsen der Kinematik in positive und negative Richtung entsprechend der Einstellungen zu vertippen.



Bandachsen gehören nicht zur Achsgruppe und können daher nicht vertippt werden.

Einlernbetrieb (TeachIn) Visualisierung

Die Visualisierung **TeachIn** bietet die Möglichkeit Punkte zu teachen.

The interface is titled "TeachIn". It includes a "Clear error" button. Below the title, there is a "Kin Nr" dropdown menu currently showing "KINEMATICS_%d". To the right are "Warning" and "Error" status indicators, both currently inactive. The main area is split into two columns: "Command values" on the left and "Actual values" on the right. Each column has a "PointName" input field (placeholder %s) and a "Point" section with 16 input fields for coordinates (placeholder %2f for Command, %8.2f for Actual). Below these is a "CoordSystem" input field (placeholder %s) and a grid of buttons for coordinate systems: MCS, ACS, BCS, PCS1, PCS2, PCS3, PCS4, PCS5. Further down are "AllowOverwrite" (TRUE/FALSE) and "SaveBeltValues" (TRUE/FALSE) settings. At the bottom left is a "FileName" input field (placeholder %s), an "Option" input field (placeholder %s), and buttons for "New", "Append", and "Append Overwrite". At the bottom right is a "Control" section with "Point" operations (Write, Read, Delete, ReadNext) and "PointFile" operations (Save, Delete, Load, and a large "Execute" button). The bottom of the interface has three navigation buttons: "Kin Overview", "System Overview", and "Setup".

Abb. 6-10: Einlernbetrieb (TeachIn) Visualisierung

Im ersten Teil des Abschnitts "Command values" werden notwendige Eingaben für das Schreiben, Lesen und Löschen von Punkten definiert:

- PointName - Name des Punktes
- Point - 16 Koordinaten des Punktes
- CoordSystem - Koordinatensystem, in welchem der Punkt gültig ist
- AllowOverwrite - Überschreiben von Punkten zulassen/verweigern
- SaveBeltValues - Bandwerte im Punkt abspeichern/nicht abspeichern

Im zweiten Teil des Abschnitts "Command values" werden notwendige Eingaben für das Speichern, Laden und Löschen von Punktedateien definiert:

- FileName - Name der Punktedatei
- Option - Optionen zum Speichern und Laden der Punktedatei (nähere Informationen siehe E_KIN_TEACH_IN_POINT_FILE_OPTIONS)

Nach dem Lesen eines Punktes bzw. nächsten Punktes werden die entsprechenden Ausgaben im Abschnitt "Actual values" ausgegeben. Es wird der Name, die Koordinaten sowie das Koordinatensystem des Punktes bzw. des nächsten Punktes dargestellt.

Kinematik-Interface

Im Abschnitt "Control" kann über die Schaltflächen **Write**, **Read**, **ReadNext**, **Delete** (für Punkte) und **Save**, **Delete**, **Load** (für Punktedateien) das gewünschte Kommando gewählt und über die Schaltfläche **Execute** ausgelöst werden.

PCS-Konfiguration (ProductCoordSystem) Visualisierung

Die Visualisierung **ProductCoordSystem** dient der Konfiguration eines Produktkoordinatensystems.

Abb. 6-11: PCS-Konfiguration Visualisierung

Durch Angabe des Ursprungs (Origin) und der Art des Koordinatensystems (CoordType) kann das entsprechende Produktkoordinatensystem konfiguriert werden. Die Auswahl der Koordinatensystemnummer erfolgt hierbei über "PCS Nr.". Mithilfe dieser Koordinatensystemnummer wird intern auf die Kinematik-Interface-Struktur `arKinCtrl_gb[].Admin.PCSConfig` und `arKinStatus_gb[].Admin.PCSStatus` zugegriffen.



Die Konfiguration des Produktkoordinatensystems erfolgt intern mit den Funktionsbausteinen `MC_SetCartesianTransform` bzw. `ML_SetCylindricTransform`. Das Auslesen der aktuellen Konfiguration geschieht über den Funktionsbaustein `ML_RobotGetTrafoValues`.

6.2.4 Betriebsarten

Überblick

Es gibt zwei Methoden eine Betriebsart zu aktivieren:

1. Auswahl über Enum-Werte

Zuweisen eines Wertes vom TYPE MB_KINEMATICS_MODE:

```
arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordAH;
```

2. Benutzung des Bit-Zugriffs

Setzen eines Bits über die Bit-Zugriffsfunktionalität vom TYPE MB_KINEMATICS_MODE_BITS.

```
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_AH:= TRUE;
```



Es darf immer nur 1 Bit in `arKinCtrl_gb[].Admin._OpMode` gesetzt sein. Das Setzen mehrerer Bits verursacht einen Fehler im Kinematik-Interface.

Bevor eine Betriebsart aktiviert werden kann, müssen jedem Attribut zuerst Werte zugewiesen werden. Alle Attribute haben Standardwerte. Einige haben Werte ungleich Null, während andere als 0 definiert sind und, aufgrund der speziellen Anforderungen, ihnen ein Wert zugewiesen werden muss.



Nur die Attribute (z. B. Point, Velocity), die benutzt werden oder deren Standardwert geändert wurde, müssen definiert werden, bevor der aktuelle Betriebsartenwechsel ausgeführt wird.

Der folgende Abschnitt beschreibt die Betriebsarten, die vom Kinematik-Interface unterstützt werden und die Attribute, die zugewiesen werden können.



Das Kinematik-Interface nutzt intern das Achs-Interface, um einige Umschaltungen durchzuführen (z. B. Gruppieren/Degruppieren aller Achsen, Referenzieren von Achsen) .

Folgende Betriebsarten werden unterstützt:

- ModeCoordAB, [Kinematik Bereit, Antriebe Bereit auf Seite 336](#)
- ModeCoordAH, [Kinematik Bereit, Antriebe Halt auf Seite 336](#)
- ModeCoordHoming, [Kinematik Referenzieren auf Seite 337](#)
- ModeCoordPosDirAbs, [Absolutes PTP Positionieren auf Seite 337](#)
- ModeCoordPosDirRel, [Relatives PTP Positionieren auf Seite 338](#)
- ModeCoordPosLinAbs, [Absolutes Lineares Positionieren auf Seite 339](#)
- ModeCoordPosLinRel, [Relatives Lineares Positionieren auf Seite 340](#)
- ModeCoordPosCircAbs, [Absolutes Zirkulares Positionieren auf Seite 341](#)
- ModeCoordPosCircRel, [Relatives Zirkulares Positionieren auf Seite 342](#)
- ModeCoordPosJumpAbs, [Absolutes Jump Positionieren auf Seite 343](#)
- ModeCoordStopping, [Kinematik Stopp auf Seite 344](#)
- ModeCoordInterrupted, [Betriebsart Interrupted auf Seite 345](#)
- ModeCoordContinue, [Betriebsart Continue auf Seite 345](#)

Kinematik-Interface

- ModeCoordExternalFB, [Betriebsart Externer Funktionsbaustein auf Seite 346](#)
- ModeCoordCyclic, [Zyklische Positionssollwertvorgabe auf Seite 346](#)

Kinematik Bereit, Antriebe Bereit

Die Aktivierung dieser Betriebsart degruppert die Kinematik. Alle der Kinematik zugehörigen Achsen werden in AB (Antrieb Bereit) geschaltet. In dieser Betriebsart ist das Kinematik-Interface deaktiviert und die Kinematik befindet sich im PLCOpen-Zustand "StandStil incomplete".

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordAB;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_AB := TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	Admin.StopDeceleration	REAL	999	Ja
	Admin.StopJerk	REAL	9999	Ja
arKinStatus_gb[]	Admin.MODE_COORD_AB	BOOL		entfällt

Tab. 6-3: Attribute Kinematik Bereit, Antriebe Bereit



Bei der Auswahl der Betriebsart "ModeCoordAB" gibt das Kinematik-Interface in der Statusstruktur die Betriebsart "ModeCoordUngrouped" aus!

Kinematik Bereit, Antriebe Halt

Die Aktivierung dieser Betriebsart degruppert die Kinematik. Alle der Kinematik zugehörigen Achsen werden in AH (Antrieb Halt) geschaltet.

In dieser Betriebsart ist das Kinematik-Interface deaktiviert und die Kinematik befindet sich im PLCOpen-Zustand "StandStil incomplete". Die Achsen werden in den PLCOpen-Zustand "StandStil" überführt, d. h. sie sind mithilfe des Achs-Interfaces oder separater PLCOpen-Funktionsbausteine kommandierbar.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordAH;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_AH := TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	Admin.StopDeceleration	REAL	999.0	Ja
	Admin.StopJerk	REAL	9999.0	Ja
arKinStatus_gb[]	Admin.MODE_COORD_AH	BOOL		entfällt

Tab. 6-4: Attribute Kinematik Bereit, Antriebe Halt



Bei der Auswahl der Betriebsart "ModeCoordAH" gibt das Kinematik-Interface in der Status-Struktur die Betriebsart "ModeCoordUn-grouped" aus!

Kinematik Referenzieren

Die Aktivierung dieser Betriebsart führt das Referenzieren aller, der Kinematik zugehörigen Achsen, aus. Dieses Referenzieren erfolgt über den Setup-Mode des Achs-Interface.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordHoming;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_HOMING := TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	Admin.HomingSequence	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	MB_MAX_KIN_AXIS_NUMBER(0.0)	Ja
arKinStatus_gb[]	Admin.MODE_COORD_HOMING	BOOL		entfällt
	Admin.HomingAck	BOOL		entfällt

Tab. 6-5: Attribute Kinematik Referenzieren



Ist die Kinematik bei Auswahl der Betriebsart "ModeCoordHoming" bereits gruppiert, so ist ein erneutes Referenzieren nicht möglich und das Kinematik-Interface gibt eine Fehlermeldung aus.

Absolutes PTP Positionieren

Mit Aktivierung dieser Betriebsart wird eine absolute PTP-Bewegung auf eine vorher festgelegte Zielposition ausgeführt.

Die Kinematik wird gruppiert und geht in den PLCOpen-Zustand "GroupMoving" über.

Folgendes Kommando aktiviert die Betriebsart:

Kinematik-Interface

```

arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordPosDirAbs;
oder
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_POS_DIR_ABS :=
TRUE;

```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	Ja
	CoordMode.Velocity	REAL	10.0	Nein
	CoordMode.Acceleration	REAL	10.0	Nein
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100	Nein
	CoordCoordModeCtrl.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Nein
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	Nein
	CoordMode.SetMode	ML_SET_MODE	SM_TO	Nein
	CoordMode.BlendingRadius	REAL	0.0	Nein
arKinStatus_gb[]	Admin.MODE_COORD_POS_DIR_ABS	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-6: Attribute Absolutes PTP Positionieren

Relatives PTP Positionieren

Mit Aktivierung dieser Betriebsart wird eine relative PTP-Bewegung um eine vorher festgelegte Positionsdivergenz ausgeführt.

Die Kinematik wird gruppiert und geht in den PLCOpen-Zustand "GroupMoving" über.

Folgendes Kommando aktiviert die Betriebsart:

```

arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordPosDirRel;
oder
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_POS_DIR_REL :=
TRUE;

```


Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	Ja
	CoordMode.Velocity	REAL	10.0	Nein
	CoordMode.Acceleration	REAL	10.0	Nein
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100	Nein
	CoordCoordModeCtrl.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Nein
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	Nein
	CoordMode.SetMode	ML_SET_MODE	SM_TO	Nein
	CoordMode.BlendingRadius	REAL	0.0	Nein
arKinStatus_gb[]	Admin.MODE_COORD_POS_DIR_REL	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-7: Attribute Relatives PTP Positionieren

Absolutes Lineares Positionieren

Mit Aktivierung dieser Betriebsart wird eine absolute Linearbewegung auf eine vorher festgelegte Zielposition ausgeführt.

Die Kinematik wird gruppiert und geht in den PLCOpen-Zustand "GroupMoving" über.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosLinAbs;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_LIN_ABS :=  
TRUE;
```

Kinematik-Interface

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	Ja
	CoordMode.Velocity	REAL	10.0	Nein
	CoordMode.Acceleration	REAL	10.0	Nein
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100.0	Nein
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Nein
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	Nein
	CoordMode.SetMode	ML_SET_MODE	SM_TO	Nein
	CoordMode.BlendingRadius	REAL	0.0	Nein
arKinStatus_gb[]	Admin.MODE_COORD_POS_LIN_ABS	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-8: Attribute Absolutes Lineares Positionieren

Relatives Lineares Positionieren

Mit Aktivierung dieser Betriebsart wird eine relative Linearbewegung um eine vorher festgelegte Positionsdivergenz ausgeführt.

Die Kinematik wird gruppiert und geht in den PLCOpen-Zustand "GroupMoving" über.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosLinRel;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_LIN_REL := TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	Ja
	CoordMode.Velocity	REAL	10.0	Nein
	CoordMode.Acceleration	REAL	10.0	Nein
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100.0	Nein
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Nein
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	Nein
	CoordMode.SetMode	ML_SET_MODE	SM_TO	Nein
	CoordMode.BlendingRadius	REAL	0.0	Nein
arKinStatus_gb[]	Admin.MODE_COORD_POS_LIN_REL	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-9: Attribute Relatives Lineares Positionieren

Absolutes Zirkulares Positionieren

Mit Aktivierung dieser Betriebsart wird eine absolute Zirkularbewegung auf eine vorher festgelegte Zielposition und um einen ebenfalls festgelegten Zwischenpunkt ausgeführt.

Die Kinematik wird gruppiert und geht in den PLCOpen-Zustand "GroupMoving" über.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosCircAbs;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_CIRC_ABS := TRUE;
```

Kinematik-Interface

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.AuxPoint	ARRAY [1..16] OF REAL	16(0.0)	Nein
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	Ja
	CoordMode.Velocity	REAL	10.0	Nein
	CoordMode.Acceleration	REAL	10.0	Nein
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100.0	Nein
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Nein
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	Nein
	CoordMode.SetMode	ML_SET_MODE	SM_TO	Nein
	CoordMode.BlendingRadius	REAL	0.0	Nein
	CoordMode.CircularCircMode	ML_CIRCMODES	CM_BOARDER	Nein
arKinStatus_gb[]	Admin.MODE_COORD_POS_CIRC_ABS	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-10: Attribute Absolutes Zirkulares Positionieren

Relatives Zirkulares Positionieren

Mit Aktivierung dieser Betriebsart wird eine relative Zirkularbewegung um eine vorher festgelegte Positionsdivergenz und um eine ebenfalls festgelegte Zwischenpositionsdivergenz ausgeführt.

Die Kinematik wird gruppiert und geht in den PLCOpen-Zustand "GroupMoving" über.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosCircRel;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_CIRC_REL := TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.CircularAuxPoint	ARRAY [1..16] OF REAL	16(0.0)	Nein
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	Ja
	CoordMode.Velocity	REAL	10.0	Nein
	CoordMode.Acceleration	REAL	10.0	Nein
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100.0	Nein
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Nein
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	Nein
	CoordMode.SetMode	ML_SET_MODE	SM_TO	Nein
	CoordMode.BlendingRadius	REAL	0.0	Nein
	CoordMode.CircularCircMode	ML_CIRCMODES	CM_BOARDER	Nein
arKinStatus_gb[]	Admin.MODE_COORD_POS_CIRC_REL	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-11: Attribute Relatives Zirkulares Positionieren

Absolutes Jump Positionieren

Mit Aktivierung dieser Betriebsart wird eine absolute Jump-Bewegung auf eine vorher festgelegte Zielposition ausgeführt.

Die Kinematik wird gruppiert und geht in den PLCOpen-Zustand "GroupMoving" über.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordPosJumpAbs;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_POS_JUMP_ABS := TRUE;
```

Kinematik-Interface

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	Ja
	CoordMode.Velocity	REAL	10.0	Nein
	CoordMode.Acceleration	REAL	10.0	Nein
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100.0	Nein
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Nein
	CoordMode.SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	Nein
	CoordMode.SetMode	ML_SET_MODE	SM_TO	Nein
	CoordMode.BlendingRadius	REAL	0.0	Nein
	CoordMode.StartHeight	REAL	0.0	Nein
	CoordMode.MaxHeight	REAL	0.0	Nein
	CoordMode.EndHeight	REAL	0.0	Nein
arKinStatus_gb[]	Admin.MODE_COORD_POS_JUMP_ABS	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-12: Attribute Absolutes Jump Positionieren

Kinematik Stopp

Die Aktivierung dieser Betriebsart stoppt die Kinematik unter Aufrechterhaltung der Gruppierung.

Die Kinematik geht in den PLCOpen-Zustand "Stopping" über.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordStopping;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_STOPPING := TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	AdminCtrl.StopDeceleration	REAL	999.0	Ja
	AdminCtrl.StopJerk	REAL	9999.0	Ja
arKinStatus_gb[]	Admin.MODE_COORD_STOPPING	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-13: Attribute Kinematik Stopp

Betriebsart "Interrupted"

Die Aktivierung dieser Betriebsart unterbricht die Bewegung der Kinematik unter Aufrechterhaltung der Gruppierung.

Die Kinematik bleibt im PLCOpen-Zustand "GroupMoving".

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordInterrupted;
```

oder

```
arKinCtrl_gb[ ].Admin._OpMode.b.MODE_COORD_INTERRUPTED := TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100.0	Nein
arKinStatus_gb[]	Admin.MODE_COORD_INTERRUPTED	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-14: Attribute Betriebsart "Interrupted"

Betriebsart "Continue"

Die Aktivierung dieser Betriebsart startet eine zuvor unterbrochene Bewegung.

Die Kinematik befindet sich im PLCOpen-Zustand "GroupMoving".

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[ ].Admin._OpMode.en:= ModeCoordContinue;
```

oder

Kinematik-Interface

```
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_CONTINUE :=  
TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
arKinStatus_gb[]	Admin.MODE_COORD_CONTINUE	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-15: Attribute Betriebsart "Continue"

Betriebsart "Externer Funktionsbaustein"

Die Aktivierung dieser Betriebsart gruppiert die Kinematik und wartet auf einen externen Bewegungsfehl.

Die Kinematik geht in den PLCOpen-Zustand "StandStill Complete" über.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordExternalFB;
```

oder

```
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_EXTERNAL_FB :=  
TRUE;
```

Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch gescannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
arKinStatus_gb[]	Admin.MODE_COORD_EXTERNAL_FB	BOOL		entfällt

Tab. 6-16: Attribute Betriebsart "Externer Funktionsbaustein"

Zyklische Positionssollwertvorgabe

Mit Aktivierung dieser Betriebsart wird die zyklische Positionssollwertvorgabe aktiviert.

Die Kinematik wird gruppiert und geht in den PLCOpen-Zustand "GroupMoving" über.

Folgendes Kommando aktiviert die Betriebsart:

```
arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordCyclic;
```

oder

```
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_CYCLIC := TRUE;
```


Die folgende Tabelle enthält die Attribute, die von dieser Betriebsart unterstützt werden:

Element	Name	Typ	Standard	Zyklisch ge- scannt
arKinCtrl_gb[]	Admin.ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OPMODE	NoAction	Ja
	CoordMode.Point	ARRAY [1..16] OF REAL	16(0.0)	Ja
	CoordMode.Deceleration	REAL	10.0	Nein
	CoordMode.Jerk	REAL	100.0	Nein
	CoordMode.CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Nein
	CoordMode.SetMode	ML_SET_MODE	SM_TO	Nein
	CoordMode.BlendingRadius	REAL	0.0	Nein
	CoordMode.CyclicMode	ML_CYCLIC_MODE	CYCLIC_MODE_POS_ABS	Nein
	CoordMode.CyclicChannelType	ML_CYCLIC_CHANNEL_MODE_SUBTYPE	ML_CYCLIC_CHANNEL_MODE_LINEAR	Nein
	CoordMode.CyclicVelocity	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER MBER(0.0)]	Nein
arKinStatus_gb[]	Admin.MODE_COORD_POS_CYCLIC	BOOL		entfällt
	Admin.CmdDone	BOOL		entfällt
	Admin.InBuffer	BOOL		entfällt

Tab. 6-17: Attribute der zyklischen Positionssollwertvorgabe

6.2.5 Dialogbasierte Kinematikkommandierung

Zur komfortablen Kommandierung einer Kinematik steht für alle GAT-Programmiervorlagen von Bosch Rexroth die **dialogbasierte Kinematikkommandierung** zur Verfügung. Diese Kommandierung ermöglicht die Eingabe aller notwendigen Sollwerte der ausgewählten Kinematik in einem Dialog und die Erzeugung des entsprechenden SPS-Codes. Somit kann eine Kinematik ohne Kenntnisse über die genaue Syntax fehlerfrei und zeitsparend kommandiert werden.

Voraussetzungen

Vor der erstmaligen Verwendung der dialogbasierten Kinematikkommandierung müssen folgende Voraussetzungen gegeben sein:

- PlugIn "Kinematik-Interface" muss hinzugefügt sein
- Der Kinematik-Container muss hinzugefügt sein (nicht beim AchsInterface-Beispielprojekt)
- Alle Kinematiken, die der Unit/dem Modul angehören, müssen über den GAT-Wizard zugewiesen sein (nicht beim AchsInterface-Beispielprojekt)
- Alle Achsen der Kinematik(en), die der Unit/dem Modul zugehörig sind, müssen der entsprechenden Unit/dem entsprechenden Modul zugewiesen sein (nicht beim AchsInterface-Beispielprojekt)

Kinematik-Interface

Hinzufügen des Kinematik-Containers

Das Hinzufügen des Kinematik-Containers erfolgt über den GAT-Wizard. In Abhängigkeit der gewählten GAT-Variante kann hier über das Kontext-Menü der Unit bzw. des Moduls "Kinematik-Container hinzufügen" gewählt werden. Dies ist allerdings nur möglich, wenn bereits das PlugIn "Kinematik-Interface" hinzugefügt wurde.

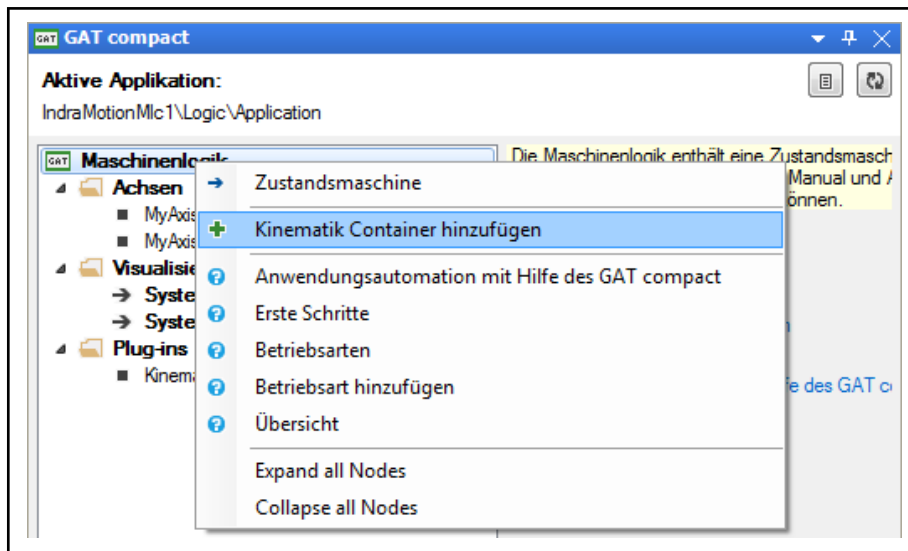


Abb. 6-12: Hinzufügen des Kinematik-Containers

Aufruf der ST-Code Vorlagen

Über das Kontextmenü "ST-Code Vorlagen ..." erhält man Zugriff auf die dialogbasierte Kommandierung. Anders als bei der Achskommandierung steht dieser Dialog im gesamten ST-Programmcode (ST - "Structured Text") zur Verfügung.

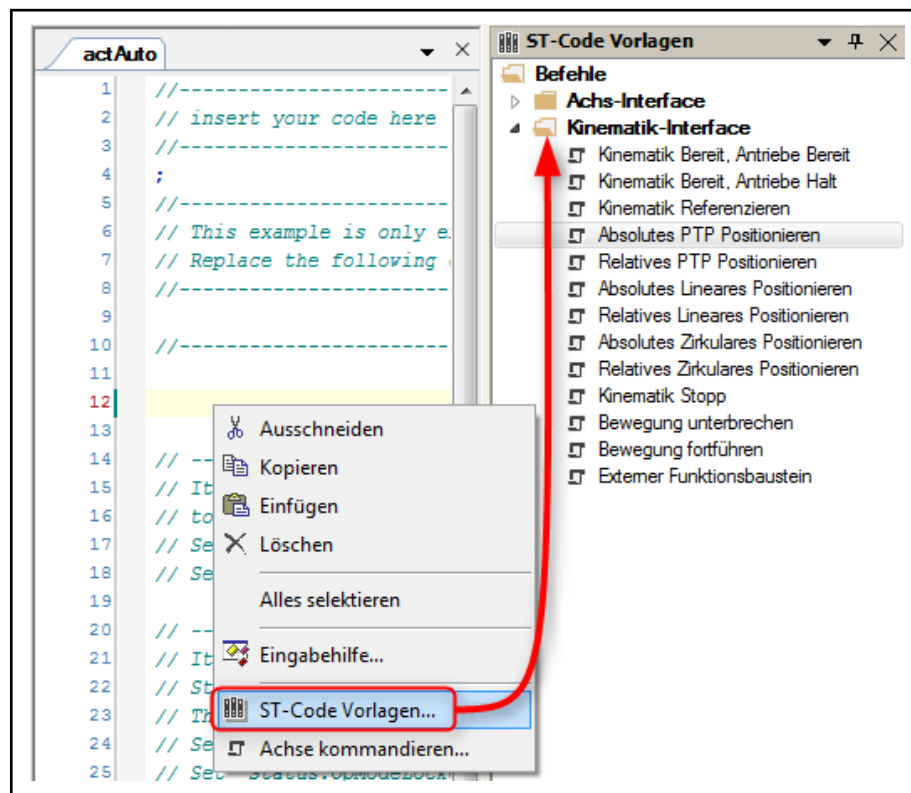


Abb. 6-13: Aufruf der dialogbasierten Kinematikkommandierung

Durch Doppelklick auf eine der Vorlagen öffnet sich ein weiteres Fenster, in welchem alle für die Betriebsart notwendigen Werte, vorgegeben werden können.

ST-Code erzeugen

Absolutes Lineares Positionieren
Geben Sie die erforderlichen Parameter ein.
Der Code wird an die aktuelle Cursorposition im Editor eingefügt.

Absolutes Lineares Positionieren

Eigenschaft	Wert	Standardwert
Group*	Kinematics2	
Point	arStartPoint	[0.0, 0.0, 0.0, ...]
Velocity	25	10.0
Acceleration	2000	100.0
Deceleration	2000	☐ 100.0
Jerk		0.0
CoordSystem		CS_MACHINE_COORDINATES
SlopeType		ST_BLOCK_SLOPE
SetMode		SM_TO
BlendingRadius		0.0
ChangeOpMode		NoAction

Kommentar:

☒ Quittierungscode erzeugen

Fertigstellen Abbrechen Hilfe

Abb. 6-14: Absolutes lineares Positionieren über die dialogbasierte Kinematikkommandierung

Für Eigenschaften, welche mit einem * markiert sind, ist eine Wertzuweisung erforderlich. Alle weiteren Eigenschaften bedürfen nicht zwingend einer Wertzuweisung. In diesem Fall wirken die Standardwerte und es erfolgt keine Codeerzeugung. Die Vorgabe von Werten kann durch direkte Eingabe oder auch durch SPS-Variablen erfolgen. Auch die Eingabehilfe (F2) kann genutzt werden. Das optionale Hinzufügen von Quittierungscode kann durch die Checkbox "Quittierungscode erzeugen" aktiviert oder deaktiviert werden.

In Abhängigkeit davon, an welcher Stelle der erzeugte SPS-Code eingefügt werden soll und welche GAT-Variante zum Einsatz kommt, werden die globalen Strukturen des Kinematik-Interfaces oder der Kinematik-Container (Kinematics) angesprochen. Dies wird automatisch durch die dialogbasierte Kinematikkommandierung erkannt.

Nach erfolgter Eingabe notwendiger Werte und Betätigen des Buttons **Fertigstellen** wird der SPS-Code an die aktuelle Cursor-Position im Programmcode eingefügt (im Folgenden gezeigt an der Unit-Aktion "actAuto" eines GAT Dezentral-Projektes).

Kinematik-Interface

```

12 Kinematics.Kinematics2.Ctrl.CoordMode.Point := arStartPoint;
13 Kinematics.Kinematics2.Ctrl.CoordMode.Velocity := 25;
14 Kinematics.Kinematics2.Ctrl.CoordMode.Acceleration := 2000;
15 Kinematics.Kinematics2.Ctrl.CoordMode.Deceleration := 2000;
16 Kinematics.Kinematics2.Ctrl.Admin._OpMode.en := ModeCoordPosLinAbs;
17 (Code example to check if the commanded operation mode is acknowledged
18 // Note: it can be necessary to wait one PLC cycle before checking the acknowledge
19 (e.g. when the previous OpMode was the same)
20 IF (Kinematics.Kinematics2.Ctrl.Admin._OpModeAck = ModeCoordPosLinAbs AND
21     Kinematics.Kinematics2.Ctrl.Admin.CmdDone (* or Admin.InBuffer *)) THEN
22     // commanded operation mode is acknowledged
23     ;
24 END_IF
25 *)

```

Abb. 6-15: Automatisch erstellter Programmcode der Vorlage "Absolutes Lineares Positionieren"

Dieser Programmcode entspricht der Syntax der Kinematikkommandierung im GAT-Umfeld in Verbindung mit dem Kinematik-Container. Der Quittierungscode wird auskommentiert eingefügt und kann durch den Anwender individuell angepasst werden.

Anwendererweiterung

Die dialogbasierte Kinematikkommandierung unterstützt auch durch den Anwender programmierte Erweiterungen des Kinematik-Interfaces. Diese Erweiterungen können, äquivalent zur dialogbasierten Achskommandierung, ebenfalls in die dialogbasierte Kinematikkommandierung eingebunden werden. Näher Informationen hierzu sind der GAT-Dokumentation zu entnehmen (siehe **Rexroth IndraLogic XLC IndraMotion MLC 14VRS Generic Application Template**, DOK-XLCMLC-TF*GAT**V14-APRS-DE-P, R911341486).

6.2.6 Kinematik-Interface Anwendererweiterung

Die `arKinCtrl_gb[]` und `arKinStatus_gb[]` Strukturen können durch den Anwender erweitert werden, um das Kinematik-Interface an spezielle Applikationen anzupassen.

Dieser Abschnitt zeigt, wie das Kinematik-Interface durch Hinzufügen einer neuen Funktionalität erweitert wird. Es wird davon ausgegangen, dass die im Kinematik-Interface bereits vorbereiteten Erweiterungsstrukturen verwendet werden. Es wird im Folgenden nur beschrieben, welche Änderungen in den POUs und Strukturen notwendig sind.

Hinweise zur Implementation von Anwendererweiterungen

Die Anwendererweiterungen wurden mit Hilfe der objektorientierten Erweiterungen von IndraLogic 2G implementiert.

Dabei sind einige Besonderheiten zu beachten:

- Der Funktionsbaustein **TE_KinematicsInterface** ist vom Basis-Funktionsbaustein "MB_KinInterfaceType01" abgeleitet. Über das Schlüsselwort "SUPER" kann der Basis-Funktionsbaustein bzw. dessen Methoden/Aktionen aufgerufen werden. Zum Beispiel wird an diversen Stellen über `SUPER^.mSetError(...)`; die Methode `mSetError` von `MB_KinInterfaceType01` aufgerufen, um Fehler in das Diagnosesystem einzutragen
- Innerhalb des Funktionsbaustein **TE_KinematicsInterface** kann auf die Daten von `arKinCtrl_gb` über die Eingänge `AdminCtrlExt`, `CoordModeCtrlExt`, usw., bzw. auf `arAxisStatus_gb` über `AdminStatusExt`, `DiagStatusExt`, usw. zugegriffen werden



Die Eingänge AdminCtrl (ohne *Ext) usw. gehören zum Basis-Funktionsbaustein und sollten nicht genutzt werden, da hier keine Anwendererweiterungen zur Verfügung stehen.

- Die Eingänge des Funktionsbaustein TE_KinematicsInterface sind als Referenzen definiert. In der Methode "mInitExtension" werden die Referenzen einmalig initialisiert und müssen dann beim zyklischen Aufruf des Funktionsbausteins nicht mehr übergeben werden

Erweiterung des MB_KinInterfaceType01

Erweitern der Kinematik-Interface-Struktur MB_KinInterfaceType01

Wenn die Strukturen TE_KINEMATICS_CONTROL_TYPE01 und TE_KINEMATICS_STATUS_TYPE01 mit zusätzlichen Unterstrukturen erweitert werden sollen, sind folgende zusätzliche Schritte zur nachfolgend beschriebenen Vorgehensweise notwendig:

1. Neue Elemente in TE_KINEMATICS_CONTROL_TYPE01 bzw. TE_KINEMATICS_STATUS_TYPE01 eintragen
2. Die zusätzlichen Eingänge am FB TE_KinematicsInterface als REFERENCE TO hinzufügen
3. In der Methode "mInitExtension" des FB TE_KinematicsInterface die Referenzen initialisieren



Wenn die Unterstrukturen TE_KINEMATICS_ADMINISTRATION, TE_KINEMATICS_COORDINATED, usw. erweitert werden sollen, ist nur Schritt 1 des zuvor beschriebenen Arbeitsablaufes für die jeweilige Struktur notwendig. In dieser Struktur ist die Erweiterung nach (`* add your extensions below here | Erweiterungen hier einfuegen*`) einzufügen. Danach steht der neu angelegte Ein- oder Ausgang über das entsprechende *Ext-Strukturelement zur Verfügung.

Erweitern des Funktionsbausteins TE_KinematicsInterface

Der letzte Schritt im Ablauf der Anwendererweiterung ist, den Funktionsbaustein TE_KinematicsInterface so zu erweitern, dass der Programmcode der Anwendererweiterung eingefügt werden kann.

1. Im Projektbaum, mit rechtem Mausklick **TE_KinematicsInterface** und **Hinzufügen ► Aktion** wählen.
2. Geben Sie der neuen Aktion einen zu Ihrer Erweiterung passenden Namen.
3. Fügen Sie Ihren Programmcode in die neue Aktion ein.
Nutzen Sie für den Zugriff auf die Strukturen `arKinCtrl_gb[]` und `arKinStatus_gb[]` die Eingänge mit dem Suffix `*Ext` (z. B. `AdminCtrlExt`, `DiagStatusExt`, usw).
4. Rufen Sie im Funktionsbaustein TE_KinematicsInterface die neue Aktion auf.
5. Übersetzen Sie das Projekt neu und überprüfen Sie es auf Programmierfehler.
6. Laden Sie das Projekt in die Steuerung.

Die neuen Eingangs- und Ausgangselemente sind nun ein Teil der Kinematik-Interface-Struktur und können über die UserVarGlobal-Variablen betrachtet werden.

Erweiterung des MB_KinInterfaceType11

Bestehendes Interface erweitern

Die folgenden Schritte zeigen, wie das Kinematik-Interface um die Funktionalität "Homing Sequence" erweitert werden kann. Die Erweiterung betrifft ein bereits bestehendes Interface und ist im Kinematik-Interface-Beispielprojekt und allen GAT-Projekten auf offener Codebasis enthalten. Zum Nachvollziehen der folgenden Schritte sollte parallel der Code des Beispielprojektes verglichen werden. Sinngemäß können die nachfolgenden Schritte auch für andere Erweiterungen genutzt werden.

- Das Interface ITF_KINEMATICS_ADMINISTRATION_EXT um die Property "HomingSequence" erweitern. Die "Get"-Teile und "Set"-Teile bleiben hierbei leer
- Unterhalb des Funktionsbausteines TE_KinematicsInterface einen neuen Ordner "HomeMode" anlegen
- Unterhalb des neuen Ordners "HomeMode" die o. g. Property "HomingSequence" hinzufügen und entsprechenden Programmcode implementieren
- Die zu implementierende Set Property prüft die Änderung des Eingangswertes und übernimmt den Wert auf einer lokalen Variablen
- Die zu implementierende Get Property kopiert den Wert der lokalen Variable in Richtung der Anwendung
- In der Deklaration des Funktionsbausteins TE_KinematicsInterface die notwendigen zusätzlichen lokalen Variablen hinzufügen
- Zusätzliche Aktion "actModeHome" implementieren
- Zusätzliche Aktion "actModeHome" im Funktionsbaustein TE_KinematicsInterface aufrufen

Neues Interface hinzufügen

Die nachfolgenden Schritte zeigen, wie das Kinematik-Interface um ein neues Interface erweitert wird.

- Neues Interface unterhalb des "Control"-Ordners hinzufügen
- Neue Properties unterhalb des neuen Interfaces hinzufügen
- Diese Properties auch unterhalb des FBs TE_KinematicsInterface hinzufügen
- In der Deklaration des FBs TE_KinematicsInterface die Implementierung des neuen Interfaces über das Schlüsselwort IMPLEMENTS <Interface-Name> eintragen
- Get und Set Property unterhalb des FBs TE_KinematicsInterface mit entsprechendem Programmcode implementieren
- Unterhalb des FBs TE_KinematicsInterface weitere Aktion hinzufügen
- In Datenstruktur TE_KINEMATICS_CONTROL_TYPE11 neues Interface hinzufügen
- In der Methode mInitExtension des FBs TE_KinematicsInterface das neue Interface initialisieren
- In der Deklaration des FBs TE_KinematicsInterface die notwendigen zusätzlichen lokalen Variablen hinzufügen
- Optional: Statusstruktur erweitern. Vorgehensweise wie bei den Erweiterungen des Type01 FBs

6.2.7 Umstellung auf ereignisgesteuertes Kinematik-Interface (Typ 11)

Das Kinematik-Interface Plugin enthält im Standard das Kinematik-Interface mit zyklischer Arbeitsweise (MB_KinInterfaceType01).

Zur Reduzierung der Laufzeit und besseren Performance steht das ereignisgesteuerte Kinematik-Interface zur Verfügung (Type11).

Bestehende SPS-Applikationen mit dem Kinematik-Interface Plugin (Type01) können über folgende Schritte auf das ereignisgesteuerte Kinematik-Interface (Type11) umgestellt werden.



Das ereignisgesteuerte Kinematik-Interface kann nur in Kombination mit dem ereignisgesteuerten Achs-Interface genutzt werden.

Für die Umstellung auf das ereignisgesteuerte Achs-Interface siehe Umstellung auf ereignisgesteuertes Achs-Interface (Dokumentation IndraLogic XLC IndraMotion MLC 14VRS Generic Application Template).

1. IndraWorks-Projektarchiv als Sicherheitskopie erstellen
2. Plug-in "Kinematik-Interface" im GAT-Wizard löschen (wenn nicht bereits durch Umstellung auf ereignisgesteuertes Achs-Interface gelöscht)
3. Plug-In "Kinematik-Interface (ereignisgesteuert)" mit Hilfe des GAT-Wizards hinzufügen
4. Projekt speichern, übersetzen und laden
5. Funktion überprüfen

6.3 Kinematik-Interface HMI Template

6.3.1 Einleitung

Um das Kinematik-Interface in Verbindung mit dem Visualisierungsgerät VH2110 nutzen zu können, wurde das GAT-HMI-Template um Kinematik-Funktionalitäten erweitert. Das Template beinhaltet grundlegende Funktionen wie den Status der Kinematik, das Vertippen und Positionieren von Kinematiken und deren Achsen sowie das Teach von Punkten. Es ist bereits für alle GAT-Varianten (Central, Decentral und Compact) einschaltfertig vorbereitet und kann bei Bedarf individuell erweitert und modifiziert werden.

Das Kinematik-Interface HMI Template basiert auf dem GAT-HMI-Template und erweitert dieses. Daher werden im folgenden nur kinematikspezifische Visualisierungsanteile und deren Funktion beschrieben. Nähere Informationen zum allgemeinen GAT-HMI-Template sind der Dokumentation GAT-HMI-Template zu entnehmen.



Das Kinematik-Interface-HMI-Template ist **nur** für das Visualisierungsgerät VH2110 in der Auflösung 1280 x 800 verfügbar!

Zur Bearbeitung und Ausführung der WinStudio-Visualisierung wird mindestens folgende Entwicklungsumgebungslicenz oder größer benötigt:

WinStudio-Entwicklungslizenz, max. 1.500 Variablen (Mat.-Nr. R911323596).

Voraussetzungen

Durch das Hinzufügen des Visualisierungsgerätes VH2110 und des GAT-HMI-Templates wird automatisch die Anteile des Kinematik-Interface HMI

Kinematik-Interface

Templates hinzugefügt. Um diese nutzen zu können müssen folgende Voraussetzungen erfüllt sein:

- Das Projekt muss auf einer der GAT-Varianten Central, Decentral oder Compact basieren
- Die Version des hinzugefügten Plug-ins "Kinematik-Interface" muss größer oder gleich 14.8.0.0 sein
- Der Funktionsbaustein "HMI_KinInterface" muss zyklisch aufgerufen werden (bspw. in einer zyklischen Aktion wie "actCyclic...")
- Am Aufruf des Funktionsbausteins "HMI_KinInterface" muss die IP-Adresse des Visualisierungsgerätes angegeben sein
- Die UDP-Kommunikation muss auf dem HMI-Gerät aktiviert und die IP-Adresse der Steuerung eingegeben sein (zwecks Auslesen des Potentiometers)

Beim Aufruf des Funktionsbausteins "HMI_KinInterface" ist zu beachten, dass die Initialisierung des Kinematik-Interfaces abgeschlossen ist (`TE_KinInterfaceType01Prog.InitDone = TRUE`), bevor der Funktionsbaustein aktiviert wird.

Kommunikation zwischen HMI und SPS

Die Kommunikation zwischen der HMI und der Steuerung erfolgt über die Variablen der globale Variablenliste "Global_HMI_KinInterface".

Variable	Beschreibung
stKinHMICtrl_gb	Struktur zur Ansteuerung des Kinematik-Interfaces
stKinHMIStatus_gb	Struktur zur Ausgabe des Status des Kinematik-Interface
uiKinHMINumber_gb	aktuell ausgewählte Kinematiknummer
arKinHMIActive_gb	Array aller aktiven Kinematiken
iKinHMIMaxNumber	höchste, auszuwählende Kinematiknummer

Tab. 6-18: Aufbau von "Global_HMI_KinInterface"

Die Ein- und Ausgabeelemente des HMI Templates sind nicht direkt mit den Strukturen des Kinematik-Interfaces verbunden. Hierfür dienen die beiden Strukturen "stKinHMICtrl_gb" und "stKinHMIStatus_gb", welche mittels Referenzierung mit den Strukturen des Kinematik-Interfaces ("arKinCtrl_gb" und "arKinStatus_gb") verbunden sind. Wird über die HMI eine andere Kinematik ausgewählt, so werden die HMI-Strukturen neu auf die entsprechende Kinematik-Interface-Strukturen umreferenziert. Dies ermöglicht es eine Kinematik sowohl über die SPS als auch über die HMI gleichzeitig zu bedienen.

Welche Kinematik gerade bedient werden soll steuert die Variable "uiKinHMI-Number_gb". Diese Variable stellt für die Referenzierung der HMI-Strukturen den Index für die Kinematik-Interface-Strukturen dar. Somit werden bspw. bei der Wahl der zweiten Kinematik (`uiKinHMINumber_gb = 2`) die zweiten Elemente von "arKinCtrl_gb" und "arKinStatus_gb" mit "arKinHMICtrl_gb" und "arKinHMIStatus_gb" referenziert. Welchen Wert die Variable besitzt, d.h. welche Kinematik somit momentan ausgewählt ist, wird direkt durch die Umschaltung über die HMI bestimmt. Allerdings sind nur die Kinematiken bzw. deren Nummern auswählbar, welche auch aktiv sind.

Zur Identifizierung der aktiven Kinematiken dient das Array "arKinHMIActive_gb". Die Elemente dieses Arrays besitzen bei existierender Kinematik(nummer) der Wert `True`, anderenfalls `False`. Die maximale Größe des Arrays entspricht dem Wert der Variable "iKunHMIMaxNumer", welche die höchste auszuwählende Kinematiknummer darstellt.

6.3.2 KinInterface-Masken

Kinematikstatus

Das Untermenü "Kinematics Status" im Bedienbereich "Production" stellt den Status der aktuell ausgewählten Kinematik dar. Zu sehen ist hierbei

- der Kinematikname
- die Position des TCPs im gewählten Koordinatensystem
- die Geschwindigkeit des TCPs
- die Verschiebung des gewählten Koordinatensystems zum Maschinenkoordinatensystem (MCS)
- der Typ des gewählten Koordinatensystems
- der Zustand der Kinematik
- der aktuelle Override.

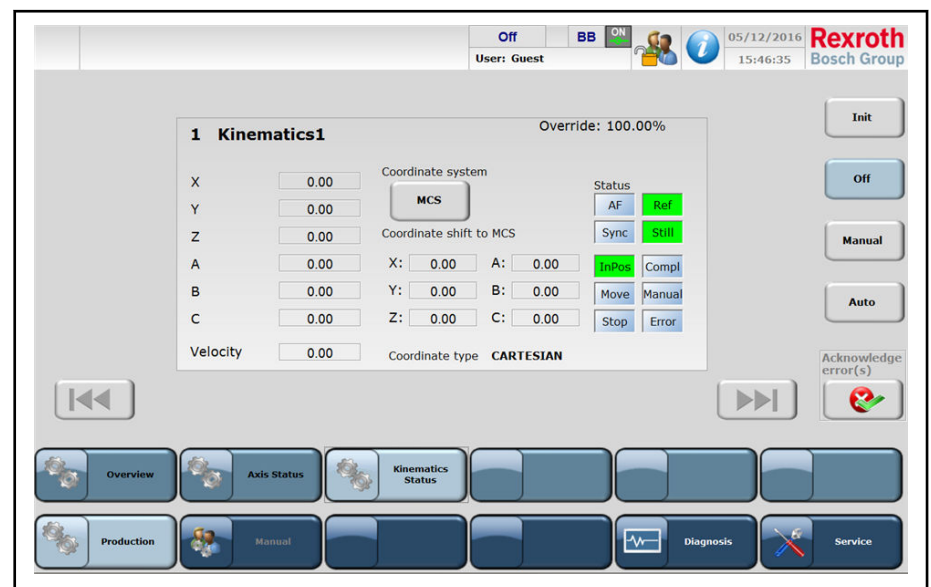


Abb. 6-16: Kinematik-Interface HMI Template - Kinematics Status

Über den Button "Coordinate system", welches das aktuell gewählte Koordinatensystem anzeigt, kann die Wahl des Koordinatensystems geändert werden. Ist ein kartesisches Koordinatensystem ausgewählt (MCS, BCS, PCSx), so wird die Position des TCPs mit "X Y Z A B C" dargestellt. Ist hingegen das Achskoordinatensystem (ACS) gewählt, so wird die Position des TCPs entsprechend der Achsnamen beschriftet und die Anzahl der dargestellten Positionswerte reduziert sich auf die Anzahl der Achsen.

Sind im Projekt weitere Kinematiken konfiguriert, so werden Weiterschaltbuttons sichtbar sein. Hierüber kann zur nächsten Kinematik umgeschaltet werden. Nicht vorhandene Kinematiknummern werden hierbei übersprungen. So ist es möglich, dass die Kinematiknummer von 3 auf 5 springt, insofern zwei Kinematiken mit den Nummern 3 und 5 konfiguriert sind. Ist die erste oder letzte Kinematik erreicht, so wird der entsprechende Weiterschaltbutton unsichtbar.



Der aktuelle eingestellt Wert des Overrides wird beim Umschalten der Kinematik für die folgende Kinematik ebenfalls übernommen!

Kinematik-Interface

Screen-Zuordnung	Name im Template
Header	Header
MainScreen	Production_KinStatus
SideBar	Production_SideBar
Navigation oben	Production_Navigation
Navigation unten	Main_Navigation

Tab. 6-19: Verwendete Template-Screens

Kinematik vertippen

Das Untermenü "Setup Kinematics" im Bedienbereich "Manual" enthält alle Bedienelemente zum manuellen Vertippen der Kinematik und deren Achsen. Der Zugriff auf dieses Menü ist nur in der Betriebsart "Manual" möglich.

Das Hauptbild dieses Untermenüs teilt sich in zwei Bereiche: Der obere Bereich entspricht zu 100% dem Bild "Kinematics Status" aus dem Bedienbereich "Production". Über den unteren Bereich ist es möglich die Tipp-Konfiguration zu verändern, den Tipp-Modus zu aktivieren sowie die Kinematikachsen zu referenzieren.

Desweiteren befinden sich am linken und rechten Rand des Hauptbildes die beiden Sidebars, dessen Button in Abhängigkeit des gewählten Koordinatensystems mit "X Y Z A B C" bzw. der Achsenamen der Kinematikachsen bezeichnet sind. Hierüber ist es möglich die Kinematik zu vertippen, insofern der Tipp-Modus aktiviert und die Kinematik referenziert wurde.

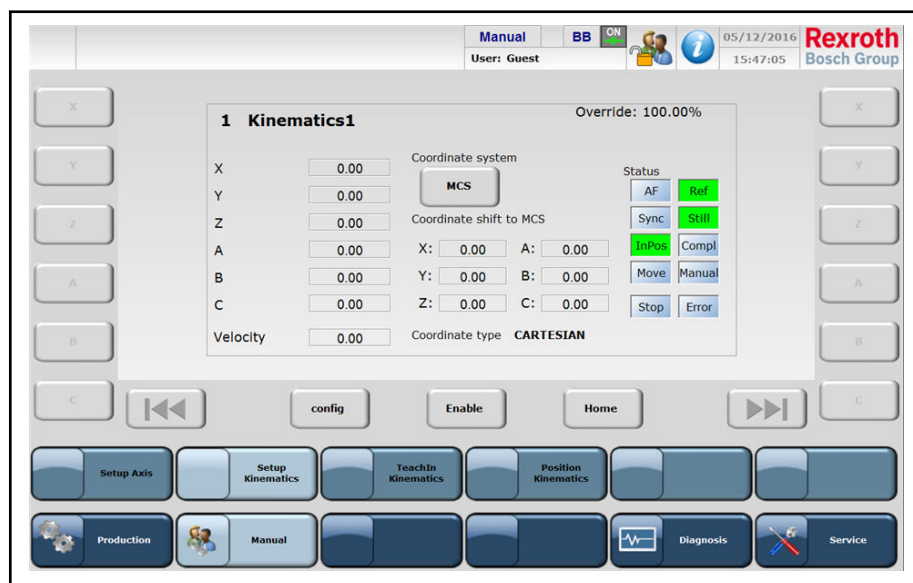


Abb. 6-17: Kinematik-Interface HMI Template - Setup Kinematics

Eingaben, welche über die Buttons **config** und **Enable** getätigt werden, wirken sich direkt auf die Kinematik-Interface-Struktur `arKinCtrl_gb[index].SetupMode` der entsprechend gewählten Kinematik aus. Desweiteren aktiviert der Button **Home** die Betriebsart "ModeCoordHoming" und alle Kinematikachsen werden nacheinander referenziert. Letzteres ist nur bei deaktiviertem Tipp-Modus möglich!



Der aktuelle eingestellt Wert des Overrides wird beim Umschalten der Kinematik für die folgende Kinematik ebenfalls übernommen!

Screen-Zuordnung	Name im Template
Header	Header
MainScreen	Manual_Kinematics_Main Manual_Kin_Jog_SideBar_Bottom
SideBar	Manual_Kin_Jog_SideBar_Left Manual_Kin_Jog_SideBar_Right
Navigation oben	Manual_Navigation
Navigation unten	Main_Navigation

Tab. 6-20: Verwendete Template-Screens

Punkte teachen

Das Untermenü "TeachIn Kinematics" im Bedienbereich "Manual" enthält alle Bedienelemente zum Schreiben, Lesen und Löschen von Punkten sowie Speichern, Laden und Löschen von Punktedateien. Der Zugriff auf dieses Menü ist nur in der Betriebsart "Manual" möglich.

Das Hauptbild dieses Untermenüs teilt sich in zwei Bereiche: Der obere Bereich entspricht zu 100% dem Bild "Kinematics Status" aus dem Bedienbereich "Production". Über den unteren Bereich ist es möglich die TeachIn-Konfiguration zu verändern sowie einen geteachten Punkt anzufahren.

Desweiteren befinden sich am linken und rechten Rand des Hauptbildes die beiden Sidebars, über welche die entsprechende Aktion ausgelöst werden kann. Die linke Sidebar beinhaltet hierbei das Schreiben, Lesen und Löschen von Punkten. Über die rechte Sidebar hingegen können Punktedateien gespeichert, geladen oder gelöscht werden.

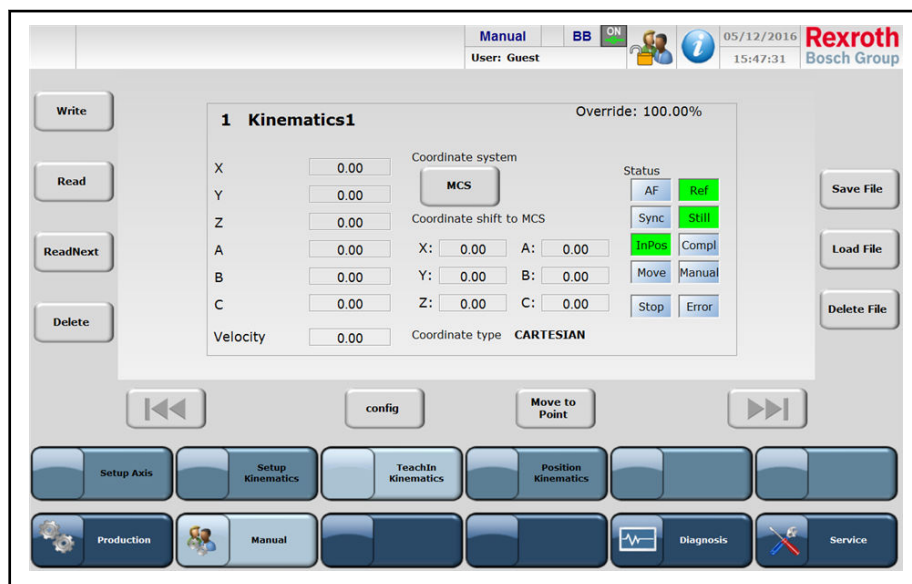


Abb. 6-18: Kinematik-Interface HMI Template - TeachIn Kinematics

Über den Button **config** gelangt man in die TeachIn-Konfiguration, welche in Abhängigkeit der gewählten Aktion zweierlei Funktion besitzt. Soll ein Punkt geschrieben/gelöscht oder eine Punktedatei gespeichert/geladen/gelöscht werden, so kann hier der Punkt, Punktname, Dateiname, das Koordinatensystem sowie die Punkte- und Dateioptionen angepasst werden. Die Eingabe des Punktes ist entweder manuell oder über den Button **Get current position** möglich.

Kinematik-Interface

Will man hingegen einen (nächsten) Punkt lesen so dient dieses Menü als Darstellung des Punktes. Durch Eingabe des Punktnamens und Betätigung des Buttons **Read** oder **ReadNext** wird der Punkt, Punktnamen sowie das Koordinatensystem mit den eingelesenen Werten aktualisiert.

Desweiteren können in der TeachIn-Konfiguration die "Move to Point"-Parameter angepasst werden. Diese werden - zusammen mit dem Punkt - durch Betätigung des Buttons **Move to Point** auf die Kinematik-Interface-Struktur `arKinCtrl_gb[index].CoordMode` geschrieben und die Kinematik wird in die Betriebsart "CoordModePosDirAbs" versetzt. Der Punkt wird somit mittels PTP-Befehl angefahren.



Der aktuelle eingestellt Wert des Overrides wird beim Umschalten der Kinematik für die folgende Kinematik ebenfalls übernommen!

Screen-Zuordnung	Name im Template
Header	Header
MainScreen	Manual_Kinematics_Main Manual_Kin_TeachIn_SideBar_Bottom
SideBar	Manual_Kin_TeachIn_SideBar_Left Manual_Kin_TeachIn_SideBar_Right
Navigation oben	Manual_Navigation
Navigation unten	Main_Navigation

Tab. 6-21: Verwendete Template-Screens

Kinematik positionieren

Das Untermenü "Position Kinematics" im Bedienbereich "Manual" enthält alle Bedienelemente zum Verfahren der Kinematik und deren Achsen. Der Zugriff auf dieses Menü ist nur in der Betriebsart "Manual" möglich.

Das Hauptbild dieses Untermenüs teilt sich in zwei Bereiche: Der obere Bereich entspricht zu 100% dem Bild "Kinematics Status" aus dem Bedienbereich "Production". Über den unteren Bereich ist es möglich die Positionierkonfiguration zu verändern.

Desweiteren befinden sich am linken und rechten Rand des Hauptbildes die beiden Sidebars, über welche die entsprechende Aktion ausgelöst werden kann. Die linke SideBar beinhaltet hierbei das Starten und Stoppen von Bewegungsvorgängen sowie die Abschaltung der Kinematik. Über die rechte SideBar hingegen können Bewegungen unterbrochen oder fortgesetzt werden.

Kinematik-Interface

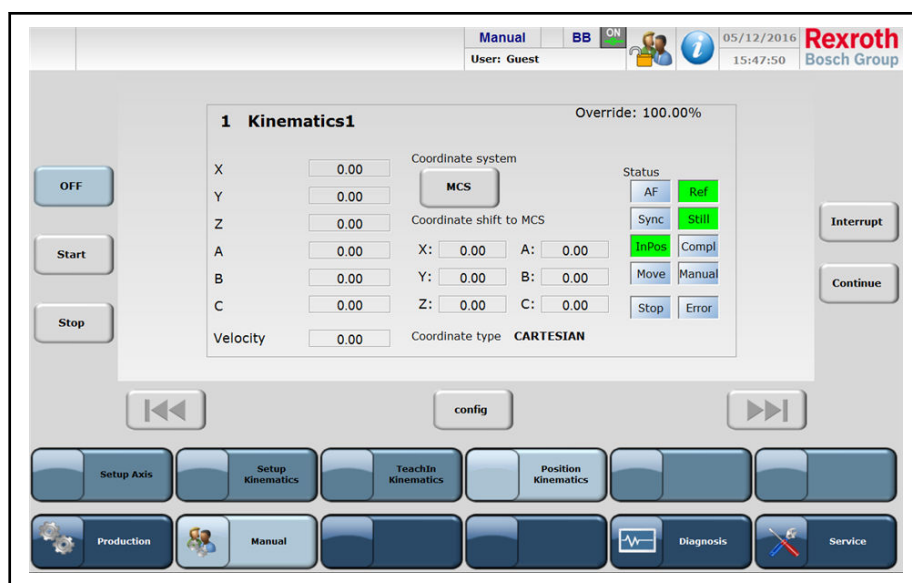


Abb. 6-19: Kinematik-Interface HMI Template - Position Kinematics

Eingaben, welche über den Button **config** getätigt werden, wirken sich direkt auf die Kinematik-Interface-Struktur `arKinCtrl_gb[index].CoordMode` der entsprechend gewählten Kinematik aus. Durch Betätigung des Buttons **Start** wird eine Bewegung mit diesen Werten ausgeführt. Mehrmaliges Betätigen mit gleichem Interpolationstyp bewirkt ein Retriggern des Kinematik-Interfaces.

Bereits gestartete Bewegungen können mittels der Buttons **Stop**, **Interrupt** und **Continue** gestoppt, unterbrochen bzw. fortgeführt werden. Die Kinematik wird hierbei in die entsprechende Betriebsart **ModeCoordStopping**, **ModeCoordInterrupted** bzw. **ModeCoordContinue** versetzt.



Der aktuelle eingestellt Wert des Overrides wird beim Umschalten der Kinematik für die folgende Kinematik ebenfalls übernommen!

Screen-Zuordnung	Name im Template
Header	Header
MainScreen	Manual_Kinematics_Main Manual_Kin_Jog_SideBar_Bottom
SideBar	Manual_Kin_Jog_SideBar_Left Manual_Kin_Jog_SideBar_Right
Navigation oben	Manual_Navigation
Navigation unten	Main_Navigation

Tab. 6-22: Verwendete Template-Screens

7 ML_KinTech.library

7.1 Einführung und Übersicht



Die Bibliothek benötigt das ROB (Robotik) Funktionspaket der Steuerung.

Programmorganisationseinheiten von ML_KinTech

Anwendungsgebiet (Ordner in der Bibliothek)	
POUs	Beschreibung
Globale Konstanten (KinInterface/Const)	
Global_MB_KinInterfaceVars, Seite 361	Beinhaltet Informationen über die Achsen, welche der Kinematik zugeordnet sind Beinhaltet Informationen über die Robot-Control-Version der Kinematik Ermöglicht den parallelen Zugriff auf das Achs-Interface sowohl aus dem Kinematik-Interface als auch von einer Anwender-Applikation
KinInterface/FB	
ML_KinematicsInitType01, Seite 363	Wird zur Initialisierung des Kinematik-Interfaces für jede Kinematik benutzt. Der Funktionsbaustein muss nur einmal beim Programmstart bzw. bei jeder Phasenumschaltung vom Parametriermodus in den Betriebsmodus aufgerufen werden
ML_KinematicsInterfaceType01, Seite 366	Wird zur Konfiguration des Kinematik-Interfaces für jede Kinematik benutzt. Der Funktionsbaustein muss zyklisch (im Motion-Takt oder langsamer als der Motion-Takt) aufgerufen werden, solange man sich im Betriebsmodus befindet
ML_KinematicsInterfaceType11, Seite 369	Wird zur Konfiguration des Kinematik-Interfaces für jede Kinematik benutzt. Der Funktionsumfang entspricht dem ML_KinematicsInterfaceType01 , wobei dieser Funktionsbaustein ereignisgesteuert und damit performanceoptimiert arbeitet. Der Funktionsbaustein muss zyklisch (im Motion-Takt oder langsamer als der Motion-Takt) aufgerufen werden, solange man sich im Betriebsmodus befindet
	Kinematik-Interface-Betriebsarten, Übersicht, Seite 373 Ordner Kin_Interface/Type/Status, Übersicht, Seite 388 Ordner Kin_Interface/Type/Control, Übersicht, Seite 376
KinJog/FB	
ML_KinJog, Seite 394	Dient zum Vertippen einzelner Achsen oder Achsgruppen einer Kinematik mit der Robot-Control V2 FB verwendet ML_JOGMODE, Seite 397

Tab. 7-1: Bibliothek ML_KinTech

7.2 KinInterface/Const

7.2.1 Global_MB_KinInterfaceVars

Die Bibliothek ML_KinTech enthält die globale Variablenliste **Global_MB_KinInterfaceVars**.

ML_KinTech.library

VAR_GLOBAL Global_MB_KinInterfaceVars

Name	Datentyp	Kommentar
arK8ConfigList_gb	ARRAY [1..MB_MAX_KIN_NUMBER] OF Kin_Axis_Config_Elements	Achskonfigurationsliste, Parameter K-0-0008
arpAdminAxisCtrl_gb	ARRAY [1..MB_MAX_AXIS_NUMBER] OF POINTER TO MB_AXIS_ADMINISTRATION	Zeiger auf die Achs-Interface-Struktur arAxisCtrl_gb[].Admin
arpCoordModeAxisCtrl_gb	ARRAY [1..MB_MAX_AXIS_NUMBER] OF POINTER TO MB_AXIS_COORDINATED	Zeiger auf die Achs-Interface-Struktur arAxisCtrl_gb[].CoordMode
arpAdminAxisStatus_gb	ARRAY [1..MB_MAX_AXIS_NUMBER] OF POINTER TO MB_AXIS_ADMIN_STATUS	Zeiger auf die Achs-Interface-Struktur arAxisStatus_gb[].Admin
arpDiagAxisStatus_gb	ARRAY [1..MB_MAX_AXIS_NUMBER] OF POINTER TO MB_AXIS_DIAGNOSIS	Zeiger auf die Achs-Interface-Struktur arAxisStatus_gb[].Diag

Die Bibliothek definiert folgende Konstanten:

1. **arK8ConfigList_gb** ([Kin_Axis_Config_Elements](#), Seite 372):

Diese Array-Konstante beinhaltet Informationen über die Achsen, welche der Kinematik zugeordnet sind (K-0-0008):

- **uiNumberOfKinAxis**: Gesamtanzahl der Achsen
- **uiNumberOfMoveAxis**: Anzahl der Positionier- und Orientierungsachsen
- **uiNumberOfBeltAxis**: Anzahl der Bandachsen
- **arAxesConfig**: Achskonfiguration (Achstyp, Steuerungsnummer, Achsnummer)

2. **arpAxisCtrlAdmin_gb**:

Diese Pointer-Konstante zeigt auf die AdminCtrl-Struktur des Achs-Interfaces (MB_AXIS_ADMINISTRATION). Sie ermöglicht den parallelen Zugriff auf das Achs-Interface sowohl aus dem Kinematik-Interface als auch von einer Anwender-Applikation.

3. **arpAxisCtrlCoordMode_gb**:

Diese Pointer-Konstante zeigt auf die CoordModeCtrl-Struktur des Achs-Interfaces (MB_AXIS_COORDINATED). Sie ermöglicht den parallelen Zugriff auf das Achs-Interface sowohl aus dem Kinematik-Interface als auch von einer Anwender-Applikation.

4. **arpAxisStatusAdmin_gb**:

Diese Pointer-Konstante zeigt auf die AdminStatus-Struktur des Achs-Interfaces (MB_AXIS_ADMIN_STATUS). Sie ermöglicht den parallelen Zugriff auf das Achs-Interface sowohl aus dem Kinematik-Interface als auch von einer Anwender-Applikation.

5. **arpAxisStatusDiag_gb**:

Diese Pointer-Konstante zeigt auf die DiagStatus-Struktur des Achs-Interfaces (MB_AXIS_DIAGNOSIS). Sie ermöglicht den parallelen Zugriff auf das Achs-Interface sowohl aus dem Kinematik-Interface als auch von einer Anwender-Applikation.

7.3 KinInterface/FB

7.3.1 ML_KinematicsInitType01

Kurzbeschreibung Der Funktionsbaustein **ML_KinematicsInitType01** wird zur Initialisierung des Kinematik-Interfaces für jede Kinematik benutzt.

Der Funktionsbaustein muss nur einmal beim Programmstart bzw. bei jeder Phasenumschaltung vom Parametrier- in den Betriebsmodus aufgerufen werden.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/FB

Tab. 7-2: Referenztable ML_KinematicsInitType01

Schnittstellenbeschreibung

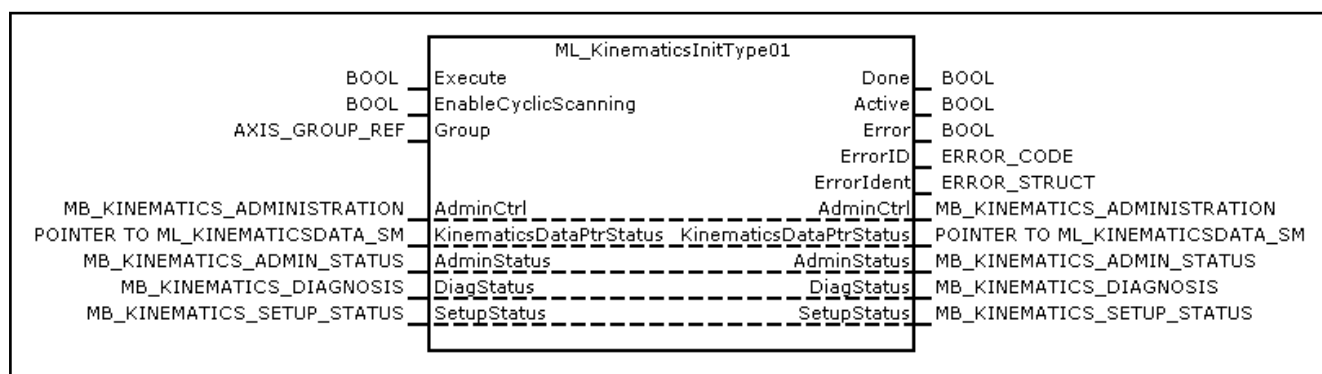


Abb. 7-1: Funktionsbaustein ML_KinematicsInitType01

I/O-Typ	Name	Datentyp	Kommentar
VAR_INPUT	Execute	BOOL	Start der Initialisierung durch positive Flanke
	EnableCyclicScanning	BOOL	Zyklische Überprüfung und Übertragung der geänderten Eingänge für die aktuelle Betriebsart
	Group	AXIS_GROUP_REF	Zu initialisierende Kinematik im Kinematik-Interface
VAR_OUTPUT	Done	BOOL	Wird gesetzt, wenn der FB die Bearbeitung beendet hat
	Active	BOOL	Wird gesetzt, wenn der FB aktiv ist (nicht im Leerlaufbetrieb)
	Error	BOOL	Zeigt an, dass ein Fehler in der FB-Instanz aufgetreten ist
	ErrorID	ERROR_CODE	Kurzer Hinweis zur Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detaillierte Information zum Fehler Fehlerbehandlung, Seite 366

ML_KinTech.library

I/O-Typ	Name	Datentyp	Kommentar
VAR_IN_OUT	AdminCtrl	MB_KINEMATICS_ADMINISTRATION	Verwaltung der Achse arKinCtrl_gb[].Admin MB_KINEMATICS_ADMINISTRATION, Seite 377
	KinematicsDataPtrStatus	POINTER TO ML_KINEMATICS_DATA_SM	Zeiger auf KinData [] dieser Kinematik
	AdminStatus	MB_KINEMATICS_ADMIN_STATUS	Status Verwaltung der Kinematik arKinStatus_gb[].Admin MB_KINEMATICS_ADMIN_STATUS, Seite 389
	DiagStatus	MB_KINEMATICS_DIAGNOSIS	Diagnoseinformationen der Kinematik arKinStatus_gb[].Diag MB_KINEMATICS_DIAGNOSIS, Seite 393
	SetupStatus	MB_KINEMATICS_SETUP_STATUS	Status Tipp-/Einrichtfunktion der Kinematik arKinStatus_gb[].SetupMode MB_KINEMATICS_SETUP_STATUS, Seite 394

Tab. 7-3: Schnittstellenvariablen ML_KinematicsInitType01



Es ist nicht möglich anstelle der einzelnen Strukturelemente "AdminCtrl", "KinematicsDataPtrStatus", etc. die kompletten Strukturen "MB_Kinematics_Control_Type01" und "MB_Kinematics_Status_Type01" dem Funktionsbaustein zu übergeben.

Somit könnten in der Anwender-Applikation keine speziellen Erweiterungen zum bestehenden Code hinzugefügt werden.

Deshalb werden die benötigten Elemente von [MB_KINEMATICS_CONTROL_TYPE01, Seite 377](#), und [MB_KINEMATICS_STATUS_TYPE01, Seite 388](#), als separate Eingänge übergeben.

Nähere Details siehe "Kinematik-Interface Anwender-Erweiterung".

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Generic Application Template, DOK-XLCMLC-TF*GAT**V14-APRS-DE-P, R911341486

oder

Rexroth IndraMotion MLC 14VRS Robot-Control V2, DOK-MLC***-ROCO****V14-RERS-DE-P, R911341586

Funktionsbeschreibung

Der Funktionsbaustein ML_KinematicsInitType01 initialisiert die folgenden Strukturelemente mit Standardwerten:

Administration Control	
AdminCtrl._OpMode.en	ModeCoordAB
AdminCtrl.ClearError	FALSE
AdminCtrl.EnableCyclicScanning	EnableCyclicScanning
Administration Status	
AdminStatus._OpModeAck	ModeCoordUngrouped

AdminStatus.Active	FALSE
AdminStatus.Name	''
AdminStatus.TrafoNb	999
AdminStatus.CmdDone	FALSE
AdminStatus.InBuffer	FALSE
AdminStatus.ParamLevelAck	FALSE
AdminStatus.ConveyorBelt[].BeginLength	0.0
AdminStatus.ConveyorBelt[].TotalLength	0.0
AdminStatus.ConveyorBelt[].ConveyorBeltOrigin[]	0.0
AdminStatus.ConveyorBelt[].CoordType	CS_TYPE_CARTESIAN
AdminStatus.ConveyorBelt[].ErrorReactionType	ML_BELT_ERROR_DESYNC_WITH_LIMITS
AdminStatus.ConveyorBelt[].ErrorReactionDesyncTime	T#300MS
AdminStatus.ConveyorBelt[].ErrorReactionDeceleration	1000.0
AdminStatus.ConveyorBelt[].ErrorReactionJerk	0.0
AdminStatus.ConveyorBelt[].Done	FALSE
AdminStatus.PCSStatus[].Origin[]	0.0
AdminStatus.PCSStatus[].CoordType	CS_TYPE_CARTESIAN
AdminStatus.PCSStatus[].Done	FALSE
AdminStatus.KinTransformPLCDone	FALSE
Diagnosis	
DiagStatus.Error	FALSE
DiagStatus.ErrorID	NONE_ERROR
DiagStatus.ErrorIdent.Table	NO_TABLE_USED
DiagStatus.ErrorIdent.Additional1	0
DiagStatus.ErrorIdent.Additional2	0
DiagStatus.Number	0
DiagStatus.Message	'reading not activated'
Setup Status	
SetupStatus.EnableAck	FALSE
KinematicsDataPtrStatus	POINTER TO KinData[] der Kinematik

Tab. 7-4: Strukturelemente initialisiert durch ML_KinematicsInitType01

ML_KinTech.library

Der Funktionsbaustein **ML_KinematicsInitType01** liest folgende Parameter aus und schreibt die Werte auf die folgenden Strukturelemente:

K-Parameter (Name)	Strukturelement
K-0-0031 (Transformationsschema)	AdminStatus.TrafoNb
K-0-0002 (Kinematikbezeichnung)	AdminStatus.Name
K-0-0008 (Achskonfigurationsliste)	nur für interne Zwecke benötigt

Tab. 7-5: Parameter ausgelesen durch **ML_KinematicsInitType01**

EnableCyclicScanning: Der Eingang "EnableCyclicScanning" aktiviert die zyklische Überprüfung und Übertragung der geänderten Eingänge für die aktuelle Betriebsart. Dies ist relevant für die Werte in `arKinCtrl_gb[]` mit der Bemerkung: "Zyklisch überprüft und übertragen bei Änderung.". Der Standardwert ist TRUE.

Group: Der Eingang "Group" gibt an, für welche Kinematik die Initialisierung durchgeführt werden soll.

Fehlerbehandlung Der Funktionsbaustein erzeugt in `DiagStatus.ErrorIdent.Additional1/Additional2` für die Tabelle `F_RELATED_TABLE`, 16#0170 die folgenden Fehlermeldungen:

ErrorID	Additional1	Additional2	Beschreibung
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0000	Ungültiger Zustand des Funktionsbausteins
INPUT_INVALID_ERROR, (16#0001)	16#2600	16#0000	Ungültige Kinematiknummer "Group.GroupNo"
INPUT_INVALID_ERROR, (16#0001)	16#2600	16#0001	Ungültige Steuerungsnummer "Group.CntrlNo". Diese muss auf LOCAL_CNTRL gesetzt sein.

Tab. 7-6: Fehlercodes **ML_KinematicsInitType01**

7.3.2 ML_KinematicsInterfaceType01

Kurzbeschreibung Der Funktionsbaustein **ML_KinematicsInterfaceType01** wird zur Konfiguration des Kinematik-Interfaces für jede Kinematik benutzt.

Dieser Funktionsbaustein muss zyklisch (im Motion-Takt oder langsamer als der Motion-Takt) aufgerufen werden, solange man sich im Betriebsmodus befindet.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/FB

Tab. 7-7: Referenztable **ML_KinematicsInitType01**

Interfacebeschreibung

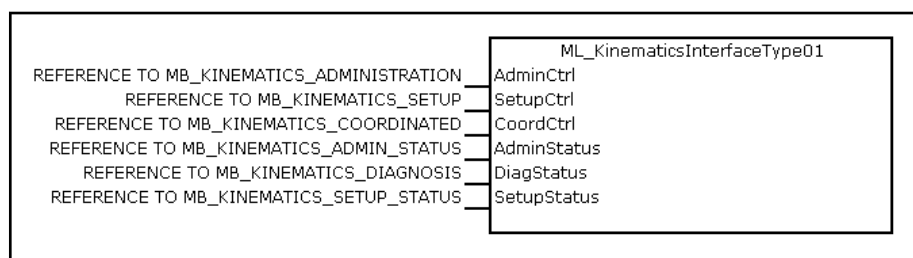


Abb. 7-2: Funktionsbaustein **ML_KinematicsInterfaceType01**

I/O-Typ	Name	Datentyp	Kommentar
VAR_INPUT	AdminCtrl	REFERENCE TO MB_KINEMATICS_ADMINISTRATION	Verwaltung der Achse arKinCtrl_gb[].Admin MB_KINEMATICS_ADMINISTRATION, Seite 377
	SetupCtrl	REFERENCE TO MB_KINEMATICS_SETUP	Einrichtbetrieb arKinCtrl_gb[].SetupMode MB_KINEMATICS_SETUP, Seite 383
	CoordCtrl	REFERENCE TO MB_KINEMATICS_COORDINATED	Betriebsart koordinierte Bewegung arKinCtrl_gb[].CoordMode MB_KINEMATICS_COORDINATED, Seite 385
	AdminStatus	REFERENCE TO MB_KINEMATICS_ADMIN_STATUS	Status Verwaltung der Achse arKinStatus_gb[].Admin MB_KINEMATICS_ADMIN_STATUS, Seite 389
	DiagStatus	REFERENCE TO MB_KINEMATICS_DIAGNOSIS	Status Diagnose der Achse arKinStatus_gb[].Diag MB_KINEMATICS_DIAGNOSIS, Seite 393
	SetupStatus	REFERENCE TO MB_KINEMATICS_SETUP_STATUS	Status Tipp-/Einrichtbetrieb arKinStatus_gb[].SetupMode MB_KINEMATICS_SETUP_STATUS, Seite 394

Tab. 7-8: Schnittstellenvariablen ML_KinematicsInterfaceType01



Es ist nicht möglich anstelle der einzelnen Strukturelemente "AdminCtrl", "SetupCtrl", etc. die kompletten Strukturen "MB_Kinematics_Control_Type01" und "MB_Kinematics_Status_Type01" dem Funktionsbaustein zu übergeben.

Somit könnten in der Anwender-Applikation keine speziellen Erweiterungen zum bestehenden Code hinzugefügt werden.

Deshalb werden die benötigten Elemente von [MB_KINEMATICS_CONTROL_TYPE01, Seite 377](#), und [MB_KINEMATICS_STATUS_TYPE01, Seite 388](#), als separate Eingänge übergeben.

Nähere Details siehe "Kinematik-Interface Anwender-Erweiterung".

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Generic Application Template, DOK-XLCMLC-TF*GAT**V14-APRS-DE-P, R911341486

oder

Rexroth IndraMotion MLC 14VRS Robot-Control V2, DOK-MLC***-ROCO****V14-RERS-DE-P, R911341586

Zur Performanceoptimierung sind die Strukturen statt als VAR_IN_OUT als VAR_INPUT mit REFERENCE TO angeschlossen. Damit reicht es einmalig die Eingänge zu initialisieren. Beim zyklischen Aufruf des Funktionsbausteins müssen damit die Strukturen nicht übergeben werden.

ML_KinTech.library

Fehlerbehandlung Der Funktionsbaustein erzeugt in DiagStatus.ErrorIdent.Additional1/Additional2 für die Tabelle F_RELATED_TABLE, 16#0170 die folgenden Fehlermeldungen:

ErrorID	Additional1	Additional2	Beschreibung
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0000	Ungültiger interner Zustand
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0001	Ungültiger Zustand des Funktionsbausteins
RESOURCE_ERROR, (16#0003)	16#2600	16#0001	Ungültige Steuerungsnummer "Group.CntrlNo". Diese muss auf LOCAL_CNTRL gesetzt sein
DEVICE_ERROR, (16#0008)	16#2601	16#0000	Kinematik meldet einen Fehler in der KinData-Struktur
INPUT_INVALID_ERROR, (16#0001)	16#2602	Nummer des OpMode	Ungültige Betriebsart ausgewählt
INPUT_INVALID_ERROR, (16#0001)	16#2603	Nummer des OpMode	Ungültige, nicht unterstützte Betriebsart aktiviert
INPUT_RANGE_ERROR, (16#0006)	16#2604	16#0000	Mindestens eine Eingangsreferenz (AdminCtrl, AdminStatus, usw.) ist nicht gesetzt
ACCESS_ERROR, (16#0004)	16#2605	16#0000	Referenzieren der Achse(n) nicht möglich und nicht notwendig
ACCESS_ERROR, (16#0004)	16#2605	Nummer der zu referenzierenden Achse	Referenzieren der Achse(n) nicht möglich
RESOURCE_ERROR, (16#0003)	16#2606	16#0000	Betriebsart wurde während des SetupModes verändert
OTHER_ERROR, (16#7FFE)	16#2607	16#0000	Kinematik konnte nicht gruppiert werden

Tab. 7-9: Fehlercodes ML_KinematicsInterfaceType01

Zusätzlich können die Fehler folgender intern aufgerufener Funktionsbausteine durchgereicht werden:

MB_GroupReadParameter	ML_GroupReadNextPoint
MB_GroupReadStringParameter	ML_GroupReadPoint
MC_GroupContinue	ML_GroupSavePointFile
MC_GroupInterrupt	ML_GroupWritePoint
MC_GroupReset	ML_KinJog
MC_GroupStop	ML_SetBeltConfiguration
MC_MoveLinearAbsolute	ML_SetBeltErrorReaction
MC_MoveLinearRelative	ML_SetCylindricTransform
MC_MoveDirectAbsolute	ML_RobotGetBeltConfiguration
MC_MoveDirectRelative	ML_RobotGetBeltErrorReaction
MC_MoveCircularAbsolute	ML_RobotGetTrafoValues

MC_MoveCircularRelative	ML_RobotMoveJumpAbsolute
MC_SetCartesianTransform	ML_RobotOpenCyclicChannel
MC_SetKinTransform	ML_RobotOpenDirectCyclicChannel
ML_GroupDeletePointFile	ML_RobotWriteCyclicChannel
ML_GroupDeletePoints	TP
ML_GroupLoadPointFile	TON

Tab. 7-10: Quellen durchgereichter Fehlercodes ML_KinematicsInterfaceType01

7.3.3 ML_KinematicsInterfaceType11

Kurzbeschreibung Der Funktionsbaustein **ML_KinematicsInterfaceType11** wird zur Konfiguration des Kinematik-Interfaces für jede Kinematik benutzt. Der Funktionsumfang entspricht dem ML_KinematicsInterfaceType01, wobei dieser Funktionsbaustein ereignisgesteuert und damit performanceoptimiert arbeitet.

Dieser Funktionsbaustein muss zyklisch (im Motion-Takt oder langsamer als der Motion-Takt) aufgerufen werden, solange man sich im Betriebsmodus befindet.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/FB

Tab. 7-11: Referenztabelle ML_KinematicsInitType11

Interfacebeschreibung

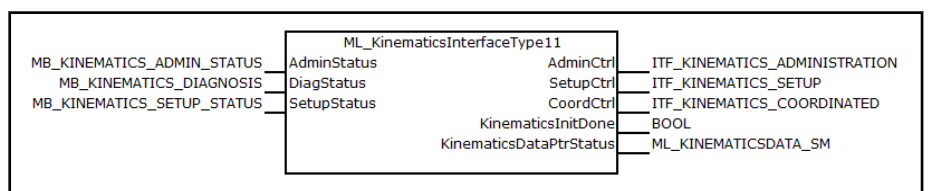


Abb. 7-3: Funktionsbaustein ML_KinematicsInterfaceType11

I/O-Typ	Name	Datentyp	Kommentar
VAR_INPUT	AdminStatus	REFERENCE TO MB_KINEMATICS_ADMIN_STATUS	Status Verwaltung der Achse arKinStatus_gb[].Admin MB_KINEMATICS_ADMIN_STATUS, Seite 389
	DiagStatus	REFERENCE TO MB_KINEMATICS_DIAGNOSIS	Status Diagnose der Achse arKinStatus_gb[].Diag MB_KINEMATICS_DIAGNOSIS, Seite 393
	SetupStatus	REFERENCE TO MB_KINEMATICS_SETUP_STATUS	Status Tipp-/Einrichtbetrieb arKinStatus_gb[].SetupMode MB_KINEMATICS_SETUP_STATUS, Seite 394

ML_KinTech.library

I/O-Typ	Name	Datentyp	Kommentar
VAR_OUT-PUT	AdminCtrl	ITF_KINEMATICS_ADMINISTRATION	Verwaltung der Achse arKinCtrl_gb[].Admin Entspricht inhaltlich dem Typ MB_KINEMATICS_ADMINISTRATION , Seite 377
	SetupCtrl	ITF_KINEMATICS_SETUP	Einrichtbetrieb arKinCtrl_gb[].SetupMode Entspricht inhaltlich dem Typ MB_KINEMATICS_SETUP , Seite 383
	CoordCtrl	ITF_KINEMATICS_COORDINATED	Betriebsart koordinierte Bewegung arKinCtrl_gb[].CoordMode Entspricht inhaltlich dem Typ MB_KINEMATICS_COORDINATED , Seite 385
	KinematicsInitDone	BOOL	Initialisierung der Kinematik ist abgeschlossen
	KinematicsDataPtrStatus	POINTER TO ML_KINEMATICS_DATA_SM	Zeiger auf KinData-Schnittstelle der Kinematik (ML_Robot.library) - Eingabe .KinematicsDataPtrStatus^ für Auswahlmenü ML_KINEMATICS_DATA_SM, ML_Base, enthält die Informationen der korrespondierenden Kinematik, auf die per Shared Memory zugegriffen werden kann

Tab. 7-12: Schnittstellenvariablen ML_KinematicsInterfaceType11



Der Funktionsbaustein nutzt sogenannte Interfaces zur Vorgabe der Betriebsarten und Sollwerte.

Zur Performanceoptimierung sind die Statusstrukturen statt als VAR_IN_OUT als VAR_INPUT mit REFERENCE TO angeschlossen. Damit ist eine einmalige Initialisierung der Eingänge ausreichend und die Strukturen müssen beim zyklischen Aufruf des Funktionsbausteins nicht übergeben werden.

Fehlerbehandlung

Der Funktionsbaustein erzeugt in DiagStatus.ErrorIdent.Additional1/Additional2 für die Tabelle F_RELATED_TABLE, 16#0170 die folgenden Fehlermeldungen:

ErrorID	Additional1	Additional2	Beschreibung
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0000	Ungültiger interner Zustand
STATE_MACHINE_ERROR, (16#0005)	16#0006	16#0001	Ungültiger Zustand des Funktionsbausteins
RESOURCE_ERROR, (16#0003)	16#2600	16#0001	Ungültige Steuerungsnummer "Group.CntrlNo". Diese muss auf LOCAL_CNTRL gesetzt sein
DEVICE_ERROR, (16#0008)	16#2601	16#0000	Kinematik meldet einen Fehler in der KinData-Struktur
INPUT_INVALID_ERROR, (16#0001)	16#2602	Nummer des OpMode	Ungültige Betriebsart ausgewählt

ErrorID	Additional1	Additional2	Beschreibung
INPUT_INVALID_ERROR, (16#0001)	16#2603	Nummer des OpMode	Ungültige, nicht unterstützte Betriebsart aktiviert
INPUT_RANGE_ERROR, (16#0006)	16#2604	16#0000	Mindestens eine Eingangsreferenz (AdminCtrl, AdminStatus, usw.) ist nicht gesetzt
ACCESS_ERROR, (16#0004)	16#2605	16#0000	Referenzieren der Achse(n) nicht möglich und nicht notwendig
ACCESS_ERROR, (16#0004)	16#2605	Nummer der zu referenzierenden Achse	Referenzieren der Achse(n) nicht möglich
RESOURCE_ERROR, (16#0003)	16#2606	16#0000	Betriebsart wurde während des SetupModes verändert
OTHER_ERROR, (16#7FFE)	16#2607	16#0000	Kinematik konnte nicht gruppiert werden

Tab. 7-13: Fehlercodes ML_KinematicsInterfaceType11

Zusätzlich können die Fehler folgender intern aufgerufener Funktionsbausteine durchgereicht werden:

MB_GroupReadParameter	ML_GroupReadNextPoint
MB_GroupReadStringParameter	ML_GroupReadPoint
MC_GroupContinue	ML_GroupSavePointFile
MC_GroupInterrupt	ML_GroupWritePoint
MC_GroupReset	ML_KinJog
MC_GroupStop	ML_SetBeltConfiguration
MC_MoveLinearAbsolute	ML_SetBeltErrorReaction
MC_MoveLinearRelative	ML_SetCylindricTransform
MC_MoveDirectAbsolute	ML_RobotGetBeltConfiguration
MC_MoveDirectRelative	ML_RobotGetBeltErrorReaction
MC_MoveCircularAbsolute	ML_RobotGetTrafoValues
MC_MoveCircularRelative	ML_RobotMoveJumpAbsolute
MC_SetCartesianTransform	ML_RobotOpenCyclicChannel
MC_SetKinTransform	ML_RobotOpenDirectCyclicChannel
ML_GroupDeletePointFile	ML_RobotWriteCyclicChannel
ML_GroupDeletePoints	TP
ML_GroupLoadPointFile	TON

Tab. 7-14: Quellen durchgereichter Fehlercodes ML_KinematicsInterfaceType11

ML_KinTech.library

7.3.4 KinInterface/FB/DataTypes

Kin_Axis_Config_Elements

Diese Struktur-Konstante beinhaltet Informationen über die Achsen, welche der Kinematik zugeordnet sind ([K-0-0008](#)):

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/FB/DataTypes

Tab. 7-15: Referenztable Kin_Axis_Config_Elements

STRUCT Kin_Axis_Config_Elements

Name	Datentyp	Initialwert	Kommentar
uiNumberOfKinAxes	UINT	0	Anzahl der Achsen, welche der Kinematik zugeordnet sind
uiNumberOfMoveAxes	UINT	0	Anzahl der Positionier- und Orientierungsachsen, welche der Kinematik zugeordnet sind
uiNumberOfBeltAxes	UINT	0	Anzahl der Bandachsen, welche der Kinematik zugeordnet sind
arAxesConfig	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF ML_KIN_AXES_CONFIG_DATA		Achskonfiguration (Achstyp, Steuerungsnummer, Achsnummer)

Die Struktur verwendet

- ML_Robot.library: ML_KIN_AXES_CONFIG_DATA

Kin_Axis_Type

Der Aufzählungstyp nennt die möglichen Achstypen: (keine Achse, Positionierachse, Orientierungsachse, Bandachse)

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/FB/DataTypes

Tab. 7-16: Referenztable Kin_Axis_Type

ENUM Kin_Axis_Type_

Name	Datentyp	Initialwert	Kommentar
NoAxisCfg	UINT	0	keine Achse
PositionAxisCfg	UINT	8	Positionierachse
OrientationAxisCfg	UINT	16	Orientierungsachse
BeltAxisCfg	UINT	32	Bandachse

7.4 KinInterface/Type

7.4.1 Kinematik-Interface-Strukturen, Überblick

Das Kinematik-Interface stellt eine Datenschnittstelle zur komfortablen Bedienung einer Kinematik zur Verfügung. Die Datenstrukturen zum Kinematik-Interface werden im Folgenden beschrieben.



Das Kinematik-Interface ist so vorbereitet, dass es durch den Anwender erweiterbar ist.

Um diese Erweiterungen zu ermöglichen, ist es nötig, eigene Strukturen im Anwenderprojekt zu definieren, welche die Strukturen der Bibliothek erweitern.

Die erweiterten Strukturen sind durch den Präfix "TE_" gekennzeichnet.

Wenn also eine Struktur, z. B.

TE_KINEMATICS_ADMIN_STATUS

heißt, ist diese eine erweiterte Struktur der

MB_KINEMATICS_ADMIN_STATUS.

Informationen zu der Struktur sind unter dem Namen **MB_KINEMATICS_ADMIN_STATUS** zu finden.

7.4.2 Kinematik-Interface-Betriebsarten, Übersicht

Es gibt zwei Methoden um eine Betriebsart zu aktivieren:

1. Auswahl über Enum-Werte

Zuweisen eines Wertes vom TYPE **MB_KINEMATICS_MODE**:

```
arKinCtrl_gb[].Admin._OpMode.en:= ModeCoordAH;
```

2. Benutzung des Bit-Zugriffs

Setzen eines Bits über die Bit-Zugriffsfunktionalität.

```
arKinCtrl_gb[].Admin._OpMode.b.MODE_COORD_AH:= TRUE;
```



Es darf immer nur 1 Bit in **arKinCtrl_gb[].Admin._OpMode** gesetzt sein. Das Setzen mehrerer Bits verursacht einen Fehler im Kinematik-Interface.

Bevor eine Betriebsart aktiviert werden kann, müssen jedem Attribut zuerst Werte zugewiesen werden. Alle Attribute haben Standardwerte. Einige haben Werte ungleich Null, während andere als 0 definiert sind und aufgrund der speziellen Anforderungen, ihnen ein Wert zugewiesen werden muss.



Nur die Attribute (z. B. Point, Velocity), die benutzt werden oder deren Standardwert geändert wurde, müssen definiert werden, bevor der aktuelle Betriebsartenwechsel ausgeführt wird.



Das Kinematik-Interface nutzt intern das Achs-Interface, um einige Umschaltungen durchzuführen (z. B. Gruppieren/Degruppieren aller Achsen, Referenzieren von Achsen) .

Siehe auch

- [MB_KINEMATICS_MODE_UNION](#), Seite 374
- [MB_KINEMATICS_MODE_BITS](#), Seite 374

- [MB_KINEMATICS_MODE](#), Seite 375

7.4.3 MB_KINEMATICS_MODE_UNION

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type

Tab. 7-17: Referenztabelle MB_KINEMATICS_MODE_UNION

Kurzbeschreibung **MB_KINEMATICS_MODE_UNION** ist ein Vereinigungsdatentyp (UNION), der vom **_OpMode**-Element, welches in der MB_KINEMATICS_ADMINISTRATION-Struktur enthalten ist, benutzt wird.

Die Steuerung von **_OpMode** ist über beide Datentypen möglich, da diese auf der gleichen Speicherstelle liegen.

UNION MB_KINEMATICS_MODE_UNION

Name	Datentyp	Kommentar
en	MB_KINEMATICS_MODE	Betriebsarten als Enumeration vom Typ MB_KINEMATICS_MODE , Seite 375,
b	MB_KINEMATICS_MODE_BITS	Betriebsarten als Bit-Struktur vom Typ MB_KINEMATICS_MODE_BITS , Seite 374

- en Zugriff auf die Betriebsart über einen Aufzählungswert
- b Zugriff auf die Betriebsart über eine Bitstruktur



Abb. 7-4: MB_KINEMATICS_MODE_UNION im Überblick

7.4.4 MB_KINEMATICS_MODE_BITS

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type

Tab. 7-18: Referenztabelle MB_KINEMATICS_MODE_BITS

Kurzbeschreibung Die Struktur MB_KINEMATICS_MODE_BITS beinhaltet die Bitstruktur für das Element **_OpMode** der Struktur [MB_KINEMATICS_ADMINISTRATION](#), Seite 377, und für die Struktur [MB_KINEMATICS_ADMIN_STATUS](#), Seite 389.

STRUCT MB_KINEMATICS_MODE_BITS

Name	Datentyp	Initialwert	Kommentar
MODE_COORD_AB	BIT	FALSE	Kinematik bereit, Antrieb bereit, OpMode(Ack).0
MODE_COORD_AH	BIT	FALSE	Kinematik bereit, Antrieb halt, OpMode(Ack).1
MODE_COORD_HOMING	BIT	FALSE	Referenzieren, AH, AxisInterface -> SetupMode, OpMode(Ack).2
MODE_COORD_POS_DIR_ABS	BIT	FALSE	Absolutes PTP Positionieren, AF, MC_MoveDirectAbsolute, OpMode(Ack).3
MODE_COORD_POS_DIR_REL	BIT	FALSE	Relatives PTP Positionieren, AF, MC_MoveDirectRelative, OpMode(Ack).4
MODE_COORD_POS_LIN_ABS	BIT	FALSE	Absolutes lineares Positionieren, AF, MC_MoveLinearAbsolute, OpMode(Ack).5
MODE_COORD_POS_LIN_REL	BIT	FALSE	Relatives lineares Positionieren, AF, MC_MoveLinearRelative, OpMode(Ack).6
MODE_COORD_POS_CIRC_ABS	BIT	FALSE	Absolutes zirkulares Positionieren, AF, MC_MoveCircularAbsolute, OpMode(Ack).7
MODE_COORD_POS_CIRC_REL	BIT	FALSE	Relatives zirkulares Positionieren, AF, MC_MoveCircularRelative, OpMode(Ack).8
MODE_COORD_POS_JUMP_ABS	BIT	FALSE	Absolutes Jump Positionieren, AF, ML_RobotMoveJumpAbsolute OpMode(Ack).9
MODE_COORD_STOPPING	BIT	FALSE	Kinematik Stopp, AH, MC_GroupStop, OpMode(Ack).10
MODE_COORD_INTERRUPTED	BIT	FALSE	Betriebsart "Interrupted", AF, MC_GroupInterrupt, OpMode(Ack).11
MODE_COORD_CONTINUE	BIT	FALSE	Betriebsart "Continue", AF, MC_GroupContinue, OpMode(Ack).12
MODE_COORD_EXTERNAL_FB	BIT	FALSE	Betriebsart "Externer Funktionsbaustein", Leistung ist zugeschaltet und dann übernimmt der externe FB die Steuerung, OpMode(Ack).13
MODE_COORD_CYCLIC	BIT	FALSE	Zyklische Positionssollwertvorgabe, AF, ML_RobotWriteCyclicChannel, OpMode(Ack).14

7.4.5 MB_KINEMATICS_MODE

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type

Tab. 7-19: Referenztablelle MB_KINEMATICS_MODE

Kurzbeschreibung Der Aufzählungstyp MB_KINEMATICS_MODE (ENUM) beinhaltet die Konstanten für das Element **_OpMode** der Struktur [MB_KINEMATICS_ADMINISTRATION](#), Seite 377,

ML_KinTech.library

und das Element **_OpModeAck** der Struktur [MB_KINEMATICS_ADMIN_STATUS](#), Seite 389.

Der Aufzählungstyp wird von den Strukturen [MB_KINEMATICS_MODE_UNION](#), Seite 374, und [MB_KINEMATICS_ADMIN_STATUS](#) genutzt.

STRUCT MB_KINEMATICS_MODE

Name	Datentyp	Initialwert	Kommentar
ModeCoordUngrouped	UINT	16#0	Alle Achsen in AB oder AH/AF
ModeCoordAB	UINT	16#1	AB, alle Achsen drehmomentenfrei
ModeCoordAH	UINT	16#2	AH, alle Achsen drehmomentenbehaftet, Endzustand 'standstill' (AB->AH oder AF->AH)
ModeCoordHoming	UINT	16#4	Referenzieren, AH, AxisInterface -> SetupMode
ModeCoordPosDirAbs	UINT	16#8	Absolutes PTP Positionieren, AF, MC_MoveDirectAbsolute
ModeCoordPosDirRel	UINT	16#10	Relatives PTP Positionieren, AF, MC_MoveDirectRelative
ModeCoordPosLinAbs	UINT	16#20	Absolutes lineares Positionieren, AF, MC_MoveLinearAbsolute
ModeCoordPosLinRel	UINT	16#40	Relatives lineares Positionieren, AF, MC_MoveLinearRelative
ModeCoordPosCircAbs	UINT	16#80	Absolutes zirkulares Positionieren, AF, MC_MoveCircularAbsolute
ModeCoordPosCircRel	UINT	16#100	Relatives zirkulares Positionieren, AF, MC_MoveCircularRelative
ModeCoordPosJumpAbs	UINT	16#200	Absolutes Jump Positionieren, AF, ML_RobotMoveJumpAbsolute
ModeCoordStopping	UINT	16#400	Kinematik stopp, AH, MC_GroupStop
ModeCoordInterrupted	UINT	16#800	Betriebsart "Interrupted", AF, MC_GroupInterrupt
ModeCoordContinue	UINT	16#1000	Betriebsart "Continue", AF, MC_GroupContinue
ModeCoordExternalFB	UINT	16#2000	Betriebsart "Externer Funktionsbaustein", Leistung ist zugeschaltet und dann übernimmt der externe FB die Steuerung
ModeCoordCyclic	UINT	16#4000	Zyklische Positionssollwertvorgabe, AF, ML_RobotWriteCyclicChannel

7.4.6 KinInterface/Type/Control

Ordner Kin_Interface/Type/Control, Übersicht

MB_KINEMATICS_CONTROL_TYPE01 auf Seite 377			
	MB_KINEMATICS_ADMINISTRATION auf Seite 377		
		MB_KINEMATICS_MODE_UNION auf Seite 374	
		MB_KINEMATICS_CONVEYOR_BELT auf Seite 380	
		MB_KINEMATICS_TEACH_IN auf Seite 380	
			MB_KINEMATICS_TEACH_IN_MODE auf Seite 382
		MB_KINEMATICS_ADMIN_CHANGE_OPMODE auf Seite 383	
		MB_KINEMATICS_PCS_CONFIG auf Seite 383	
	MB_KINEMATICS_SETUP auf Seite 383		
	MB_KINEMATICS_COORDINATED auf Seite 385		

Tab. 7-20: Struktureller Aufbau von MB_KINEMATICS_CONTROL_TYPE01

MB_KINEMATICS_CONTROL_TYPE01

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-21: Referenztablette MB_KINEMATICS_CONTROL_TYPE01

Kurzbeschreibung Auf die MB_KINEMATICS_CONTROL_TYPE01-Datenstruktur wird über die arKinCtrl_gb[]-Instanz zugegriffen.

STRUCT MB_KINEMATICS_CONTROL_TYPE01

Name	Datentyp	Kommentar
Admin	MB_KINEMATICS_ADMINISTRATION	Verwaltung der Kinematik
SetupMode	MB_KINEMATICS_SETUP	Tipp-/Einrichtfunktionen
CoordMode	MB_KINEMATICS_COORDINATED	Koordinierbetriebsart (Linear, Zirkular, PTP; absolut, relativ)

MB_KINEMATICS_ADMINISTRATION

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-22: Referenztablette MB_KINEMATICS_ADMINISTRATION

Kurzbeschreibung Die MB_KINEMATICS_ADMINISTRATION-Struktur ist eine Unterstruktur von MB_KINEMATICS_CONTROL_TYPE01 auf Seite 377 und enthält Steuerungsvariablen und Sollwerte für die Verwaltung der Kinematik.

STRUCT MB_KINEMATICS_ADMINISTRATION

Name	Datentyp	Initialwert	Kommentar
_OpMode	MB_KINEMATICS_MODE_UNION	STRUCT (en := ModeCoordAb)	Zustandsgesteuert Vor Anwahl der Betriebsart die Eingangsdaten für die Betriebsart beschreiben - Betriebsartenanwahl (ENUMs): z. B. ..._OpMode.en := ModeCoordAB - Betriebsartenanwahl (BIT-Variablen): z. B. ..._OpMode.b.MODE_COORD_AB := TRUE Hinweis: Immer nur 1 Eingang setzen!
Group	AXIS_GROUP_REF	STRUCT (CntrlNo := RIL_CommonTypes.LOCAL_CNTRL, GroupNo := NO_KINEMATICS)	Kinematik-Referenz
KinTransformPLC	MC_KIN_REF	0	Zeiger auf das Transformations-Interface der Kinematik
DiagNbrRefreshTime	TIME	TIME#200ms	Auffrischrate von arKinStatus[].Diag.Number - Ausgeschaltet, wenn Null - Liest Antriebsparameter, wenn >= T#0ms

ML_KinTech.library

Name	Datentyp	Initialwert	Kommentar
StopDeceleration	REAL	999.0	Bremsverzögerung Positiver Wert, (Einheit: z. B. in [rad/s ²] je nach Parametrierung)
StopJerk	REAL	9999.0	Verzögerungsruck Positiver Wert, (Einheit: z. B. in [rad/s ³] je nach Parametrierung)
ClearError	BOOL	FALSE	Flankengesteuert Löscht Kinematikfehler mit MC_GroupReset()
VelocityOverride	REAL	100	Vorschubkorrektur Wird auf den Parameter K-0-0240 geschrieben
AccelerationOverride	REAL	100	Beschleunigungskorrektur Wird auf den Parameter K-0-0242 geschrieben
DecelerationOverride	REAL	100	Verzögerungskorrektur Wird auf den Parameter K-0-0243 geschrieben
EnableCyclicScanning	BOOL	TRUE	Zustandsgesteuert Zuschalten des bedarfsweisen Scannens von Eingangsdaten in der aktuellen Betriebsart Betrifft Größen in arKinCtrl_gb mit Anm. "Hinweis: Wird permanent zyklisch verarbeitet"
ParamLevel	BOOL	FALSE	Umschalten der Achsen in den Parametriermodus 1, <i>Parameter A-0-0024 Bit 3</i> - TRUE: aktiviert den Parametriermodus - FALSE: deaktiviert den Parametriermodus
RetriggerOpMode	BOOL	FALSE	Flankengesteuert Neustart der gleichen Betriebsart, z. B. Mode-CoordPosLinAbs mit gleichen Werten
ConveyorBelt	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF MB_KINEMATICS_CONVEYOR_BELT		Konfiguration eines Transportbandes
TeachIn	MB_KINEMATICS_TEACH_IN		Teachen/Speichern von Punkten MB_KINEMATICS_TEACH_IN auf Seite 380
ChangeOpMode	MB_KINEMATICS_ADMIN_CHANGE_OP-MODE	NoAction	Verhalten nach einem Betriebsartenwechsel MB_KINEMATICS_ADMIN_CHANGE_OPMODE auf Seite 383
PCSConfig	ARRAY [1..ML_MAX_KIN_PCS_NUMBER] OF MB_KINEMATICS_PCS_CONFIG		Konfiguration eines Produkt-Koordinatensystems MB_KINEMATICS_PCS_CONFIG auf Seite 383

OpMode	Anwahl der Betriebsart Nähere Informationen zum Betriebsartenwechsel siehe Kinematik-Interface-Betriebsarten, Übersicht auf Seite 373.  Das Setzen mehrerer Bits in <code>arKinCtrl_gb[].Admin._OpMode.b</code> verursacht einen Fehler.
KinTransformPLC	Mithilfe dieses Elementes ist es möglich, benutzerdefinierte Transformationen anzumelden. Legt man einen Transformationsfunktionsbaustein an, so wird dieser in der SPS angemeldet.  Die Anmeldung einer benutzerdefinierten Transformation ist nur im Parametriermodus P2 möglich. Wird versucht, eine Transformation im Betriebsmodus BB anzumelden, bleibt dies ohne Resultat.  Wird das Kinematik-Interface erstmalig mit einer benutzerdefinierten Transformation gestartet, so gibt die Steuerung den Fehler "Transformation nicht angemeldet" aus. Dies ist dadurch begründet, dass das IMC-Interface versucht in den Betriebsmodus zu schalten. Nach erfolgreicher Anmeldung der Transformation lässt sich der Fehler quittieren.
EnableCyclicScanning	Dieses zustandsgesteuerte Element steuert das zyklische Lesen von <code>arKinCtrl_gb[].CoordMode.Point</code>. Mit Setzen des Elements wird mit jeder neuen Position eine neue Bewegung gestartet und somit ein neuer Satz in den Satzpuffer gestellt.
ParamLevel	Mit diesem flankengesteuerten Element können alle, der Kinematik zugehörigen Achsen in den Parametriermodus versetzt werden. Durch Rücksetzen des Elements werden die Achsen wieder in den ursprünglichen Zustand versetzt.  Ist der Parametriermodus aktiv, wird im Projektbaum bei den zugehörigen Achsen "PM" angezeigt.
RetriggerOpMode	Dieses flankengesteuerte Element bewirkt einen Neustart der aktuell verwendeten Betriebsart. Wird das Element gesetzt, so schaltet das Kinematik-Interface intern nach "ModeCoordExternalFB" und anschließend wieder in die ursprüngliche Betriebsart. Somit ist beispielsweise ein mehrmaliges relatives Verfahren der Kinematik möglich, ohne Positionswerte zu ändern.
ConveyorBelt	Zugriff auf die Attribute zur Konfiguration von Bändern (MB_KINEMATICS_CONVEYOR_BELT auf Seite 380).  Maximal sind 16 Bänder pro Kinematik konfigurierbar.
PCSConfig	Zugriff auf die Attribute zur Konfiguration von Produkt-Koordinatensystemen (MB_KINEMATICS_PCS_CONFIG auf Seite 383).  Maximal sind 16 Produkt-Koordinatensysteme pro Kinematik konfigurierbar. • Ordner Kin_Interface/Type/Control, Übersicht auf Seite 376

ML_KinTech.library

MB_KINEMATICS_CONVEYOR_BELT

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-23: Referenztable MB_KINEMATICS_CONVEYOR_BELT

Kurzbeschreibung

Die Struktur MB_KINEMATICS_CONVEYOR_BELT dient der Konfiguration eines Bandes.

Die in dieser Struktur enthaltenen Elemente entsprechen den Eingängen der Funktionsbausteine (ML_Robot.library)

- ML_SetBeltConfiguration
- ML_SetBeltErrorReaction

Nähere Informationen sind der entsprechenden Dokumentation zu entnehmen.

STRUCT MB_KINEMATICS_CONVEYOR_BELT

Name	Datentyp	Initialwert	Kommentar
Execute	BOOL	FALSE	Flankengesteuert Ausführen der Bandkonfiguration
ConveyorBeltOrigin	ARRAY [1..6] OF REAL	[6(0.0)]	Definiert Position und Verdrehung des Bandes relativ zum WCS
TotalLength	REAL	0.0	Definiert Länge des bandsynchronen Verfahrbereichs ab der Startposition
BeginLength	REAL	0.0	Definiert die Länge des spätesten Starten einer bandsynchronen Bewegung ab der Startposition
Axis	AXIS_REF		Referenz der Bandachse
ErrorReactionType	ML_BELT_ERROR_REACTION_MODE	ML_BELT_ERROR_NO_SPECIFIC_REACTION	Definiert den Typ der Fehlerreaktion • ML_BELT_ERROR_NO_SPECIFIC_REACTION • ML_BELT_ERROR_DESYNC_OVER_TIME • ML_BELT_ERROR_DESYNC_WITH_LIMITS
ErrorReactionDesyncTime	TIME	TIME#0ms	Definiert Zeit zum Absynchronisieren
ErrorReactionDeceleration	REAL	0.0	Definiert Bremsverzögerung zum Absynchronisieren
ErrorReactionJerk	REAL	0.0	Definiert Bremsruck zum Absynchronisieren

- [Ordner Kin_Interface/Type/Control, Übersicht auf Seite 376](#)

MB_KINEMATICS_TEACH_IN

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-24: Referenztable MB_KINEMATICS_TEACH_IN

Kurzbeschreibung *Die Struktur MB_KINEMATICS_TEACH_IN dient dem*

- Schreiben eines einzelnen Punktes (ML_GroupWritePoint, ML_Robot)
- Lesen eines einzelnen Punktes (ML_GroupReadPoint, ML_Robot)
- Lesen des nächsten Punktes (ML_GroupReadNextPoint, ML_Robot)
- Löschen eines oder mehrerer Punkte(s) (ML_GroupDeletePoints, ML_Robot)
- Speichern einer Punktedatei (ML_GroupSavePointFile, ML_Robot)
- Laden einer Punktedatei (ML_GroupLoadPointFile, ML_Robot)
- Löschen einer Punktedatei (ML_GroupDeletePointFile, ML_Robot)

Nähere Informationen sind der Dokumentation des entsprechenden Funktionsbausteins zu entnehmen.

Siehe Rexroth IndraMotion MLC 14VRS Robot-Control V2, DOK-MLC***-ROCO****V14-RERS-DE-P, R911341586.

STRUCT MB_KINEMATICS_TEACH_IN

Name	Datentyp	Initialwert	Kommentar
Execute	BOOL	FALSE	Flankengesteuert Ausführen der Bearbeitung
Mode	MB_KINEMATICS_TEACH_IN_MODE	WritePoint	Art der Bearbeitung eines Punktes oder einer Punktedatei MB_KINEMATICS_TEACH_IN_MODE , Seite 382
FileName	STRING	"	<i>Name der Datei</i> • Eingang des FBs ML_GroupSavePointFile • Eingang des FBs ML_GroupLoadPointFile • Eingang des FBs ML_GroupDeletePointFile
PointName	STRING(19)	"	<i>Name des Punktes</i> • Eingang des FBs ML_GroupWritePoint • Eingang des FBs ML_GroupReadPoint • Eingang des FBs ML_GroupReadnextPoint • Eingang des FBs ML_GroupDeletePoints
Point	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER(0.0)]	<i>Punkte-Array</i> • Eingang des FBs ML_GroupWritePoint
CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	<i>Koordinatensystem des Punktes (ML_Robot.library)</i> • Eingang des FBs ML_GroupWritePoint <i>Mögliche Koordinatensysteme</i> • CS_MACHINE_COORDINATES • CS_AXES_COORDINATES • CS_BASE_COORDINATES • CS_PRODUCT_COORDINATES_n (n = 1..16)

ML_KinTech.library

Name	Datentyp	Initialwert	Kommentar
Option	E_KIN_TEACH_IN_PO INT_FILE_OPTIONS	E_GROUP_TEACH_IN _NEW	<i>Option zum Laden und Speichern von Punkte- dateien</i> - Eingang des FBs ML_GroupSavePointFile - Eingang des FBs ML_GroupLoadPointFile <i>Mögliche Optionen (ML_Robot.library)</i> - E_GROUP_TEACH_IN_NEW - E_GROUP_TEACH_IN_APPEND - E_GROUP_TEACH_IN_APPEND_OVER- WRITE
AllowOverwrite	BOOL	FALSE	<i>Erlauben des Überschreibens existierender Punkte</i> Eingang des FBs ML_GroupWritePoint
SaveBeltValues	BOOL	FALSE	<i>Speichern der Bandwerte des Punktes</i> Eingang des FBs ML_GroupWritePoint

- [Ordner Kin_Interface/Type/Control, Übersicht auf Seite 376](#)

MB_KINEMATICS_TEACH_IN_MODE

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-25: Referenztable MB_KINEMATICS_TEACH_IN_MODE

Der Aufzählungstyp nennt die möglichen Zugriffsarten der Betriebsart "TEACH_IN".

ENUM MB_KINEMATICS_TEACH_IN_MODE

Name	Datentyp	Initialwert	Kommentar
WritePoint	UINT	0	Schreiben eines Punktes
ReadPoint	UINT	1	Lesen eines Punktes
ReadNextPoint	UINT	2	Lesen eines nächsten Punktes
DeletePoints	UINT	4	Löschen eines Punktes
SavePointFile	UINT	8	Speichern einer Punktedatei
LoadPointFile	UINT	10	Laden einer Punktedatei
DeletePointFile	UINT	12	Löschen einer Punktedatei

- [Ordner Kin_Interface/Type/Control, Übersicht auf Seite 376](#)

MB_KINEMATICS_ADMIN_CHANGE_OPMODE

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-26: Referenztabelle MB_KINEMATICS_ADMIN_CHANGE_OPMODE
 Der Aufzählungstyp beschreibt das Verhalten nach einem Betriebsartenwechsel.

ENUM MB_KINEMATICS_TEACH_IN_MODE

Name	Datentyp	Initialwert	Kommentar
NoAction	UINT	16#0	Führt den aktuellen Satz aus und kommandiert den nächsten Satz
StopMovement	UINT	16#1	Stoppt den aktuellen Satz und startet den nächsten Satz

- [Ordner Kin_Interface/Type/Control, Übersicht auf Seite 376](#)

MB_KINEMATICS_PCS_CONFIG

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-27: Referenztabelle MB_KINEMATICS_PCS_CONFIG

Kurzbeschreibung Die MB_KINEMATICS_PCS_CONFIG-Struktur ist eine Unterstruktur von [MB_KINEMATICS_CONTROL_TYPE01, Seite 377](#) und enthält Variablen zur Konfiguration des Produkt-Koordinatensystems (PCS) einer Kinematik.

STRUCT MB_KINEMATICS_PCS_CONFIG

Name	Datentyp	Initialwert	Kommentar
Execute	BOOL	FALSE	Start Konfiguration des PCS
Origin	ARRAY [1..6] OF REAL	[6 (0)]	<i>Orientierung des PCS im MCS</i> [TransX, TransY, TransZ, RotationX, RotationY, RotationZ]
CoordType	ML_KIN_CS_TYPE	CS_TYPE_CARTESIAN	<i>Typ des PCS</i> - Kartesisches Koordinatensystem - Zylindrisches Koordinatensystem

- [Ordner Kin_Interface/Type/Control, Übersicht auf Seite 376](#)

MB_KINEMATICS_SETUP

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-28: Referenztabelle MB_KINEMATICS_SETUP

Kurzbeschreibung Die MB_KINEMATICS_SETUP-Struktur ist eine Unterstruktur von [MB_KINEMATICS_CONTROL_TYPE01, Seite 377](#) und enthält Steuerungsvariablen für das Vertippen einer Kinematik.

ML_KinTech.library

STRUCT MB_KINEMATICS_SETUP

Name	Datentyp	Initialwert	Kommentar
Enable	BOOL	FALSE	Freigabe Tipp-/Einrichtbetrieb
CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	<i>Mögliche Koordinatensysteme (ML_Robot.library)</i> • CS_MACHINE_COORDINATES • CS_AXES_COORDINATES • CS_BASE_COORDINATES • CS_PRODUCT_COORDINATES_n (n = 1..16)
JogMode	ML_JOGMODE	JM_STEP	Tippart ML_JOGMODE, Seite 397
JogPlus	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF BOOL	[MB_MAX_KIN_AXIS_NUMBER(FALSE)]	<i>Tippen</i> • Inkrementell, wenn JogMode = JM_CONT -> FALSE • Kontinuierlich, wenn JogMode = JM_STEP -> TRUE
JogMinus	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF BOOL	[MB_MAX_KIN_AXIS_NUMBER(FALSE)]	<i>Tippen</i> • Inkrementell, wenn JogMode = JM_CONT -> FALSE • Kontinuierlich, wenn JogMode = JM_STEP -> TRUE
Distance	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER(1.0)]	Tippdistanz
Velocity	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER(10.0)]	Tippgeschwindigkeit
Acceleration	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER(10.0)]	Tippbeschleunigung
Deceleration	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER(10.0)]	Tippverzögerung
Jerk	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER(10.0)]	Tippruck

Distance Dieses Element beinhaltet die Distanz für das inkrementelle Tippen im Achs- und kartesischen Koordinatensystem.



Beim Tippen mit dem Kinematik-Interface wird nur mit KIN_JOG_STEP_SMALL verfahren und die Schrittweite über "Distance" bestimmt.

Velocity Dieses Element beinhaltet die langsame Geschwindigkeit für das kontinuierliche Tippen im Achs- und kartesischen Koordinatensystem.



Beim Tippen mit dem Kinematik-Interface wird nur mit KIN_JOG_CONT_SLOW verfahren und die Verfahrensgeschwindigkeit über "Velocity" bestimmt.

- [Ordner Kin_Interface/Type/Control, Übersicht auf Seite 376](#)

MB_KINEMATICS_COORDINATED

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Control

Tab. 7-29: Referenztable MB_KINEMATICS_COORDINATED

Kurzbeschreibung

Die MB_KINEMATICS_COORDINATED-Struktur ist eine Unterstruktur von MB_KINEMATICS_CONTROL_TYPE01, Seite 377, und enthält Steuerungsvariablen und Sollwerte für die Positionierbetriebsarten.

Die in dieser Struktur enthaltenen Elemente entsprechen den Eingängen der Funktionsbausteine:

- MC_MoveLinearAbsolute, ML_Robot
- MC_MoveLinearRelative, ML_Robot
- MC_MoveDirectAbsolute, ML_Robot
- MC_MoveDirectRelative, ML_Robot
- MC_MoveCircularAbsolute, ML_Robot
- MC_MoveCircularRelative, ML_Robot
- ML_RobotMoveJumpAbsolute, ML_Robot
- ML_RobotOpenCyclicChannel, ML_Robot
- ML_RobotOpenDirectCyclicChannel, ML_Robot
- ML_RobotWriteCyclicChannel, ML_Robot

Nähere Informationen sind der entsprechenden Dokumentation zu entnehmen: Rexroth IndraMotion MLC 14VRS, Robot-Control V2, DOK-MLC***-ROCO****V14-RERS-DE-P, R911341586.

STRUCT MB_KINEMATICS_COORDINATED

Name	Datentyp	Initialwert	Kommentar
Point	BOOL	FALSE	Zielposition, absolut oder relativ (Einheit: z. B. in [Grad] je nach Parametrierung) (negativ oder positiv) Freigabe mit den Positionierbetriebsarten Hinweis: Wird permanent zyklisch verarbeitet (EnableCyclicScanning = TRUE)
Velocity	REAL	10.0	Positioniergeschwindigkeit, (Einheit: z. B. in [U/min] je nach Parametrierung) (positiv eingeben) Freigabe mit Positionierbetriebsarten
Acceleration	REAL	10.0	Beschleunigung, (Einheit: z. B. in [rad/s ²] je nach Parametrierung) (immer positiv) Freigabe mit Positionierbetriebsarten

ML_KinTech.library

Name	Datentyp	Initialwert	Kommentar
Deceleration	REAL	10.0	Verzögerung, (Einheit: z. B. in [rad/s ²] je nach Parametrierung) (immer positiv) Freigabe mit Positionierbetriebsarten
Jerk	REAL	0.0	Ruck, (Einheit: z. B. in [rad/s ³] je nach Parametrierung) (immer positiv) Freigabe mit Positionierbetriebsarten
CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	<i>Mögliche Koordinatensysteme (ML_Robot.library)</i> - CS_MACHINE_COORDINATES - CS_AXES_COORDINATES - CS_BASE_COORDINATES - CS_PRODUCT_COORDINATES_n (n = 1..16)
SlopeType	ML_SLOPE_TYPE	ST_BLOCK_SLOPE	Slope-Art der Bewegung (BlockSlope, ProgramSlope or ContinuousSlope) ML_Robot.library Freigabe mit Positionierbetriebsarten
SetMode	ML_SET_MODE	SM_TO	Definiert ob die Bewegung am Endpunkt endet oder überschiffen wird (to oder via) ML_Robot.library Freigabe mit Positionierbetriebsarten
BlendingRadius	REAL	0.0	Überschleifradius (Unit: [%] vom Parameter A-0-0230) Freigabe mit Positionierbetriebsarten
CircularAuxPoint	ARRAY [1..16] OF REAL	[16(0.0)]	Zielposition, zirkular (Einheit: z. B. in [Grad] je nach Parametrierung) (negativ oder positiv) Freigabe mit Positionierbetriebsart "Zirkular"
CircularCircMode	ML_CIRCMODES	CM_BOARDER	Definiert die Interpretation des Eingangs "Aux-Point" ML_Robot.library Freigabe mit Positionierbetriebsart "Zirkular"
CyclicMode	ML_CYCLIC_MODE	CYCLIC_MODE_POS_ABS	Wahl der zyklischen Sollwertvorgabe - Zyklische Sollwertvorgabe als absolute Position - Zyklische Sollwertvorgabe erfolgt als relatives Delta zur letzten kommandierten Position in jedem Takt - Zyklische Sollwertvorgabe erfolgt als Geschwindigkeit

Name	Datentyp	Initialwert	Kommentar
CyclicChannelType	ML_CYCLIC_CHANNEL_MODE_SUBTYPE	ML_CYCLIC_CHANNEL_MODE_LINEAR	Kanaltyp - Eingang des Funktionsbausteins ML_RobotWriteCyclicChanne - ML_CYCLIC_CHANNEL_MODE_LINEAR - ML_CYCLIC_CHANNEL_MODE_DIRECT
CyclicVelocity	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER(0.0)]	Zyklische Geschwindigkeitsvorgabe - Wertinterpretation abhängig von Cyclic-Mode - wird nur mit CYCLIC_MODE_VEL verwendet
StartHeight	REAL	0.0	Starthöhe des MoveJump ML_Robot.library Freigabe mit Positionierbetriebsart "Jump"
MaxHeight	REAL	0.0	Maximalhöhe des MoveJump ML_Robot.library Freigabe mit Positionierbetriebsart "Jump"
EndHeight	REAL	0.0	Endhöhe des MoveJump ML_Robot.library Freigabe mit Positionierbetriebsart "Jump"

- [Ordner Kin_Interface/Type/Control, Übersicht auf Seite 376](#)

ITF_KINEMATICS-Strukturen

Das eventgesteuerte Kinematik-Interface (ML_KinematicsInterfaceType11) nutzt sogenannte Interfaces zur Vorgabe der Betriebsarten und Sollwerte. Dies hat den Vorteil, dass das Schreiben von Werten auf die Control-Struktur eine sofortige Aktion auslöst, welche in den Get- und Set-Methoden der einzelnen Properties bzw. Strukturelemente hinterlegt ist. Der für das eventgesteuerte Kinematik-Interface angelegte MB_KINEMATICS_CONTROL_TYPE11 nutzt diese Interfaces. Der Aufbau dieser Struktur entspricht dem Aufbau des MB_KINEMATICS_INTERFACE_TYPE01, wie an folgender Grafik verdeutlicht.

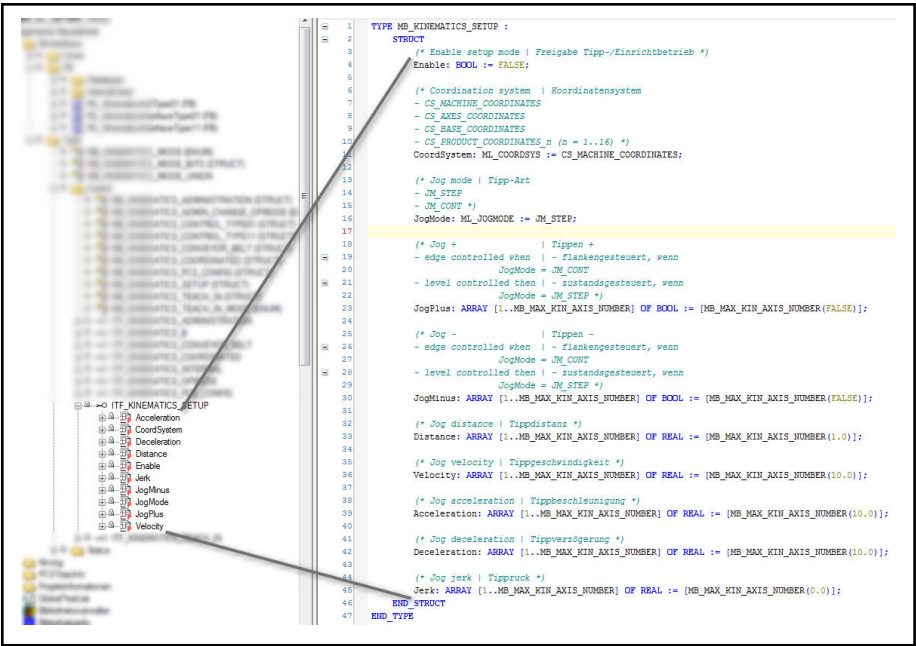


Abb. 7-5: Vergleich ITF_KINEMATICS_SETUP mit MB_KINEMATICS_SETUP
Aufgrund der identischen Namensgebung der Properties ändert sich an der Kommandierung des Kinematik-Interface zwischen Typ01 und Typ11 nichts. Bereits vorhandener Programmcode kann eins zu eins übernommen werden.

7.4.7 KinInterface/Type/Status
Ordner Kin_Interface/Type/Status, Übersicht

MB_KINEMATICS_STATUS_TYPE01 auf Seite 388		
	MB_KINEMATICS_ADMIN_STATUS auf Seite 389	
		MB_KINEMATICS_ADMIN_STATUS auf Seite 389
	MB_KINEMATICS_TEACH_IN_STATUS auf Seite 392	
		MB_KINEMATICS_PCS_STATUS auf Seite 392
	MB_KINEMATICS_DIAGNOSIS auf Seite 393	
	MB_KINEMATICS_SETUP_STATUS auf Seite 394	

Tab. 7-30: MB_KINEMATICS_STATUS_TYPE01, struktureller Aufbau

MB_KINEMATICS_STATUS_TYPE01

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Status

Tab. 7-31: Referenztable MB_KINEMATICS_STATUS_TYPE01

Kurzbeschreibung Auf die MB_KINEMATICS_STATUS_TYPE01-Struktur wird über die arKin-Status_gb[]-Instanz zugegriffen.

STRUCT MB_KINEMATICS_STATUS_TYPE01

Name	Datentyp	Initialwert	Kommentar
PtrKinData	POINTER TO ML_KINEMATICS_DATA_SM	0	Zeiger auf KinData-Schnittstelle der Kinematik (ML_Robot.library) - Eingabe .PtrKinData^ für Auswahlmenü
Admin	MB_KINEMATICS_ADMIN_STATUS		Kinematik-Status MB_KINEMATICS_ADMIN_STATUS, Seite 389
Diag	MB_KINEMATICS_DIAGNOSIS		Diagnose-Informationen der Kinematik - Fehler, Warnung, usw. MB_KINEMATICS_DIAGNOSIS, Seite 393
SetupMode	MB_KINEMATICS_SETUP_STATUS		Einrichtfunktionen MB_KINEMATICS_SETUP_STATUS, Seite 394

- [Ordner Kin_Interface/Type/Status, Übersicht, Seite 388](#)

MB_KINEMATICS_ADMIN_STATUS

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Status

Tab. 7-32: Referenztable MB_KINEMATICS_ADMIN_STATUS

Kurzbeschreibung

Die Struktur MB_KINEMATICS_ADMIN_STATUS ist eine Unterstruktur von [MB_KINEMATICS_STATUS_TYPE01](#) auf [Seite 388](#) und enthält den Ausgangsstatus für die Verwaltung der Kinematik.

STRUCT MB_KINEMATICS_ADMIN_STATUS

Name	Datentyp	Initialwert	Kommentar
_OpModeAck	MB_KINEMATICS_MODE	ModeCoordUngrouped	Dieses Element gibt den Status der Betriebsart aus
MODE_COORD_AB	BIT	FALSE	Diese Elemente dienen lediglich als Platzhalter und sind nur in der Control-Struktur zulässig
MODE_COORD_AH	BIT	FALSE	Diese Elemente dienen lediglich als Platzhalter und sind nur in der Control-Struktur zulässig
MODE_COORD_HOMING	BIT	FALSE	Diese Elemente geben den Status der Betriebsart als Bit aus. Ist keines der Bits gesetzt, so signalisiert dies den Status "ModeCoordUngrouped" AH, AxisInterface -> SetupMode, OpMode(Ack).2
MODE_COORD_POS_DIR_ABS	BIT	FALSE	Absolutes PTP Positionieren, AF, MC_MoveDirectAbsolute, OpMode(Ack).3
MODE_COORD_POS_DIR_REL	BIT	FALSE	Relatives PTP Positionieren, AF, MC_MoveDirectRelative, OpMode(Ack).4
MODE_COORD_POS_LIN_ABS	BIT	FALSE	Absolutes lineares Positionieren, AF, MC_MoveLinearAbsolute, OpMode(Ack).5
MODE_COORD_POS_LIN_REL	BIT	FALSE	Relatives lineares Positionieren, AF, MC_MoveLinearRelative, OpMode(Ack).6

ML_KinTech.library

Name	Datentyp	Initialwert	Kommentar
MODE_COORD_POS_CIRC_ABS	BIT	FALSE	Absolutes zirkulares Positionieren, AF, MC_MoveCircularAbsolute, OpMode(Ack).7
MODE_COORD_POS_CIRC_REL	BIT	FALSE	Relatives zirkulares Positionieren, AF, MC_MoveCircularRelative, OpMode(Ack).8
MODE_COORD_POS_JUMP_ABS	BIT	FALSE	Absolutes Jump Positionieren, AF, ML_RobotMoveJumpAbsolute, OpMode(Ack).9
MODE_COORD_STOPPING	BIT	FALSE	Kinematik Stopp, AH, MC_GroupStop, OpMode(Ack).10
MODE_COORD_INTERRUPTED	BIT	FALSE	Betriebsart "Interrupted", AF, MC_GroupInterrupt, OpMode(Ack).11
MODE_COORD_CONTINUE	BIT	FALSE	Betriebsart "Continue", AF, MC_GroupContinue, OpMode(Ack).12
MODE_COORD_EXTERNAL_FB	BIT	FALSE	Betriebsart "Externer Funktionsbaustein", Leistung ist zugeschaltet und dann übernimmt der externe FB die Steuerung, OpMode(Ack).13
MODE_COORD_CYCLIC	BIT	FALSE	Zyklische Positionssollwertvorgabe, AF, ML_RobotWriteCyclicChannel, OpMode(Ack).14
Active	BOOL	FALSE	TRUE: Kinematik aktiviert Hinweis: Wird bei der Initialisierung von FB ML_KinematicsInitType01 gesetzt
Name	STRING(40)	"	Name der Kinematik Hinweis: Wird bei der Initialisierung von FB ML_KinematicsInitType01 gesetzt
TrafoNb	DINT	999	Nummer des aktuellen Transformationsschemas Hinweis: Wird bei der Initialisierung von FB ML_KinematicsInitType01 gesetzt
CmdDone	BOOL	FALSE	Letztes Kommando ist abgeschlossen Gültig für alle Positionierbetriebsarten
InBuffer	BOOL	FALSE	Letztes Kommando ist im Satzpuffer Gültig für alle Positionierbetriebsarten
ParamLevelAck	BOOL	FALSE	Achsen befinden sich im Parametriermodus 1, Parameter A-0-0024, Bit 3 ist aktiv - TRUE: Parametriermodus aktiviert - FALSE: Parametriermodus deaktiviert
ConveyorBelt	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF MB_KINEMATICS_CONV_BELT_STATUS		Status der Konfiguration von Bändern
KinTransformPLCDone	BOOL	FALSE	Benutzerdefinierte Transformation wurde erfolgreich angemeldet

Name	Datentyp	Initialwert	Kommentar
TeachIn	MB_KINEMATICS_TEACH_IN_STATUS		Status des TeachIn und Speichern von Punkten MB_KINEMATICS_TEACH_IN_STATUS auf Seite 392
PCSStatus	ARRAY [1..ML_MAX_KIN_PCS_NUMBER] OF MB_KINEMATICS_PCS_STATUS		Status des konfigurierten Produktkoordinatensystems

- [Ordner Kin_Interface/Type/Status, Übersicht auf Seite 388](#)

MB_KINEMATICS_CONV_BELT_STATUS

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Status

Tab. 7-33: Referenztable MB_KINEMATICS_CONV_BELT_STATUS

Kurzbeschreibung Die Struktur MB_KINEMATICS_CONV_BELT_STATUS gibt die aktuelle Konfiguration eines Bandes wieder.

Die in dieser Struktur enthaltenen Elemente entsprechen den Ausgängen der folgenden Funktionsbausteine (ML_Robot.library):

- ML_RobotGetBeltConfiguration
- ML_RobotGetBeltErrorReaction

Nähere Informationen sind der entsprechenden Dokumentation zu entnehmen.

STRUCT MB_KINEMATICS_CONV_BELT_STATUS

Name	Datentyp	Initialwert	Kommentar
ConveyorBeltOrigin	ARRAY [1..6] OF REAL	[6(0.0)]	Gibt die Position und Verdrehung des Bandes relativ zum MCS zurück
TotalLength	REAL	0.0	Gibt die Länge des bandsynchronen Verfahrbereichs ab der Startposition zurück
BeginLength	REAL	0.0	Gibt die Länge des spätesten Starten einer bandsynchronen Bewegung ab der Startposition zurück
CoordType	ML_KIN_CS_TYPE	CS_TYPE_CARTESIAN	Gibt den Typ des Koordinatensystems zurück
ErrorReactionType	ML_BELT_ERROR_REACTION_MODE	ML_BELT_ERROR_NO_SPECIFIC_REACTION	<i>Gibt den Typ der Fehlerreaktion zurück:</i> • ML_BELT_ERROR_NO_SPECIFIC_REACTION • ML_BELT_ERROR_DESYNC_OVER_TIME • ML_BELT_ERROR_DESYNC_WITH_LIMITS

ML_KinTech.library

Name	Datentyp	Initialwert	Kommentar
ErrorReactionDesync-Time	TIME	T#300ms	Gibt die Zeit zum Absynchronisieren zurück
ErrorReactionDeceleration	REAL	0.0	Gibt die Bremsverzögerung zum Absynchronisieren zurück
ErrorReactionJerk	REAL	0.0	Gibt den Bremsruck zum Absynchronisieren zurück
Done	BOOL	FALSE	Konfiguration eines Transportbandes erfolgreich abgeschlossen

- [Ordner Kin_Interface/Type/Status, Übersicht auf Seite 388](#)

MB_KINEMATICS_TEACH_IN_STATUS

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Status

Tab. 7-34: Referenztabelle MB_KINEMATICS_SETUP_STATUS

Kurzbeschreibung

MB_KINEMATICS_TEACH_IN_STATUS repräsentiert die Statusstruktur zu MB_KINEMATICS_TEACH_IN.

STRUCT MB_KINEMATICS_TEACH_IN_STATUS

Name	Datentyp	Initialwert	Kommentar
TeachIn-ModeDone	BOOL	FALSE	Status des letzten Teach-Befehls • TRUE: Bearbeitung war erfolgreich • FALSE: Bearbeitung ist fehlgeschlagen oder wurde nicht gestartet
Point	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	[MB_MAX_KIN_AXIS_NUMBER(0.0)]	Zuletzt gelesener Punkt
CoordSystem	ML_COORDSYS	CS_MACHINE_COORDINATES	Koordinatensystem des zuletzt gelesenen Punktes
PointNameOut	STRING(19)	"	Name des nächsten Punktes

- [Ordner Kin_Interface/Type/Status, Übersicht auf Seite 388](#)

MB_KINEMATICS_PCS_STATUS

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Status

Tab. 7-35: Referenztabelle MB_KINEMATICS_PCS_STATUS

Kurzbeschreibung

Die MB_KINEMATICS_PCS_STATUS-Struktur ist eine Unterstruktur von MB_KINEMATICS_STATUS_TYPE01 auf Seite 388 und enthält Informationen zur Konfiguration des Produkt-Koordinatensystems (PCS) der Kinematik.

STRUCT MB_KINEMATICS_SETUP_STATUS

Name	Datentyp	Initialwert	Kommentar
InBuffer	BOOL	FALSE	Konfiguration eines PCS im Satzpuffer
Done	BOOL	FALSE	Konfiguration eines PCS erfolgreich abgeschlossen
Coord-Type	ML_KIN_CS_TYPE	CS_TYPE_CARTESIAN	Gibt Typ des konfigurierten PCS zurück: - Kartesisches Koordinatensystem - Zylindrisches Koordinatensystem
Origin	ARRAY [1..6] OF REAL	[6 (0)]	Gibt Position und Orientierung des PCS zurück: [TransX, TransY, TransZ, RotationX, RotationY, RotationZ]

- [Ordner Kin_Interface/Type/Status, Übersicht auf Seite 388](#)

MB_KINEMATICS_DIAGNOSIS

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Status

Tab. 7-36: Referenztabelle MB_KINEMATICS_DIAGNOSIS

Kurzbeschreibung

Die MB_KINEMATICS_DIAGNOSIS-Struktur ist eine Unterstruktur von [MB_KINEMATICS_STATUS_TYPE01 auf Seite 388](#) und enthält Warnungs- und Fehlerausgänge.

STRUCT MB_KINEMATICS_DIAGNOSIS

Name	Datentyp	Initialwert	Kommentar
Warning	BOOL	FALSE	Dieses Element stellt eine Kopie von <code>KinData[].Kin_Warning</code> oder von <code>arAxisStatus_gb[].Diag.Warning</code> dar und signalisiert, dass die Kinematik oder deren zugehörigen Achsen eine Warnung ausgibt
Error	BOOL	FALSE	Dieses Element signalisiert, dass die Kinematik, deren zugehörige Achsen oder ein interner Funktionsbaustein des Kinematik-Interfaces einen Fehler meldet
ErrorID	ERROR_CODE	NONE_ERROR	Dieses Element gibt den Fehlertyp der Kinematik, einer zugehörigen Achse oder eines internen Funktionsbausteins aus
ErrorIdent	ERROR_STRUCT	STRUCT (Table := NO_TABLE_USED, Additional1 := 0, Additional2 := 0)	Dieses Element gibt die Fehler-ID der Kinematik, einer zugehörigen Achse oder eines internen Funktionsbausteins aus
Number	DWORD	16#0	Dieses Element gibt die Diagnosenummer der Kinematik, einer zugehörigen Achse oder eines internen Funktionsbausteins aus
Message	STRING(60)	"	Dieses Element gibt die Diagnosemeldung der Kinematik, einer zugehörigen Achse oder eines internen Funktionsbausteins aus

Mit `arKinCtrl_gb[].Admin.DiagNbrRefreshTime` wird die Aktualisierungsrate des Elementes "Number" gesteuert.

Sobald sich das Element "Number" ändert, wird der Text "Message" neu gelesen.

Mit `arKinCtrl_gb[].Admin.DiagNbrRefreshTime` kann auch das Lesen der Diagnoseinformationen abgeschaltet werden (`DiagNbrRefreshTime := T#0ms`).

- [Ordner Kin_Interface/Type/Status, Übersicht auf Seite 388](#)

MB_KINEMATICS_SETUP_STATUS

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinInterface/Type/Status

Tab. 7-37: Referenztabelle MB_KINEMATICS_SETUP_STATUS

Kurzbeschreibung Die MB_KINEMATICS_SETUP_STATUS-Struktur ist eine Unterstruktur von [MB_KINEMATICS_STATUS_TYPE01 auf Seite 388](#) und enthält Informationen über das Vertippen der Kinematik.

STRUCT MB_KINEMATICS_SETUP_STATUS

Name	Datentyp	Initialwert	Kommentar
EnableAck	BOOL	FALSE	Signalisiert, ob der Einrichtungsbetrieb aktiv ist (= TRUE) oder nicht (= FALSE)

- [Ordner Kin_Interface/Type/Status, Übersicht auf Seite 388](#)

7.5 KinJog

7.5.1 KinJog/FB

ML_KinJog

Kurzbeschreibung Der Funktionsbaustein **ML_KinJog** dient dem Vertippen einzelner Achsen oder Achsgruppen einer Kinematik. Dieser Funktionsbaustein muss zyklisch aufgerufen werden.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinJog/FB

Tab. 7-38: Referenztabelle ML_KinJog

Schnittstellenbeschreibung

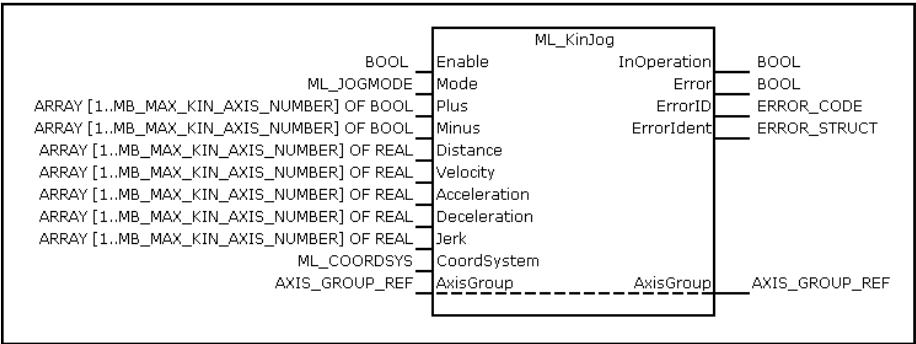


Abb. 7-6: Funktionsbaustein ML_KinJog

I/O-Typ	Name	Datentyp	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe des Bausteins
	Mode	ML_JOGMODE	Angabe der Tippart (inkrementell, kontinuierlich) ML_JOGMODE, Seite 397
	Plus	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF BOOL	Enthält Koordinatenelemente, die in positiver Richtung vertippt werden können
	Minus	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF BOOL	Enthält Koordinatenelemente, die in negativer Richtung vertippt werden können
	Distance	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	Angabe der Tipp-Distanz für jede Achse
	Velocity	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	Angabe der Tipp-Geschwindigkeit für jede Achse
	Acceleration	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	Angabe der Beschleunigung für den Tipp-Betrieb für jede Achse
	Deceleration	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	Angabe der Bremsverzögerung für den Tipp-Betrieb für jede Achse
	Jerk	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL	Angabe des Rucks für den Tipp-Betrieb für jede Achse
	CoordSystem	ML_COORDSYS	Angabe des Tipp-Koordinatensystems
VAR_OUTPUT	InOperation	BOOL	Der Baustein arbeitet an seiner eigentlichen Aufgabe
	Error	BOOL	Zeigt an, dass ein Fehler in der FB-Instanz aufgetreten ist
	ErrorID	ERROR_CODE	Kurzer Hinweis zur Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detaillierte Information zum Fehler
VAR_IN_OUT CONSTANT	AxisGroup	AXIS_GROUP_REF	Referenz zur Kinematik, welche vertippt werden soll

Tab. 7-39: Schnittstellenvariablen ML_KinJog

Name	Typ	Min-Wert	Max-Wert	Standardwert	Übernahme
Enable	BOOL			FALSE	kontinuierlich
Mode	ML_JOGMODE			JM_STEP	kontinuierlich
Plus	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF BOOL			16(FALSE)	kontinuierlich
Minus	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF BOOL			16(FALSE)	kontinuierlich
Distance	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL			16(0.0)	steigende Flanke an "Plus" bzw. "Minus"
Velocity	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL			16(0.0)	steigende Flanke an "Plus" bzw. "Minus"

ML_KinTech.library

Name	Typ	Min-Wert	Max-Wert	Standardwert	Übernahme
Acceleration	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL			16(0.0)	steigende Flanke an "Plus" bzw. "Minus"
Deceleration	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL			16(0.0)	steigende Flanke an "Plus" bzw. "Minus"
Jerk	ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL			16(0.0)	steigende Flanke an "Plus" bzw. "Minus"
CoordSystem	ML_COORDSYS			CS_MACHINE_COORDINATES	steigende Flanke an "Plus" bzw. "Minus"

Tab. 7-40: Eingangsverhalten ML_KinJog

Funktionsbeschreibung

Mithilfe des Funktionsbausteins ML_KinJog ist es möglich, eine Kinematik mit der Robot-Control V2 zu vertippen. Hierbei entspricht das Vertippen im kartesischen Koordinatenraum einer linearen Verfahrbewegung, wohingegen das Vertippen im Achskoordinatenraum einer PTP Verfahrbewegung gleicht. Nach dem erfolgreichen Start des Funktionsbausteins gruppiert dieser automatisch die zur Kinematik zugehörigen Achsen.

Mode Dieser Eingang gibt die Art des Vertippens an (inkrementell, kontinuierlich).

Der Eingang kann folgende Werte annehmen:

- **JM_STEP:** inkrementelles Tippen
- **JM_CONT:** kontinuierliches Tippen

Plus Dieser Eingang aktiviert die Verfahrbewegung in positive Richtung für die entsprechende Achse.

Minus Dieser Eingang aktiviert die Verfahrbewegung in negative Richtung für die entsprechende Achse.

Distance, Velocity, Acceleration, Deceleration, Jerk

Diese Eingänge definieren die Tipp-Distanz, Tipp-Geschwindigkeit, Tipp-Beschleunigung sowie Tipp-Ruck für inkrementelles und kontinuierliches Tippen. Die Werte werden in Abhängigkeit des gewählten Koordinatensystems als absolute (MCS, BCS, PCS) bzw. prozentuale Werte (ACS) interpretiert. Im Falle der prozentualen Interpretation beziehen sich die Werte auf die entsprechenden Grenzwerte der Achse.

CoordSystem Dieser Eingang gibt das Koordinatensystem an, in welchem vertippt werden soll.

Der Eingang kann folgende Werte annehmen:

- **CS_MACHINE_COORDINATES:** Maschinen-Koordinatensystem
- **CS_AXES_COORDINATES:** Achs-Koordinatensystem
- **CS_BASE_COORDINATES:** Basis-Koordinatensystem
- **CS_GRIPPER_COORDINATES:** Greifer-Koordinatensystem
- **CS_PRODUCT_COORDINATES_1:** Produkt-Koordinatensystem 1
- **CS_PRODUCT_COORDINATES_2:** Produkt-Koordinatensystem 2
- ...
- **CS_PRODUCT_COORDINATES_16:** Produkt-Koordinatensystem 16

Fehlerbehandlung

Der Funktionsbaustein erzeugt in DiagStatus.ErrorIdent.Additional1/Additional2 für die Tabelle F_RELATED_TABLE, 16#0170 die folgenden Fehlermeldungen:

ErrorID	Additional1	Additional2	Beschreibung
RESOURCE_ERROR, (16#0003)	16#26FF	16#0000	ungültige Robot-Control-Version

Tab. 7-41: Fehlercodes ML_KinJog

7.5.2 KinJog/Type

ML_JOGMODE

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek/Ordner
IndraMotion MLC 14VRS Funktionspaket ROB	ML_KinTech/KinJog/Type

Tab. 7-42: Referenztable ML_KinematicsInitType01

Kurzbeschreibung

Der Aufzählungstyp "ML_JOGMODE" enthält die zur Auswahl stehenden Arten zum Vertippen einer Kinematik.

Der Aufzählungstyp wird durch folgenden Funktionsbaustein verwendet:

- ML_KinJog

ENUM ML_JOGMODE

Name	Datentyp	Initialwert	Kommentar
JM_STEP	INT	0	Inkrementelles Tippen
JM_CONT	INT	1	Kontinuierliches Tippen

8 Anwendungsbeispiele

8.1 Anwendungsbeispiele, Übersicht

Im Folgenden soll anhand verschiedener Beispiele die Anwendung der Robot-Control gezeigt werden.

8.2 Einsatz von Koordinatensystemen

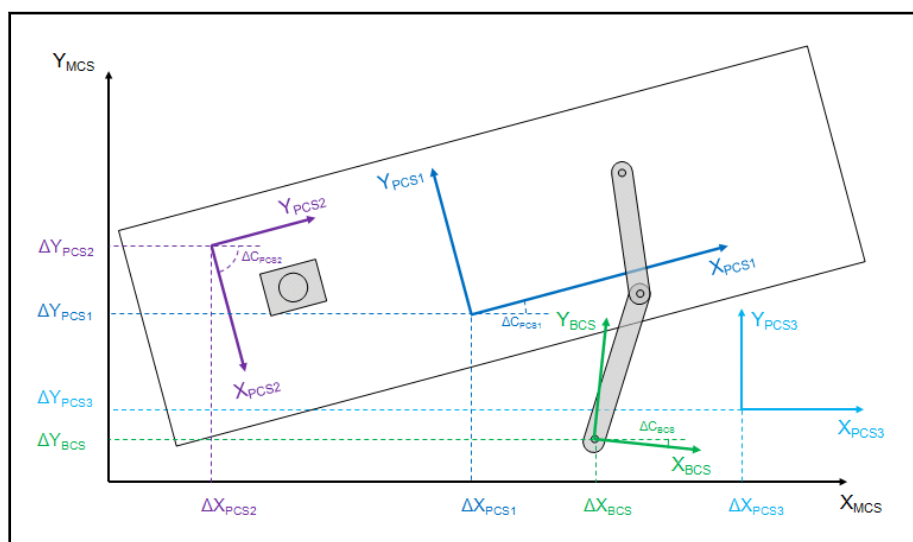


Abb. 8-1: Anwendungsbeispiel Koordinatensysteme

PCS1: Koordinatensystem des Bandes

Um ein Produkt synchron zu einem Band aufnehmen zu können, ist ein PCS in Bandrichtung definiert.

Die Verschiebung und Verdrehung des PCS1 zum MCS kann über [Set Belt Configuration auf Seite 38](#) mit $(\Delta X_{PCS1}, \Delta Y_{PCS1}, 0, 0, 0, \Delta C_{PCS1})$ definiert werden.

PCS2: Koordinatensystem der Kamera

Um die Position von Produkten einfach erfassen zu können, ist ein PCS als Koordinatensystem für die Kamera definiert.

Die Verschiebung und Verdrehung des PCS2 zum MCS kann über [Set Cartesian Transformation auf Seite 38](#) mit $(\Delta X_{PCS2}, \Delta Y_{PCS2}, 0, 0, 0, \Delta C_{PCS2})$ definiert werden.

PCS3: Koordinatensystem der Palette

Um die Produkte einfach auf eine Palette ablegen zu können, ist ein PCS als Koordinatensystem für die Palette definiert.

Die Verschiebung und Verdrehung des PCS3 zum MCS kann über [Set Cartesian Transformation auf Seite 38](#) mit $(\Delta X_{PCS3}, \Delta Y_{PCS3}, 0, 0, 0, 0)$ definiert werden.

BCS: Koordinatensystem des Roboters

Der Roboter ist als Scara (9) im Parameter [K-0-0031](#) definiert.

Die Verschiebung und Verdrehung des BCS zum MCS kann über den Parameter [K-0-0042](#) mit $(\Delta X_{BCS}, \Delta Y_{BCS}, 0, 0, 0, \Delta C_{BCS})$ definiert werden.

Ablauf

Die Kamera erfasst Produkte, die vom Band genommen und auf die Palette abgelegt werden sollen.

Die Kamera erfasst das Produkt und liefert die Position im PCS2.

Die Position im PCS2 (X_{PCS2}, Y_{PCS2}) wird für die bandsynchrone Bewegung mit dem Bandwert (B1) des Bandes zum Zeitpunkt der Kameraaufnahme versehen ($X_{PCS2}, Y_{PCS2}, B1$).

Anwendungsbeispiele

Diese Position wird mit "[Move Linear Absolute auf Seite 37](#)" kommandiert.

Es wird gewartet, bis das Produkt den bandsynchronen Arbeitsbereich erreicht hat. Danach wird die Position angefahren (inkl. Aufsynchronisieren auf das Band).

Ist das Produkt aufgenommen, kann eine Ablageposition auf der Palette direkt im PCS3 (X_{PCS3} , Y_{PCS3} , 0) mit "Move Linear Absolute" kommandiert werden.

Da dieser Punkt ohne Bandwert kommandiert wird, findet automatisch ein Absynchronisieren vom Band statt.

Die notwendigen Umrechnungen zwischen den Koordinatensystemen erfolgen automatisch.

8.3 Bandsynchronität

Ausgangssituation

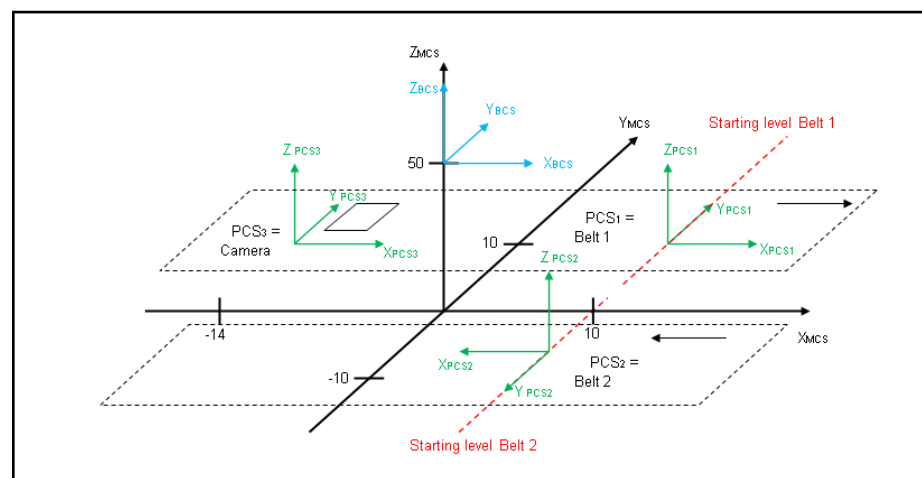


Abb. 8-2: Anwendungsbeispiel Bandsynchron

Eine Kamera erfasst Produkte, die von einem Band entnommen und auf ein zweites Band abgelegt werden sollen.

Die Kamera liefert die (X, Y)-Positionen des Produkts in einem eigenen Produkt-Koordinatensystem (PCS_3).

Die beiden Bänder stellen zwei weitere Produkt-Koordinatensysteme (PCS_1 und PCS_2) dar.

Die Verschiebung und Verdrehung des Kamera-Koordinatensystems (PCS_3) gegenüber dem **MCS** wird mit Hilfe des Kommandos [Set Cartesian Transformation auf Seite 38](#) $(-14, 10, 0, 0, 0, 0)_{PCS3}$ definiert.

Die ersten drei angegebenen Werte stehen hierbei für die Verschiebung des Koordinatensystems in X-, Y- und Z-Richtung, die letzten drei Werte stehen für die Verdrehung um die X-, Y- bzw. Z-Achse.

Weiterhin werden die beiden Bänder mit Hilfe des Kommandos [Set Belt Configuration auf Seite 38](#) $(10, 10, 0, 0, 0, 0)_{PCS1}$, $(10, -10, 0, 0, 0, 180)_{PCS2}$ konfiguriert und somit gegenüber dem **MCS** verdreht und verschoben.

Mit diesem Kommando wird ebenfalls die im Bild dargestellte Startebene festgelegt.

Erst wenn das erfasste Produkt diese Ebene passiert, befindet es sich im bandsynchronen Arbeitsbereich und kann aufsynchronisiert werden. Um Wartezeiten zu vermeiden, kann alternativ die adaptive Interpolation aktiviert

Anwendungsbeispiele

werden (siehe [Adaptive Motion auf Seite 74](#)). Die Startebenen der beiden Bänder müssen nicht zwingend identisch sein!

Für dieses Beispiel ist eine Delta-Kinematik über den Parameter [K-0-0031](#) = 20 definiert und mit dem Parameter [K-0-0042](#) (0, 0, 50, 0, 0, 0)_{BCS} in positive Z-Richtung verschoben worden.

Die von der Kamera gelieferte Position des Produktes (X_{PCS3}, Y_{PCS3}) muss mit dem aktuellen Bandwert zum Zeitpunkt der Kameraaufnahme versehen werden, um einen zeitlichen Bezug herstellen zu können.

Der aktuelle Bandwert wird durch den Parameter [K-0-0100](#) an der Position der jeweiligen Achse bereitgestellt.

Der so entstandene, zeitlich bezogene Punkt kann direkt mittels eines Bewegungskommandos (z. B. [Move Linear Absolute auf Seite 37](#)) angefahren werden.

Die MLC wartet in diesem Fall so lange, bis das Produkt die Startebene passiert und somit den bandsynchronen Arbeitsbereich erreicht hat. Zu diesem Zeitpunkt wird das Aufsynchronisieren gestartet und der Punkt angefahren.

Die notwendigen Berechnungen zur Ermittlung der aktuellen Lage des Produktes in Bezug zum Band wird automatisch durch die MLC durchgeführt und kann durch die Funktion [Move Point auf Seite 38](#) ausgelesen werden.

Die Kinematik bleibt so lange bandsynchron bis ein weiterer Bewegungsbefehl mit einem Bandwert von 0 (für absolute Bewegungen) bzw. 1.000.000 (für relative Bewegungen) angefahren wird.

In diesem Fall findet automatisch ein Absynchronisieren vom Band statt.

Berechnungsbeispiel

Die allgemeine Abbildungsvorschrift zur Bandsynchronisation lautet:

$$B_{act} - B_{product} + X_{PCS1} > 0$$

B_{act} aktueller Bandwert

$B_{Product}$ Bandwert zum Zeitpunkt der Kameraaufnahme

X_{PCS1} X-Position des Produktes im PCS₁

Abb. 8-3: Abbildungsvorschrift zur Bandsynchronisation

Die Delta-Kinematik sei mit 3 Positionierachsen, 1 Orientierungsachse und 2 Bandachsen konfiguriert.

Die Kamera hat beispielsweise ein Produkt mit der Position (10, 10, 0, 0, **35**, 0)_{PCS3} aufgenommen.

Aus der Sicht des Bandes, d. h. des Produkt-Koordinatensystems 1, befindet sich das Produkt an Position (-**14**, 10, 0, 0, 35, 0)_{PCS1}.

Zu einem weiteren Berechnungszeitpunkt besitzt das Band beispielsweise einen aktuellen Bandwert von **40**, wodurch die Vorschrift wie folgt lautet:

$$40 - 35 + (-14) = -9 < 0$$

Dies bedeutet, dass die Kinematik noch 10 Bandwerte wartet, bis sie sich auf den Punkt aufsynchronisiert.

Nach diesen 10 Bandwerten ist die Abbildungsvorschrift erfüllt, und das Produkt hat die Startebene durchlaufen.

8.4 Benutzerdefinierte Transformation

Dieses Beispiel implementiert die 2-Achs-Delta Slide-Transformation (Transformationsschema 45) als benutzerdefinierte Transformation.

Anwendungsbeispiele

Dies dient nur als Beispiel: Sollte man das Transformationsschema 45 benötigen, kann es natürlich direkt verwendet werden, ohne das Transformationschema selbst in der SPS zu implementieren.

Als Konfiguration gelten die gleichen Randbedingungen wie in [2-Achs-Delta Slide auf Seite 156](#) angegeben sind.

Implementieren des Funktionsbausteins

Als erstes muss ein Funktionsbaustein implementiert werden, der das Interface MC_KIN_REF implementiert.

Deklaration

```
FUNCTION_BLOCK DeltaSlide IMPLEMENTS MC_KIN_REF
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
    L1: REAL := 0;
    L2: REAL := 0;
    Gamma: REAL := 0;
    Ymin, Ymax: REAL := 0;
    GRAD_RAD_FAKTOR : REAL := 0.01745329251994329; // (2*PI)/360
END_VAR
```

In diesem Beispiel werden einige Variablen deklariert, auf die drei Interface-Methoden zugreifen.

Implementieren der Initialisierungsmethode

Die Initialisierungsmethode führt folgende Prüfungen und Berechnung durch:

- Prüfung der Längen 1 und 2 im Parameter "[K-0-0040](#), Transformation, Längen der Mechanik"
- Prüfung des Faktors 1 im Parameter "[K-0-0041](#), Transformation, Faktoren"
- Prüfung, ob mindestens 2 Positionierachsen im Parameter "[K-0-0008](#), Achskonfigurationsliste" konfiguriert wurden
- Prüfung, ob die beiden Positionierachsen translatorisch absolut gewichtet sind
- Berechnung des minimalen und maximalen Y-Wertes im Basis-Koordinatensystem

Initialisierung

```
METHOD Initialisation : INT
VAR_INPUT
    AxesConfigData : REFERENCE TO ARRAY [1..16] OF ML_KIN_AXES_CONFIG_DATA;
    TransformationLengths : REFERENCE TO ARRAY [1..16] OF REAL;
    TransformationFactors : REFERENCE TO ARRAY [1..16] OF REAL;
END_VAR

VAR
    Axis1_A_0_0021: DWORD;
    Axis2_A_0_0021: DWORD;
END_VAR

// arm lengths ok?
IF (TransformationLengths[1] > 0.0) THEN
    L1 := TransformationLengths[1];
ELSE
    Initialisation := 1;
    RETURN;
END_IF

IF (TransformationLengths[2] >= 0.0) THEN
    L2 := TransformationLengths[2];
ELSE
    Initialisation := 2;
    RETURN;
END_IF
```


Anwendungsbeispiele

```
IF ((TransformationFactors[1] >= 0.0) AND
    (TransformationFactors[1] < 360.0)) THEN
    Gamma := TransformationFactors[1];
ELSE
    Initialisation := 3;
    RETURN;
END_IF

IF ((AxesConfigData[1].Configuration <> ML_KIN_AXIS_TYPE_POSITION) OR
    (AxesConfigData[2].Configuration <> ML_KIN_AXIS_TYPE_POSITION)) THEN
    Initialisation := 4;
    RETURN;
END_IF

Axis1_A_0_0021 := DV_Axis[AxesConfigData[1].AxisNumber].A_0_0021;
Axis2_A_0_0021 := DV_Axis[AxesConfigData[1].AxisNumber].A_0_0021;

//Axes must be translatory
IF (Axis1_A_0_0021.29 = TRUE) THEN
    Initialisation := 5;
    RETURN;
END_IF
IF (Axis2_A_0_0021.29 = TRUE) THEN
    Initialisation := 5;
    RETURN;
END_IF

//Axes must be absolute
IF (Axis1_A_0_0021.28 = TRUE) THEN
    Initialisation := 5;
    RETURN;
END_IF
IF (Axis2_A_0_0021.28 = TRUE) THEN
    Initialisation := 5;
    RETURN;
END_IF

Ymin := - L1 - (COS((180 - Gamma)* GRAD_RAD_FAKTOR)*L2);
Ymax := SIN((180 - Gamma) * GRAD_RAD_FAKTOR)*L2;

Initialisation := 0;
```

Implementieren der Vorwärts-Transformation

Die Vorwärts-Transformation berechnet aus Koordinaten im Achs-Koordinatensystem Koordinaten im Basis-Koordinatensystem.

Zu beachten sind die Fehlerfälle, wenn die Achsen zu weit auseinander stehen oder verkehrt herum stehen.

Vorwärts-Transformation

```
METHOD Forward : INT
VAR_INPUT
    OriginalCoordinates : REFERENCE TO ARRAY [1..16] OF REAL;
    JointCoordinates    : REFERENCE TO ARRAY [1..16] OF REAL;
END_VAR
VAR
    Alpha : REAL;      //d_Angle_AC_X
    Beta  : REAL;      //d_Angle_CTCP_X
    Gamma_Temp : REAL; //d_Angle_L1_L2_Temp;
    Cx      : REAL;
    Cy      : REAL;
END_VAR

IF (JointCoordinates[2] - JointCoordinates[1]) < 0.0 THEN
    Forward := 1; // arms wrong
    RETURN;
END_IF

Cx := (JointCoordinates[1] + JointCoordinates[2]) / 2;

IF (L1*L1 - (Cx - JointCoordinates[1])*(Cx - JointCoordinates[1])) <= 0.0 THEN
    Forward := 2; //arms too short
    RETURN;
END_IF

Cy := - SQRT(L1*L1 - (Cx - JointCoordinates[1])*(Cx - JointCoordinates[1]));
Alpha := ACOS((Cx - JointCoordinates[1])/L1);

Gamma_Temp := 180 - Gamma; (* angle correction *)
Beta := Alpha - (Gamma_Temp*GRAD_RAD_FAKTOR);
```

Anwendungsbeispiele

```
//make new base coordinates
OriginalCoordinates[1] := Cx + (COS(Beta)*L2);
OriginalCoordinates[2] := Cy - (SIN(Beta)*L2);

Forward := 0;
```

Implementieren der Rückwärts-Transformation

Die Rückwärts-Transformation berechnet aus Koordinaten im Basis-Koordinatensystem Koordinaten im Achs-Koordinatensystem.

Zu beachten sind die Fehlerfälle, wenn der Punkt nicht erreicht werden kann.

Rückwärts-Transformation

```
METHOD Inverse : INT
VAR_INPUT
    _OriginalCoordinates : REFERENCE TO ARRAY [1..16] OF REAL;
    _JointCoordinates    : REFERENCE TO ARRAY [1..16] OF REAL;
END_VAR
VAR
    ATCP, AB : REAL;
    Beta : REAL; //d_Angle_ATCP_X
    Delta : REAL; //d_Angle_AC_ATCP
    Alpha : REAL; //d_Angle_AC_X
    Ax, Bx : REAL;
    TCPx, TCPy : REAL;
    GRAD_RAD_FAKTOR : REAL := 0.01745329251994329; // (2*PI)/360
END_VAR

IF (Ymin > OriginalCoordinates[2] OR Ymax <= OriginalCoordinates[2]) THEN
    Inverse := 1; // unreachable
    RETURN;
END_IF

TCPx := OriginalCoordinates[1];
TCPy := OriginalCoordinates[2];

ATCP := SQRT((L1*L1) + (L2*L2) - (2 * L1 * L2 * COS(Gamma* GRAD_RAD_FAKTOR))); // Law of cosines
Ax := TCPx - SQRT(ATCP*ATCP - TCPy*TCPy);

Beta := ATAN2(-TCPy, (TCPx-Ax));
Delta := ACOS(-(L2*L2) - (L1*L1) - (ATCP * ATCP)) / (2 * L1 * ATCP); // Law of cosines
Alpha := Beta + Delta;
AB := 2 * L1 * COS(Alpha);

Bx := Ax + AB;

_JointCoordinates[1] := Ax;
_JointCoordinates[2] := Bx;

Inverse := 0;
```

Anmelden der Transformation an eine Kinematik

Es wird angenommen, dass eine Kinematik mit dem Namen "K1" existiert. Für diese Kinematik soll die Transformation "DeltaSlide" angemeldet werden.

Zu beachten ist, dass die **Instanz** myDeltaSlide an der Kinematik angemeldet wird und nicht der Funktionsbausteintyp DeltaSlide.

Auch für jede weitere Kinematik ist immer eine eigene **Instanz** des Funktionsbaustein DeltaSlide zu verwenden.

Verwendung

```
PROGRAM PlcProg
VAR
    myDeltaSlide : DeltaSlide;
    setKinTrafo : MC_SetKinTransform;
    initTransformation : BOOL;
END_VAR

IF(initTransformation) THEN
    setKinTrafo.KinTransform := myDeltaSlide;
    setKinTrafo.Execute := TRUE;
    setKinTrafo(AxisGroup := K1);
    IF(setKinTrafo.Done) THEN
        initTransformation := FALSE;
        setKinTrafo.Execute := FALSE;
        setKinTrafo(AxisGroup := K1);
    END_IF
END_IF
```

```
END_IF
IF(SetKinTrafo.Error) THEN
    initTransformation := FALSE;
END_IF
END_IF
```

8.5 Erweiterte benutzerdefinierte Transformation

Dieses Beispiel implementiert die 3-Achs-Scara-Transformation (Transformationsschema 9) als erweiterte benutzerdefinierte Transformation.

Dies dient nur als Beispiel: Sollte man das Transformationsschema 9 benötigen, kann es natürlich direkt verwendet werden, ohne das Transformationsschema selbst in der SPS zu implementieren.

Als Konfiguration gelten die gleichen Randbedingungen wie sie in [3-Achs-Scara mit optional 1 Orientierungsachse auf Seite 119](#) angegeben sind.

Implementieren des Funktionsbausteins

Als erstes muss ein Funktionsbaustein implementiert werden, der das Interface MC_KIN_REF_EXT implementiert.

Deklaration

```
FUNCTION_BLOCK Scara IMPLEMENTS MC_KIN_REF_EXT
VAR_INPUT
END_VAR
VAR_OUTPUT
END_VAR
VAR
    c_boLeftArm: BOOL := FALSE;
    dL1 : LREAL := 0.0;
    dL2 : LREAL := 0.0;
    FLT_EPSILON: LREAL := 1.192092896E-07;
    GRAD_RAD_FAKTOR: LREAL := 0.01745329251994329; (* (2*PI)/360 *)
    RAD_GRAD_FAKTOR: LREAL := (1.0/GRAD_RAD_FAKTOR);
    PI_GANZ : LREAL := 3.1415926535897928;
END_VAR
```

Implementieren der GetUsedAxes-Methode

In diesem Beispiel werden einige Variablen deklariert, auf die die Interface-Methoden zugreifen.

Die GetUsedAxes-Methode gibt die Freiheitsgrade der Transformation zurück. In unserem Fall hat die Transformation ihre Freiheitsgrade in der X-Y-Ebene. Die zusätzliche Orientierungsachse (Verdrehung um C) wird nicht als Orientierungsachse angegeben, sondern als Hilfsachse. Da diese Achse immer definiert ist, muss sie hier nicht mit angegeben werden.

GetUsedAxes-Methode

```
METHOD GetUsedAxes : UINT
GetUsedAxes := ML_KIN_USED_AXES.X_AXIS OR ML_KIN_USED_AXES.Y_AXIS;
```

Implementieren der Initialisierungsmethode

Die Initialisierungsmethode führt folgende Prüfungen und Berechnung durch:

- Prüfung der Längen 1 und 2 im Parameter "[K-0-0040](#), Transformation, Längen der Mechanik"
- Prüfung, ob mindestens 2 Positionierachsen im Parameter "[K-0-0008](#), Achskonfigurationsliste" konfiguriert wurden
- Prüfung, ob die beiden Positionierachsen rotatorisch absolut gewichtet sind

Initialisierung

```
METHOD Initialisation : UDINT
VAR_INPUT
    AxesConfigData : REFERENCE TO ARRAY [1..16] OF ML_KIN_AXES_CONFIG_DATA;
    TransformationLengths : REFERENCE TO ARRAY [1..16] OF LREAL;
    TransformationFactors : REFERENCE TO ARRAY [1..16] OF LREAL;
END_VAR
VAR
```

Anwendungsbeispiele

```

    axisScalingA1: DWORD;
    axisScalingA2: DWORD;
END_VAR

IF (TransformationLengths[1] > 0.0) THEN
    dL1 := TransformationLengths[1];
ELSE
    Initialisation := 16#F12C008A; // trafo values wrong
    RETURN;
END_IF

IF (TransformationLengths[2] > 0.0) THEN
    dL2 := TransformationLengths[2];
ELSE
    Initialisation := 16#F12C008A; // trafo values wrong
    RETURN;
END_IF

IF (AxesConfigData[1].Configuration <> ML_KIN_AXIS_CONFIG.ML_KIN_AXIS_TYPE_POSITION) OR
    (AxesConfigData[2].Configuration <> ML_KIN_AXIS_CONFIG.ML_KIN_AXIS_TYPE_POSITION) THEN
    Initialisation := 16#F22B0025; // too less axes
    RETURN;
END_IF

axisScalingA1 := DV_Axis[AxesConfigData[1].AxisNumber].A_0_0021;
axisScalingA2 := DV_Axis[AxesConfigData[2].AxisNumber].A_0_0021;

IF axisScalingA1.28 OR NOT(axisScalingA1.29) OR // axis 1 must be absolute rotatory
    axisScalingA2.28 OR NOT(axisScalingA2.29) THEN // axis 2 must be absolute rotatory
    Initialisation := 16#F22B0024; // axis has wrong scaling
    RETURN;
END_IF

```

Implementieren der DetermineMemory-Methode

Die DetermineMemory-Methode bestimmt die Armigkeit des Scara-Roboters. Aus den Achskoordinaten wird errechnet, ob die Kinematik rechtsarmig oder linksarmig ist. Der Eingang "TrafoChoise" wird ignoriert. Wenn der Wert "TRUE" ist, könnte man z. B. die Armigkeit beim nächsten Rückwärtstrafo verwenden, bei dem sich die erste Achse am wenigsten bewegt. Dadurch könnte man zeitlich kürzere Bewegungen erreichen, wenn man davon ausgeht, dass Achse 2 dynamischer ist als Achse 1. Dies ist hier jedoch nicht implementiert.

Determine Memory-Methode

```

METHOD DetermineMemory : UDINT
VAR_INPUT
    ACS      : REFERENCE TO ARRAY [1..16] OF LREAL;
    TrafoChoise : BOOL;
END_VAR

IF( ACS[2] > FLT_EPSILON ) THEN
    c_boLeftArm := FALSE;
END_IF

IF( ACS[2] < -FLT_EPSILON ) THEN
    c_boLeftArm := TRUE;
END_IF

```

Implementieren der Vorwärts-Transformation

Die Vorwärts-Transformation berechnet aus Koordinaten im Achs-Koordinatensystem Koordinaten im Basis-Koordinatensystem.

Zu beachten sind die Fehlerfälle und dass die Hilfsachse nicht kopiert werden muss. Dies wird vom System selbständig durchgeführt!

Vorwärts-Transformation

```

METHOD Forward : UDINT
VAR_INPUT
    BCS      : REFERENCE TO ARRAY [1..16] OF LREAL;
    ACS      : REFERENCE TO ARRAY [1..16] OF LREAL;
END_VAR

VAR
    dPHI1 : LREAL;
    dPHI2 : LREAL;
    dPHI11 : LREAL;
    dPHI12 : LREAL;

```

```

END_VAR

dPHI1 := ACS[1];
dPHI2 := ACS[2];

IF ( (ABS(dPHI1) > 179.0)
    OR (ABS(dPHI2) > 179.0) ) THEN
  Forward := 16#F22B0021; // Angles out of range
  RETURN;
END_IF

dPHI11 := dPHI1 * GRAD_RAD_FAKTOR;
dPHI12 := ( dPHI2 * GRAD_RAD_FAKTOR ) + dPHI11;

BCS[1] := ( dL1 * COS( dPHI11 ) ) + ( dL2 * COS( dPHI12 ) );
BCS[2] := ( dL1 * SIN( dPHI11 ) ) + ( dL2 * SIN( dPHI12 ) );

```

Implementieren der Rückwärts-Transformation

Die Rückwärts-Transformation berechnet aus Koordinaten im Basis-Koordinatensystem Koordinaten im Achs-Koordinatensystem.

Zu beachten sind die Fehlerfälle und dass die Hilfsachse nicht kopiert werden muss. Dies wird vom System selbständig durchgeführt!

Die Hilfsfunktion $\text{atan2}(y, x)$ ist im System nicht definiert und muss selbst nach der Definition des atan2 implementiert werden (ein Beispiel hierfür kann dem anschließenden Abschnitt entnommen werden).

Rückwärts-Transformation

```

METHOD Inverse : UDINT
VAR_INPUT
  BCS : REFERENCE TO ARRAY [1..16] OF LREAL;
  ACS : REFERENCE TO ARRAY [1..16] OF LREAL;
END_VAR
VAR
  dLengthTCP : LREAL; //length of the distance from the TCP to the origin
  dGamma : LREAL; //angle of the triangle; leads to phi2
  dBeta : LREAL; //angle of the triangle; leads to phi1
  dTheta : LREAL; //pitch angle of the distance from the TCP to the origin; leads to phi1
  dTemp : LREAL; //temporary variable
  dPHI1 : LREAL; //angle of the motor in degrees
  dPHI2 : LREAL; //angle of the motor in degrees
  dphi11 : LREAL; //angle of the motor in radian; d_phi12 = d_phi1 + d_phi2
  dphi21 : LREAL; //angle of the motor in radian; d_phi12 = d_phi1 + d_phi2

END_VAR

(*****
  Get Length of TCP-Origin
  *****)
dLengthTCP := SQRT(BCS[1]*BCS[1] + BCS[2]*BCS[2]);
IF ( (dLengthTCP - FLT_EPSILON) > (dL1 + dL2) ) THEN
  Inverse := 16#F22B0022; // too far away
  RETURN;
END_IF

IF ( dLengthTCP > (dL1 + dL2) ) THEN
  dLengthTCP := dL1 + dL2;
END

(*****
  Calc d_Gamma
  *****)
dTemp := ( (dL1 * dL1) + (dL2 * dL2) - (dLengthTCP * dLengthTCP) )
        / ( 2.0 * dL1 * dL2 );
IF ( ABS(dTemp) > 1.0 ) THEN
  Inverse := 16#F12C0479; // singular position: acos is not defined for |x| > 1, catch possible failure
  RETURN;
END_IF
dGamma := ACOS(dTemp);

(*****
  Calc d_Beta
  *****)
IF (dLengthTCP < FLT_EPSILON) THEN
  Inverse := 16#F12C0479; // to near to base point
  RETURN;
END_IF

```

Anwendungsbeispiele

```

dTemp := ( (dL1 * dL1) + (dLengthTCP * dLengthTCP) - (dL2 * dL2) )
         / ( 2.0 * dL1 * dLengthTCP );
IF ( ABS(dTemp) > 1.0 ) THEN
    inverse := 16#F12C0479; // singular position: acos is not defined for |x| > 1, catch possible failure
    RETURN;
END IF
dBeta := ACOS(dTemp);

(*****
  Calc d_Theta
  *****)
dTheta := atan2( BCS[2], BCS[1] );

(*****
  Set the angles considering to the arm
  *****)
IF ( c_boLeftArm ) THEN
    //left arm
    dphi1 := dBeta + dTheta;
    IF ( dphi1 > PI_GANZ ) THEN
        dphi1 := dphi1 - ( 2.0 * PI_GANZ );
    END_IF

    dphi2 := dGamma - PI_GANZ;
    IF(dphi2 > 0.0) THEN //prevent the change of the arm
        dphi2 := -FLT_EPSILON; //may occur of inaccuracy of the calculation
    END_IF
ELSE
    //right arm
    dphi1 := dTheta - dBeta;
    IF ( dphi1 < -PI_GANZ ) THEN
        dphi1 := dphi1 + ( 2.0 * PI_GANZ );
    END_IF

    dphi2 := PI_GANZ - dGamma;
    IF(dphi2 < 0.0) THEN //prevent the change of the arm
        dphi2 := +FLT_EPSILON; //may occur of inaccuracy of the calculation
    END_IF
END_IF

(*****
  Transform angles to Grad
  *****)
dPHI1 := dphi1 * RAD_GRAD_FAKTOR;
dPHI2 := dphi2 * RAD_GRAD_FAKTOR;

(*****
  Check angles
  *****)
IF ( (ABS(dPHI1) > 179.0)
    OR (ABS(dPHI2) > 179.0) ) THEN
    inverse := 16#F22B0021; // Angles out of range
    RETURN;
END_IF

ACS[1] := dPHI1;
ACS[2] := dPHI2;

```

**Beispielimplementierung der
atan2-Funktion**

Im folgenden ist eine beispielhafte atan2-Funktion gezeigt, welche für die Rückwärts-Transformation genutzt werden kann.

atan2-Funktion

```

FUNCTION ATAN2 : REAL
VAR_INPUT
    y : REAL;
    x : REAL;

END_VAR
VAR
END_VAR
VAR_CONSTANT
    lrPI: LREAL := 3.14159265359;
END_VAR

IF x > 0 THEN
    ATAN2 := ATAN(y / x);
END_IF

IF x < 0 THEN

```

```
IF y >= 0 THEN
  ATAN2 := lrPI + ATAN(y / x);
ELSE
  ATAN2 := -lrPI + ATAN(y / x);
END_IF
END_IF

IF x = 0 THEN
  ATAN2 := y / ABS(y) * (lrPI / 2);
END_IF
```

GetMechanic-Methode

Die GetMechanic-Methode soll zu einem Punkt im Achskoordinatensystem die dazugehörigen mechanischen Daten berechnen. Beim Scara soll die mechanische Ansicht aus 3 Kugeln (BASE, MID, TCP) und 2 Zylindern (BASE zu MID, MID zu TCP) bestehen. Hier ist zu beachten, dass der Objektname "TCP" von den meisten Visualisierungen zum Aufzeichnen des Pfades verwendet wird. Für die Größe der Stäbe und Kugeln wird die Transformationslänge 2 ([K-0-0040](#)) herangezogen. Zur Vereinfachung wird die Funktion ML_AddMechanicData aus der ML_Robot-Bibliothek verwendet.

GetMechanic-Methode

```
METHOD GetMechanic : UDINT
VAR_INPUT
  _ACS      : REFERENCE TO ARRAY [1..16] OF LREAL;
  MechanicData : REFERENCE TO ML_KIN_MECHANIC_DATA;
  MechanicDataSize : UDINT;
  MechanicDataRetSize : REFERENCE TO UDINT;
  UpdateOnly : BOOL;
END_VAR
VAR
  dphil1 : LREAL; //angle of the motor in radian
  dphil2 : LREAL; //angle of the motor in radian; d_phil2 = dphil1 + dphil2
  TCPx : LREAL; //TCP point
  TCPy : LREAL; //TCP point
  MIDx : LREAL; //MID point
  MIDy : LREAL; //MID point
  MID2TCPx : LREAL;
  MID2TCPy : LREAL;

  fRodRadius : LREAL := dL2 / 100.0;
  fSphereSize : LREAL := fRodRadius * 4.0;
  xMID2TCP : INT;
END_VAR

(*****
compute angles
*****
)
dphil1 := ACS[1] * GRAD_RAD_FAKTOR;
dphil2 := ( ACS[2] * GRAD_RAD_FAKTOR ) + dphil1;

(*****
Get the MID
*****
)
MIDx := dL1 * COS( dphil1 );
MIDy := dL1 * SIN( dphil1 );

(*****
Get the TCP
*****
)
TCPx := ( dL1 * COS( dphil1 ) ) + ( dL2 * COS( dphil2 ) );
TCPy := ( dL1 * SIN( dphil1 ) ) + ( dL2 * SIN( dphil2 ) );

MID2TCPx := TCPx - MIDx;
MID2TCPy := TCPy - MIDy;

IF NOT(UpdateOnly) THEN
  ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
    KIN_MECH_CMD_CREATE_SPHERE, 'BASE', 0.0, 0.0, 0.0);
  ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
    KIN_MECH_CMD_CREATE_SPHERE, 'MID', 0.0, 0.0, 0.0);
  ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
    KIN_MECH_CMD_CREATE_SPHERE, 'TCP', 0.0, 0.0, 0.0);
  ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
    KIN_MECH_CMD_CREATE_ROD, 'ROD1', 0.0, 0.0, 0.0);
  ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
    KIN_MECH_CMD_CREATE_ROD, 'ROD2', 0.0, 0.0, 0.0);
  ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
```

Anwendungsbeispiele

```

        KIN_MECH_CMD_SIZE, 'BASE', fSphereSize, fSphereSize, fSphereSize);
ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_SIZE, 'MID', fSphereSize, fSphereSize, fSphereSize);
ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_SIZE, 'TCP', fSphereSize, fSphereSize, fSphereSize);
ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_SIZE, 'ROD1', fRodRadius, fRodRadius, dL1);
ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_SIZE, 'ROD2', fRodRadius, fRodRadius, dL2);
END_IF

ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_POSITION, 'TCP', TCPx, TCPy, 0.0);
ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_POSITION, 'MID', MIDx, MIDy, 0.0);
ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_POSITION, 'ROD2', MIDx, MIDy, 0.0);
ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_DIRECTION, 'ROD1', MIDx, MIDy, 0.0);
ML_AddMechanicData(MechanicData, MechanicDataSize, MechanicDataRetSize,
        KIN_MECH_CMD_DIRECTION, 'ROD2', MID2TCPx, MID2TCPy, 0.0);

```

Anmelden der Transformation an eine Kinematik

Es wird angenommen, dass eine Kinematik mit dem Namen "K1" existiert. Für diese Kinematik soll die Transformation "Scara" angemeldet werden.

Zu beachten ist, dass die **Instanz** myScara an der Kinematik angemeldet wird und nicht der Funktionsbaustein Typ Scara.

Auch für jede weitere Kinematik ist immer eine eigene **Instanz** des Funktionsbaustein Scara zu verwenden.

Verwendung

```

PROGRAM PLCTrafoExt2
VAR
    myScara                : Scara;
    setKinTrafo             : MC_SetKinTransformExt;
    initTransformation      : BOOL;
END_VAR

IF(initTransformation) THEN
    setKinTrafo.KinTransform := myScara;
    setKinTrafo.Execute      := TRUE;
    setKinTrafo.AxisGroup   := K1;
    IF(setKinTrafo.Done) THEN
        initTransformation := FALSE;
        setKinTrafo.Execute := FALSE;
        setKinTrafo.AxisGroup := K1;
    END_IF
    IF(setKinTrafo.Error) THEN
        initTransformation := FALSE;
    END_IF
END_IF

```

8.6 Erweiterung eines Punkt

Sinnvoll wird das Zuschalten der Positionierungsachsen, wenn eine FlexStamper-Kinematik in Verbindung mit der Bewegungsart Jump verwendet werden soll. Die Bewegungsart Jump kann nicht verwendet werden, da die FlexStamper Transformation nur in der X-Y-Ebene liegt und eine Jump-Bewegung nur in Z-Richtung kommandiert werden kann.

Durch Verdrehung des Roboters um die X-Achse (A) um 90° im Parameter [K-0-0042](#) wird der Roboter in den X-Z-Raum gebracht. Dadurch kann zwar der MoveJump verwendet werden, aber die Z-Koordinate des Endpunkt nicht kommandiert werden, da nur die X und Y Koordinaten durch die Transformation definiert sind. Nach Setzen des Bit 7 des Parameter [K-0-0099](#) wird die Kommandierung aller Positionierungsachsen freigeschaltet.

Transformationsschema	Punkt	Zuschalten rungsachsen	Positionie-
FlexStamper (39)	(X, Y, H1)	(X, Y, Z, H1)	

8.7 Bandsynchronität mit "Adaptive Motion"

Im folgenden Beispiel wird schematisch dargestellt, wie ein bandsynchroner Punkt P mit und ohne "Adaptive Motion" angefahren wird.

Die Koordinaten des kommandierten Punkts P zum Zeitpunkt t_0 sind in den folgenden Zeichnungen identisch.

In der Zeichnung [Abb. 8-4 "Punkt P wird ohne Adaptive Motion angefahren"](#) auf [Seite 411](#) wird der Punkt ohne "Adaptive Motion" angefahren.

- Dazu wird der Punkt P zum Zeitpunkt t_0 vom Anwender kommandiert.
- Da sich der synchrone Endpunkt der Bewegung noch vor der Startebene befindet, wartet die Kinematik bis der Punkt die Startebene passiert hat
- Hat der kommandierte Endpunkt P der Bewegung den bandsynchronen Arbeitsbereich zum Zeitpunkt t_1 erreicht, startet die Bewegung. Die Kinematik beginnt zu synchronisieren
- Zum Zeitpunkt t_2 ist die Bewegung abgeschlossen und die Kinematik ist synchron zum Endpunkt P auf dem Band

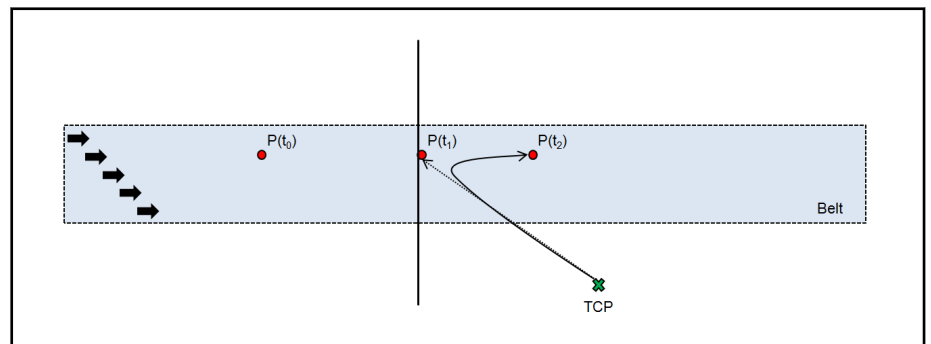


Abb. 8-4: Punkt P wird ohne "Adaptive Motion" angefahren

In der Zeichnung [Abb. 8-5 "Punkt P wird mit Adaptive Motion angefahren"](#) auf [Seite 412](#) wird der Punkt mit "Adaptive Motion" angefahren.

- Dazu wird wie zuvor der Punkt P zum Zeitpunkt t_0 vom Anwender kommandiert
- Obwohl der synchrone Endpunkt der Bewegung noch vor der Startebene ist, beginnt die Kinematik sofort zum Zeitpunkt t_0 mit der Bewegung. Die Wartezeit der Kinematik entfällt komplett
- Hat der kommandierte Endpunkt P der Bewegung den bandsynchronen Arbeitsbereich zum Zeitpunkt t_1 erreicht, endet auch die Bewegung. Die Kinematik ist genau zu diesem Zeitpunkt synchron zum Endpunkt P auf dem Band

Anwendungsbeispiele

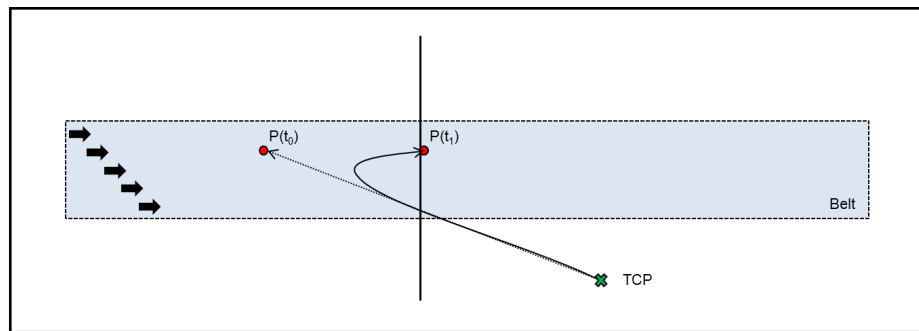


Abb. 8-5: Punkt P wird mit "Adaptive Motion" angefahren

Fazit Da die Kinematik mit "Adaptive Motion" im selben Augenblick den Zielpunkt erreicht, in dem die Kinematik ohne "Adaptive Motion" mit der Bewegung startet, wird die Zeit zwischen dem Zeitpunkt t_1 und dem Zeitpunkt t_2 komplett eingespart.

Somit kann durch die Funktionalität "Adaptive Motion" ein zeitoptimales Verhalten erreicht werden, welches neben einer der Mechanik schonenden Bewegung zusätzlich eine Einsparungen an Energie ermöglicht.

8.8 Bestimmung der Kommandopuffergröße

Im folgenden Beispiel werden 100 Kommandos je 1mm in derselben Richtung in Verbindung mit Cont_Slope verwendet. Durch den Cont_Slope werden diese Kommandos zu einer Bewegung zusammengefasst, bleiben aber logisch als einzelne Kommandos erhalten. Der Cont_Slope stellt zusätzlich sicher, dass zum letzten bekannten Zielpunkt die Geschwindigkeit abgebaut werden kann (Stillstand).

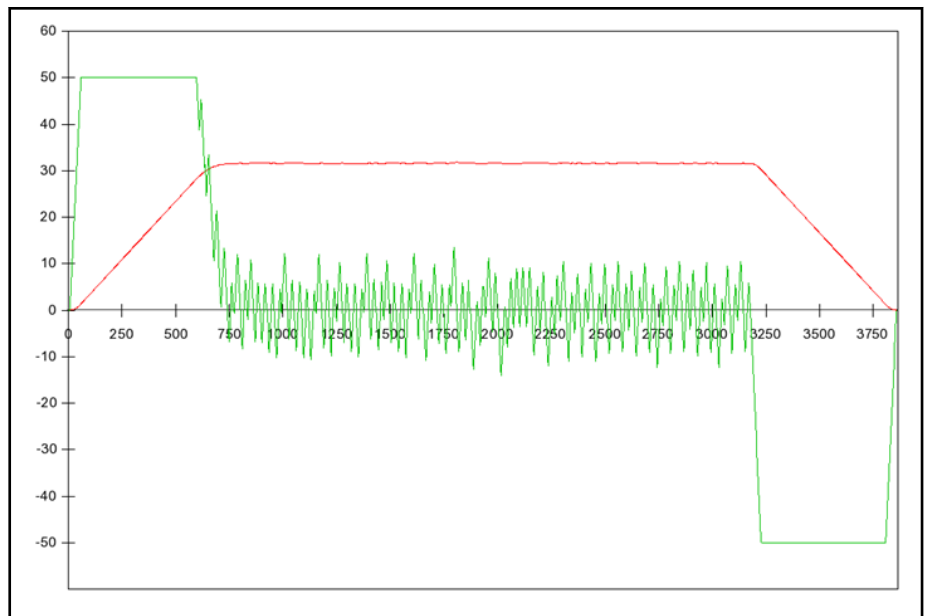
Fall 1: Satzpuffer von 12

Im Parameter [K-0-0097](#), Maximale Kommandopuffergröße wird 12 eingetragen. Dadurch können sich maximal 12 Kommandos gleichzeitig im Satzpuffer befinden.

Weitere Kommandos können erst abgesetzt werden, wenn ein Kommando beendet ist. Werden jetzt Kommandos abgesetzt, können nur die ersten 12 Kommandos direkt in den Kommandopuffer geschrieben werden. Weitere Kommandos müssen warten. Solange genügend Bremsweg bis zum letzten bekannten Zielpunkt vorhanden ist, kann beschleunigt werden.

Kann durch die Kommandopufferbegrenzung kein neues Kommando in den Kommandopuffer gestellt werden, muss ab einem gewissen Punkt abgebremst werden (bzw. geringer beschleunigt werden), um am letzten bekannten Zielpunkt anzuhalten. Beim Abbremsen wird jedoch ein Kommando fertig. Es ist wieder Platz für ein neues Kommando. Der zur Verfügung stehende Bremsweg verlängert sich. Es kann daher wieder beschleunigt werden. Dies setzt sich solange fort bis alle Sätze abgearbeitet sind.

Anwendungsbeispiele



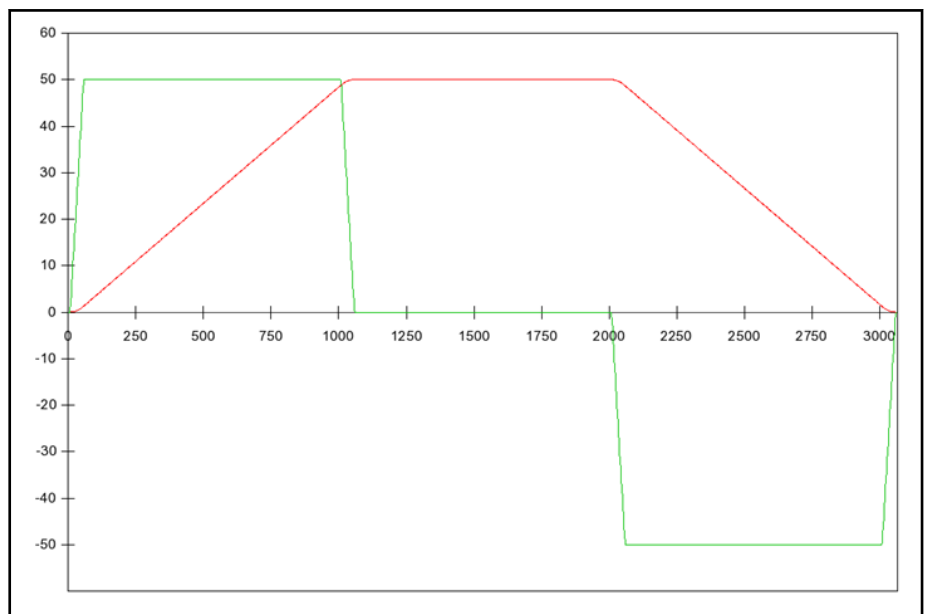
Rot Geschwindigkeit
 Grün Beschleunigung

Abb. 8-6: Zeitdiagramm Kommandopuffergröße 12

Als Folge ergibt sich eine Geschwindigkeit, die kleiner als die programmierte Geschwindigkeit ist. Weiterhin ist die resultierende Geschwindigkeit nicht konstant. Immer wenn Platz für ein neues Kommando im Kommandopuffer ist, steigt die Geschwindigkeit, wenn der Bremsweg erreicht ist, sinkt sie wieder.

Fall 2: Satzpuffer von 100

Im Parameter [K-0-0097](#), Maximale Kommandopuffergröße wird 100 eingetragen. Dadurch können alle 100 Kommandos in den Kommandopuffer geschrieben werden. Die komplette Pfadlänge ist bekannt. Es kann auf dem Pfad so lange beschleunigt werden, wie es die maximale Pfadgeschwindigkeit zulässt.



Rot Geschwindigkeit
 Grün Beschleunigung

Abb. 8-7: Zeitdiagramm Kommandopuffergröße 100

Anwendungsbeispiele

Als Folge ergibt sich ein "glatter" Geschwindigkeitsverlauf. Da eine höhere Geschwindigkeit erreicht wird, wird der Zielpunkt schneller erreicht.

8.9 Kinematik-Synchronisation mit zyklischer Sollwertvorgabe

Mithilfe der zyklischen Sollwertvorgabe ist es möglich, die Bewegung einer Kinematik individuell zu gestalten. So kann mithilfe zyklisch generierter, absoluter oder relativer Sollwerte positions- oder geschwindigkeitsgenau verfahren werden. Damit ist beispielsweise eine Master-Slave-Beziehung zwischen zwei Kinematiken realisierbar.

HINWEIS

Kinematik folgt dem Sollwert!

Vorgegebene Sollwertsprünge können zu einem Fehler führen, wenn die Kinematik (z. B. aufgrund mangelnder Dynamik einzelner Achsen) dem Sollwert nicht folgen kann! Daher sollten die beiden zu synchronisierenden Kinematiken im gewählten Koordinatensystem an derselben Position stehen!

Die Vorgabe zyklischer Sollwerte kann auf zwei verschiedene Art und Weise erfolgen. Beim Kommando "Open Cyclic Channel" werden die Sollwerte im Maschinenkoordinatensystem (MCS) aufaddiert. Ist hingegen eine Addition im Achskoordinatensystem (ACS) notwendig, so kommt das Kommando "Open Direct Cyclic Channel" zum Einsatz.

Das hier dargestellte Beispiel nutzt das Kommando "Open Direct Cyclic Channel", da somit auch Armigkeitswechsel ermöglicht werden. Dies bedeutet, dass die Bewegungen der synchronisierten Kinematik als inkrementell kleine PTP-Bewegungen ausgeführt werden.

Deklarationsteil

```
FUNCTION BLOCK RobotSync
VAR_INPUT
    Enable      : BOOL := FALSE;
    Deceleration : REAL := 1000;
    Jerk        : REAL := 10000;
    CoordSystem : ML_COORDSYS := CS_AXES_COORDINATES;
    SetMode     : ML_SET_MODE := SM_TO;
    BlendingRadius : REAL := 0.0;
    MasterGroup  : AXIS_GROUP_REF;
    SlaveGroup   : AXIS_GROUP_REF;
END_VAR
VAR_OUTPUT
    InOperation : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : ERROR_CODE := NONE_ERROR;
    ErrorIdent  : ERROR_STRUCT := (Table := NO_TABLE_USED, Additional1 := 16#0000, Additional2 := 16#0000);
END_VAR
VAR
    bOpen      : BOOL := FALSE;
    bWrite      : BOOL := FALSE;
    bKinStop    : BOOL := FALSE;
    bStarted    : BOOL := FALSE;
    bStopped    : BOOL := FALSE;
    uiState     : UINT := IDLE;

    fbOpenDirectCyclicChannel : ML_RobotOpenDirectCyclicChannel;
    fbWriteCyclicChannel      : ML_RobotWriteCyclicChannel;
    fbReadCyclicChannel       : ML_RobotReadCyclicChannel;
    fbGroupStop               : MC_GroupStop;
    fbRobotTransformPoint     : ML_RobotTransformPoint;
END_VAR
VAR CONSTANT
    IDLE :UINT := 0;
    INIT :UINT := 10;
    OPEN :UINT := 20;
    SYNC :UINT := 30;
    STOP :UINT := 40;
    RESET :UINT := 50;
END_VAR
```

Programmteil

```
(* start/stop the cyclical command value specification *)
IF Enable = TRUE AND bStarted = FALSE THEN
    uiState := INIT;
    bStarted := TRUE;
    bStopped := FALSE;
ELSEIF Enable = FALSE AND bStopped = FALSE THEN
    IF KinData[SlaveGroup.GroupNo].Kis_GroupMoving THEN
        uiState := STOP;
    ELSE
        uiState := RESET;
    END_IF
    bStopped := TRUE;
    bStarted := FALSE;
    Error := FALSE;
END_IF

(* state machine *)
CASE uiState OF
    INIT: (* initialization *)
        bWrite := TRUE;
        IF fbWriteCyclicChannel.InOperation AND fbRobotTransformPoint.InOperation THEN
            uiState := OPEN;
        END_IF

    OPEN: (* open cyclical channel *)
        bOpen := TRUE;
        IF fbOpenDirectCyclicChannel.Active THEN
            InOperation := TRUE;
            uiState := SYNC;
        END_IF

    SYNC: (* write cyclical values *)
        IF fbOpenDirectCyclicChannel.Done THEN
            InOperation := FALSE;
        END_IF

    STOP: (* close cyclical channel *)
        bKinStop := TRUE;
        IF fbGroupStop.Done THEN
            uiState := RESET;
        END_IF

    RESET: (* reset internal variables *)
        bOpen := FALSE;
        bWrite := FALSE;
        bKinStop := FALSE;
        InOperation := FALSE;
        uiState := IDLE;
END_CASE

(* functions blocks for cyclical command value specification *)
fbRobotTransformPoint(
    Enable := bWrite,
    InPoint := KinData[MasterGroup.GroupNo].arPositionJointCoord_i,
    InCoordSystem := CS_AXES_COORDINATES,
    OutCoordSystem := CoordSystem,
    AxisGroup := SlaveGroup);

fbWriteCyclicChannel(
    Enable := bWrite,
    Value := fbRobotTransformPoint.OutPoint,
    CoordSystem := CoordSystem,
    CyclicMode := CYCLIC_MODE_POS_ABS,
    AxisGroup := SlaveGroup);

fbOpenDirectCyclicChannel(
    Execute := bOpen,
    Deceleration := Deceleration,
    Jerk := Jerk,
    SetMode := SetMode,
    BlendingRadius := BlendingRadius,
    AxisGroup := SlaveGroup);

fbReadCyclicChannel(
    Enable := fbWriteCyclicChannel.InOperation,
    AxisGroup := SlaveGroup);

(* function block for stopping of the cyclical command value specification *)
fbGroupStop(
    Execute := bKinStop,
```

Anwendungsbeispiele

```

Deceleration:= Deceleration,
Jerk         := Jerk,
AxisGroup    := SlaveGroup);

(* error handling *)
IF fbOpenDirectCyclicChannel.Error THEN
  Error      := TRUE;
  ErrorID    := fbOpenDirectCyclicChannel.ErrorID;
  ErrorIdent := fbOpenDirectCyclicChannel.ErrorIdent;
  uiState    := RESET;
ELSIF fbRobotTransformPoint.Error THEN
  Error      := TRUE;
  ErrorID    := fbRobotTransformPoint.ErrorID;
  ErrorIdent := fbRobotTransformPoint.ErrorIdent;
  uiState    := RESET;
ELSIF fbWriteCyclicChannel.Error THEN
  Error      := TRUE;
  ErrorID    := fbWriteCyclicChannel.ErrorID;
  ErrorIdent := fbWriteCyclicChannel.ErrorIdent;
  uiState    := RESET;
ELSIF fbReadCyclicChannel.Error THEN
  Error      := TRUE;
  ErrorID    := fbReadCyclicChannel.ErrorID;
  ErrorIdent := fbReadCyclicChannel.ErrorIdent;
  uiState    := RESET;
ELSIF fbGroupStop.Error THEN
  Error      := TRUE;
  ErrorID    := fbGroupStop.ErrorID;
  ErrorIdent := fbGroupStop.ErrorIdent;
  uiState    := RESET;
END_IF

```

Eingang	Beschreibung
Enable	Aktivierung der Synchronisation
Deceleration	Verzögerung für Desynchronisation
Jerk	Ruck für Desynchronisation
CoordSystem	Koordinatensystem
SetMode	Überschleifart für Desynchronisation
BlendingRadius	Überschleifradius für Desynchronisation
MasterGroup	Master-Kinematik
SlaveGroup	Slave-Kinematik

Tab. 8-1: Beschreibung der Eingänge

Ausgang	Beschreibung
InOperation	Kinematik ist synchronisiert
Error	Kinematik ist aufgrund eines Fehlers nicht synchronisiert
ErrorID	Kurzhinweis auf die Fehlerursache
ErrorIdent	Detailinformation über den Fehler

Tab. 8-2: Beschreibung der Ausgänge

Mit steigender Flanke am Eingang "Enable" wird die aktuelle absolute Position der Master-Kinematik im angegebenen Koordinatensystem in den zyklischen Sollwertkanal geschrieben. Mit dem anschließenden Öffnen des zyklischen Sollwertkanals wird die Kinematik in den Zustand "Group Moving" versetzt und fährt den zuvor geschriebenen Sollwert in einem Zyklus an. Daher sollten vor der Synchronisation beide Kinematik im gewählten Koordinatensystem an derselben Stelle stehen, damit die Slave-Kinematik an dieser Stelle nicht "springt"!

Bewegt man nun die Master-Kinematik, so folgt ihr die Slave-Kinematik. Mittels des Ausgangs "InOperation" wird signalisiert, dass die Kinematiken syn-

chronisiert sind. Wird eine Synchronisation aufgrund eines Fehler verhindert, so zeigt dies der Ausgang "Error". Detaillierte Hinweise zum Fehler können den Ausgängen "ErrorID" und "ErrorIdent" entnommen werden.

Über die weiteren Eingänge "Deceleration", "Jerk", "SetMode" und "BlendingRadius" wird die Desynchronisation und somit das Verlassen der zyklischen Sollwertvorgabe bestimmt. Setzt man während der synchronen Phase ein beliebiges Bewegungskommando für die Slave-Kinematik ab (z. B. Move Direct Absolut), so wird diese mit der angegebenen Verzögerung und Ruck abgebremst. Ist desweiteren ein Überschleifradius angegeben und steht die Überschleifart auf "SM_VIA", so wird das neue Bewegungskommando überschleifend verfahren, insofern ein PTP-Bewegungskommando abgesetzt wurde (siehe [Überschleifmöglichkeiten bei verschiedenen Bewegungsarten auf Seite 61](#)). Ist die Desynchronisation abgeschlossen, so wird der Ausgang "In-Operation" zurückgesetzt.

8.10 Sequenzielle Programmierung mit dem Kinematik-Interface

Unter Beachtung gewisser Randbedingungen ist es möglich, mit dem Kinematik-Interface einen sequenziellen Bewegungsablauf zu realisieren. Anhand eines Beispiels wird im Folgenden erläutert, wie dies möglich ist.

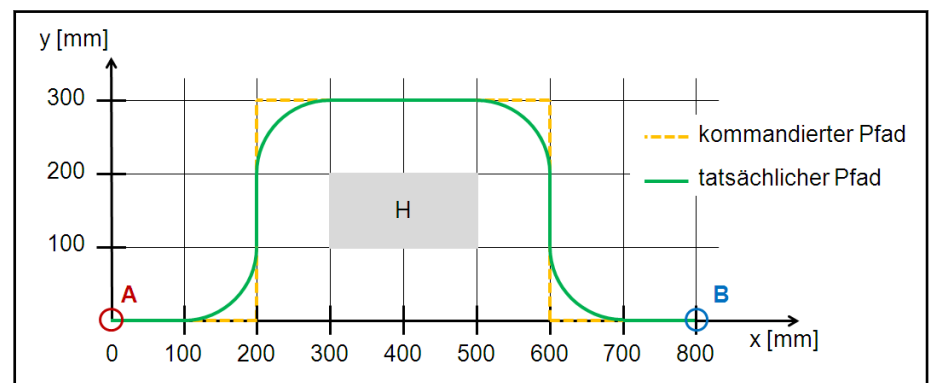


Abb. 8-8: Sequenzielle Programmierung mit dem Kinematik-Interface - Bewegungspfad

Eine Kinematik (in diesem Fall ein Karteser) soll von einem Punkt A zu einem Punkt B um ein Hindernis H fahren. Hierzu wird der in obiger Abbildung dargestellte Pfad vorgegeben.

Die hieraus resultierenden Positionen sind:

- A(0,0)
- P1(200,0)
- P2(200,300)
- P3(600,300)
- P4(600,0)
- B(800,0)

Die Positionen sollen mittels linearer, absoluter Verfahrensbewegungen überschleifend angefahren werden. Um dies mithilfe des Kinematik-Interfaces zu realisieren sind folgende Programmzeilen notwendig. Die Grenzwerte für Geschwindigkeit, Beschleunigung, Verzögerung und Ruck sind hierbei frei gewählt.

Deklarationsteil

```
VAR
  uiState:      UINT := STATE_IDLE;
  bStartMove:   BOOL := FALSE;
  rVelocity:    REAL := 0.0;
```

Anwendungsbeispiele

```

rAcceleration: REAL := 0.0;
rDeceleration: REAL := 0.0;
rJerk: REAL := 0.0;
arPoint1: ARRAY[1..16] OF REAL := [200.0, 0.0, 14(0.0)];
arPoint2: ARRAY[1..16] OF REAL := [200.0, 300.0, 14(0.0)];
arPoint3: ARRAY[1..16] OF REAL := [600.0, 300.0, 14(0.0)];
arPoint4: ARRAY[1..16] OF REAL := [600.0, 0.0, 14(0.0)];
arPointB: ARRAY[1..16] OF REAL := [800.0, 0.0, 14(0.0)];
END_VAR

VAR CONSTANT
STATE_IDLE: UINT := 1;
STATE_MOVE1: UINT := 2;
STATE_MOVE2: UINT := 3;
STATE_MOVE3: UINT := 4;
STATE_MOVE4: UINT := 5;
STATE_MOVE5: UINT := 6;
END_VAR

```

Programmteil

```

CASE uiState OF
STATE_IDLE:
    IF bStartMove THEN
        bStartMove := FALSE;
        uiState := STATE_MOVE1;
    END_IF

STATE_MOVE1:
    (* Verhalten nach einem Betriebsartenwechsel festlegen;
       muss bei überschleifenden Bewegungen immer auf "NoAction" stehen *)
    arKinCtrl_gb[1].Admin.ChangeOpMode := NoAction;

    (* arPoint1 = Position P1 festlegen *)
    arKinCtrl_gb[1].CoordMode.Point := arPoint1;

    (* Grenzwerte festlegen *)
    arKinCtrl_gb[1].CoordMode.Velocity := 100;
    arKinCtrl_gb[1].CoordMode.Acceleration := 1000;
    arKinCtrl_gb[1].CoordMode.Deceleration := 1000;
    arKinCtrl_gb[1].CoordMode.Jerk := 10000;

    (* Koordinatensystem festlegen *)
    arKinCtrl_gb[1].CoordMode.CoordSystem := CS_MACHINE_COORDINATES;

    (* SetMode festlegen *)
    arKinCtrl_gb[1].CoordMode.SetMode := SM_VIA;

    (* SlopeType festlegen *)
    arKinCtrl_gb[1].CoordMode.SlopeType := ST_CONTINUOUS_SLOPE;

    (* Überschleifradius festlegen *)
    arKinCtrl_gb[1].CoordMode.BlendingRadius := 100;

    (* Lineares Verfahrkommando absetzen;
       Satz wird in den Satzpuffer gestellt *)
    arKinCtrl_gb[1].Admin._OpMode.en := ModeCoordPosLinAbs;

    (* vor nächstem Schritt: Ausgang "InBuffer" abfragen *)
    IF arKinStatus_gb[1].Admin.InBuffer THEN
        uiState := STATE_MOVE2;
    END_IF

STATE_MOVE2:
    (* arPoint2 = Position P2 festlegen *)
    arKinCtrl_gb[1].CoordMode.Point := arPoint2;

    (* es müssen nur die Eingangsstrukturen beschrieben werden, welche
       sich geändert haben *)
    arKinCtrl_gb[1].CoordMode.Velocity := 20;

    (* ein weiteres Absetzen eines linearen Verfahrkommandos ist nicht notwendig,
       da dies durch die Änderung des Positionswertes automatisch geschieht *)

    (* vor nächstem Schritt: Ausgang "InBuffer" abfragen *)
    IF arKinStatus_gb[1].Admin.InBuffer THEN
        uiState := STATE_MOVE3;
    END_IF

STATE_MOVE3:
    (* arPoint3 = Position P3 festlegen *)
    arKinCtrl_gb[1].CoordMode.Point := arPoint3;

```



```
(* vor nächstem Schritt: Ausgang "InBuffer" abfragen *)
IF arKinStatus_gb[1].Admin.InBuffer THEN
    uiState := STATE_MOVE4;
END_IF

STATE_MOVE4:
(* arPoint4 = Position P4 festlegen *)
arKinCtrl_gb[1].CoordMode.Point := arPoint4;

(* vor nächstem Schritt: Ausgang "InBuffer" abfragen *)
IF arKinStatus_gb[1].Admin.InBuffer THEN
    uiState := STATE_MOVE5;
END_IF

STATE_MOVE5:
(* arPointB = Position B festlegen *)
arKinCtrl_gb[1].CoordMode.Point := arPointB;

(* Ende *)
uiState := STATE_IDLE;

END_CASE
```

Die Satzvorbereitung, d. h. die Abarbeitung der Schrittkette erfolgt erheblich schneller als die Ausführung der Bewegung. Somit ist es möglich, dass die Schrittkette bereits im STATE_IDLE steht, obwohl die Kinematik noch in der Bewegung ist.



Die Status-Ausgänge des Kinematik-Interfaces beziehen sich immer auf den **zuletzt** abgesetzten Verfahrbefehl, in diesem Beispiel auf die Bewegung zum Punkt B. Es gibt somit keine Möglichkeit den Status der vorhergehenden Bewegungen auszuwerten.

8.11 Anmeldung einer benutzerdefinierten Transformation mit dem Kinematik-Interface

Bei der Verwendung einer benutzerdefinierten Transformation, d. h. der Transformationsnummer 1100 ist es notwendig vor der Phasenschaltung nach **BB** die Transformation in der SPS anzumelden. Diese Anmeldung kann in einfacher Art und Weise über das Kinematik-Interface realisiert werden. Hierfür ist es lediglich notwendig, den Transformations-Funktionsbaustein an den Eingang "Admin.KinTransform" des Kinematik-Interfaces anzulegen.

Folgendes Kommando startet die Anmeldung der Transformation ("fbTransformation" entspricht einer Instanz des Transformations-Funktionsbausteins):

```
arKinCtrl_gb[1].Admin.KinTransformPLC:= fbTransformation;
```



Die Anmeldung ist nur im Parametriermodus **P2** möglich. Wird in einer anderen Phase eine Transformation an den Eingang des Kinematik-Interfaces angelegt, so startet die Anmeldung erst wenn Phase 2 erreicht ist.



Wird das Kinematik-Interface erstmalig mit einer benutzerdefinierten Transformation angemeldet, so gibt die Steuerung den Fehler "Transformation nicht angemeldet" aus. Dies ist dadurch begründet, dass das IMC-Interface versucht in den Betriebsmodus zu schalten. Nach erfolgreicher Anmeldung der Transformation lässt sich der Fehler quittieren.

War die Anmeldung der Transformation in der SPS erfolgreich, so wird dies signalisiert über den Statusausgang

```
arKinStatus_gb[1].Admin.KinTransformPLCDone
```


9 Parameter

9.1 Parameter, Allgemeines

Das Parametersystem der IndraMotion MLC unterstützt die Implementation von Kinematiken auf der Steuerung:

- Grenzwerte und achsbezogene Daten werden in A-Parametern der jeweiligen Achse abgelegt
- Zur Verwaltung der Daten **aller** Kinematiken einer Steuerung werden [Steuerungsbezogene Parameter auf Seite 428](#) angelegt:
- Zur Verwaltung der Daten **einer** Kinematik in der Steuerung werden [Kinematikbezogene Parameter auf Seite 429](#) angelegt. Jeder Parameter existiert genau einmal pro Kinematik

9.2 Achsbezogene Parameter

9.2.1 Achsbezogene Parameter, Allgemeines

Grenzwerte und achsbezogene Daten zur Unterstützung einer Robot-Control V2-Kinematik werden in A-Parametern der jeweiligen Achse abgelegt.

9.2.2 A-0-0028, Fahrbereichsgrenzschalter

Funktion In diesem Parameter wird das Signalverhalten der Fahrbereichsgrenzschalter-Eingänge und das Verhalten des Antriebs bezüglich Fahrbereichsüberschreitung festgelegt.

Aufbau

Bit	Bezeichnung/ Funktion
0	Signalverhalten Fahrbereichsgrenzschalter Eingänge (nur reale Achsen) 0: nicht invertiert, hohe Spannung wird als Fahrbereich überschritten gewertet, 1: invertiert, niedrige Spannung wird als Fahrbereich überschritten gewertet.
1	Aktivierung Fahrbereichsgrenzschalter (nur reale Achsen) 0: nicht aktiv, 1: aktiv.
2	Behandlung der Fahrbereichsüberschreitung 0: als Fehler (gemäß Fehlerreaktion), 1: als fatale Warnung (Stillsetzung).
3..15	reserviert

9.2.3 A-0-0029, Positionspolaritäten

Funktion Nur für reale Achsen gilt:

Mit den ersten 4 Bits dieses Parameter können die Polaritäten der angegebenen Positionsdaten invertiert werden.

Die Polaritäten werden außerhalb der Regelstrecke umgeschaltet, d. h. am Eingang und Ausgang der Regelstrecke.

Der Parameter wirkt hierzu auf den S-0-0055 des Antriebsgerätes.

Parameter



Die Festlegung der Positionspolarität muss bei der Erstinbetriebnahme einer Achse vor der Herstellung des Maschinen-Nullpunktbezugs für die Messsysteme durchgeführt werden, da sich bei Änderung der Polarität andere Positionswerte ergeben.

Bei rotatorischen (drehenden) Motoren gilt:

- Drehung der Motorabtriebswelle im Uhrzeigersinn bei positiver Positionssollwertdifferenz und positiver Polarität.

Bei Linearmotoren gilt:

- Bewegung des Primärteils in Richtung der Kabelanschluss-Seite bei positiver Positionssollwertdifferenz und positiver Polarität.

Siehe auch: [A-0-0048](#) und [A-0-0049](#)

Funktion Mit dem Bit 4 dieses Parameters werden die Software-Positionsgrenzwert, "A-0-0030, Positionsgrenzwert, positiv" bzw. "A-0-0031, Positionsgrenzwert, negativ" aktiviert bzw. deaktiviert.

Aufbau

Bit	Bezeichnung/ Funktion
0	Positionssollwert 0: Polarität positiv 1: Polarität negativ
1	Positionssollwert additiv 0: Polarität positiv 1: Polarität negativ
2	Positionswert 1 0: Polarität positiv 1: Polarität negativ
3	Positionswert 2 0: Polarität positiv 1: Polarität negativ
4	Positionsgrenzwert 0: nicht aktiv, 1: aktiv.
15..5	reserviert

9.2.4 A-0-0030, Positionsgrenzwert, positiv

Funktion Der erlaubte Verfahrbereich einer Achse kann durch Positionsgrenzwerte festgelegt werden. Mit diesem Parameter wird der maximal erlaubte positive Positionswert (positive Fahrerrichtung) begrenzt. In diesen wird der erste nicht mehr erlaubte Positionswert eingetragen.

Der positive Grenzwert ist nur aktiv, wenn alle Positionsdaten auf den Achsnullpunkt bezogen sind, d.h. die Achse referenziert ist "A-0-0021, Achsstatus, Bit 10".

Verwendung Wenn die "A-0-2200, Sollposition" der Achse auf oder außerhalb des positiven Positionsgrenzwertes gesetzt ist, wird das Fehlerbit im "A-0-0021, Achsstatus, Bit 9" TRUE und erzeugt den Fehler "F2112053, Zielposition außerhalb des Verfahrbereiches".

Erreicht oder überschreitet der "A-0-0100, Positionswert" den "[A-0-0030](#), Positionsgrenzwert, positiv" oder den "[A-0-0031](#), Positionsgrenzwert, negativ", setzt die Achse das Fehlerbit im "A-0-0021, Achsstatus, Bit 9" und erzeugt den Fehler "F2116029, Positionsgrenzwert positiv überschritten" bzw. "F2116030, Positionsgrenzwert negativ überschritten".

Siehe auch Funktionsbeschreibung: "Positionsgrenzwerte".

9.2.5 A-0-0031, Positionsgrenzwert, negativ

Funktion Der erlaubte Verfahrbereich einer Achse kann durch Positionsgrenzwerte festgelegt werden. Mit diesem Parameter wird der maximal erlaubte negative Positionswert (negative Verfahrrichtung) begrenzt. In diesen wird der erste nicht mehr erlaubte Positionswert eingetragen.

Der negative Grenzwert ist nur aktiv, wenn alle Positionsdaten auf den Achsnullpunkt bezogen sind, d.h. die Achse referenziert ist "A-0-0021, Achsstatus, Bit 10".

Verwendung Wenn die "A-0-2200, Sollposition" der Achse auf oder außerhalb des negativen Positionsgrenzwertes gesetzt ist, wird das Fehlerbit im "A-0-0021, Achsstatus, Bit 9" TRUE und erzeugt den Fehler "F2112053, Zielposition außerhalb des Verfahrbereiches".

Erreicht oder überschreitet der "A-0-0100, Positionswert" den "[A-0-0030](#), Positionsgrenzwert, positiv" oder den "[A-0-0031](#), Positionsgrenzwert, negativ", setzt die Achse das Fehlerbit im "A-0-0021, Achsstatus, Bit 9" und erzeugt den Fehler "F2116029, Positionsgrenzwert, positiv, überschritten" bzw. "F2116030 Positionsgrenzwert, negativ, überschritten".

Siehe auch Funktionsbeschreibung: "Positionsgrenzwerte".

9.2.6 A-0-0032, Geschwindigkeitsgrenzwert, positiv

Funktion Der Parameter beschreibt die maximal zulässige Geschwindigkeit in positiver Richtung.

Verwendung Wenn der Parameter "A-0-0102, Geschwindigkeitswert" den Parameter "[A-0-0032](#), Geschwindigkeitsgrenzwert, positiv" überschreitet und der Wert des Parameters kleiner ist als der Inhalt des "[A-0-0035](#), Geschwindigkeitsgrenzwert, bipolar", setzt die Achse das Warnungsbit "A-0-0021, Achsstatus, bit 8" und erzeugt die Warnung "E0112061, Sollgeschwindigkeit größer positiver Grenzwert".

9.2.7 A-0-0033, Geschwindigkeitsgrenzwert, negativ

Funktion Der Parameter beschreibt die maximal zulässige Geschwindigkeit in negativer Richtung.

Verwendung Wenn der Parameter "A-0-0102, Geschwindigkeitswert" den Parameter "[A-0-0033](#), Geschwindigkeitsgrenzwert, negativ" unterschreitet und der Wert des Parameters größer ist als der Inhalt des "[A-0-0035](#), Geschwindigkeitsgrenzwert, bipolar", setzt die Achse das Warnungsbit "A-0-0021, Achsstatus, bit 8" und erzeugt die Warnung "E0112062, Sollgeschwindigkeit kleiner negativer Grenzwert".

9.2.8 A-0-0034, Beschleunigungsgrenzwert, bipolar

Funktion Der Parameter beschreibt die maximal zulässige Beschleunigung symmetrisch in beide Richtungen (Beschleunigen und Bremsen).

Verwendung Wenn der Parameter "A-0-0103, Beschleunigungswert" den Parameter "[A-0-0034](#), Beschleunigungsgrenzwert, bipolar" überschreitet, setzt die Achse das Warnungsbit "A-0-0021, Achsstatus, bit 8" und erzeugt die Warnung "E0112039, Maximale Beschleunigung überschritten".

Parameter

9.2.9 A-0-0035, Geschwindigkeitsgrenzwert, bipolar

Funktion Der Parameter beschreibt die maximal zulässige Geschwindigkeit symmetrisch in beide Richtungen.

Verwendung Wenn der Parameter "A-0-0102, Geschwindigkeitswert" den Parameter "A-0-0035, Geschwindigkeitsgrenzwert, bipolar" überschreitet und der Wert des Parameters kleiner ist als der Inhalt "A-0-0032, Geschwindigkeitsgrenzwert, positiv", setzt die Achse das Warnungsbit "A-0-0021, Achsstatus, bit 8" und erzeugt die Warnung "E0112061, Sollgeschwindigkeit grösser positiver Grenzwert" oder "E0112062, Sollgeschwindigkeit kleiner negativer Grenzwert".

9.2.10 A-0-0036, Ruckgrenzwert, bipolar

Funktion Der Parameter beschreibt die maximal zulässige Beschleunigungsänderung pro Zeit (= Ruck) symmetrisch in beide Richtungen (Beschleunigen und Bremsen).

Dieser Grenzwert wirkt gegen den kommandierten Ruck im Parameter "A-0-0216, Ruck".



Der Parameter begrenzt die Beschleunigungsänderung pro Zeit bei "Antrieb-Halt" und bei Kommandos, die sich selbst Positions-sollwerte generieren.

Der Wert "0" schaltet den Ruckgrenzwert ab.

Er wird beeinflusst durch den Parameter "A-0-0053, Wichtungsart für Beschleunigungsdaten".

9.2.11 A-0-0038, Drehmoment-/Kraftgrenzwert, bipolar

Funktion Durch diesen Parameter kann der Drehmoment-/Kraftsollwert zum Schutz für mechanische Komponenten im Antriebsstrang auf zulässige Maximalwerte begrenzt werden.

Der Parameter wirkt bipolar, d.h. positive und negative Sollwerte werden auf den eingetragenen Wert begrenzt. Die Begrenzung wirkt auf die Summe aller Drehmomentsollwerte, d.h. auf die Summe der Sollwertanteile des Geschwindigkeitsreglerausgangs und der Sollwertanteile aus beschleunigungsabhängigen Vorsteuerungen ("S-0-0348, Verstärkung Beschleunigungsvorsteuerung"; Geschw.-Regelkreis: "P-0-1126, Beschleunigungsvorsteuerung").

Die Einheit von "A-0-0038" ist abhängig von der eingestellten Wichtung.

Siehe auch Parameterbeschreibung "A-0-0050, Wichtungsart für Drehmoment-/Kraftdaten"



Der wirksame Grenzwert wird in "P-0-0049, Wirksamer Momenten-/Kraft-Sollwert" angezeigt. Dieser Wert kann gegenüber "A-0-0038" aufgrund eines kleineren Wertes in "P-0-0109, Spitzendrehmoment-/ Kraftbegrenzung" oder aufgrund einer Strombegrenzung durch entsprechende Auslastung von Verstärker oder Motor eingeschränkt sein.

Soll dem Antrieb im stationären Betrieb (konstante Drehzahl) ein geringeres Drehmoment zur Verfügung stehen als bei Beschleunigungsvorgängen, so kann dies über entsprechende Werte von "S-0-0082, Drehmoment-/ Kraft-Grenzwert positiv" und "S-0-0083, Drehmoment-/ Kraft-Grenzwert negativ" in Verbindung mit Beschleunigungsvorsteuerung (S-0-0348 oder P-0-1126) erreicht werden.

Der Drehmoment-/Kraft-Grenzwert bipolar wirkt auch auf "C1300 Kommando: Fahren auf Festanschlag".

Der Wert von "A-0-0038" ist ein Kriterium für die Quittierung von "C1300".

Siehe auch Funktionsbeschreibung "Drehmoment-/Kraftregelung".

9.2.12 A-0-0039, Drehmoment-/Kraftgrenzwert, positiv

Funktion Der Parameter beschreibt das maximal zulässige Kraft/Drehmoment in positiver Richtung.



Ab einer Antriebsfirmware HDx-20 (Hydraulischer Antrieb) wird der Parameter "A-0-0039, Drehmoment-/ Kraftgrenzwert, positiv" auf den "S-0-0550, Drehmoment-/Kraft-Sollwert positiv 2" synchronisiert.

Verwendung Wenn der Parameter "A-0-0110, Drehmoment-/Kraftistwert" den Parameter "A-0-0039, Drehmoment-/Kraftgrenzwert, positiv" überschreitet und der Wert des Parameters kleiner ist als der Inhalt "A-0-0038, Drehmoment-/Kraftgrenzwert, bipolar", setzt die Achse das Warnungsbit "A-0-0021, Achsstatus, bit 8" und erzeugt die Warnung "E0112057, Drehmoment-/Kraftgrenzwert positiv überschritten".

9.2.13 A-0-0040, Drehmoment-/Kraftgrenzwert, negativ

Funktion Der Parameter beschreibt das maximal zulässige Kraft/Drehmoment in negativer Richtung.



Ab einer Antriebsfirmware HDx-20 (Hydraulischer Antrieb) wird der Parameter "A-0-0040, Drehmoment-/ Kraftgrenzwert, negativ" auf den "S-0-0551, Drehmoment-/Kraft-Sollwert negativ 2" synchronisiert.

Verwendung Wenn der Parameter "A-0-0110, Drehmoment-/Kraftistwert" den Parameter "A-0-0040, Drehmoment-/Kraftgrenzwert, negativ" unterschreitet und der Wert des Parameters größer ist als der Inhalt des "A-0-0038, Drehmoment-/ Kraftgrenzwert, bipolar", setzt die Achse das Warnungsbit "A-0-0021, Achsstatus, bit 8" und erzeugt die Warnung "E0112058, Drehmoment-/Kraftgrenzwert negativ unterschritten".

9.2.14 A-0-0048, Geschwindigkeitspolaritäten

Funktion In diesem Parameter kann die Polarität der Geschwindigkeitsdaten auf die Anwendung bezogen umgeschaltet werden.

Parameter

Die Polarität wird nicht innerhalb, sondern außerhalb (am Eingang und Ausgang) einer Regelstrecke umgeschaltet.

Aufbau

Bit	Bezeichnung/ Funktion
0	Geschwindigkeitssollwert 0: Polarität positiv 1: Polarität negativ
1	Positionssollwert additiv 0: Polarität positiv 1: Polarität negativ
2	Geschwindigkeitsistwert 0: Polarität positiv 1: Polarität negativ
15..3	reserviert



Die Bits 1 und 2 sind Kopien von Bit 0.

Änderungen sind nur in Bit 0 wirksam. Unterschiedliche Einstellungen der einzelnen Bits sind nicht zulässig!

Bei **rotatorischen (drehenden) Motoren** gilt: Drehung der Motorabtriebswelle im Uhrzeigersinn führt zu positivem Geschwindigkeitsistwert (positive Polarität).

Bei **Linearmotoren** gilt: Bewegung des Primärteils in Richtung der Kabelanschluss-Seite führt zu positivem Geschwindigkeitsistwert (positive Polarität).

9.2.15 A-0-0049, Drehmoment-/Kraft-Polaritäten

Funktion In diesem Parameter kann die Polarität der angegebenen Drehmoment-/Kraftdaten auf die Anwendung bezogen invertiert werden.

Die Polarität wird außerhalb der Regelstrecke umgeschaltet, d.h. am Eingang und Ausgang der Regelstrecke.

Aufbau

Bit	Bezeichnung/ Funktion
0	Drehmomentsollwert 0: Polarität positiv 1: Polarität negativ
1	Drehmomentsollwert additiv 0: Polarität positiv 1: Polarität negativ
2	Drehmomentistwert 0: Polarität positiv 1: Polarität negativ
15..3	reserviert



Die Bits 1 und 2 sind Kopien von Bit 0.

Änderungen sind nur in Bit 0 wirksam. Unterschiedliche Einstellungen der einzelnen Bits sind nicht zulässig!

Bei **rotatorischen (drehenden) Motoren** gilt: Drehung der Motorabtriebswelle im Uhrzeigersinn bei positivem Drehmoment und positiver Polarität.

Bei **Linearmotoren** gilt: Bewegung des Primärteils in Richtung der Kabelanschluss-Seite bei positiver Kraft und positiver Polarität.

9.2.16 A-0-0224, Schnellhalt-Verzögerung

- Funktion** In diesem Parameter wird eingestellt, mit welchem maximalen Verzögerungswert die Achse bei
- Schnellhalt mit Rampe
- stillgesetzt wird.
- Die Einheit dieses Parameters wird durch "A-0-0053, Wichtungsart für Beschleunigungsdaten" festgelegt.
- Siehe auch Funktionsbeschreibung: "Fehlerreaktion"

9.2.17 A-0-0225, Schnellhalt-Ruck

- Funktion** In diesem Parameter wird eingestellt, mit welchem maximalen Ruckwert die Achse bei
- Schnellhalt mit Rampe
- stillgesetzt wird.
- Die Einheit dieses Parameters wird durch "A-0-0053, Wichtungsart für Beschleunigungsdaten" festgelegt.
- Siehe auch Funktionsbeschreibung: "Fehlerreaktion".



Dieser Parameter ist nur wirksam falls für die Achse "Interpolation in der Steuerung" gewählt wurde.

9.2.18 A-0-0240, Feedrate-Override

- Funktion** In diesem Parameter wird eingestellt, mit welchem Faktor die Geschwindigkeit der Achse verändert wird.
- Ein Wert von 100% bedeutet keine Veränderung.
- Bei einem Faktor von 0% wird die Warnung "E0112055, Feedrate-Override A-0-0240 = 0" ausgegeben.

9.2.19 A-0-0241, Feedrate-Override-Steuerwort

- Funktion** Dieser Parameter wirkt auf den Feedrate-Override, ohne dass der Feedrate-Override-Parameter A-0-0240 verändert wird.

Aufbau

Bit	Bezeichnung/ Funktion
0	reserviert
1	Feedrate-Override = 0
15-2	reserviert

Verwendung

Bit 1:

Mit dem Setzen des Bit 1 wird ein Feedrate-Override von 0 wirksam, ohne dass der eigentliche Parameterwert A-0-0240 verändert wird.



Feedrate-Override = 0 bedeutet "Halt", jedoch in diesem Fall ohne die Warnung "E0112055, Feedrate-Override A-0-0240 = 0"

Parameter

9.3 Steuerungsbezogene Parameter

9.3.1 Steuerungsbezogene Parameter, Allgemeines

Zur Verwaltung der Daten **aller** Kinematiken einer Steuerung werden nachstehende C-Parameter angelegt:

9.3.2 C-0-0123, Maximale Anzahl an Kinematiken

Funktion Dieser Parameter gibt die maximale Anzahl an Kinematiken an.

9.3.3 C-0-2480, Konfigurierte Kinematikenanzahl

Funktion Dieser Parameter zeigt die aktuelle Anzahl konfigurierter Kinematiken an.

9.3.4 C-0-2482, Liste der nicht verknüpften Achsen

Funktion Dieser Listenparameter zeigt die Achsen an, die noch keiner Kinematik zugeordnet sind.

Ist keine Kinematik konfiguriert, so entspricht diese Liste dem Parameter "C-0-0483, Liste der konfigurierten Achsen".

Die Zuordnung der Achsen zu den Kinematiken erfolgt über den Parameter "[K-0-0008](#), Achskonfigurationsliste".



Eine Achse kann nur einer Kinematik zugeordnet werden.

9.3.5 C-0-2483, Liste der konfigurierten Kinematiken

Funktion Dieser Parameter zeigt die konfigurierten Kinematiken des Systems.

Verwendung Beispiel:

3 Kinematiken (Kinematik 1, 2 und 3) sind im System konfiguriert.

[C-0-2483](#) zeigt:

Element	Wert	Kommentar
1	1	Kinematik 1 ist konfiguriert
2	2	Kinematik 2 ist konfiguriert
3	3	Kinematik 3 ist konfiguriert

9.3.6 C-0-2484, Kinematikkonfigurationsliste

Funktion Dieser Parameter definiert das Kinematiksystem. Er konfiguriert die Kinematiken. Es kann nur auf K-Parameter von Kinematiken zugegriffen werden, die hier konfiguriert sind.

Die Zuordnung der Achsen zur Kinematik erfolgt über den Parameter "[K-0-0008](#), Achskonfigurationsliste" (Beispiel K01: [K-0-0008](#)).

Aufbau Gültige Werte einer Kinematikkonfiguration:

Wert	Kommentar
0	keine Kinematik konfiguriert
2	Robot-Control V2-Kinematik konfiguriert

Verwendung Ein Beschreiben des Parameters "[C-0-2484](#), Kinematikkonfigurationsliste", aktualisiert die Parameter "[C-0-2480](#), Konfigurierte Kinematikenanzahl" und "[C-0-2483](#), Liste der konfigurierten Kinematiken".

Beispiel

Es sollen drei Kinematiken mit den logischen Nummer 1, 2 und 4 konfiguriert werden.

[C-0-2484](#) zeigt:

Element	Wert	Kommentar
1	2	Kinematik 1 ist konfiguriert
2	2	Kinematik 2 ist konfiguriert
3	0	Keine Kinematik 3 konfiguriert
4	2	Kinematik 4 konfiguriert

9.4 Kinematikbezogene Parameter

9.4.1 Kinematikbezogene Parameter, Allgemeines

Zur Verwaltung der Daten **einer** Robot-Control V2-Kinematik in der Steuerung werden nachstehende K-Parameter angelegt. Jeder Parameter existiert genau einmal pro Kinematik.

9.4.2 K-0-0001, Logische Kinematiknummer

Funktion Aus diesem Parameter kann die logische Kinematiknummer gelesen werden.

9.4.3 K-0-0002, Kinematikbezeichnung

Funktion In diesem Parameter steht der Name der Kinematik (z.B. Robi_1).

Mit diesem Namen wird automatisch im SPS-Programm eine globale Variable vom Typ AXIS_GROUP_REF angelegt.



Folgende Einschränkungen gelten:

- A...Z, a...z, 0..9, '_',
- erstes Zeichen muss ein Buchstabe sein,
- "Robi_1" identisch "ROBI_1".

9.4.4 K-0-0003, Maximale Bahngeschwindigkeit

Funktion Maximal programmierbare Bahngeschwindigkeit in Maschinenkoordinaten (MCS) am Tool Center Point (TCP).

Die Eingabe erfolgt in Abhängigkeit vom Parameter [K-0-0054](#) in mm/s oder in/s.

Der Parameter begrenzt die maximale Bahngeschwindigkeit der Kinematik in den Bewegungsarten Linear und Circular .

Es findet keine Begrenzung in der Bewegungsart PTP statt.

Wird der Wert des Parameters erreicht, so wird die Warnung E02C08CF gemeldet und beim Unterschreiten wieder zurückgesetzt.

9.4.5 K-0-0005, Software-Endschalter, positiv, MCS

Funktion Mit dem Parameter werden die Werte der positiven Software-Endschalter in Maschinenkoordinaten (MCS) festgelegt.

Die Eingabe erfolgt in Abhängigkeit vom Parameter [K-0-0054](#) in Millimeter [mm] oder Inch [in] bzw. in Grad.

Parameter

9.4.6 K-0-0006, Software-Endschalter, negativ, MCS

Funktion Mit dem Parameter werden die Werte der negativen Software-Endschalter in Maschinenkoordinaten (MCS) festgelegt.

Die Eingabe erfolgt in Abhängigkeit vom Parameter [K-0-0054](#) in Millimeter [mm] oder Inch [in] bzw. in Grad.

9.4.7 K-0-0007, Achsanzahl der Kinematik

Funktion Dieser Parameter enthält die aktuelle Achsanzahl der Kinematik.

Der Wert ergibt sich aus "[K-0-0008](#), Achskonfigurationsliste".

Er enthält die Summe aller dort konfigurierten Achsen und Bänder.

Der Parameter "[K-0-0070](#), Bandanzahl der Kinematik" enthält nur die Summe der Bänder.

9.4.8 K-0-0008, Achskonfigurationsliste

Funktion Dieser Parameter konfiguriert die Achsen für die Kinematik.

Aufbau Der Parameter ist ein Listenparameter mit folgender Dreifach-Struktur:

- Element 1: Achskonfiguration der Kinematikachse 1
- Element 2: Steuerungsnummer
- Element 3: zugeordnete Achse zur Kinematikachse 1
- Element 4: Achskonfiguration der Kinematikachse 2
- Element 5: Steuerungsnummer
- Element 6: zugeordnete Achse zur Kinematikachse 2
- ...

Die Elementanzahl muss geradzahlig sein.

Wert	Kommentar
0	keine Achse konfiguriert
8	Positionierachse konfiguriert
16	Orientierungsachse konfiguriert
32	Bandachse konfiguriert

Tab. 9-1: Gültige Werte einer Achskonfiguration

Wert	Kommentar
0	eigene Steuerung
1-64	Steuerungsnummer im C2C-Verbund

Tab. 9-2: Gültige Werte für die Steuerungsnummer

Verwendung**Aufbau der Dreifachliste**

Die Liste darf keine Lücken enthalten.

Einer Kinematik muss mindestens eine Positionierungsachse zugeordnet sein.

Orientierungsachsen müssen hinter den Positionierungsachsen liegen.

Bandachsen müssen hinter Positionierungsachsen und Orientierungsachsen liegen.

Abhängig vom eingestellten Parameter "[K-0-0031](#), Transformationsschema" wird eine unterschiedliche minimale Anzahl von Positionierungsachsen und

Parameter

Orientierungsachsen benötigt (siehe gewünschte Transformationsbeschreibung).

Es dürfen maximal vier Bandachsen definiert sein.

Die Dreifachliste darf in Summe maximal 16 Achsen konfigurieren.

Hilfen bei der Zuordnung

Jede Achse darf höchstens einer Kinematik als Positionierungsachse oder Orientierungsachse zugeordnet werden.

Im Parameter "C-0-2482, Liste der nicht verknüpften Achsen" sind die Achsen aufgelistet, die noch keiner Kinematik zugeordnet wurden.

Als Bandachse konfigurierte Kinematikachsen beeinflussen den Parameter "C-0-2482, Liste der nicht verknüpften Achsen" nicht.

Bandachsen können aus der "C-0-0483, Liste der konfigurierten Achsen" ausgewählt und mehreren Kinematiken zugewiesen werden.

Einfluss auf andere Parameter

Die Anzahl der Dreifach-Einträge in der Liste bestimmt den Parameter "K-0-0007, Achsanzahl der Kinematik" .

Die Anzahl der in der Liste konfigurierten Bandachsen bestimmt den Parameter "K-0-0070, Bandanzahl der Kinematik" .



Ist bei den Kinematikachsen die "A-0-0059, Wichtungsart für Lagdaten" auf Moduloformat eingestellt, wird auch geprüft, ob der "A-0-0045, Modulowert" größer Null und ganzzahlig ist.

Bei einer Robot-Control V2- Kinematik wird zusätzlich überprüft, ob die Wichtungen der zugeordneten Achsen zulässig sind.

Ist dies nicht der Fall, wird der Parameter "K-0-0008, Achskonfigurationsliste" in den Parameter "K-0-0011, Liste der ungültigen K-Parameter" eingetragen und eine fehlerhafte Konfiguration angezeigt (F0160040, Fehler im Cross-check K-Parameter bzw. F22B0024, Achse hat falsche Wichtung für Transformation).

Beispiel:

Die Kinematik benutzt das Transformationsschema 10 (SCARA) mit vier Achsen.

Achse 1, 2 und 3 sind Positionierachsen.

Achse 4 ist eine Orientierungsachse.

K-0-0008 zeigt:

Element	Wert	Kommentar
1	8	Kinematikachse 1: Positionierachse
2	0	Steuerungsnummer 0: eigene Steuerung
3	1	Achse mit logischer Achsnummer 1 zugeordnet
4	8	Kinematikachse 2: Positionierachse
5	0	Steuerungsnummer 0: eigene Steuerung
6	2	Achse mit logischer Achsnummer 2 zugeordnet
7	8	Kinematikachse 3: Positionierachse
8	0	Steuerungsnummer 0: eigene Steuerung

Parameter

Element	Wert	Kommentar
9	3	Achse mit logischer Achsnummer 3 zugeordnet
10	16	Kinematikachse 4: Orientierungsachse
11	0	Steuerungsnummer 0: eigene Steuerung
12	4	Achse mit logischer Achsnummer 4 zugeordnet

Für Positionier- und Orientierungsachsen stehen folgende Achstypen zur Verfügung (siehe C-0-0484):

- reale Achse,
- virtuelle Achse,
- Reglerachse.

Für Bandachsen stehen zusätzlich noch zur Verfügung:

- Geberachse und
- Verbundachse (Querverbund sercos II).

9.4.9 K-0-0009, Oszi: Liste aller oszilloskopierbaren K-Parameter

Funktion Dieser Parameter enthält die Liste aller oszilloskopierbaren K-Parameter.

9.4.10 K-0-0010, Liste aller K-Parameter

Funktion In dieser Liste stehen alle K-Parameter. Der Eintrag erfolgt im ID-Nummern-Format.

Die Einträge sind nach Bedeutung und Zusammengehörigkeit sortiert, also nicht nach Ident-Nummern oder alphabetischer Reihenfolge.

In der Reihenfolge dieser Einträge werden K-Parameter abgespeichert.

Nachstehend das ID-Nummern-Format:

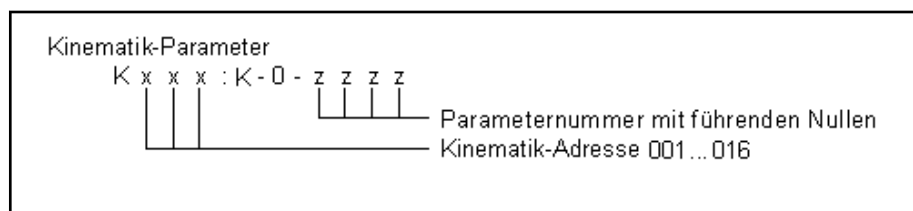


Abb. 9-1: Aufbau eines K-Parameters

Beispiel:

K001: [K-0-0002](#) - Kinematik 1; Parameter [K-0-0002](#).

9.4.11 K-0-0011, Liste aller ungültigen K-Parameter

Funktion Diese Liste zeigt alle ungültigen K-Parameter im Ergebnis

- eines Quersummentestes, z.B. nach einem Firmware-Update, oder
- eines Plausibilitätstestes, z.B. im Ergebnis einer illegalen Betriebsart.

Der Eintrag erfolgt im ID-Nummern-Format.

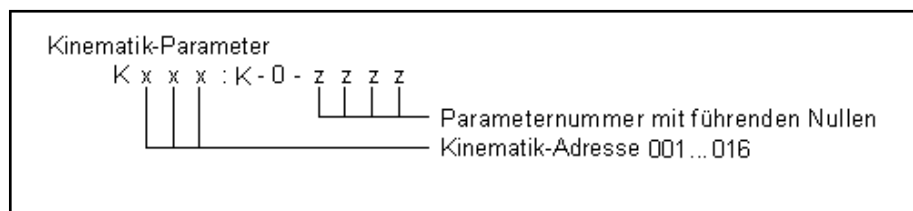


Abb. 9-2: Aufbau eines K-Parameters

Verwendung K001: [K-0-0008](#) - Kinematik 1; Parameter [K-0-0008](#).

9.4.12 K-0-0012, Prüfsumme aller K-Parameter

Funktion Beim Auslesen dieses Parameters wird die Prüfsumme über alle Parameterwerte gebildet, deren Identnummern im Parameter "[K-0-0013](#), IDN-Liste der K-Prüfsummenparameter" aufgelistet sind.

9.4.13 K-0-0013, IDN-Liste der K-Prüfsummenparameter

Funktion In diesen Parameter werden die Identnummern der Parameter eingetragen, deren Checksumme gebildet werden soll.

Die Checksummenbildung wird mit dem Auslesen des Parameters "[K-0-0012](#), Prüfsumme über alle K-Parameter" ausgelöst.

Im Defaultzustand sind in [K-0-0013](#) noch keine Identnummern eingetragen.

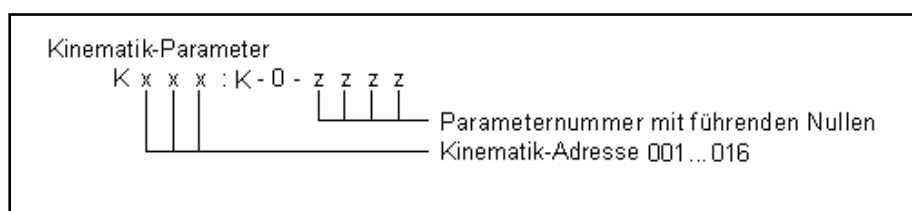


Abb. 9-3: Aufbau eines K-Parameters

Verwendung K001: [K-0-0008](#) - Kinematik 1; Parameter [K-0-0008](#).

9.4.14 K-0-0016, Projektidentifikationsnummer

Funktion IndraWorks generiert in diesem Parameter eine eindeutige Zeichenkette zur weltweiten Identifikation des Projektes.

Die Zeichenkette ist eineindeutig für jedes Projekt.



Dieser Parameter ist nur für interne Verwendung vorgesehen.

9.4.15 K-0-0019, Kinematiktyp

Funktion Dieser Parameter zeigt den Typ der Kinematik an.

Aufbau

Bit	Inhalt	Name des Bits
2..0	001: Robot-Control V1 010: Robot-Control V2	Kinematiktyp
31..3	reserviert	

Tab. 9-3: K-0-0019 Kinematiktyp

9.4.16 K-0-0020, Kinematikdiagnosemeldung

Funktion Dieser Parameter enthält die aktuelle Diagnosemeldung der Kinematik (Meldung, Warnung oder Fehler).

Wenn Bit 8 oder Bit 9 des Parameters "[K-0-0021](#), Kinematikstatus" auf 1 gesetzt ist, enthält dieser Parameter den Warnungs- (Bit 8) bzw. Fehlertext (Bit 9).

Aus der Sicht der SPS sind diese Informationen für die Kinematik "" verfügbar als

- KinData[*].dwKinStatus_i.8= TRUE, Diagnosemeldung in [K-0-0020](#) ist eine Warnungstext bzw.

Parameter

- KinData[*].dwKinStatus_i.9= TRUE, Diagnosemeldung in [K-0-0020](#) ist ein Fehlertext.



Hat die Kinematik keinen Fehler oder Warnung, so wird der aktuelle Zustand der Kinematik in Textform angezeigt.

9.4.17 K-0-0021, Kinematikstatus

Funktion Dieser Parameter beschreibt den aktuellen Status der Kinematik.

Dabei handelt es sich um interne Systeminformationen, die zum Beispiel über SPS-Funktionsbausteine ausgelöst werden. Es handelt sich dabei nicht um Informationen über den Zustand der Kinematik-Zustandsmaschine.

Aufbau

Bit	Inhalt	Name des Bits
5	Alle Achsen der Kinematik stehen. (Bandachsen werden nicht berücksichtigt).	Kin_StandStill
6	Sollposition = Istposition Die programmierte Endposition ist erreicht.	Kin_InPosition
7	Bandsynchron	Kin_BeltSynchronous
8	Warnung	Kin_Warning
9	Fehler	Kin_Error
10	Homing erforderlich, nicht alle Achsen sind referenziert.	Kin_NotHomed
11	Überschleifen	Kin_Blending
12	Transformierte Maschinenkoordinaten sind ungültig	Kin_TrafoValuesInvalid
14	Alle Achsen betriebsbereit (AH oder AF)	Kin_DrivesOn
17	Fehlerreaktion aktiviert	Kin_ErrorReaction
20	Voll Bandsynchron	Kin_FullyBeltSynchronous
21	BeginLength erreicht	Kin_BeginLengthReached
22	Warten auf bandsynchrone Startbedingung	Kin_WaitBeltStartCondition
28	Spline für Überschleifbewegung wird berechnet	
29	Spline ist gültig	

Tab. 9-4: K-0-0021 Kinematikstatus

Verwendung Bit 5: Alle Achsen der Kinematik stehen

Dieses Bit signalisiert, dass die durch die Kinematik positionierten Achsen in Stillstand sind (A-0-0021, Bit 5). Hierbei wird das Stillstandsfenster A-0-0220 berücksichtigt).

Bit 6: Die programmierte Endposition ist erreicht

Es existiert kein aktives Bewegungskommando.

Bit 7: Bandsynchron

Es wird teil- oder vollsynchron zu einem Band verfahren.

Bit 8: Warnung

Es steht eine Warnung an.

Bit 9: Fehler

Es steht ein Fehler an.

Bit 10: Homing erforderlich

Mindestens eine Achse der Kinematik ist nicht referenziert.

Bit 11: Überschleifen

Es wird gerade überschleift.

Bit 12: Transformierte Maschinenkoordinaten sind ungültig

Die in der KinData angezeigten aktuellen Maschinenkoordinaten der Kinematik sind nicht gültig. Dies ist zum Beispiel der Fall, wenn nicht alle Achsen der Kinematik zugeordnet sind oder die Transformation einen Fehler hat.

Bit 14: Alle Achsen Betriebsbereit

Alle Achsen der Kinematik sind betriebsbereit (AH oder AF).

Bit 17: Fehlerreaktion aktiv

Es ist gerade eine Fehlerreaktion aktiv.

Bit 20: Voll Bandsynchron

Es wird vollsynchron zu einem Band verfahren.

Bit 21: BeginLength erreicht (Bandsynchron)

Die Kinematik ist bandsynchron und die "Begin Length" wurde überschritten.

Bit 22: Warten auf Startbedingung (Bandsynchron)

Es wird auf die bandsynchrone Startbedingung gewartet.

Bit 28: Spline für Überschleifbewegung wird berechnet

Es wird gerade der Spline für die Überschleifbewegung berechnet.

Bit 29: Spline ist gültig

Der für die Überschleifbewegung berechnete Spline ist gültig. Das Bit bleibt so lange aktiv bis der Spline abgefahren wurde.

9.4.18 K-0-0022, Erweiterter Kinematikstatus

Funktion Dieser Parameter beschreibt den aktuellen Zustand der Kinematik-Zustandsmaschine mit Hilfe mehrerer Statusbits. Jeder Zustand ist dabei einer Statusbitkombination im Parameter [K-0-0022](#) zugeordnet.

	Kinematik Zustandsmaschine ist in
Bit 0: Kis_Errorstop	"Errorstop" oder "Errorstop Stopping"
Bit 1: Kis_Stopping	"Errorstop Stopping" oder "Stopping"
Bit 3: Kis_Incomplete	"Standstill Incomplete"
Bit 7: Kis_StandStill	"Standstill Complete" oder "Standstill Incomplete"
Bit 10: Kis_ManualMotion	"Manual Motion"
Bit 12: Kis_GroupMoving	"Group Moving"

Aufbau

9.4.19 K-0-0023, Kinematikdiagnosenummer

Funktion Wenn die Bits 8 oder 9 des Parameters "[K-0-0021](#), Kinematikstatus" gesetzt sind, kann in diesem Parameter die aktuelle Diagnosenummer gelesen werden.

Parameter

9.4.20 K-0-0028, Geschwindigkeitslimit Hilfsachsen

Funktion In diesem Parameter kann ein zusätzlicher Grenzwert für die Geschwindigkeit für jede Hilfsachse angegeben werden.

Je nach Bewegungsart hat der Parameter folgende Auswirkungen:

- Bei Linear und Circular wird das Minimum aus dem Geschwindigkeitsgrenzwert in diesem Parameter und dem Einzelachsgrenzwert der Hilfsachse gebildet. Dieses Minimum wird für die Begrenzung der Achsbewegung verwendet.
- Bei Punkt zu Punkt wird nur der Einzelachsgrenzwert der Hilfsachse für die Begrenzung der Achsbewegung verwendet.



Ob eine Hilfsachse verwendet wird, kann im Transformationschema nachgeschaut werden.



Da dieser Parameter in der Satzvorbereitung verwendet wird, wirkt ein neuer Wert nur auf Kommandos, die noch nicht durch die Satzvorbereitung vorbereitet wurden.

Aufbau

Element	Kommentar
1	1. Hilfsachse des Transformationsschemas (falls vorhanden)
2	2. Hilfsachse des Transformationsschemas (falls vorhanden)
...	...

9.4.21 K-0-0029, Beschleunigungslimit Hilfsachsen

Funktion In diesem Parameter kann ein zusätzlicher Grenzwert für die Beschleunigung für jede Hilfsachse angegeben werden.

Je nach Bewegungsart hat der Parameter folgende Auswirkungen:

- Bei Linear und Circular wird das Minimum aus dem Beschleunigungsgrenzwert in diesem Parameter und dem Einzelachsgrenzwert der Hilfsachse gebildet. Dieses Minimum wird für die Begrenzung der Achsbewegung verwendet.
- Bei Punkt zu Punkt wird nur der Einzelachsgrenzwert der Hilfsachse für die Begrenzung der Achsbewegung verwendet.



Ob eine Hilfsachse verwendet wird, kann im Transformationschema nachgeschaut werden.



Da dieser Parameter in der Satzvorbereitung verwendet wird, wirkt ein neuer Wert nur auf Kommandos, die noch nicht durch die Satzvorbereitung vorbereitet wurden.

Aufbau

Element	Kommentar
1	1. Hilfsachse des Transformationsschemas (falls vorhanden)

Parameter

2	2. Hilfsachse des Transformationsschemas (falls vorhanden)
...	...

9.4.22 K-0-0030, Rucklimit Hilfsachsen

Funktion In diesem Parameter kann ein zusätzlicher Grenzwert für den Ruck für jede Hilfsachse angegeben werden.

Je nach Bewegungsart hat der Parameter folgende Auswirkungen:

- Bei Linear und Circular wird das Minimum aus dem Ruckgrenzwert in diesem Parameter und dem Einzelachsgrenzwert der Hilfsachse gebildet. Dieses Minimum wird für die Begrenzung der Achsbewegung verwendet.
- Bei Punkt zu Punkt wird nur der Einzelachsgrenzwert der Hilfsachse für die Begrenzung der Achsbewegung verwendet.



Ob eine Hilfsachse verwendet wird, kann im Transformationsschema nachgeschaut werden.



Da dieser Parameter in der Satzvorbereitung verwendet wird, wirkt ein neuer Wert nur auf Kommandos, die noch nicht durch die Satzvorbereitung vorbereitet wurden.

Aufbau

Element	Kommentar
1	1. Hilfsachse des Transformationsschemas (falls vorhanden)
2	2. Hilfsachse des Transformationsschemas (falls vorhanden)
...	...

9.4.23 K-0-0031, Transformationsschema

Funktion

In diesem Parameter wird das Transformationsschema eingestellt.

Weitere Einstellungen für das Transformationsschema werden in den Parametern [K-0-0040](#) und [K-0-0041](#) vorgenommen werden.

9.4.24 K-0-0034, Maximale Bahnbeschleunigung (bipolar)

Funktion Maximal programmierbare Bahnbeschleunigung in Maschinenkoordinaten (MCS) am Tool Center Point (TCP).

Die Eingabe erfolgt in Abhängigkeit vom Parameter [K-0-0054](#) in mm/s² oder in/s².

Der Parameter begrenzt die maximale Bahnbeschleunigung der Kinematik in den Bewegungsarten Linear und Circular.

In der Bewegungsart PTP findet keine Begrenzung statt.

Wird der Wert des Parameters erreicht, so wird die Warnung E02C08CE gemeldet und beim Unterschreiten wieder zurückgesetzt.

Parameter

9.4.25 K-0-0035, Maximaler Bahnruck (bipolar)

Funktion Maximal programmierbarer Bahnruck in Maschinenkoordinaten (MCS) am Tool Center Point (TCP).

Die Eingabe erfolgt in Abhängigkeit vom Parameter [K-0-0054](#) in mm/s³ oder in/s³.

Der Parameter begrenzt den maximalen Bahnruck der Kinematik in den Bewegungsarten Linear und Circular. Es findet keine Begrenzung in der Bewegungsart PTP statt.

Wird der Wert des Parameters erreicht so wird die Warnung E02C08CD gemeldet und beim Unterschreiten wieder zurückgesetzt.

9.4.26 K-0-0037, Ruck für Move-Kommandos während Verzögerungsphase

Funktion Dieser Parameter gibt den Ruck für die Verzögerungsphase an. Dabei wird der prozentuale Anteil des programmierten Rucks für die Verzögerungsphase verwendet.

9.4.27 K-0-0040, Transformation, Längen der Mechanik

Funktion

In diesem Parameter werden Längen für das Transformationsschema im Parameter [K-0-0031](#) konfiguriert.

Die Bedeutung der einzelnen Werte hängt vom eingestellten [K-0-0031](#), Transformationsschema ab.

9.4.28 K-0-0041, Transformation, Faktoren

Funktion

In diesem Parameter werden die Faktoren für das Transformationsschema im Parameter [K-0-0031](#) konfiguriert.

Die Bedeutung der einzelnen Werte hängt vom eingestellten [K-0-0031](#), Transformationsschema ab.

9.4.29 K-0-0042, Verschiebung des BCS-Systems

Funktion In diesem Parameter werden die Verschiebung und die Verdrehung des Basiskoordinatensystems (BCS) der Kinematik gegenüber dem Maschinenkoordinatensystem (MCS) eingestellt.

Die Angabe erfolgt dabei im Maschinenkoordinatensystem.

Aufbau Dieser Parameter ist ein Listenparameter mit sechs Werten:

Element	Bedeutung
1	Verschiebung in kartesischer X-Richtung [in Millimeter bzw. Inch (siehe K-0-0054)]
2	Verschiebung in kartesischer Y-Richtung [in Millimeter bzw. Inch (siehe K-0-0054)]
3	Verschiebung in kartesischer Z-Richtung [in Millimeter bzw. Inch (siehe K-0-0054)]
4	Verdrehung um die kartesische X-Achse [in Grad] (siehe Hinweis)
5	Verdrehung um die kartesische Y-Achse [in Grad] (siehe Hinweis)
6	Verdrehung um die kartesische Z-Achse [in Grad] (siehe Hinweis)

Verwendung Die ersten drei Werte geben die Lage des Koordinatensystem-Ursprungs aus Sicht des Maschinen-Koordinatensystem an. Die Elemente 4 bis 6 geben die Rotation des Koordinatensystems an.



Die Verdrehung des Basiskoordinatensystems ist nur in den Freiheitsgraden möglich die das Transformationschema unterstützt.

So besitzt z.B. das Scara-Transformationsschema als Orientierung nur den Freiheitsgrad um die Z-Achse. Es kann also nur eine Verdrehung des Basiskoordinatensystems um die Z-Achse vorgenommen werden.

9.4.30 K-0-0050, Distanzeinheiten MCS

Funktion In diesem Text-Parameter sind die Einheiten für Distanzen und Orientierung für jedes Element eines Listenparameters im Maschinen-Koordinatensystem (MCS), mit Kommata getrennt, hinterlegt.

- Nur lesbar - wird automatisch ermittelt.
- Für jede Koordinatenachse (x, y, z, alpha, beta, gamma) ein Eintrag (mit Kommata getrennt)
- Einheit abhängig vom Trafo-Typ [K-0-0031](#) und der verwendeten Achsen [K-0-0008](#).
 - für x-, y-, z-Achsen: mm bzw inch (siehe [K-0-0054](#))
 - für alpha, beta, gamma: Grad

Verwendung *Beispiele*

1. Karteser mit X, Y, Z, C

[K-0-0050](#) = „mm,mm,mm,Grad“ bzw. "inch,inch,inch,Grad" (siehe [K-0-0054](#))

2. Scara mit Hub-Drehspindel (X, Y, Z, C)

[K-0-0050](#) = „mm,mm,mm,Grad“ bzw. "inch,inch,inch,Grad" (siehe [K-0-0054](#))

3. Karteser mit X, Y, Z, C, B

[K-0-0050](#) = „mm,mm,mm,Grad,Grad“ bzw. "inch,inch,inch,Grad,Grad" (siehe [K-0-0054](#))

4. Karteser mit X, Y, C

[K-0-0050](#) = „mm,mm,Grad“ bzw. "inch,inch,Grad" (siehe [K-0-0054](#))

5. Delta mit C-Achse und 2 Bändern

[K-0-0050](#) = „mm,mm,mm,Grad“ bzw. "inch,inch,inch,Grad" (siehe [K-0-0054](#))



Die Bänder werden in den Listen für das Maschinen-Koordinatensystem (MCS) nicht berücksichtigt.

9.4.31 K-0-0051, Geschwindigkeitseinheiten MCS

Funktion In diesem Text-Parameter sind die Einheiten für Geschwindigkeiten für jedes Element eines Listenparameters, mit Kommata getrennt, hinterlegt.

- Nur lesbar - wird automatisch ermittelt.
- Für jede Koordinatenachse (x, y, z, alpha, beta, gamma) ein Eintrag (mit Kommata getrennt)
- Einheit abhängig vom Trafo-Typ [K-0-0031](#) und der verwendeten Achsen [K-0-0008](#).

Parameter

- für x-, y-, z-Achsen: mm/s bzw inch/s (siehe [K-0-0054](#))
- für alpha, beta, gamma: Grad/s

Verwendung *Beispiele*

1. Karteser mit X, Y, Z, C

[K-0-0051](#) = „mm/s,mm/s,mm/s,Grad/s“ bzw. "inch/s,inch/s,inch/s,Grad/s" (siehe [K-0-0054](#))

2. Scara mit Hub-Drehspindel (X, Y, Z, C)

[K-0-0051](#) = „mm/s,mm/s,mm/s,Grad/s“ bzw. "inch/s,inch/s,inch/s,Grad/s" (siehe [K-0-0054](#))

3. Karteser mit X, Y, Z, C, B

[K-0-0051](#) = „mm/s,mm/s,mm/s,Grad/s,Grad/s“ bzw. "inch/s,inch/s,inch/s,Grad/s,Grad/s" (siehe [K-0-0054](#))

4. Karteser mit X, Y, C

[K-0-0051](#) = „mm/s,mm/s,Grad/s“ bzw. "inch/s,inch/s,Grad/s" (siehe [K-0-0054](#))

5. Delta mit C-Achse und 2 Bändern

[K-0-0051](#) = „mm/s,mm/s,mm/s,Grad/s“ bzw. "inch/s,inch/s,inch/s,Grad/s" (siehe [K-0-0054](#))



Die Bänder werden in den Listen für das Maschinen-Koordinatensystem (MCS) nicht berücksichtigt.

9.4.32 K-0-0052, Beschleunigungseinheiten MCS

Funktion In diesem Text-Parameter sind die Einheiten für Beschleunigung bzw. Bremsverzögerung für jedes Element eines Listenparameters, mit Kommata getrennt, hinterlegt.

- Nur lesbar - wird automatisch ermittelt.
- Für jede Koordinatenachse (x, y, z, alpha, beta, gamma) ein Eintrag (mit Kommata getrennt)
- Einheit abhängig vom Trafo-Typ [K-0-0031](#) und der verwendeten Achsen [K-0-0008](#).
 - für x-, y-, z-Achsen: mm/s² bzw inch/s²
 - für alpha, beta, gamma: Grad/s²

Verwendung *Beispiele*

1. Karteser mit X, Y, Z, C

[K-0-0052](#) = „mm/s²,mm/s²,mm/s²,Grad/s²“ bzw. "inch/s²,inch/s²,inch/s²,Grad/s²" (siehe [K-0-0054](#))

2. Scara mit Hub-Drehspindel (X, Y, Z, C)

[K-0-0052](#) = „mm/s²,mm/s²,mm/s²,Grad/s²“ bzw. "inch/s²,inch/s²,inch/s²,Grad/s²" (siehe [K-0-0054](#))

3. Karteser mit X, Y, Z, C, B

[K-0-0052](#) = „mm/s²,mm/s²,mm/s²,Grad/s²,Grad/s²“ bzw. "inch/s²,inch/s²,inch/s²,Grad/s²,Grad/s²" (siehe [K-0-0054](#))

4. Karteser mit X, Y, C

[K-0-0052](#) = „mm/s²,mm/s²,Grad/s²“ bzw. "inch/s²,inch/s²,Grad/s²" (siehe [K-0-0054](#))

5. Delta mit C-Achse und 2 Bändern

K-0-0052 = „mm/s²,mm/s²,mm/s²,Grad/s²,“ bzw. "inch/s²,inch/s²,inch/s²,Grad/s²" (siehe **K-0-0054**)



Die Bänder werden in den Listen für das Maschinen-Koordinatensystem (MCS) nicht berücksichtigt.

9.4.33 K-0-0053, Ruckeinheiten MCS

Funktion In diesem Text-Parameter sind die Einheiten für Ruck für jedes Element eines Listenparameters, mit Kommata getrennt, hinterlegt.

- Nur lesbar - wird automatisch ermittelt.
- Für jede Koordinatenachse (x, y, z, alpha, beta, gamma) ein Eintrag (mit Kommata getrennt)
- Einheit abhängig vom Trafo-Typ **K-0-0031** und der verwendeten Achsen **K-0-0008**.
 - für x-, y-, z-Achsen: mm/s³ bzw inch/s³
 - für alpha, beta, gamma: Grad/s³

Verwendung *Beispiele*

1. Karteser mit X, Y, Z, C

K-0-0053 = „mm/s³,mm/s³,mm/s³,Grad/s³“ bzw. "inch/s³,inch/s³,inch/s³,Grad/s³" (siehe **K-0-0054**)

2. Scara mit Hub-Drehspindel (X, Y, Z, C)

K-0-0053 = „mm/s³,mm/s³,mm/s³,Grad/s³“ bzw. "inch/s³,inch/s³,inch/s³,Grad/s³" (siehe **K-0-0054**)

3. Karteser mit X, Y, Z, C, B

K-0-0053 = „mm/s³,mm/s³,mm/s³,Grad/s³,Grad/s³“ bzw. "inch/s³,inch/s³,inch/s³,Grad/s³,Grad/s³" (siehe **K-0-0054**)

4. Karteser mit X, Y, C

K-0-0053 = „mm/s³,mm/s³,Grad/s³“ bzw. "inch/s³,inch/s³,Grad/s³" (siehe **K-0-0054**)

5. Delta mit C-Achse und 2 Bändern

K-0-0053 = „mm/s³,mm/s³,mm/s³,Grad/s³“ bzw. "inch/s³,inch/s³,inch/s³,Grad/s³" (siehe **K-0-0054**)



Die Bänder werden in den Listen für das Maschinen-Koordinatensystem (MCS) nicht berücksichtigt.

9.4.34 K-0-0054, Maßeinheit für Kinematikdaten

Funktion Dieser Parameter bestimmt die Wichtungseinheit der Kinematik.

Aufbau Folgende Einstellungen können vorgenommen werden:

Bit	Bezeichnung/ Funktion
0	Einheit 0: Millimeter [mm]. 1: Inch [in].
15..1	reserviert

Parameter

9.4.35 K-0-0070, Bandanzahl der Kinematik

- Funktion** Dieser Parameter enthält die aktuelle Bandanzahl der Kinematik.
- Der Wert ergibt sich aus "K-0-0008, Achskonfigurationsliste". Er enthält nur die Summe der dort konfigurierten Bänder.
- Der Parameter "K-0-0007, Achsanzahl der Kinematik" enthält die Summe aller Achsen und Bänder.

9.4.36 K-0-0090, Taskpriorität Satzvorbereitung

- Funktion** In diesem Parameter wird die Priorität der Satzvorbereitung angepasst.
- Taskpriorität Satzvorbereitung*
- 20 - Die Satzvorbereitung wird bevorzugt vor der Kommunikation abgearbeitet
 - 21 - Die Priorität der Satzvorbereitung liegt auf dem Level der Kommunikation und wird gleichberechtigt zu dieser abgearbeitet (Round Robin)
 - 22 - Default: Die Satzvorbereitung wird nach der Kommunikation abgearbeitet



Dieser Parameter sollte nur von erfahrenen Anwendern angepasst werden, da das Systemverhalten auf den Defaultwert ausgelegt ist.

9.4.37 K-0-0095, Bewegungsidentifikationsnummer

- Funktion** Dieser Parameter gibt die aktuelle Bewegungsidentifikationsnummer (**MoveID**) an.
- Der ganzzahlige Teil verweist auf die Nummer der Bewegungsanweisung.
- Der Dezimalteil zeigt das Fortschreiten in % der Bewegung an.
- Die **MoveID** kann manuell auf einen ganzzahligen Wert gesetzt werden.



Bei **Jog** wird die **MoveID** nicht erhöht.

Die **MoveID** sollte nicht mit "=="-Operator getestet werden.

9.4.38 K-0-0096, Höchstwert der Kommandopuffergröße

- Funktion** Dieser Parameter zeigt den Höchstwert der Kommandopuffergröße an, der erreicht wurde. Ein Zurücksetzen des Wertes ist durch Schreiben kleinerer Werte möglich.

9.4.39 K-0-0097, Maximale Kommandopuffergröße

- Funktion** Dieser Parameter gibt die maximale Anzahl von Kommandos an, die im Satzpuffer gleichzeitig gespeichert werden können.

9.4.40 K-0-0098, Aktuelle Kommandopuffergröße

- Funktion** Dieser Parameter zeigt die aktuelle Anzahl von Kommandos im Satzpuffer an.

9.4.41 K-0-0099, Kinematikkonfiguration

- Funktion** Dieser Parameter konfiguriert das Verhalten der Kinematik.

Parameter

Aufbau

Bit	Bezeichnung/ Funktion
0	Aktiviert das automatische Umschalten von CONT_SLOPE auf BLOCK_SLOPE
1	Aktiviert den Achsmodulo jeder Bandachse als Bandmodulo
2	Aktiviert den vollen Radius für das Überschleifen mit BLOCK_SLOPE
3	Aktiviert die Option mit mehreren Bändern gleichzeitig synchron zu sein
4	Aktiviert die Option, dass mehr als zwei Kommandos parallel im BLOCK_SLOPE aktiv sein können
5	Deaktiviert alle Wichtungsüberprüfungen der Achsen der Kinematik
6	Aktiviert das adaptive Interpolieren für bandsynchrone Bewegungen
7	Aktiviert das Kommandieren mit dem vollen Positionsvektor (X,Y,Z)
8	Aktiviert das Kommandieren mit dem vollen Orientierungsvektor (A,B,C)
9..31	reserviert

Verwendung

Bit 0: Aktiviert das automatische Umschalten von CONT_SLOPE auf BLOCK_SLOPE

Ist das automatische Umschalten aktiviert, dann wird auf die Überschleifart BLOCK_SLOPE umgeschaltet wenn das Überschleifen mit CONT_SLOPE fehlgeschlagen ist.

Bit 1: Aktiviert den Achsmodulo jeder Bandachse als Bandmodulo

Ist Bit 1 gesetzt wird der Achsmodulo jeder Bandachse als Bandmodulowert verwendet. Dabei ist zu berücksichtigen, dass der Bandarbeitsbereich maximal Bandmodulo/2 sein darf.

Bit 2: Aktiviert den vollen Radius für das Überschleifen mit BLOCK_SLOPE

Ist Bit 2 gesetzt wird der Überschleifradius für BLOCK_SLOPE nicht mehr auf das Minimum von Radius und Pfaddistanz/2, des kürzeren Satzes, begrenzt.

Bit 3: Aktiviert die Option mit mehreren Bändern gleichzeitig synchron zu sein

Ist Bit 3 gesetzt besteht die Möglichkeit zu mehreren Bändern gleichzeitig synchron zu sein. Die Bänder müssen linear unabhängig zu einander stehen, d. h. in einem Winkel von 90 Grad zu einander verdreht sein.

Bit 4: Aktiviert die Option, dass mehr als zwei Kommandos parallel im BLOCK_SLOPE aktiv sein können

Ist Bit 4 gesetzt können mehr als zwei Kommandos parallel im BLOCK_SLOPE aktiv sein. Dadurch werden mehr als 2 Kommandos überschleift. Ein drittes Kommando wird auch dann gestartet, wenn es im Überschleifradius ist und das erste Kommando noch nicht beendet wurde. Hier ist zu beachten, dass die resultierende Bahn durch das Überschleifen von mehr als zwei Kommandos sehr undefiniert ist. z.B. können sich zwei gegenläufige Sätze aufheben.

Bit5: Deaktiviert alle Wichtungsüberprüfungen der Achsen der Kinematik

Ist Bit 5 gesetzt werden alle Wichtungsüberprüfungen der Achsen der Kinematik abgeschaltet. Die Wichtung der Achsen muss dann nicht mehr zur Transformation passen. Es ist möglich nicht translatorisch gewichtete Achsen für lineare Bänder zu verwenden.

Parameter

**VORSICHT****Nicht erwartete Achsbewegungen**

Die Deaktivierung der Wichtungsprüfung kann zu unerwarteten Achsbewegungen führen.

Bit6: Aktiviert das adaptive Interpolieren für bandsynchrone Bewegungen

Ist Bit 6 gesetzt werden, bandsynchrone Bewegungen, bei Bedarf adaptiv interpoliert, d.h. Warte- bzw. Stillstandszeiten der Kinematik, bis der vollsynchrone Endpunkte einer Bewegung den bandsynchronen Arbeitsbereich erreicht hat, entfallen. Der vollsynchrone Endpunkt wird von der Kinematik im gleichen Augenblick erreicht, sobald dieser den bandsynchronen Arbeitsbereich erreicht hat. Die Interpolation der bandsynchronen Bewegung findet in Abhängigkeit des vollsynchrone Endpunkts statt.

Bit7: Aktiviert das Kommandieren mit dem vollen Positionsvektor (X,Y,Z)

Ist Bit 7 gesetzt, können Punkte mit allen Positionskoordinaten kommandiert werden. Der Vektor wird in diesem Modus nicht mehr auf die von der Kinematik benötigten Koordinaten beschränkt.

Bit8: Aktiviert das Kommandieren mit dem vollen Orientierungsvektor (A,B,C)

Ist Bit 8 gesetzt, können alle Freiheitsgrade der Orientierung kommandiert werden. Dadurch wird es möglich, Kinematiken über eine Transformation hinaus im Raum zu verdrehen.

9.4.42 K-0-0100, Aktuelle Position, Kartesische Maschinenkoordinaten

Funktion Der Parameter zeigt die aktuelle Sollposition der Kinematik im Maschinen-Koordinatensystem (MCS) an.

Die aktuelle Position besteht je nach Transformationsschema ([K-0-0031](#)) und Achskonfiguration ([K-0-0008](#)) aus einer Liste von kartesischen Raumkoordinaten, der Orientierung und den Bandzählern.

9.4.43 K-0-0102, Aktuelle Bahngeschwindigkeit

Funktion Dieser Parameter liefert die aktuelle Bahngeschwindigkeit der Kinematik am TCP.

Es werden nur die Positionierungsachsen berücksichtigt, nicht die Orientierungs- oder Bandachsen.

Die Einheit ist mm/s bzw. in/s (siehe [K-0-0054](#)).

9.4.44 K-0-0103, Aktuelle Bahnbeschleunigung

Funktion Dieser Parameter liefert die aktuelle Bahnbeschleunigung der Kinematik am TCP.

Es werden nur die Positionierungsachsen berücksichtigt, nicht die Orientierungs- oder Bandachsen.

Die Einheit ist mm/s² bzw. in/s².

9.4.45 K-0-0104, Aktueller Bahnruck

Funktion Dieser Parameter liefert den aktuellen Bahnruck der Kinematik am TCP.

Es werden nur die Positionierungsachsen berücksichtigt, nicht die Orientierungs- oder Bandachsen.

Die Einheit ist mm/s³ bzw. in/s³.

9.4.46 K-0-0150, Interpolierte Sollposition

Funktion Der Parameter zeigt die aktuelle interpolierte Sollposition der Kinematik im Maschinen-Koordinatensystem (MCS) an.
Die aktuelle Position besteht je nach Transformationsschema ([K-0-0031](#)) und Achskonfiguration ([K-0-0008](#)) aus einer Liste von kartesischen Raumkoordinaten, der Orientierung und den Bandzählern.

9.4.47 K-0-0152, Interpolierte Bahngeschwindigkeit

Funktion Dieser Parameter liefert die aktuelle interpolierte Bahngeschwindigkeit der Kinematik am TCP.
Es werden nur die Positionierungsachsen berücksichtigt, nicht die Orientierungs- oder Bandachsen.
Die Einheit ist mm/s bzw. in/s (siehe [K-0-0054](#)).

9.4.48 K-0-0153, Interpolierte Bahnbeschleunigung

Funktion Dieser Parameter liefert die aktuelle interpolierte Bahnbeschleunigung der Kinematik am TCP.
Es werden nur die Positionierungsachsen berücksichtigt, nicht die Orientierungs- oder Bandachsen.
Die Einheit ist mm/s² bzw. in/s².

9.4.49 K-0-0154, Interpolierter Bahnruck

Funktion Dieser Parameter liefert den aktuellen interpolierten Bahnruck der Kinematik am TCP.
Es werden nur die Positionierungsachsen berücksichtigt, nicht die Orientierungs- oder Bandachsen.
Die Einheit ist mm/s³ bzw. in/s³.

9.4.50 K-0-0223, Auswahl Fehlerreaktion

Funktion Mit diesem Parameter kann das Verhalten der Kinematik im Fehlerfall festgelegt werden.

Aufbau

Bit	Bezeichnung/ Funktion
0	0: Es werden die Grenzwerte der aktiven Kommandos verwendet 1: Es werden die Grenzwerte der Parametern K-0-0224 , Nothaltverzögerung und K-0-0225 , Nothaltruck verwendet
1..31	Reserviert

Verwendung Bit0= 0: Es werden Grenzwerte des aktiven Kommandos verwendet

Die Kinematik hält im Fehlerfall mit den Grenzwerten des aktiven Kommandos an. Wird während der Fehlerreaktion ein anderes Kommando aktiv (z.B. durch Überschleifen), werden die Grenzwerte des neuen aktiven Kommandos verwendet.

Bit0= 1: Es werden die Grenzwerte der Parametern [K-0-0224](#), Nothaltverzögerung und [K-0-0225](#), Nothaltruck verwendet

Die Kinematik hält im Fehlerfall mit den Grenzwerten der Parameter [K-0-0224](#), Nothaltverzögerung und [K-0-0225](#), Nothaltruck an. Diese Grenzwerte werden nur verwendet, wenn sie größer als das jeweils aktive Kommando sind. Dabei wird die Nothaltverzögerung und der Nothaltruck separat betrachtet: ist die programmierte Verzögerung kleiner als die Nothaltverzögerung

Parameter

rung, so wird die Nothaltverzögerung benutzt, ist der programmierte Ruck größer als der Nothaltruck, wird der programmierte Ruck verwendet.

9.4.51 K-0-0224, Nothaltverzögerung

Funktion Der Parameter legt die Nothaltverzögerung fest. Der Parameter ist nur wirksam, wenn im Parameter [K-0-0223](#), Auswahl Fehlerreaktion, das Bit 0 gesetzt ist.

9.4.52 K-0-0225, Nothaltruck

Funktion Der Parameter legt den Nothaltruck fest. Der Parameter ist nur wirksam, wenn im Parameter [K-0-0223](#), Auswahl Fehlerreaktion, das Bit 0 gesetzt ist.

9.4.53 K-0-0240, Feedrate-Override

Funktion

Mit diesem Parameter kann die Sollgeschwindigkeit zur Laufzeit prozentual zum programmierten Wert geändert werden.

Ein Wert von 100% bedeutet, dass mit der vorgegebenen (programmierten) Geschwindigkeit die Achsen verfahren werden.

Der Prozentwert kann zwischen 0% und 600% eingestellt werden.



Bei 0% bleiben die Achsen stehen, ohne den Zustand der Kinematik-Zustandsmaschine zu ändern.

9.4.54 K-0-0241, Feedrate-Override-Steuerwort

Funktion Mit diesem Parameter wird der Feedrate-Override gesteuert.

Aufbau

Bit	Bezeichnung/ Funktion
0	reserviert
1	Bit 1 = 0: Feedrate-Override im K-0-0240 ist wirksam, Bit 1 = 1: Vorschub Halt, der Wert in K-0-0240 bleibt unverändert.
15..2	reserviert

Verwendung

Bit 1:

Mit Setzen des Bit 1 wird der Vorschub Halt wirksam.

9.4.55 K-0-0242, Acceleration-Override

Funktion

Mit diesem Parameter kann die Beschleunigung zur Laufzeit prozentual zum programmierten Wert geändert werden.

Ein Wert von 100% bedeutet, dass mit der vorgegebenen (programmierten) Beschleunigung die Achsen verfahren werden.

Der Prozentwert kann zwischen 1% und 600% eingestellt werden.



Das Setzen von [K-0-0242](#) wirkt in der Satzvorverarbeitung, also nicht sofort.

9.4.56 K-0-0243, Deceleration-Override

Funktion

Mit diesem Parameter kann die Bremsverzögerung zur Laufzeit prozentual zum programmierten Wert geändert werden.

Parameter

Ein Wert von 100% bedeutet, dass mit der vorgegebenen (programmierten) Bremsverzögerung die Achsen verfahren werden.

Der Prozentwert kann zwischen 1% und 600% eingestellt werden.



Das Setzen von [K-0-0243](#) wirkt in der Satzvorverarbeitung, also nicht sofort.

Kleine Werte in [K-0-0243](#) führen zu sehr langen Bremswegen.

9.4.57 K-0-0250, Continuous Slope Vorhaltezeit

Funktion Über diesen Parameter lässt sich die Vorhaltezeit für die Berechnung der Überschleifbewegung bei CONT_SLOPE angeben.

Verwendung Aus der Vorhaltezeit und der Pfadgeschwindigkeit wird eine Vorhaldedistanz gebildet.

Wird durch das späte Absetzen eines Verfahrbefehls das späte Überschleifen aktiviert, wird die aktuelle Position gespeichert.

Ist am Ende der Berechnung der Überschleifbewegung der Abbiegepunkt (Vorhaldedistanz + gespeicherte aktuelle Position) überschritten, wird ein Fehler geworfen.

Ist der Abbiegepunkt nicht überschritten, wird die Überschleifbewegung (bei genügend großem Überschleifradius) am Abbiegepunkt gestartet.

9.4.58 K-0-0815, Kilometerzähler

Funktion Dieser Parameter zeigt den verfahrenen Weg des "Tool Center Points" (TCP) im Kilometern an.

9.4.59 K-0-1000, Kommando: Basisparameter der Kinematik laden

Funktion Dieser Parameter wird als Kommando zum Laden der Basisparameter einer Kinematik genutzt.

Verwendung *Von IndraWorks MLC aus:*

- Wizard "Erstinbetriebnahme" im **Kontextmenü der Kinematik** ► **Erstinbetriebnahme** ► **Schritt 1: Basisparameter laden...**
- Schaltfläche "Basisparameter laden" im **Kontextmenü der Kinematik** ► **Parameter** ► **Basisparameter laden....**

Vom Feldbus aus:

- Eingabe des Binärwertes "0000000000000011" (=16#0003) in den Parameter ([K-0-1000](#)).

Dies löst den Ladevorgang der Basisparameter der Kinematik aus.

- Nach dem Ladevorgang setzt die MLC den Parameter automatisch wieder auf "16#00" zurück.

9.4.60 K-0-2100, Liste der zu sichernden K-Parameter

Funktion In dieser Liste stehen alle K-Parameter, die bei einer Parametersicherung zu berücksichtigen sind. Der Eintrag erfolgt im ID-Nummern-Format.

Die Einträge sind nach Bedeutung und Zusammengehörigkeit sortiert, also nicht nach Ident-Nummern oder alphabetischer Reihenfolge.

In der Reihenfolge dieser Einträge werden K-Parameter abgespeichert.

Nachstehend das ID-Nummern-Format:

Parameter

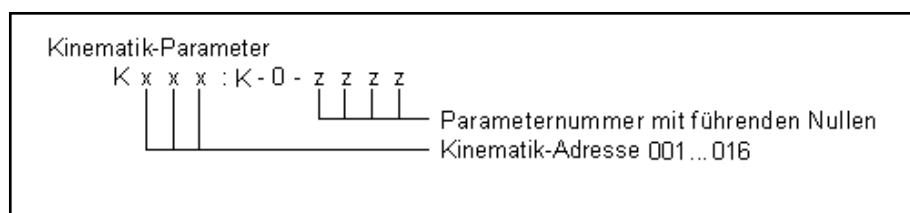


Abb. 9-4: Aufbau eines K-Parameters

Beispiel:

K001: [K-0-0002](#) - Kinematik 1; Parameter [K-0-0002](#).

10 Diagnosen

Das Diagnosesystem der IndraMotion MLC unterstützt die Arbeit mit Kinematiken auf der Steuerung.

Die vollständige Sammlung der Meldungen, Warnungen und Fehler ist enthalten in

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnosen

DOK-XLCMLC-DIAG****V14-RERS-DE-P, R911341480

11 Anbindung von Robotik-Fremdsystemen

11.1 Allgemeine Information

In der allgemeinen Automatisierung werden häufig spezielle Robotikanwendungen benötigt, welche nahtlos in den Automatisierungsprozess einer Automationsanlage integriert werden müssen. Um den Anwendern ein einfaches Engineering zu ermöglichen, bietet Bosch Rexroth mit der Motion Logic Steuerung (MLC) eine Lösung an, die die Vorteile spezialisierter Roboterteuerungen mit der Vielseitigkeit der Bosch Rexroth Multiachssteuerungen vereint. Somit entsteht ein einheitliches, durchgängiges "Look and feel", dass es dem Anwender ermöglicht, die gesamte Anlage mit einem Engineering Tool (IndraWorks) zu programmieren und zu parametrieren. Damit lässt sich der Aufwand für das Engineering, sowie für die Inbetriebnahme, deutlich reduzieren und vereinfachen.

11.2 uniVAL drive

11.2.1 Übersicht

Mit uniVAL drive bietet die Firma Stäubli eine, am Automatisierungsmarkt etablierte, anschlussfertige Lösung an, die es ermöglicht, die gesamte Stäubli Roboterpalette über den standardisierten Echtzeitbus Sercos an überlagerte Multiachssteuerungen anzubinden. Dem Anwender erschließt sich auf diese Weise die Möglichkeit, Stäubli-Roboter mit den bewährten Bosch Rexroth Steuerungssystemen zu programmieren, sowie über IndraWorks zu parametrieren.

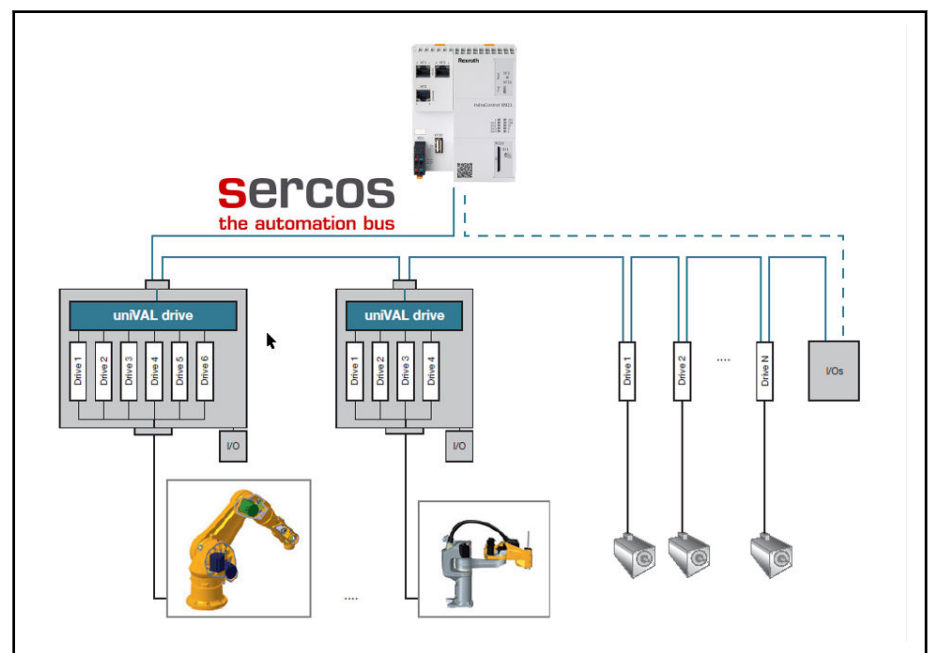


Abb. 11-1: Schematischer Topologieaufbau Bosch Rexroth Steuerungstechnik mit uniVAL drive

11.2.2 Achsen anlegen

Um Roboter der Firma Stäubli an einer MLC betreiben zu können, müssen zunächst die zugehörigen Antriebe der Kinematik in IndraWorks konfiguriert werden.

Dafür ist wie folgt vorzugehen:

Anbindung von Robotik-Fremdsystemen

- Die MLC mit der entsprechenden Robotersteuerung von Stäubli über den Sercos-Bus verbinden, siehe [Abb. 11-1 "Schematischer Topologieaufbau Bosch Rexroth Steuerungstechnik mit uniVAL drive" auf Seite 451](#) (Robotersteuerung von Stäubli muss über die entsprechende Kommunikationsschnittstelle verfügen)
- Integration der Antriebe erfolgt als generischer SercosDrive (siehe auch Kapitel 6.3.1 "Allgemeines", Doku IndraLogic XLC IndraMotion MLC 14VRS Funktionsbeschreibung (DOK-XLCMLC-FUNC****V14-APRS-DE-P))
- Achsen in IndraWorks anlegen (siehe auch "Anlegen einer realen Achse", Doku IndraLogic XLC IndraMotion MLC 14VRS Funktionsbeschreibung (DOK-XLCMLC-FUNC****V14-APRS-DE-P))

11.2.3 Achsen konfigurieren

Die Konfiguration der einzelnen Stäubli-Roboter sind herstellerspezifisch und somit der jeweiligen Dokumentation von Stäubli zu entnehmen. Informationen hierzu finden Sie auf den entsprechenden Internetseiten des Herstellers (www.staubli.com/de/robotik/).



- Die Achsen müssen über die notwendigen A-Parameter konfiguriert werden. Wo es notwendig ist werden diese in die entsprechenden S- Parameter synchronisiert
- Bei Vorgabe der Wichtungen für die einzelnen Achsen ist die Herstellervorgabe durch Firma Stäubli zu beachten, da für einen fehlerfreien Betrieb explizite Wertvorgaben erfolgen müssen

11.2.4 Konfiguration der Kinematiktransformation

Durch die Robot-Control werden verschiedene Möglichkeiten angeboten, Transformationen zu konfigurieren. Wie eine Kinematik hinzugefügt oder Transformationen konfiguriert werden können, sind dem Kapitel [Kap. 4 "Transformationsschemen" auf Seite 111](#) zu entnehmen.

12 Umstieg von Robot-Control V1 auf Robot-Control V2

12.1 Überblick

Beim Umstieg von Robot-Control V1 auf Robot-Control V2 muss das vorliegende RCL- bzw. IEC-Programm konvertiert sowie die Konfiguration im IndraWorks-Projekt angepasst werden.

Diese Konvertierungsanleitung soll kurz beschreiben, welche Schritte hierfür durchzuführen sind. Diese Schritte sind allerdings davon abhängig, ob ein RCL- oder IEC-Programm vorliegt.

- Basierend auf einem RCL-Programm ist vorbereitend eine Funktion, ein Funktionsbaustein bzw. ein Programm zu erstellen, in dem die Befehle und Programmabläufe des RCL-Programms integriert werden. Ob eine Funktion, ein Funktionsbaustein oder ein Programm erstellt wird, ist hierbei anwendungsabhängig. Auch die Task, in der die Funktion, der Funktionsbaustein oder das Programm aufgerufen wird, ist von Fall zu Fall unterschiedlich.

Näheres kann dem Kapitel [Konvertierung von RCL-Befehlen auf Seite 454](#) und [Konvertierung von RCL-Programmen auf Seite 460](#) entnommen werden.

- Ist das Programm bereits auf IEC-Basis erstellt, so verringert sich der Aufwand für die Umstellung. Hierbei sind Eingriffe in den Programmablauf nur bei verwendeter Bandsynchronisierung bzw. bei Verwendung von Funktionsbausteinen notwendig, welche nur für die Robot-Control V1 zulässig sind.

Näheres kann dem Kapitel [Konvertierung von IEC-Befehlen auf Seite 464](#) entnommen werden.

Unabhängig davon, ob ein RCL- oder IEC-Programm vorliegt, sind zusätzlich minimale Anpassungen im IW-Projekt notwendig.

Näheres kann dem folgenden Kapitel entnommen werden.

12.2 Anpassung im IndraWorks-Projekt

Vor dem Umstieg auf Robot-Control V2 wird generell empfohlen, eine Parametersicherung durchzuführen. Somit kann die Parametrierung der ursprünglichen Kinematik nach dem Umstieg schnell und einfach wiederhergestellt werden.



Aus Sicherheitsgründen **muss** nach der Wiederherstellung der Parameter eine erneute Inbetriebnahme der Kinematik durchgeführt werden (Überprüfung der Achslängen, Achszuweisungen, Wichtung, etc.)!

Für die Umstellung auf Robot-Control V2 ist in der Konfiguration der Kinematik die Version anzupassen. Hierzu mittels Rechtsklick auf die Kinematik den Eigenschaftendialog öffnen und die Registerkarte "Konfiguration" wählen (siehe folgende Abbildung).

Umstieg von Robot-Control V1 auf Robot-Control V2

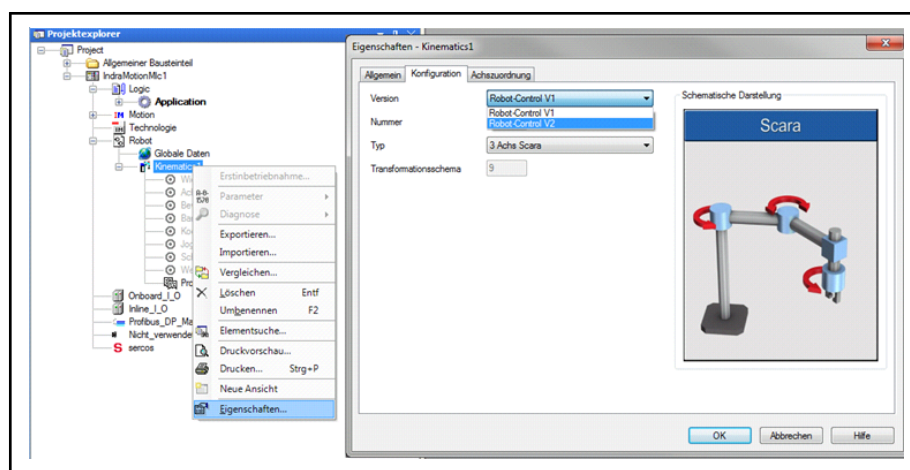


Abb. 12-1: Kinematikkonfiguration Robot-Control V2

Durch diese Anpassung der Version wird intern die bestehende Kinematik gelöscht und eine neue Kinematik angelegt. Somit müssen die zuvor gesicherten Kinematik-Parameter wiederhergestellt werden.



Alle bestehenden RCL-Programme werden gelöscht!

Anwenderdefinierte Transformation

Der Kinematiktyp "anwenderdefiniert" wird von der Robot-Control V2 nicht mehr unterstützt. Stattdessen kann in der Registerkarte "Konfiguration" im Eigenschaftendialog der Kinematik unter "Typ" die gewünschte Kinematik ausgewählt werden. Somit ist beispielsweise für eine verwendete Kinematiknummer 1007 der Typ "2-Achs-Delta (7)" auszuwählen. Aber auch in der SPS definierte Transformationen sind weiterhin verwendbar. Hierzu ist die "Benutzerdefinierte Transformation" oder auch die "Erweiterte Benutzerdefinierte Transformation" zu verwenden. Mit letzterem Kinematiktyp sind weitere neue Funktionalitäten für benutzerdefinierte Transformationen nutzbar, welche nur unter Robot-Control V2 verfügbar sind.

12.3 Konvertierung von RCL-Befehlen

Für die Programmierung von RCL-Programmen ist ein Sprachraum definiert, welcher alle Schlüsselwörter zur Kommandierung einer Kinematik beinhaltet.

Im Folgenden wird dargestellt, welche PLCopen-Funktionsbausteine diese Schlüsselwörter ersetzen bzw. wie die hinter dem Schlüsselwort stehende Funktionalität mittels PLCopen-Funktionsbausteinen nachgebildet werden kann.

Variablendeklaration

Die Deklaration notwendiger Variablen erfolgt im Deklarationsteil des Funktionsbausteins bzw. Programms. Im Folgenden sind beispielhaft Variablendeklarationen mit RCL sowie PLCopen dargestellt.

RCL	PLCopen
INTEGER: iValue	iValue: DINT; iValue: INT;
REAL: rValue	rValue: REAL;
BINARY: bValue	bValue: BOOL;
CHAR: strValue	strValue: STRING(255);

Umstieg von Robot-Control V1 auf Robot-Control V2

RCL	PLCopen
POINT: arPoint	arPoint: ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL; (Nutzung des Maschinen-Koordinatensystems)
JC_POINT: arPoint	arPoint: ARRAY [1..MB_MAX_KIN_AXIS_NUMBER] OF REAL; (Nutzung des Achs-Koordinatensystems)
BELT	Nicht notwendig
TEXT: strValue	strValue: STRING(80);
ARRAY [1..7] INTEGER: arValue	arValue: ARRAY [1..7] OF DINT; arValue: ARRAY [1..7] OF INT;
WC_FRAME	Nicht notwendig
FILE: strValue	Nutzung der Funktionsbausteine der Bibliothek SysFile
INPUT BINARY: 1 = ibValue	VAR_INPUT ibValue: BOOL; END_VAR
OUTPUT BINARY: 1 := obValue	VAR_OUTPUT obValue: BOOL; END_VAR
PUBLIC ... EXTERNAL ...	Nutzung der globalen Variablendeklaration
CONST: iValue=1, rValue=2.75	VAR CONSTANT iValue: UINT := 1; rValue: REAL := 2.75; END_VAR

Der Zusatz "DEF", der in RCL vor POINT-Variablen und JC_POINT-Variablen geschrieben werden kann, bewirkt, dass der Punkt in einer Punktedatei abgespeichert wird. Bei Nutzung der PLCopen-Funktionsbausteine muss hierfür die TeachIn-Funktionalität (ML_GroupWritePoint und ML_GroupSavePointFile) aufgerufen werden.

Funktionen

RCL- und PLCopen-Funktionen besitzen im überwiegenden Maße dieselbe Syntax oder sind durch einfache Funktionsbausteine abbildbar. Im Folgenden werden die Funktionen von RCL durch äquivalente PLCopen-Funktionen beispielhaft gegenübergestellt.

RCL	PLCopen
SIN(0.5)	SIN(0.5) -> 0.479426
COS(0.5)	COS(0.5) -> 0.877583
ATAN(0.5)	ATAN(0.5) -> 0.463648
SQRT(49)	SQRT(49) -> 7
ABS(-5)	ABS(-5) -> 5
TRUNC(1.9)	TRUNC(1.9) -> 1

Umstieg von Robot-Control V1 auf Robot-Control V2

ORD('g')	IL_AsciiToByte('g') -> 103
CHR(103)	IL_ByteToAscii(INT_TO_BYTE(103) -> g
ROUND(7.81)	REAL_TO_INT(7.81) -> 8
13 MOD 5	13 MOD 5 -> 3
END_OF_FILE	Nutzung des Funktionsbausteins SysFileEOF
ASSIGN	Nicht notwendig
INT_ASC	Siehe Programmauszug im Anschluss der Tabelle
ASC_INT	Siehe Programmauszug im Anschluss der Tabelle
SIZEOF(arValue)	SIZEOF(arValue) -> Ergebnis abhängig vom Typ der Array-Elemente arValue: ARRAY[1..5] OF INT; -> Ergebnis = 10 arValue: ARRAY[1..5] OF DINT; -> Ergebnis = 20
LIMIT_MIN, ..._MAX, ..._OFF	Entspricht den Software-Endschaltern ML_GroupSetSafeZone
JC(...), WC(...)	Entspricht einer Koordinatentransformation ML_RobotTransformPoint

Für die Funktionen "INT_ASC" und "ASC_INT" bietet die PLCopen keine äquivalenten Funktionen. Daher müssen eigene Funktionen erstellt werden.

Beispiel für die Funktion "INT_ASC":

Deklarationsteil

```
VAR
    iValue: INT := 0;
    strValue: STRING(255) := '';
    strPartofValue: STRING(255) := '';
    arStrValue: ARRAY [1..10] OF STRING := [10('')];
    iPosition: INT := 1;
    iArraySize: INT := 0;
END_VAR
```

Programmteil

```
strValue := INT_TO_STRING(iValue);
iArraySize := SIZEOF(arStrValue)/81; //81 Bytes per String element
FOR iPosition := 1 TO iArraySize DO
    strPartofValue := MID(strValue,1,iPosition);
    arStrValue[iPosition] := strPartofValue;
END_FOR
```

Beispiel für die Funktion "ASC_INT":

Deklarationsteil

```
VAR
    iValue: INT := 0;
    strValue: STRING(255) := '';
    strPartofValue: STRING(255) := '';
    arStrValue: ARRAY [1..10] OF STRING := [10('')];
    iPosition: INT := 1;
    iArraySize: INT := 0;
    iFactor: INT := 0;
END_VAR
```

Programmteil

```
strValue := '';
iArraySize := SIZEOF(arStrValue)/81; //81 Bytes per String element
FOR iPosition := 1 TO iArraySize DO
```

Umstieg von Robot-Control V1 auf Robot-Control V2

```

strPartofValue := arStrValue[iPosition];
IF STRING_TO_INT(strPartofValue) <> 0 THEN
    strValue := CONCAT(strValue, strPartofValue);
ELSIF strPartofValue = '-' THEN
    iFactor := -1;
ELSIF strPartofValue = '+' THEN
    iFactor := 1;
END_IF
END_FOR
iValue := STRING_TO_INT(strValue) * iFactor;
    
```

Spezialfunktionen

Mit den RCL-Spezialfunktionen werden Funktionen realisiert, für die im RCL-Sprachraum keine Elemente reserviert sind. Sie stellen eine Erweiterung des RCL-Sprachraums dar und müssen vor dem Aufruf, ähnlich wie eine Variable, vereinbart werden. Im Folgenden ist dargestellt durch welche PLCopen-Funktionsbausteine die Spezialfunktionen des erweiterten RCL-Sprachraums abgedeckt sind.

RCL	PLCopen
SPC_FCT:1=IOL	ML_SwitchOnPath
SPC_FCT:2=PRO	Mithilfe von ML_SwitchOnPath abbildbar
SPC_FCT:21=belt_kind	Nicht notwendig
SPC_FCT:23=clock_date	ML_GetSystemTimeUTC bzw. IL_SysTimeDate
SPC_FCT:24=sys_count	Nutzung des Parameters C-0-0423
SPC_FCT:28=belt_set	Das Rücksetzen des Bandwertes auf 0 erfolgt in Abhängigkeit vom Bit 1 des Parameters K-0-0099 automatisch bei 10.000 bzw. beim Achsmodulowert
SPC_FCT:53=belt_area	Entspricht den Eingängen "BeginLength" und "TotalLength" des FBs ML_SetBeltConfiguration
SPC_FCT:59=getBlendMode	Nutzung des Parameters K-0-0099 , Bit 2
SPC_FCT:60=setBlendMode	Auslesen des Parameters K-0-0099 , Bit 2

Randbedingungen einer Bewegung

Jeder Bewegungsbefehl benötigt gewisse Randbedingungen (Geschwindigkeit, Beschleunigung, Koordinatensystem, etc.).

Im RCL-Programm werden diese Werte über entsprechende Zusatzinformation definiert. Diese Werte werden mit den Buchstaben "V" (Bahngeschwindigkeit), "A" (Bahnbeschleunigung) usw. angegeben.

Im Folgenden soll anhand einer beispielhaften PTP- und Linearbewegung gezeigt werden, in welcher Beziehung die Randbedingungen im RCL-Programm zu den Eingängen der PLCopen-Funktionsbausteine stehen und wie diese wirken:

Umstieg von Robot-Control V1 auf Robot-Control V2

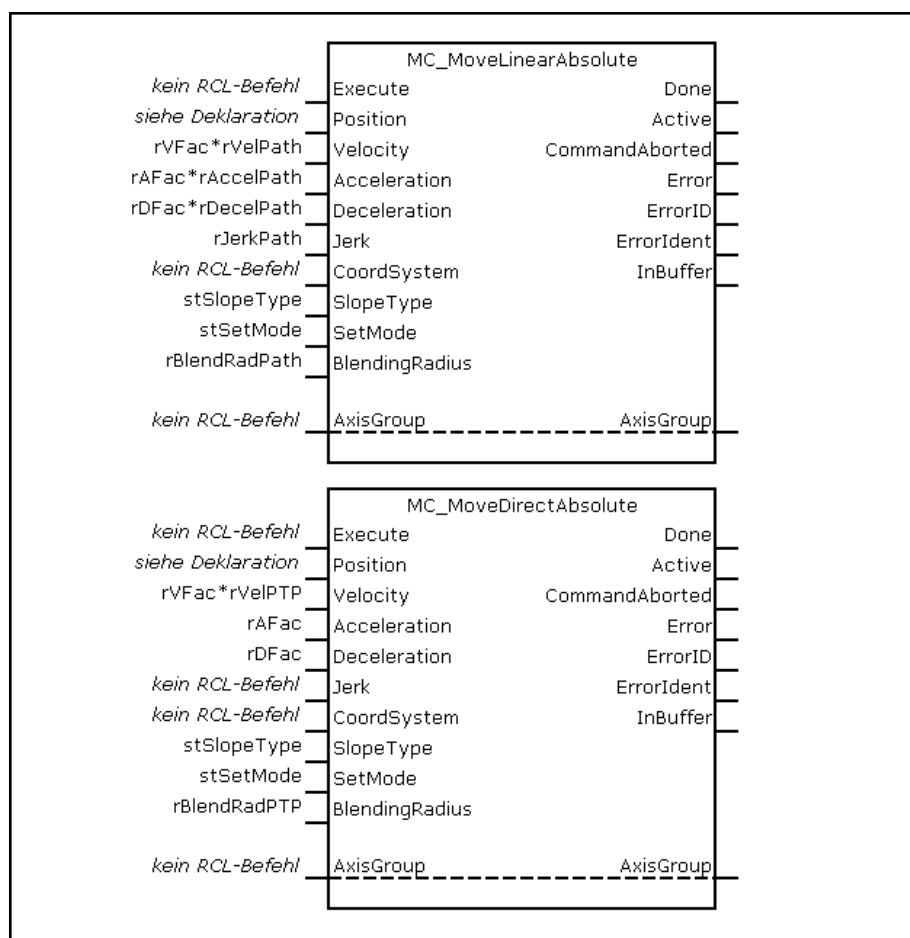


Abb. 12-2: Randbedingungen im RCL-Programm und Eingänge der PLCopen-Funktionsbausteine

RCL	PLCopen
V	rVelPath
A	rAccelPath, rDecelPath
J	rJerkPath
R	rBlendRadPath
VFACTOR	rVFac
AFACTOR	rAFac
DFACTOR	rDFac
V_PTP	rVelPTP
R_PTP	rBlendRadPTP
BLOCK_..., PROGR_..., CONT_SLOPE	stSlopeType
TO / EXACT, VIA / APPROX	stSetMode
WITH	Kein expliziter Eingang (siehe Erläuterung im Anschluss der Tabelle)
UNTIL	Kein expliziter Eingang (siehe Erläuterung im Anschluss der Tabelle)

Umstieg von Robot-Control V1 auf Robot-Control V2

Das Koordinatensystem, in welchem verfahren werden soll, wird im RCL-Programm implizit durch den Zielpunkt angegeben.

Ist der Zielpunkt als "POINT" deklariert, so entspricht dies der Angabe CS_MACHINE_COORDINATES am Eingang "CoordSystem" der PLCopen-Funktionsbausteine.

Ist der Zielpunkt hingegen als "JC_POINT" deklariert, so entspricht dies der Angabe CS_AXES_COORDINATES am Eingang "CoordSystem" der PLCopen-Funktionsbausteine.

Die Angabe des Zielpunktes erfolgt im RCL-Programm zusammen mit dem Bewegungsbefehl, wofür am PLCopen-Funktionsbaustein der Eingang "Position" (absoluter Bewegungsbefehl) bzw. "Distance" (relativer Bewegungsbefehl) steht.

Durch den RCL-Befehl **WITH** ist es möglich einem Bewegungsbefehl andere Randbedingungen mitzugeben, welche lediglich für diesen einen Befehl gelten. Bei den PLCopen-Funktionsbausteinen ist dies implizit dadurch gegeben, dass an jedem Funktionsbaustein andere Werte angegeben werden können.

Wird ein RCL-Befehl mit einer Abbruchbedingung **UNTIL** versehen, so wird die Bewegung nur solange ausgeführt, solange die Bedingung erfüllt ist. Dies ist mittels PLCopen-Funktionsbausteinen nur möglich, indem zusätzlich zum Bewegungsbefehl die Funktionsbausteine MC_GroupInterrupt und MC_GroupContinue eingesetzt werden.

Details siehe [Konvertierung von RCL-Programmen auf Seite 460](#)

Weitere Schlüsselworte

Neben den bereits genannten Schlüsselwörtern besitzt der RCL-Sprachraum weitere Befehle, welche für eine Bewegung notwendig bzw. hilfreich sind. Im Folgenden ist dargestellt, durch welche PLCopen-Funktionsbausteine diese weiteren Schlüsselwörter des RCL-Sprachraums abgedeckt sind.

RCL	PLCopen
MOVE ...	MC_MoveDirectAbsolute MC_MoveLinearAbsolute MC_MoveCircularAbsolute
MOVE_REL ...	MC_MoveDirectRelative MC_MoveLinearRelative MC_MoveCircularRelative
PAUSE	MC_GroupInterrupt
HALT	MC_GroupStop
POS	MC_GroupReadActualPosition
SYNC	Das Rücksetzen des Bandwertes auf 0 erfolgt in Abhängigkeit vom Bit 1 des Parameters K-0-0099 automatisch bei 10.000 bzw. beim Achsmodulwert
SYNCHRON	Entspricht der Angabe eines Bandwertes ≠ 0 im Zielpunkt
SYNCHRON_END	Entspricht der Angabe eines Bandwertes = 0 im Zielpunkt
WAIT ...	ML_GroupWait
WAIT UNTIL ...	Mithilfe einer IF-Abfrage abbildbar

Umstieg von Robot-Control V1 auf Robot-Control V2

12.4 Konvertierung von RCL-Programmen

In RCL-Programmen wird ein Bewegungsablauf sequentiell programmiert, d. h. das Programm wird Zeile für Zeile vorbereitet und abgearbeitet. Da PLCopen-Funktionsbausteine zyklisch aufgerufen werden, ist hier eine besondere Handhabung notwendig.

Durch das Zusammenspiel des Aktivierungseingangs "Execute" sowie der Statusausgänge "Done" und "InBuffer" kann eine sequentielle Programmierung initiiert werden.

Im Folgenden soll anhand von typischen Anwendungsbeispielen aufgezeigt werden, wie verschiedene RCL-Abläufe durch PLCopen-Funktionsbausteine realisiert werden können.

Überschleifende Bewegung mit verschiedenen Geschwindigkeiten

RCL-Programmbeispiel

```
BEGIN
BLOCK_SLOPE
V = rVelocity
A = rAcceleration
J = rJerk
R = rRadius

MOVE_LINEAR VIA arPosition
MOVE_LINEAR WITH V = rNewVelocity TO arNextPosition
PROGRAM_END
```

Nach dem Start des RCL-Programms wird der erste programmierte Bewegungsablauf automatisch gestartet sowie die nachfolgende Bewegung vorbereitend in den Satzpuffer geschrieben. Bei Nutzung der PLCopen-Funktionsbausteine muss der Start durch das Setzen des Eingangs "Execute" des ersten Bewegungsbausteins angestoßen werden. Alle weiteren Bewegungsbausteine können mit den Ausgängen des vorhergehenden Bausteins gestartet werden.

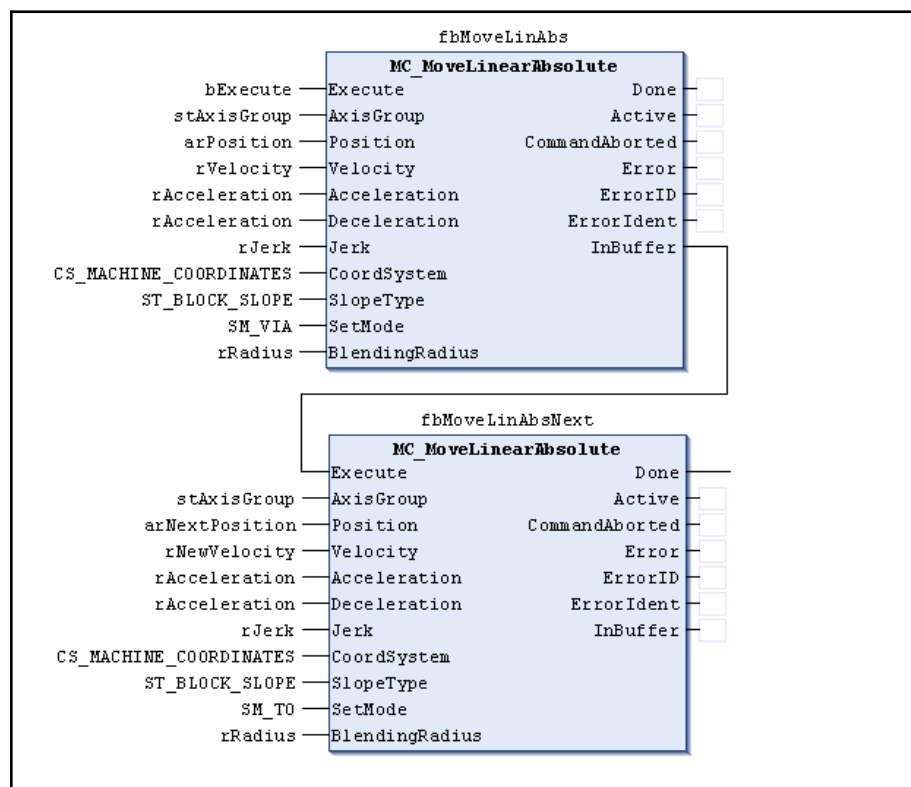


Abb. 12-3: PLCopen-Programmbeispiel

Umstieg von Robot-Control V1 auf Robot-Control V2

Ob eine Bewegung überschleifend verfahren wird oder nicht, hängt bei den PLCopen-Funktionsbausteinen davon ab, welcher Funktionsbausteinausgang für die Aktivierung des Folgebausteins genutzt wird. Wie im Beispiel (siehe obige Abbildung) zu erkennen wird der Ausgang "InBuffer" der Instanz "fbMoveLinAbs" an den Eingang des Folgebausteins gelegt. Somit wird der zweite Bewegungsbefehl abgesetzt, sobald der erste Bewegungsbefehl in den Satzpuffer geschrieben wurde. Ist hingegen eine nichtüberschleifende Bewegung gefordert, d. h. soll der Zielpunkt direkt angefahren werden (Set-Mode = TO), so ist es nicht zwingend notwendig den Ausgang "InBuffer" zu nutzen. Hierbei kann ebenso der Ausgang "Done" als Weiterschaltbedingung herangezogen werden.

Das Koordinatensystem, in welchem verfahren wird, definiert sich im RCL-Programm indirekt über den Punkt. Dieser ist im Beispiel vom Typ POINT, weshalb im Maschinen-Koordinatensystem verfahren wird. Bei Nutzung der PLCopen-Funktionsbausteine muss das Koordinatensystem am Eingang "CoordSystem" angegeben werden und entspricht hierbei "CS_MACHINE_COORDINATES".

Ebenfalls ist es möglich, die im Beispiel gezeigte Bewegung mit nur einer Instanz des Funktionsbausteins zu realisieren. Hierzu ist der Ausgang "InBuffer" bzw. "Done" durch eine entsprechende Logik auf den Eingang "Execute" der selben Instanz zurückzuführen, um diesen so mit einer positiven Flanke am Ausgang neu zu starten. Bei Nutzung des Ausgangs "InBuffer" geht allerdings jegliche Diagnosemöglichkeit des ersten Bewegungsbefehls verloren, da bei erneutem Start des Funktionsbausteins dessen Ausgänge zurückgesetzt werden und folglich für die zuletzt gestartete Bewegung gelten!

Bewegung mit Abbruchbedingung

RCL-Programmbeispiel

```
BEGIN
BLOCK_SLOPE
  V = rVelocity
  A = rAcceleration
  J = rJerk
  R = rRadius

  MOVE_LINEAR TO arPosition UNTIL bAbort = FALSE
PROGRAM_END
```

Nach dem Start des RCL-Programms wird überprüft, ob die Bedingung für den Bewegungsfehl (bAbort = FALSE) erfüllt ist. Ist dies der Fall, so startet der Bewegungsablauf. Ebenfalls wird während der Bewegung die Bedingung überwacht und die Bewegung gestoppt, sobald die Bedingung nicht mehr erfüllt ist.

Umstieg von Robot-Control V1 auf Robot-Control V2

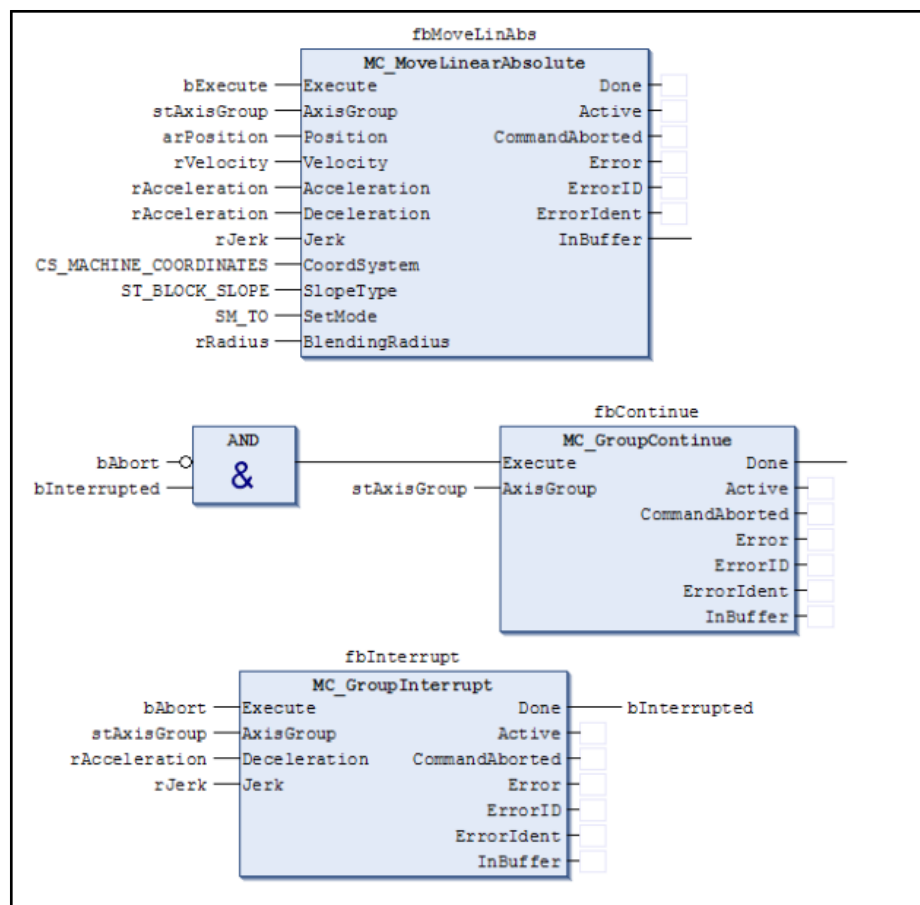


Abb. 12-4: PLCopen-Programmbeispiel nur für Robot-Control V2

Bei Nutzung der PLCopen-Funktionsbausteine müssen zusätzlich zum Bewegungsbaustein (Instanz "fbMoveLinAbs") die Funktionsbausteine MC_GroupInterrupt und MC_GroupContinue eingesetzt werden. Wie im Beispiel abgebildet, wird der Bewegungsablauf unterbrochen bzw. erst gar nicht gestartet, solange die Variable (Bedingung) "bAbort" gesetzt ist. Wird diese Variable zurückgesetzt, so wird die Bewegung fortgesetzt bzw. kann durch eine steigende Flanke am Eingang "Execute" der Instanz "fbMoveLinAbs" gestartet werden.

Das Koordinatensystem, in welchem Verfahren wird, definiert sich im RCL-Programm indirekt über den Punkt. Dieser ist im Beispiel vom Typ POINT, weshalb im Maschinen-Koordinatensystem verfahren wird. Bei Nutzung der PLCopen-Funktionsbausteine muss das Koordinatensystem am Eingang "CoordSystem" angegeben werden und entspricht hierbei "CS_MACHINE_COORDINATES".

Bandsynchronisierung und
Warte-Befehl

RCL-Programmbeispiel

```
BEGIN
BLOCK_SLOPE
  V = rVelocity
  A = rAcceleration
  J = rJerk
  R = rRadius

  SPC_FCT_21_belt_kind (rBeltNumber, BeltKind)
  SPC_FCT_53_belt_area (rBeltNumber, arBase, rFullLength, rBeginLength)

  SYNCHRON
  MOVE LINEAR TO arSyncPosition
  WAIT 5
```

Umstieg von Robot-Control V1 auf Robot-Control V2

```
SYNCHRON_END
MOVE_LINEAR TO arAsyncPosition
PROGRAM_END
```

Nach dem Start des RCL-Programms wird das Band durch die Spezialfunktionen SPC_FCT 21 und 53 konfiguriert. Anschließend startet automatisch der Synchronisationsvorgang.

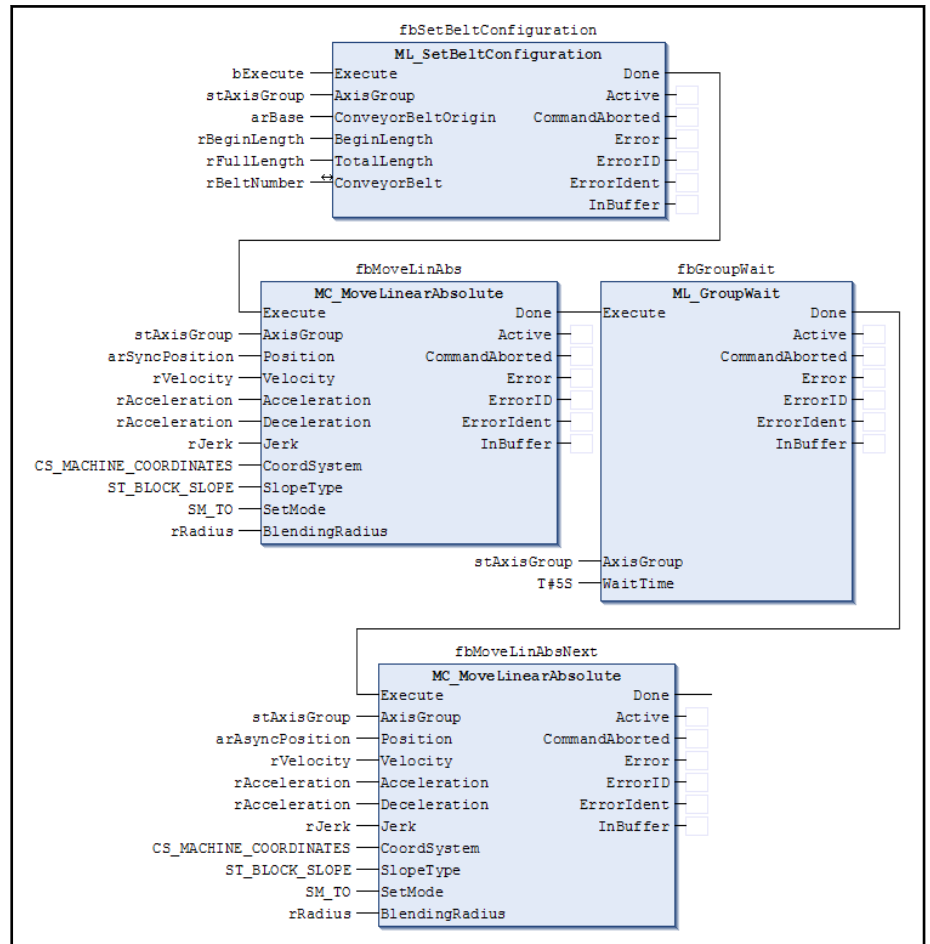


Abb. 12-5: PLCopen-Programmbeispiel nur für Robot-Control V2

Bei Nutzung der PLCopen-Funktionsbausteine erfolgt die Bandkonfiguration durch den Funktionsbaustein ML_SetBeltConfiguration. Nach Abschluss der Konfiguration startet die Synchronisierbewegung. Ist diese abgeschlossen, d. h. ist der kommandierte Punkt erreicht und die Kinematik vollsynchron zum angegebenen Band, so wird der Funktionsbaustein ML_GroupWait gestartet und die angegebene Zeit läuft ab. Anschließend wird der Absynchronisiervorgang angestoßen, wodurch die aufgebaute Bandsynchronität abgebaut und der neue Zielpunkt angefahren wird.

Nähere Informationen zum Auf- und Absynchronisieren siehe [Betrieb eines Bandes auf Seite 72](#).

In diesem Beispiel sind jegliche Weiterschaltbedingungen durch den Ausgang "Done" der Funktionsbausteine definiert. Es ist ebenfalls möglich anstelle dessen den Ausgang "InBuffer" zu nutzen, wodurch die Befehle in den Satzpuffer geschrieben und nacheinander abgearbeitet werden.

Weitere Anwendungsbeispiele zum Thema "Bandsynchronität" siehe

- [Einsatz von Koordinatensystemen auf Seite 399](#)

Umstieg von Robot-Control V1 auf Robot-Control V2

- Bandsynchronität auf Seite 400

12.5 Konvertierung von IEC-Befehlen

Bereits bestehende IEC-Programme für die Robot-Control V1 können in wenigen Schritten zu IEC-Programmen für die Robot-Control V2 konvertiert werden. Diese Konvertierung ist natürlich stark vom jeweiligen Programm abhängig. Desweiteren sind Eingriffe in das IEC-Programm nur notwendig wenn Bänder oder Funktionsbausteine, welche nur für Robot-Control V1 zulässig sind, verwendet werden.

Ersetzen von **ML_SetBeltParameter** durch **ML_SetBeltConfiguration**

Der Funktionsbaustein **ML_SetBeltParameter** ist durch den äquivalenten Funktionsbaustein **ML_SetBeltConfiguration** zu ersetzen.

Hierbei können die Variablen für die Eingänge "Execute", "Full/TotalLength", "BeginLength", "AxisGroup" und "ConveyorBelt" übernommen werden.

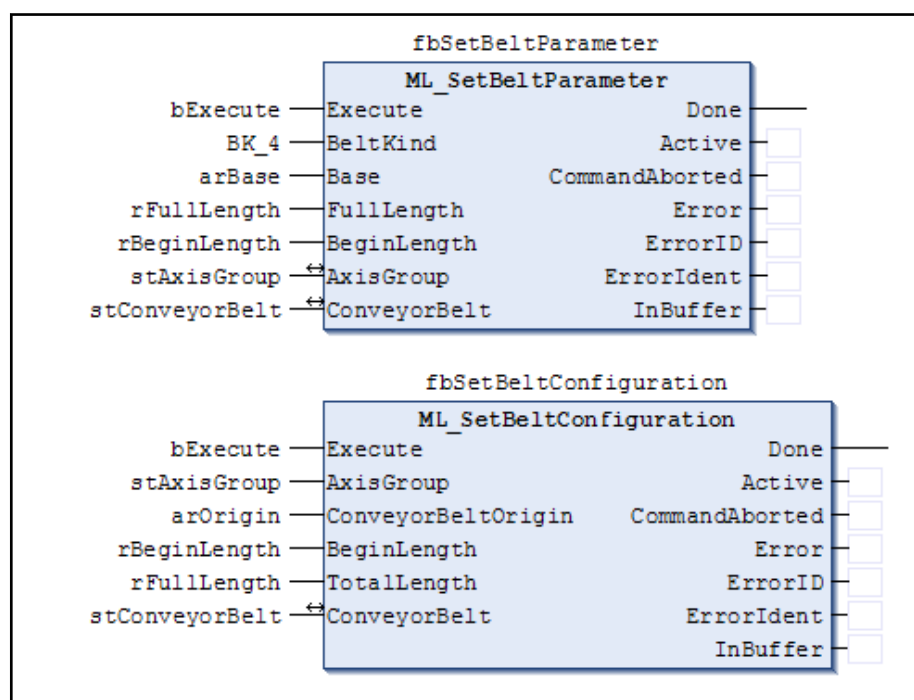


Abb. 12-6: Funktionsbausteine für Bandkonfiguration

Die Werte für den Eingang "ConveyorBeltOrigin" können allerdings nicht direkt von "Base" übernommen werden, da sich die Bedeutung und die Array-Größe geändert haben.

Definition der Position und Verdrehung des Bandes (ConveyorBeltOrigin)

Bei Verwendung der Robot-Control V1 ist die Position des Bandes durch den Eingang "Base" des Funktionsbausteins **ML_SetBeltParameter** sowie die Verdrehung über Bandkopplungsfaktoren definiert.

Bei Verwendung der Robot-Control V2 sind diese beiden Konfigurationen im Eingang "ConveyorBeltOrigin" des Funktionsbausteins **ML_SetBeltConfiguration** zusammengefasst.

Die ersten drei Werte bestimmen die Verschiebung und die letzten drei Werte die Verdrehung des Bandes relativ zum Maschinen-Koordinatensystem (MCS). Somit ist für diesen Eingang eine neue Variable vom Typ ARRAY [1..6] OF REAL anzulegen.

Ersetzen weiterer Bausteine

Alle Funktionsbausteine, welche nur für die Robot-Control V1 gültig sind, müssen entfernt bzw. ersetzt werden.

Umstieg von Robot-Control V1 auf Robot-Control V2

Wird dies nicht getan und ein Funktionsbaustein aufgerufen, welcher nicht für die Robot-Control V2 zulässig ist, gibt dieser eine entsprechende Fehlermeldung aus und der Programmablauf wird eventuell gestört.

Folgende Funktionsbausteine sollten entsprechend ersetzt werden:

Robot-Control V1	Robot-Control V2
ML_KinSetDistanceBlendOption	Nutzung des Parameters K-0-0099 , Bit 2
ML_SetBeltParameter	ML_SetBeltConfiguration
ML_KinGetDistanceBlendOption	Nutzung des Parameters K-0-0099 , Bit 2
ML_IRDProgSelect	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_IRDProgStart	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_IRDProgStop	MC_GroupStop
ML_RoCoDir	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_RoCoGetPoint	ML_GroupReadPoint
ML_RoCoGetPointList	Mithilfe von ML_GroupReadNextPoint abbildbar (Ausgang "PointNameOut" mit Eingang "PointNameIn" verbinden)
ML_RoCoSetPoint	ML_GroupWritePoint
ML_KinGetUserBit	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_KinGetUserBit_DWORD	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_KinGetUserDInt	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_KinGetUserReal	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_KinSetUserBit	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_KinSetUserDInt	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_KinSetUserPoint	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind
ML_KinSetUserReal	Nicht mehr notwendig, da RCL-Dateien nicht mehr verfügbar sind

Bei Verwendung des Kinematik-Interfaces muss das Plug-in aktualisiert werden.

Nach der Durchführung dieser Schritte ist die Umstellung von Robot-Control V1 auf Robot-Control V2 abgeschlossen und das Projekt kann neu in die Steuerung geladen werden.

13 Service und Support

Für Ihre schnelle und optimale Unterstützung verfügen wir über ein dichtes weltweites Servicenetz. Unsere Experten stehen Ihnen mit Rat und Tat zur Seite. Sie erreichen uns täglich **rund um die Uhr - auch an Wochenenden und Feiertagen**.

Service Deutschland Unser technologieorientiertes Competence Center in Lohr deckt alle Belange rund um den Service für elektrische Antriebe und Steuerungen ab.

Sie erreichen unsere **Service-Hotline** und unseren **Service-Helpdesk** unter:

Telefon: **+49 9352 40 5060**
Fax: **+49 9352 18 4941**
E-Mail: service.svc@boschrexroth.de
Internet: <http://www.boschrexroth.com>

Auf unseren Internetseiten finden Sie ergänzende Hinweise zu Service, Reparatur (z. B. Anlieferadressen) und Training.

Service weltweit Außerhalb Deutschlands nehmen Sie bitte zuerst Kontakt mit Ihrem Ansprechpartner auf. Die Hotline-Rufnummern entnehmen Sie bitte den Vertriebsadressen im Internet.

Vorbereitung der Informationen Wir können Ihnen schnell und effizient helfen, wenn Sie folgende Informationen bereithalten:

- Eine detaillierte Beschreibung der Störung und der Umstände
- Angaben auf dem Typenschild der betreffenden Produkte, insbesondere Typenschlüssel und Seriennummern
- Ihre Kontaktdaten (Telefon-, Faxnummer und E-Mail-Adresse)

Index

A

A-0-0028, Fahrbereichsgrenzscharter.....	421
A-0-0029, Positionspolaritäten.....	421
A-0-0030, Positionsgrenzwert, positiv.....	422
A-0-0031, Positionsgrenzwert, negativ.....	423
A-0-0032, Geschwindigkeitsgrenzwert, positiv.....	423
A-0-0033, Geschwindigkeitsgrenzwert, negativ.....	423
A-0-0034, Beschleunigungsgrenzwert, bipolar.....	423
A-0-0035, Geschwindigkeitsgrenzwert, bipolar.....	424
A-0-0036, Ruckgrenzwert, bipolar.....	424
A-0-0038, Drehmoment-/Kraftgrenzwert, bipolar.....	424
A-0-0039, Drehmoment-/Kraftgrenzwert, positiv.....	425
A-0-0040, Drehmoment-/Kraftgrenzwert, negativ.....	425
A-0-0048, Geschwindigkeitspolaritäten.....	425
A-0-0049, Drehmoment-/Kraft-Polaritäten.....	426
A-0-0224, Schnellhalt-Verzögerung.....	427
A-0-0225, Schnellhalt-Ruck.....	427
A-0-0240, Feederate-Override.....	427
A-0-0241, Feederate-Override-Steuerwort.....	427
Abkürzungen.....	21
Absolute Positionierung.....	48
Achs-Koordinatensystem (ACS).....	26
Achstypen.....	30
ACS.....	26
Active.....	195
Anbindung Fremdsysteme.....	451
uniVAL drive.....	451
Anregungen.....	22
ANSI Z535.6-2006.....	14
Anwender-Erweiterung.....	352
Anwendererweiterung.....	350, 351, 352
Interface-Struktur (MB_KinInterface-Type11) erweitern.....	351, 352
Anwendungsbeispiele	
Anmeldung einer benutzerdefinierten Transformation.....	419
Sequenzielle Programmierung mit dem Kinematik-Interface.....	417
Anwendungsphasen.....	13
ASC_INT, Erzeugung.....	455
Axis Coordinate System.....	26

B

Bandbetrieb	
Adaptive Motion.....	74
Allgemein.....	72
Bandkonfiguration.....	68
Bandspezifische Fehlerreaktion.....	82

Bandsynchrone Bewegung.....	67
Bandwerte.....	69
Base Coordinate System.....	26
Basis-Koordinatensystem (BCS).....	26
Basisparameter laden	
Kinematik.....	102
Bausteintypen.....	193
Bausteintypen und ihre Schnittstellen.....	194
BCS.....	26
Beschwerde.....	22
Betriebsarten.....	335
Absolutes Jump Positionieren.....	343
Absolutes Lineares Positionieren.....	339
Absolutes PTP Positionieren.....	337
Absolutes Zirkulares Positionieren.....	341
Continue.....	345
Externer Funktionsbaustein.....	346
Interrupt.....	345
Kinematik Bereit.....	336
Kinematik Referenzieren.....	337
Kinematik Stopp.....	344
Relatives Lineares Positionieren.....	340
Relatives PTP Positionieren.....	338
Relatives Zirkulares Positionieren.....	342
Zyklische Positionssollwertvorgabe.....	346
Bewegungen	
Stoppen.....	64
Unterbrechen.....	65
Warten.....	67
Wiederaufnehmen.....	66
Bewegungsart	
Circular.....	41
HalfJump.....	45
HalfJumpDown.....	46
HalfJumpUp.....	45
Jump.....	42
Linear.....	40
PTP.....	42
Bibliothek	
ML_KinTech.....	361
ML_Robot.....	185
Block_Slope.....	55

C

C-0-0123, Maximale Anzahl an Kinematiken....	428
C-0-2480, Konfigurierte Kinematikenanzahl.....	428
C-0-2482, Liste der nicht verknüpften Achsen.....	428
C-0-2483, Liste der konfigurierten Kinematiken.....	428
C-0-2484, Kinematikkonfigurationsliste.....	428
Circular.....	41
CommandAborted.....	195
Cont_Slope.....	58

Index

D

Darstellung von Informationen

Bezeichnungen und Abkürzungen.....	21
Dialogbasierte Kinematikkommandierung.....	347
Done.....	195
DV_Kin[].....	188

E

E_KIN_MECH_CMD.....	312
E_KIN_TEACH_IN_POINT_FILE_OPTIONS....	315
Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbei- tung.....	199
Ein- und Ausgänge flankengesteuerter Funktionsbausteine, permanente Bearbei- tung.....	205
Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbei- tung.....	196
Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbei- tung.....	202
Enable.....	194
Entartete Kreise.....	42
Erstinbetriebnahme	
Kinematik.....	102
Execute.....	194
ExecuteLock.....	195

F

Feedback.....	22
Fehlercodes	
MC_KIN_REF.....	306
MC_KIN_REF_EXT.....	309
MC_MoveCircularAbsolute.....	254
MC_MoveCircularRelative.....	256
MC_MoveDirectAbsolute.....	259
MC_MoveDirectRelative.....	261
MC_MoveLinearAbsolute.....	264
MC_MoveLinearRelative.....	266
MC_SetCartesianTransform.....	211
MC_SetKinTransform.....	213
MC_SetKinTransformExt.....	214
ML_GroupAllAxes.....	217
ML_GroupJogCont.....	271
ML_GroupJogStep.....	273
ML_KinematicsInitType01.....	366
ML_KinematicsInterfaceType01.....	368, 369
ML_KinematicsInterfaceType11.....	371
ML_RobotGetBeltConfiguration.....	231
ML_RobotGetBeltErrorReaction.....	233
ML_RobotGetCmdInfo.....	234
ML_RobotGetMechanic.....	236
ML_RobotGetMoveID.....	237
ML_RobotGetPerformance.....	238
ML_RobotGetSafeZones.....	239
ML_RobotGetTrafoValues.....	241

ML_RobotMoveJumpAbsolute.....	282
ML_RobotMovePoint.....	242
ML_RobotReadCyclicChannel.....	243
ML_RobotSetMoveID.....	228
ML_RobotSetPerformanceEnable.....	245
ML_RobotTransformPoint.....	246
ML_SetBeltConfiguration.....	222
ML_SetBeltErrorReaction.....	223
ML_SetCylindricTransform.....	225
ML_SetRotaryTableConfiguration.....	227
ML_SwitchOnPath.....	247
ML_UngroupAllAxes.....	215
Fehlerreaktion	
Bandspezifische.....	82
kommandierte Bewegung.....	82
Funktionsvariablenzugriff	
DV_Kin[].....	188

G

Gefahrenhinweise.....	14
Gliederung der Dokumentation.....	13
Global_MB_KinInterfaceVars.....	361
Globale Variablen.....	361
GVL_K_Param_ID.....	282
GVL_RobotConstant.....	187

H

HalfJump.....	45
HalfJumpDown.....	46
HalfJumpUp.....	45
Handbetrieb.....	49
Helpdesk.....	467
Hilfsachsen	
Circular.....	41
Linear.....	40
Hinzufügen einer Kinematik zur Systemüber- sicht.....	325
Hotline.....	467

I

InBuffer.....	207
InGear.....	194
InOperation.....	194
InSync.....	194
INT_ASC, Erzeugung.....	455
IntelliSense	
Kinematikdaten.....	187, 188
Interface-Struktur (MB_KinInterfaceType11)	
erweitern	
Funktionsbaustein erweitern.....	351
Hinweise zur Implementation.....	350
Interface erweitern.....	351, 352
Neues Interface hinzufügen.....	352
InTorque.....	194
InVelocity.....	194
ITF_KINEMATICS-Strukturen.....	387

J

Jump..... 42

K

K-0-0001, Logische Kinematiknummer..... 429
 K-0-0002, Kinematikbezeichnung..... 429
 K-0-0003, Maximale Bahngeschwindigkeit..... 429
 K-0-0005, Software-Endschalter, positiv, MCS..... 429
 K-0-0006, Software-Endschalter, negativ, MCS..... 430
 K-0-0007, Achszahl der Kinematik..... 430
 K-0-0008, Achskonfigurationsliste..... 430
 K-0-0009, Oszi: Liste aller oszilloskopierbaren K-Parameter..... 432
 K-0-0010, Liste aller K-Parameter..... 432
 K-0-0011, Liste aller ungültigen K-Parameter..... 432
 K-0-0012, Prüfsumme aller K-Parameter..... 433
 K-0-0013, IDN-Liste der K-Prüfsummenparameter..... 433
 K-0-0016, Projektidentifikationsnummer..... 433
 K-0-0019, Kinematiktyp..... 433
 K-0-0020, Kinematikdiagnosemeldung..... 433
 K-0-0021, Kinematikstatus..... 434
 K-0-0022, Erweiterter Kinematikstatus..... 435
 K-0-0023, Kinematikdiagnosenummer..... 435
 K-0-0028, Geschwindigkeitslimit Hilfsachsen... 436
 K-0-0029, Beschleunigungslimit Hilfsachsen... 436
 K-0-0030, Rucklimit Hilfsachsen..... 437
 K-0-0031, Transformationsschema..... 437
 K-0-0034, Maximale Bahnbeschleunigung (bipolar)..... 437
 K-0-0035, Maximaler Bahnruck (bipolar)..... 438
 K-0-0037, Ruck für Move-Kommandos während Verzögerungsphase..... 438
 K-0-0040, Transformation, Längen der Mechanik..... 438
 K-0-0041, Transformation, Faktoren..... 438
 K-0-0042, Verschiebung des BCS-Systems..... 438
 K-0-0050, Distanzeinheiten MCS..... 439
 K-0-0051, Geschwindigkeitseinheiten MCS..... 439
 K-0-0052, Beschleunigungseinheiten MCS..... 440
 K-0-0053, Ruckeinheiten MCS..... 441
 K-0-0054, Maßeinheit für Kinematikdaten..... 441
 K-0-0070, Bandanzahl der Kinematik..... 442
 K-0-0090, Taskpriorität Satzvorbereitung..... 442
 K-0-0095, Bewegungsidentifikationsnummer... 442
 K-0-0096, Höchstwert der Kommandopuffergröße..... 442
 K-0-0097, Maximale Kommandopuffergröße.... 442
 K-0-0098, Aktuelle Kommandopuffergröße..... 442
 K-0-0099, Kinematikkonfiguration..... 442
 K-0-0100, Aktuelle Position, Kartesische Maschinenkoordinaten..... 444
 K-0-0102, Aktuelle Bahngeschwindigkeit..... 444
 K-0-0103, Aktuelle Bahnbeschleunigung..... 444

K-0-0104, Aktueller Bahnruck..... 444
 K-0-0150, Interpolierte Sollposition..... 445
 K-0-0152, Interpolierte Bahngeschwindigkeit... 445
 K-0-0153, Interpolierte Bahnbeschleunigung... 445
 K-0-0154, Interpolierter Bahnruck..... 445
 K-0-0223, Auswahl Fehlerreaktion..... 445
 K-0-0224, Nothaltverzögerung..... 446
 K-0-0225, Nothaltruck..... 446
 K-0-0240, Feedrate-Override..... 446
 K-0-0241, Feedrate-Override-Steuerwort..... 446
 K-0-0242, Acceleration-Override..... 446
 K-0-0243, Deceleration-Override..... 446
 K-0-0250, Continuous Slope Vorhaltezeit..... 447
 K-0-0815, Kilometerzähler..... 447
 K-0-1000, Kommando: Basisparameter der Kinematik laden..... 447
 K-0-2100, Liste der zu sichernden K-Parameter..... 447
 Kin_Axis_Config_Elements..... 372
 Kin_Axis_Type..... 372
 KinData[] - Kinematikdaten im Shared Memory..... 187
 Kinematik-Interface Betriebsarten
 Überblick..... 335
 Kinematik-Interface Visualisierungen..... 323
 Kinematik-Interface-Betriebsarten
 Übersicht..... 373
 Kinematik-Interface-Strukturen
 MB_KINEMATICS_ADMIN_STATUS..... 389
 MB_KINEMATICS_ADMINISTRATION..... 377
 MB_KINEMATICS_CONTROL_TYPE01..... 377
 MB_KINEMATICS_CONV_BELT_STATUS..... 391
 MB_KINEMATICS_CONVEYOR_BELT..... 380
 MB_KINEMATICS_COORDINATED..... 385
 MB_KINEMATICS_DIAGNOSIS..... 393
 MB_KINEMATICS_MODE..... 375
 MB_KINEMATICS_MODE_BITS..... 374
 MB_KINEMATICS_MODE_UNION..... 374
 MB_KINEMATICS_PCS_CONFIG..... 383
 MB_KINEMATICS_PCS_STATUS..... 392
 MB_KINEMATICS_SETUP..... 383
 MB_KINEMATICS_SETUP_STATUS..... 394
 MB_KINEMATICS_STATUS_TYPE01..... 388
 MB_KINEMATICS_TEACH_IN..... 380
 MB_KINEMATICS_TEACH_IN_STATUS.... 392
 Kinematikdaten über Funktionsvariablenzugriff..... 188
 Kinematikinbetriebnahme..... 89
 Bandkonfiguration..... 95
 Gruppieren einer Kinematik..... 92
 Positionieren..... 94
 Produktkoordinaten..... 96
 Sicherheitsfunktionen..... 91
 SPS-Konfiguration..... 97
 Tippen..... 93
 Übersicht..... 89
 KinInterface/Const..... 361

Index

KinInterface/FB.....	363
KinInterface/FB/DataTypes.....	372
KinInterface/Type.....	373
KinInterface/Type/Control.....	376
KinInterface/Type/Status.....	388
KinJog.....	394
KinJog/FB.....	394
KinJog/Type.....	397
KinSystem_Overview_All.....	324
Kommandieren	
modulo-gewichtete Achsen.....	48
Kontinuierlicher Betrieb.....	50
Koordinaten.....	24
Koordinatensysteme.....	24
Kritik.....	22
Kundenfeedback.....	22

L

Linear.....	40

M

Machine Coordinate System.....	26
Maschinen-Koordinatensystem (MCS).....	26
MB_GroupReadListParameter.....	283
MB_GroupReadParameter.....	284
MB_GroupReadRealParameter.....	285
MB_GroupReadSercosAttribute.....	286
MB_GroupReadSercosDataStatus.....	288
MB_GroupReadStringParameter.....	289
MB_GroupWriteListParameter.....	290
MB_GroupWriteParameter.....	291
MB_GroupWriteRealParameter.....	292
MB_GroupWriteStringParameter.....	293
MB_KINEMATICS_ADMIN_CHANGE_OPMO- DE.....	383
MB_KINEMATICS_ADMIN_STATUS.....	389
MB_KINEMATICS_ADMINISTRATION.....	377
MB_KINEMATICS_CONTROL_TYPE01.....	377
MB_KINEMATICS_CONV_BELT_STATUS.....	391
MB_KINEMATICS_CONVEYOR_BELT.....	380
MB_KINEMATICS_COORDINATED.....	385
MB_KINEMATICS_DIAGNOSIS.....	393
MB_KINEMATICS_MODE.....	375
MB_KINEMATICS_MODE_BITS.....	374
MB_KINEMATICS_MODE_UNION.....	374
MB_KINEMATICS_PCS_CONFIG.....	383
MB_KINEMATICS_PCS_STATUS.....	392
MB_KINEMATICS_SETUP.....	383
MB_KINEMATICS_SETUP_STATUS.....	394
MB_KINEMATICS_STATUS_TYPE01.....	388
MB_KINEMATICS_TEACH_IN.....	380
MB_KINEMATICS_TEACH_IN_MODE.....	382
MB_KINEMATICS_TEACH_IN_STATUS.....	392
MC_GroupContinue.....	247
MC_GroupInterrupt.....	249
MC_GroupReadActualPosition.....	228
MC_GroupReset.....	208

MC_GroupStop.....	250
MC_KIN_REF.....	305
MC_KIN_REF_EXT.....	307
MC_MoveCircularAbsolute.....	251
MC_MoveCircularRelative.....	254
MC_MoveDirectAbsolute.....	257
MC_MoveDirectRelative.....	259
MC_MoveLinearAbsolute.....	261
MC_MoveLinearRelative.....	264
MC_SetCartesianTransform.....	209
MC_SetKinTransform.....	211
MC_SetKinTransformExt.....	213
MC_UngroupAllAxes.....	214
MCS.....	26
Mechanische Informationen.....	183
ML_AddMechanicData.....	215
ML_ARR_COORDS.....	317
ML_BELT_ERROR_REACTION_MODE.....	310
ML_BeltDeSynchronisationWithLimits.....	266
ML_BeltSynchronisationWithLimits.....	267
ML_CIRCMODES.....	317
ML_COORDSYS.....	317
ML_CYCLIC_MODE.....	318
ML_GroupAllAxes.....	216
ML_GroupDeletePointFile.....	295
ML_GroupDeletePoints.....	296
ML_GroupJogCont.....	269
ML_GroupJogStep.....	271
ML_GroupLoadPointFile.....	297
ML_GroupReadNextPoint.....	298
ML_GroupReadPoint.....	299
ML_GroupSavePointFile.....	301
ML_GroupSetSafeZone.....	218
ML_GroupWait.....	273
ML_GroupWritePoint.....	302
ML_IRDProgSelect, Konvertierung.....	465
ML_IRDProgStart, Konvertierung.....	465
ML_IRDProgStop, Konvertierung.....	465
ML_JOGMODE.....	397
ML_KIN_AXES_CONFIG_DATA.....	310
ML_KIN_AXIS_CONFIG.....	311
ML_KIN_CMD_INFO.....	313
ML_KIN_CS_TYPE.....	314
ML_KIN_INFO_TYPE.....	314
ML_KIN_MECHANIC_DATA.....	314
ML_KIN_PERF_DATA.....	315
ML_KIN_SAFE_ZONE_DATA.....	315
ML_KIN_SAFE_ZONE_TYPE.....	311
ML_KIN_USED_AXES.....	311
ML_KINEMATICS_DATA_SM.....	312
ML_KinematicsInitType01.....	363
ML_KinematicsInterfaceType01.....	366
ML_KinematicsInterfaceType11.....	369
ML_KinGetDistanceBlendOption, Konvertie- rung.....	465
ML_KinGetUserBit_DWORD, Konvertierung....	465
ML_KinGetUserBit, Konvertierung.....	465
ML_KinGetUserDInt, Konvertierung.....	465

ML_KinGetUserReal, Konvertierung.....	465
ML_KinInterface.....	361
ML_KinJog.....	394
ML_KinSetDistanceBlendOption, Konvertie- rung.....	465
ML_KinSetUserBit, Konvertierung.....	465
ML_KinSetUserDInt, Konvertierung.....	465
ML_KinSetUserPoint, Konvertierung.....	465
ML_KinSetUserReal, Konvertierung.....	465
ML_KinTech.library.....	361
KinInterface/Const.....	361
KinInterface/FB.....	363
KinInterface/FB/DataTypes.....	372
KinInterface/Type.....	373
KinInterface/Type/Control.....	376
KinInterface/Type/Status.....	388
ML_POINTKIND.....	315
ML_Robot	
Datentypen.....	303
Funktionen Info.....	228
Funktionsbausteine.....	190, 207
Funktionsbausteine Config.....	208
Funktionsbausteine Info.....	228
Funktionsbausteine Movement.....	247
Funktionsbausteine Parameter.....	282
Funktionsbausteine TeachIn.....	295
Parameter lesen / schreiben.....	186
ML_Robot.library.....	185
ML_RobotGetBeltConfiguration.....	229
ML_RobotGetBeltDesyncRelConfiguration.....	231
ML_RobotGetBeltErrorReaction.....	232
ML_RobotGetCmdInfo.....	233
ML_RobotGetMechanic.....	234
ML_RobotGetMoveID.....	236
ML_RobotGetPerformance.....	237
ML_RobotGetSafeZones.....	238
ML_RobotGetTrafoValues.....	239
ML_RobotMoveJumpAbsolute.....	279
ML_RobotMovePoint.....	241
ML_RobotOpenCyclicChannel.....	275
ML_RobotOpenDirectCyclicChannel.....	276
ML_RobotReadCyclicChannel.....	242
ML_RobotSetMoveID.....	227
ML_RobotSetPerformanceEnable.....	244
ML_RobotTransformPoint.....	245
ML_RobotWriteCyclicChannel.....	277
ML_RoCoDir, Konvertierung.....	465
ML_RoCoGetPoint, Konvertierung.....	465
ML_RoCoGetPointList, Konvertierung.....	465
ML_RoCoSetPoint, Konvertierung.....	465
ML_SET_MODE.....	319
ML_SetBeltConfiguration.....	220
ML_SetBeltDesyncRelConfiguration.....	219
ML_SetBeltErrorReaction.....	222
ML_SetBeltParameter, Konvertierung.....	464, 465
ML_SetCylindricTransform.....	223
ML_SetRotaryTableConfiguration.....	225
ML_SLOPE_TYPE.....	319

ML_SwitchOnPath.....	246
MLPI COM	
Robot-Control.....	37
MLPI core	
Robot-Control.....	37
ModeCoordAB.....	336
ModeCoordAH.....	336
ModeCoordContinue.....	345
ModeCoordCyclic.....	346
ModeCoordExternalFB.....	346
ModeCoordHoming.....	337
ModeCoordInterrupt.....	345
ModeCoordPosCircAbs.....	341
ModeCoordPosCircRel.....	342
ModeCoordPosDirAbs.....	337
ModeCoordPosDirRel.....	338
ModeCoordPosJumpAbs.....	343
ModeCoordPosLinAbs.....	339
ModeCoordPosLinRel.....	340
ModeCoordStopping.....	344

O

Ordner Kin_Interface/Type/Control, Über- sicht.....	376
Ordner Kin_Interface/Type/Status, Übersicht..	388

P

Parameterzugriff aus SPS-Programm.....	282
PathViewer.....	98
Kontextmenü.....	100
Navigation und Bedienung.....	99
Übersicht.....	98
PCS.....	27
Performancemessung.....	85
Positionierung	
Absolute	48
Relative	48
ProductCoordSystem.....	334
Produkt-Koordinatensystem (PCS).....	27
Programm zum Kinematik-Interface.....	323
PTP.....	42
Punkt.....	31

R

REFERENCE_TO	
Kinematikdaten.....	187, 188
Relative Positionierung.....	48
Robot-Control	
Achsbezogene Parameter.....	421
Diagnosen.....	449
Funktionen.....	37
Kinematik – Kontextmenü Diagnose.....	105
Kinematikbezogene Parameter.....	429
Kommando-Zustände.....	36
Kommandos.....	37
Parameter.....	421
Steuerungsbezogene Parameter.....	428

Index

Robot-Control V1 nach Robot-Control V2.....	453	Standardschnittstellen an Funktionsbausteinen.....	193
IEC-Befehlen.....	464	StateDiagram	
IndraWorks-Projekt.....	453	Axisgroup.....	33
RCL-Befehle.....	454	Status bei Funktionsvariablenzugriff.....	190
RCL-Programme.....	460	Support.....	467
Überblick.....	453	Symbole.....	15
Robot-Control V2		Systemübersicht Navigation.....	326
ACS-BCS-Transformationen.....	111	T	
Anbindung Fremdsysteme.....	451	TE_KinInterfaceType01Prog.....	323
Anwendungsbeispiele.....	399	Teach-in.....	50
Fehler und Fehlerreaktionen.....	81	Aufbau der Punktedatei.....	51
Funktionsbeschreibung.....	23	Lade Punktedatei.....	54
Handbetrieb.....	49	Lese nächsten Punkt.....	52
Kinematik - Kontextmenü.....	102	Lese Punkt.....	52
Kontinuierlicher Betrieb.....	50	Lösche Punkte.....	53
Oberflächenfunktionen.....	87	Lösche Punktedatei.....	54
Ordner Robot - Dialog Achslängen und Kopplung.....	108	Schreibe Punkt.....	52
Ordner Robot - Dialog Bewegungsgrenzwerte.....	108	Speichere Punktedatei.....	53
Ordner Robot - Dialog Wichtung / Maßeinheiten.....	107	Transformation	
Ordner Robot - Dialoge der Kinematik.....	107	Achsen-Koordinaten (ACS) / Basis-Koordinaten (BCS).....	28
Ordner Robot - Dialog Koordinatenverschiebung.....	109	Basis-Koordinaten (BCS) / Maschinen-Koordinaten (MCS).....	29
Performancemessung.....	85	Maschinen-Koordinaten (MCS) / Produkt-Koordinaten (PCS).....	29
Schalten auf der Bahn.....	76	Transformationen	
Schrittbetrieb.....	50	Robot-Control V2.....	28
Slope-Arten.....	55	Transformationsvorschrift	
Systemübersicht.....	23	2-Achs Presse.....	130
Transformationsschemen.....	111	2-Achs-Delta mit asymmetrischem TCP.....	148
Überschleifen.....	54	2-Achs-Delta mit optional 1 Orientierungsachse.....	113
Verfahrbereichsgrenzen.....	79	2-Achs-Delta Slide.....	156
Zyklische Sollwertvorgabe.....	83	2-Achs-Parallelogramm.....	127
Robot-Control-Anwendungsbeispiel		2-Achs-Parallelogramm invers.....	129
Adaptive Motion.....	411	2-Achs-Parallelogramm V1.....	133
Bandsynchronität.....	400	2-Achs-Parallelogramm V2.....	137
Benutzerdefinierte Transformation.....	401	3-Achs Canfield.....	173
Einsatz von Koordinatensystemen.....	399	3-Achs-Delta Linear mit optional 1 Orientierungsachse.....	138
Erweiterte benutzerdefinierte Transformation.....	405	3-Achs-Delta mit 90 Grad Winkeln.....	160
Erweiterung eines Punkts.....	410	3-Achs-Delta mit Linearachsen mit optional 1 Orientierungsachse.....	152
Kommandopuffer.....	412	3-Achs-Delta mit optional 1 Orientierungsachse.....	135
Zyklische Sollwertvorgabe.....	414	3-Achs-Flex Stamper.....	151
Robot-Control, Kommandos und Funktionen.....	35	3-Achs-Parallelogramm V3.....	157
Robot-Control, Move ID.....	62	3-Achs-Scara mit optional 1 Orientierungsachse.....	119
S		3-Achs-Twisting Crane.....	149
Schrittbetrieb.....	50	4-Achs Scara mit linearer Hubachse.....	115
Service-Hotline.....	467	4-Achs-Scara mit Kurbel.....	169
Shared Memory		4-Achs-Scara mit optional 1 Orientierungsachse.....	123
KinData[].....	187	4-Achs-Speedpicker.....	163
Shutdown.....	195		
Sicherheitshinweise.....	14		
Signalgrafik.....	14		
Signalwörter.....	14		

5-Achs-Delta.....	141
6-Achs-Karteser.....	158
Adabot.....	145
Benutzerdefiniert.....	176
Benutzerdefiniert erweitert.....	178
Karteser.....	112
Stewart-Plattform.....	165
Umschlingungskinematik.....	132, 173
Transformationsvorschriften	
Übersicht.....	28
U	
Überschleifen.....	54
Überschleifen und Bandsynchron.....	75
Überschleifradius.....	55
Umstellung ereignisgesteuertes Kinematik-	
Interface.....	353
uniVAL drive.....	451
V	
Verfahrenbereichsgrenzen.....	79
Visualisierung	
Bandkonfiguration.....	331
Einlernbetrieb	332
Kinematikübersicht.....	327
PCSkonfiguration.....	334
Positionierübersicht.....	329
Referenzierübersicht.....	328
Systemübersicht.....	324
Tipp-/Einrichtbetrieb	332
Überblick.....	323
W	
Warnhinweise.....	14
Z	
Zielgruppe.....	13
Zu dieser Dokumentation.....	13
Gültigkeit der Dokumentation.....	13
Zugriff auf Daten einer Kinematik.....	282
Zustandsdiagramm	
Kinematik.....	33
Zwischenpunkt.....	41
Zyklische Sollwertvorgabe	
Robot-Control V2.....	83

Notizen

Notizen

Bosch Rexroth AG

Electric Drives and Controls

Postfach 13 57

97803 Lohr, Deutschland

Bgm.-Dr.-Nebel-Str. 2

97816 Lohr, Deutschland

Tel. +49 9352 18 0

Fax +49 9352 18 8400

www.boschrexroth.com/electrics



R911341586