

Rexroth IndraLogic XLC IndraMotion MLC 13VRS

PLCopen-Bibliotheken

Bibliothek
R911336359

Ausgabe 04



Titel Rexroth IndraLogic XLC
IndraMotion MLC 13VRS
PLCopen-Bibliotheken

Art der Dokumentation Bibliothek

Dokumentations-Type DOK-XLCMLC-FUNLIB**V13-LI04-DE-P

Interner Ablagevermerk RS-ed362e7d1cdd5b270a6846a5012127e2-3-de-DE-9

Änderungsverlauf

Ausgabe	Stand	Bemerkung
Ausgabe 01	2012-03	Erstausgabe MLC/XLC 13VRS
Ausgabe 02	2013-02	Überarbeitung und Erweiterung "open-loop"
Ausgabe 03	2013-08	Überarbeitung und ML_Robot entfernt
Ausgabe 04	2015-02	Überarbeitung ML_PLCopen (ML_ControlWriteActualData u.a.), ML_ControllerBase (ML_Convert_DoublePrecision)

Schutzvermerk © Bosch Rexroth AG 2015

Alle Rechte bei Bosch Rexroth AG, auch für den Fall von Schutzrechtsanmeldungen. Jede Verfügungsbefugnis, wie Kopier- und Weitergaberecht, bei uns.

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 Systementwicklung Automation Motion Logic Control, KWa, HBu (vha)

Inhaltsverzeichnis

	Seite
1 Zu dieser Dokumentation.....	13
1.1 Gültigkeit der Dokumentation.....	13
1.2 Gebrauch der Sicherheitshinweise.....	13
1.2.1 Aufbau der Sicherheitshinweise.....	13
1.2.2 Erläuterung der Signalwörter und der Signalgrafik.....	14
1.2.3 Verwendete Symbole.....	15
1.3 Erforderliche und ergänzende Dokumentationen.....	15
2 Standardschnittstellen an Funktionsbausteinen.....	21
2.1 Motivation.....	21
2.2 Bausteintypen.....	21
2.3 Bausteintypen und ihre Schnittstellen.....	22
2.4 Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung.....	23
2.5 Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung.....	26
2.6 Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung.....	29
2.7 Ein- und Ausgänge flankengesteuerter Funktionsbausteine, permanente Bearbeitung.....	32
3 RIL_CommonTypes.library.....	35
3.1 RIL_CommonTypes, Überblick.....	35
3.2 Globale_Variablen.....	36
3.2.1 IL_NULL.....	36
3.3 Datentypen zur Referenzierung von Objekten bei IndraMotion-Bausteinen.....	36
3.3.1 Referenzierung von Objekten, allgemeines.....	36
3.3.2 CONTROLS.....	37
3.3.3 AXIS_REF.....	37
3.3.4 OBJECTS.....	38
3.3.5 AXIS_GROUP_REF.....	39
3.3.6 KIN_OBJECTS.....	40
3.3.7 TOUCHPROBE_REF.....	40
3.3.8 TP_OBJECTS.....	41
3.4 Datentypen zur Unterstützung von Kurvenscheiben, Motion- und FlexProfilen.....	41
3.4.1 Datentyp zum Umschaltung zwischen Kurvenscheiben, allgemeines.....	41
3.4.2 MC_CAM_ID.....	42
3.4.3 MC_STEP_MODE, MotionProfile erweitert zu FlexProfile.....	42
3.5 Datentypen zum Aufsynchronisieren in den lagegeregelten Synchron-Betriebsarten.....	66
3.5.1 Allgemeines.....	66
3.5.2 MC_START_MODE.....	66
3.5.3 MC_SYNC_DIRECTION.....	68
3.6 Datentypen zur Fehlerdiagnose bei IndraMotion-Bausteinen.....	69
3.6.1 ERROR_CODE.....	69
3.6.2 ERROR_STRUCT.....	70
3.6.3 ERROR_TABLE.....	71
3.7 Datentypen zur Wiedergabe des Datum und absoluter Zeitangaben bei IndraMotion-Bausteinen.....	71

Inhaltsverzeichnis

	Seite
3.7.1	Wiedergabe des Datum und absoluter Zeitangaben, allgemeines..... 71
3.7.2	SYS_TIME64..... 72
3.7.3	EXT_SYS_TIME_DATE..... 72
3.7.4	SYS_TIME_DATE..... 72
3.8	Datentypen zur Unterstützung von Messtaster-Funktionen..... 73
3.8.1	Datentypen zur Unterstützung von Messtaster-Funktionen, allgemeines..... 73
3.8.2	MB_DIFF_EVAL..... 73
3.8.3	MB_EXPECT_WINDOW_MODE..... 74
3.8.4	MB_INTERPOLATION_MODE..... 75
3.8.5	MB_POSITION_EVAL..... 75
3.8.6	MB_PROBE_MODULE..... 75
3.8.7	MB_PROBE_INPUT..... 76
3.8.8	MB_PROBE_DATA..... 76
3.8.9	MB_PROBE_CONFIG..... 77
3.8.10	MB_PROBE_CONFIG, Besonderheiten bei SERCOS-basierten Achsen..... 77
4	ML_Base.library..... 79
4.1	ML_Base, Überblick..... 79
4.2	Data Types - Datentypen 81
4.2.1	Data Types/ Common Types - Allgemeine Typen..... 81
	MB_IDN..... 81
4.2.2	Data Types/ Flex Structs..... 82
	Datentypen zur Unterstützung von Motion- und FlexProfile-Funktionen, Allgemeines..... 82
	MB_CHANGESTEP_MODE..... 82
	MB_EVENT_ACTION..... 83
	MB_EVENT_TRIGGER..... 84
	MB_EXECUTION_MODE..... 84
	MB_FLEX_EVENT..... 85
	MB_MASTER_TYPE..... 85
	MB_PROFILE_START_POINT..... 86
	MB_PROFILE_STEP..... 87
	MB_STEP_TYPE..... 88
	MB_SYNC_TYPE..... 88
4.2.3	Data Types/ Read Write Parameter - Parameterzugriff..... 88
	MB_SERCOS_DATA_STATUS..... 88
4.2.4	Data Types/ Shared Memory - Direktzugriff..... 89
	Datentypen zur Unterstützung des Shared Memory zwischen Motion und SPS..... 89
	Achsdaten über Shared Memory und Funktionsvariablenzugriff..... 89
	Achsdaten über Shared Memory und Funktionsvariablenzugriff, Allgemeines..... 89
	AxisData[] - Achsdaten im Shared Memory..... 91
	DV_Axis[] - Achsdaten über Funktionsvariablenzugriff..... 94
	GV_LinkAxis[] - Achsdaten über Funktionsvariablenzugriff synchron im C2C-Verbund 96
	Steuerungsdaten über Shared Memory und Funktionsvariablenzugriff..... 98
	Steuerungsdaten über Shared Memory und Funktionsvariablenzugriff, Allgemeines..... 98
	ControlData[] - Steuerungsdaten im Shared Memory..... 100
	DV_Control[] - Steuerungsdaten über Funktionsvariablenzugriff 101

Inhaltsverzeichnis

	Seite
Nockenschaltwerkdaten über Shared Memory und Funktionsvariablenzugriff.....	104
Nockenschaltwerkdaten über Shared Memory und Funktionsvariablenzugriff, Allgemeines....	104
PlsData[] - Nockenschaltwerkdaten im Shared Memory.....	105
DV_Pls[] - Nockenschaltwerkdaten über Funktionsvariablenzugriff.....	107
Messtasterdaten über Shared Memory und Funktionsvariablenzugriff.....	109
Messtasterdaten über Shared Memory und Funktionsvariablenzugriff, Allgemeines.....	109
ProbeData[] - Messtasterdaten im Shared Memory	110
DV_Probe[] - Messtasterdaten über Funktionsvariablenzugriff.....	112
Oszilloskopdaten über Funktionsvariablenzugriff.....	115
Oszilloskopdaten über Funktionsvariablenzugriff, Allgemeines.....	115
DV_Osc[] - Oszilloskopdaten über Funktionsvariablenzugriff.....	115
ML_AXISDATA_SM.....	117
ML_CONTROLDATA_SM.....	119
ML_PLSDATA_SM.....	120
ML_PROBEDATA_SM.....	120
ML_LINK.....	121
ML_SERCOS.....	121
ML_SYNC.....	121
SM_TYPES.....	122
SM_EXT_TYPES.....	122
4.3 Error-/ Diagnostics Memory - Unterstützung von Anwenderdiagnosen und Logbuch-Zugriffen.....	123
4.3.1 ML_ReadDiagLog.....	123
4.3.2 ML_SetAxisDiagEntry.....	125
4.3.3 ML_SetDiagEntry.....	126
4.3.4 E_LOG_CLASS.....	128
4.3.5 E_LOG_FACILITY.....	128
4.3.6 E_LOG_SEVERITY.....	128
4.3.7 ML_LOGBOOK.....	129
4.4 Globale Variables List - Globale Variablen-Liste.....	129
4.4.1 GVL_Base.....	129
4.4.2 GVL_BaseConstant.....	130
4.4.3 Global Variables Lists/ Global Variable Index.....	131
GVL_A_GlobalVarIdx.....	131
4.4.4 Parameter-Konstanten.....	132
Zugriff auf Daten einer Achse, Steuerung oder eines anderen Objektes.....	132
GVL_A_Param_ID.....	134
GVL_C_Param_ID.....	134
GVL_M_Param_ID.....	135
GVL_N_Param_ID.....	135
GVL_O_Param_ID.....	136
GVL_P_Param_ID.....	136
GVL_S_Param_ID.....	137
4.5 Interfaces, Funktionen und Funktionsbausteine.....	137
4.5.1 IL_SercosAttribute.....	137
4.5.2 IL_SercosElementsToParaNo.....	140
4.5.3 IL_SercosStringToParaNo.....	141

Inhaltsverzeichnis

	Seite
4.6 IP-Settings - IP-Konfiguration.....	143
4.6.1 ML_GetGateway.....	143
4.6.2 ML_GetIpAddress.....	144
4.6.3 ML_GetMacAddress.....	145
4.6.4 ML_GetSubnetMask.....	146
4.6.5 ML_SetGateway.....	148
4.6.6 ML_SetIpAddress.....	149
4.6.7 ML_SetSubnetMask.....	150
4.7 SystemTime - Systemzeit.....	151
4.7.1 IL_SetDateTime.....	151
4.7.2 ML_GetSystemTimeUTC.....	152
4.7.3 ML_SetSystemTimeUTC.....	154
4.7.4 ML_DAYSOFWEEK.....	155
4.8 Target - Zielsystem.....	156
4.8.1 ML_GetCpuLoad.....	156
4.8.2 ML_GetDiagnosisLog.....	156
4.8.3 ML_GetFirmwareString.....	157
4.8.4 ML_GetHardwareInfoCtrl.....	157
4.8.5 ML_GetInfoApplication.....	158
4.8.6 ML_GetMaxTemperature.....	159
4.8.7 ML_GetNewestDiagnosisIndex.....	160
4.8.8 ML_GetOhcFan.....	160
4.8.9 ML_GetScalingType.....	160
4.8.10 ML_GetTargetInfo.....	161
4.8.11 ML_GetTemperature.....	161
4.8.12 ML_GetWarningCpuLoad.....	162
4.8.13 ML_SetWarningCpuLoad.....	162
4.8.14 APP_INFO.....	162
4.8.15 E_SCALING_TYPE.....	162
4.8.16 INFO_APPLICATIONS.....	163
4.8.17 ML_DIAGNOSIS_LOG.....	163
4.8.18 TARGET_INFO.....	163
5 Funktionsbausteine nach PLCopen-Spezifikation.....	165
6 ML_PLCopen.library.....	169
6.1 ML_PLCopen, Allgemeines.....	169
6.2 Globale_Variablen.....	172
6.2.1 ProbeSelect.....	172
6.3 Data Types/ Data Types of TouchProbe.....	172
6.3.1 PROBECONFIG.....	172
6.4 Zustandsdiagramm einer realen Achse.....	173
6.5 Funktionsbausteine zur Einzelachssteuerung.....	176
6.5.1 MC_Power.....	176
6.5.2 MC_Power, Besonderheiten bei Sercos basierten Achsen.....	177

Inhaltsverzeichnis

	Seite
6.5.3 MC_Reset.....	177
6.5.4 MC_Reset, Besonderheiten bei SERCOS-basierten Achsen.....	178
6.5.5 MC_Stop.....	179
6.5.6 MC_Stop, Besonderheiten bei SERCOS-basierten Achsen.....	180
6.5.7 MC_MoveAbsolute.....	180
6.5.8 MC_MoveAbsolute, Besonderheiten bei SERCOS-basierten Achsen.....	183
6.5.9 MC_MoveAdditive.....	183
6.5.10 MC_MoveAdditive, Besonderheiten bei SERCOS-basierten Achsen.....	186
6.5.11 MC_MoveRelative.....	186
6.5.12 MC_MoveRelative, Besonderheiten bei SERCOS-basierten Achsen.....	189
6.5.13 MC_MoveVelocity.....	190
6.5.14 MC_MoveVelocity, Besonderheiten bei SERCOS-basierten Achsen.....	193
6.5.15 MC_TorqueControl.....	193
6.5.16 MC_TorqueControl, Besonderheiten bei Sercos basierten Achsen.....	195
6.5.17 MB_SetPositionControlMode.....	196
6.5.18 MB_Home.....	197
6.5.19 MB_Home, Besonderheiten bei Sercos basierten Achsen.....	198
6.6 Funktionsbausteine zur Multiachssteuerung	199
6.6.1 Die MotionControl-Wirkungskette.....	199
6.6.2 MB_Phasing.....	202
6.6.3 MB_PhasingSlave.....	204
6.6.4 MC_CamIn.....	206
6.6.5 MC_CamIn, Besonderheiten bei Sercos basierten Achsen.....	208
6.6.6 MC_CamOut.....	210
6.6.7 MC_CamOut, Besonderheiten bei SERCOS-basierten Achsen.....	211
6.6.8 MB_MotionProfile.....	211
6.6.9 MB_MotionProfile, Besonderheiten bei Sercos basierten Achsen.....	214
6.6.10 PLCopen-Funktionsbausteine zur Unterstützung von MotionProfiles.....	216
Allgemeines.....	216
MB_ChangeProfileSet	217
MB_ChangeProfileSet, Besonderheiten bei SERCOS-basierten Achsen.....	220
MB_ChangeProfileStep.....	222
MB_ChangeProfileStep, Besonderheiten bei SERCOS-basierten Achsen.....	224
MB_ChangeCamData.....	225
MB_ChangeCamData, Besonderheiten bei SERCOS-basierten Achsen.....	226
6.6.11 ML_FlexProfile.....	227
6.6.12 ML_FlexProfile, Besonderheiten bei sercos basierten Achsen.....	230
6.6.13 FlexProfile Umschalten und Aufsynchronisieren	233
Auswahl des Koordinatensystems.....	233
Umschalten und Aufsynchronisieren	233
Art der Abarbeitung.....	236
6.6.14 PLCopen-Funktionsbausteine zur Unterstützung von Flexprofilen.....	237
Allgemeines.....	237
MB_ChangeFlexProfileSet	238
Eigenschaften der Ereignissteuerung.....	239
MB_ChangeFlexEventSet.....	242

Inhaltsverzeichnis

	Seite
6.6.15 MC_GearIn.....	243
6.6.16 MC_GearIn, Besonderheiten bei Sercos basierten Achsen.....	244
6.6.17 MC_GearOut.....	246
6.6.18 MC_GearOut, Besonderheiten bei SERCOS-basierten Achsen.....	247
6.6.19 MB_GearInPos.....	247
6.6.20 MB_GearInPos, Besonderheiten bei Sercos basierten Achsen.....	248
6.7 Funktionsbausteine zur Achsgruppensteuerung.....	250
6.7.1 MC_AddAxisToGroup.....	250
6.7.2 MC_RemoveAxisFromGroup.....	251
6.8 Funktionsbausteine zur Nutzung des zyklischen Datenkanals.....	252
6.8.1 Allgemeines zur Nutzung des zyklischen Datenkanals.....	252
6.8.2 ML_OpenCyclicAnalogChannel.....	254
6.8.3 ML_WriteCyclicAnalog.....	255
6.8.4 ML_OpenCyclicPositionChannel.....	256
6.8.5 ML_WriteCyclicPosition.....	258
6.8.6 ML_OpenCyclicTorqueChannel.....	261
6.8.7 ML_WriteCyclicTorque.....	262
6.8.8 ML_OpenCyclicVelocityChannel.....	263
6.8.9 ML_WriteCyclicVelocity.....	264
6.9 Funktionsbausteine zur geregelten Bewegung.....	265
6.9.1 Allgemeines zur geregelten Bewegung.....	265
6.9.2 ML_ControlOn.....	266
6.9.3 ML_ControlOff.....	267
6.9.4 ML_ControlWriteActualData.....	268
6.9.5 ML_OpenCyclicAnalogChannelCtrl.....	269
6.9.6 ML_WriteCyclicAnalogCtrl.....	270
6.9.7 ML_OpenCyclicPositionChannelCtrl.....	271
6.9.8 ML_WriteCyclicPositionCtrl.....	272
6.9.9 ML_OpenCyclicTorqueChannelCtrl.....	274
6.9.10 ML_WriteCyclicTorqueCtrl.....	275
6.9.11 ML_OpenCyclicVelocityChannelCtrl.....	276
6.9.12 ML_WriteCyclicVelocityCtrl.....	277
6.10 Parameterzugriff.....	278
6.10.1 Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes.....	278
6.10.2 Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen.....	280
6.10.3 MB_Command.....	281
6.10.4 MB_Command, Besonderheiten bei SERCOS-basierten Achsen.....	284
6.10.5 MB_ReadLintParameter.....	285
6.10.6 MB_ReadLintParameter, Besonderheiten bei SERCOS-basierten Achsen.....	287
6.10.7 MB_ReadListParameter.....	287
6.10.8 MB_ReadListParameter, Besonderheiten bei SERCOS-basierten Achsen.....	289
6.10.9 MB_ReadLrealListParameter.....	290
6.10.10 MB_ReadLrealListParameter, Besonderheiten bei SERCOS-basierten Achsen.....	292
6.10.11 MB_ReadLrealParameter.....	292
6.10.12 MB_ReadLrealParameter, Besonderheiten bei SERCOS-basierten Achsen.....	293

Inhaltsverzeichnis

	Seite
6.10.13 MB_ReadMaxRealValue.....	294
6.10.14 MB_ReadMaxRealValue, Besonderheiten bei SERCOS-basierten Achsen.....	295
6.10.15 MB_ReadMaxValue.....	295
6.10.16 MB_ReadMaxValue, Besonderheiten bei SERCOS-basierten Achsen.....	297
6.10.17 MB_ReadMinRealValue.....	297
6.10.18 MB_ReadMinRealValue, Besonderheiten bei SERCOS-basierten Achsen.....	298
6.10.19 MB_ReadMinValue.....	299
6.10.20 MB_ReadMinValue, Besonderheiten bei SERCOS-basierten Achsen.....	300
6.10.21 MB_ReadName.....	300
6.10.22 MB_ReadName, Besonderheiten bei SERCOS-basierten Achsen.....	301
6.10.23 MB_ReadParameter.....	302
6.10.24 MB_ReadParameter, Besonderheiten bei SERCOS-basierten Achsen.....	303
6.10.25 MB_ReadRealParameter.....	303
6.10.26 MB_ReadRealParameter, Besonderheiten bei SERCOS-basierten Achsen.....	304
6.10.27 MB_ReadSercosAttribute.....	305
6.10.28 MB_ReadSercosAttribute, Besonderheiten bei SERCOS-basierten Achsen.....	306
6.10.29 MB_ReadSercosDataStatus.....	306
6.10.30 MB_ReadSercosDataStatus, Besonderheiten bei SERCOS-basierten Achsen.....	308
6.10.31 MB_ReadStringParameter.....	308
6.10.32 MB_ReadStringParameter, Besonderheiten bei SERCOS-basierten Achsen.....	309
6.10.33 MB_ReadUnit.....	310
6.10.34 MB_ReadUnit, Besonderheiten bei SERCOS-basierten Achsen.....	311
6.10.35 MB_WriteLintParameter.....	311
6.10.36 MB_WriteLintParameter, Besonderheiten bei SERCOS-basierten Achsen.....	312
6.10.37 MB_WriteListParameter.....	313
6.10.38 MB_WriteListParameter, Besonderheiten bei SERCOS-basierten Achsen.....	315
6.10.39 MB_WriteLrealParameter.....	315
6.10.40 MB_WriteLrealParameter, Besonderheiten bei SERCOS-basierten Achsen.....	317
6.10.41 MB_WriteParameter.....	317
6.10.42 MB_WriteParameter, Besonderheiten bei SERCOS-basierten Achsen.....	318
6.10.43 MB_WriteRealParameter.....	318
6.10.44 MB_WriteRealParameter, Besonderheiten bei SERCOS-basierten Achsen.....	319
6.10.45 MB_WriteStringParameter.....	320
6.10.46 MB_WriteStringParameter, Besonderheiten bei SERCOS-basierten Achsen.....	321
6.10.47 MC_ReadActualPosition.....	321
6.10.48 MC_ReadActualPosition, Besonderheiten bei SERCOS-basierten Achsen.....	322
6.10.49 MC_ReadAxisError.....	323
6.10.50 MC_ReadAxisError, Besonderheiten bei SERCOS-basierten Achsen.....	324
6.10.51 MC_ReadStatus.....	324
6.10.52 MC_ReadStatus, Besonderheiten bei sercos basierten Achsen.....	325
6.11 Funktionsbausteine zur Nutzung der Messtaster-Funktion.....	325
6.11.1 Daten und Funktionsbausteine zur Nutzung der Messtaster-Funktion.....	325
6.11.2 ML_AbortTrigger.....	327
6.11.3 ML_InitTouchProbe.....	328
6.11.4 ML_TouchProbe.....	334
6.11.5 ML_TouchProbeContinuous.....	337

Inhaltsverzeichnis

	Seite
6.11.6 ML_WriteExpectWindow.....	339
6.12 Zeitbedarf bei der Bearbeitung von Motion-Funktionsbausteinen in der IndraMotion MLC.....	342
6.13 Achstypen und durch sie unterstützte Funktionsbausteine.....	344
6.14 Fehlertabellen (Error_Tables).....	346
6.14.1 SERCOS_TABLE, ERROR_TABLE 16#0010	346
6.14.2 MLC_TABLE, ERROR_TABLE 16#0030	348
7 RMB_PLCOpenFieldBus.library.....	359
7.1 RMB_PLCOpenFieldBus.library, Allgemeines.....	359
7.2 Parameter-Adresskonstanten.....	359
7.2.1 Zugriff auf Daten eines (Feldbus-)Antriebs.....	359
7.2.2 GVL_S_Param_ID.....	359
7.2.3 GVL_P_Param_ID.....	360
7.3 Datatypes.....	360
7.3.1 Datatypes, Allgemeines.....	360
7.3.2 MB_IDN.....	361
7.3.3 AXIS_REF.....	361
7.4 Funktionsbausteine zur Einzelachssteuerung (Motion_FBs).....	363
7.4.1 Motion_FBs, Allgemeines.....	363
7.4.2 MC_Power.....	365
7.4.3 MC_Power, Besonderheiten der Feldbusantriebe.....	366
7.4.4 MC_Reset.....	366
7.4.5 MC_Reset, Besonderheiten der Feldbusantriebe.....	367
7.4.6 MC_Stop.....	368
7.4.7 MC_Stop, Besonderheiten der Feldbusantriebe.....	369
7.4.8 MB_Stop.....	369
7.4.9 MB_Stop, Besonderheiten der Feldbusantriebe.....	371
7.4.10 MB_Home.....	371
7.4.11 MB_Home, Besonderheiten der Feldbusantriebe.....	372
7.4.12 MC_MoveAbsolute.....	372
7.4.13 MC_MoveAbsolute, Besonderheiten der Feldbusantriebe.....	374
7.4.14 MC_MoveAdditive.....	375
7.4.15 MC_MoveAdditive, Besonderheiten der Feldbusantriebe.....	376
7.4.16 MC_MoveRelative.....	377
7.4.17 MC_MoveRelative, Besonderheiten der Feldbusantriebe.....	379
7.4.18 MC_MoveVelocity.....	380
7.4.19 MC_MoveVelocity, Besonderheiten der Feldbusantriebe.....	383
7.5 Funktionsbausteine zum Parameterzugriff (Parameter_FBs).....	383
7.5.1 Parameter_FBs, Allgemeines.....	383
7.5.2 Zugriff auf Daten eines Antriebs (IndraDrive).....	385
7.5.3 Besonderheiten bei Doppel- bzw. Mehrachsmodulen.....	386
7.5.4 MB_ReadListParameter.....	387
7.5.5 MB_ReadListParameter, Besonderheiten der Feldbusantriebe.....	390
7.5.6 MB_ReadListSegParameterPtr.....	390
7.5.7 MB_ReadListSegParameterPtr, Besonderheiten der Feldbusantriebe.....	393
7.5.8 MB_ReadMaxRealValue.....	393

Inhaltsverzeichnis

	Seite
7.5.9 MB_ReadMaxRealValue, Besonderheiten der Feldbusantriebe.....	394
7.5.10 MB_ReadMaxValue.....	395
7.5.11 MB_ReadMaxValue, Besonderheiten der Feldbusantriebe.....	396
7.5.12 MB_ReadMinRealValue.....	396
7.5.13 MB_ReadMinRealValue, Besonderheiten der Feldbusantriebe.....	397
7.5.14 MB_ReadMinValue.....	398
7.5.15 MB_ReadMinValue, Besonderheiten der Feldbusantriebe.....	399
7.5.16 MB_ReadParameter.....	399
7.5.17 MB_ReadParameter, Besonderheiten der Feldbusantriebe.....	401
7.5.18 MB_ReadRealParameter.....	401
7.5.19 MB_ReadRealParameter, Besonderheiten der Feldbusantriebe.....	402
7.5.20 MB_ReadSercosAttribute.....	403
7.5.21 MB_ReadSercosAttribute, Besonderheiten der Feldbusantriebe.....	404
7.5.22 MB_ReadStringParameter.....	404
7.5.23 MB_ReadStringParameter, Besonderheiten der Feldbusantriebe.....	405
7.5.24 MB_WriteListParameter.....	406
7.5.25 MB_WriteListParameter, Besonderheiten der Feldbusantriebe.....	408
7.5.26 MB_WriteListSegParameterPtr.....	408
7.5.27 MB_WriteListSegParameterPtr, Besonderheiten der Feldbusantriebe.....	411
7.5.28 MB_WriteParameter.....	411
7.5.29 MB_WriteParameter, Besonderheiten der Feldbusantriebe.....	412
7.5.30 MB_WriteRealParameter.....	412
7.5.31 MB_WriteRealParameter, Besonderheiten der Feldbusantriebe.....	413
7.5.32 MB_WriteStringParameter.....	414
7.5.33 MB_WriteStringParameter, Besonderheiten der Feldbusantriebe.....	415
7.5.34 MC_ReadActualPosition.....	415
7.5.35 MC_ReadActualPosition, Besonderheiten der Feldbusantriebe.....	416
7.5.36 MC_ReadAxisError.....	417
7.5.37 MC_ReadAxisError, Besonderheiten der Feldbusantriebe.....	418
7.5.38 MC_ReadStatus.....	418
7.5.39 MC_ReadStatus, Besonderheiten der Feldbusantriebe.....	420
7.6 System_FBs	420
7.7 SPS steuert Antriebe über eine Feldbusverbindung (Feldbusachsen).....	421
7.7.1 Allgemeines.....	421
7.7.2 Weiterführende Dokumentationen zu IndraDrives.....	421
7.7.3 Feldbusachsen im Projekt anlegen.....	422
7.7.4 Antriebseinstellungen für die SPS-Kommunikation.....	424
Allgemeines.....	424
Herstellen der Kommunikation zur Antriebsparametrierung.....	424
Kommunikationseinstellungen IndraDrive.....	426
7.7.5 Einbinden der notwendigen Bibliotheken.....	426
7.7.6 Adressierung von Feldbusantrieben im SPS-Projekt.....	429
AXIS_REF, Achsadressen der Feldbusantriebe.....	429
AxisData[], Achsdaten.....	430
7.7.7 E/A-Adressierung der Antriebe im SPS-Projekt.....	430

Inhaltsverzeichnis

	Seite
8 ML_ControllerBase.library	433
8.1 Einleitung.....	433
8.2 Basisklasse des Reglers.....	433
8.2.1 Basisklasse.....	433
8.2.2 Daten eines Reglers.....	434
8.2.3 Umrechnung der Daten eines Reglers.....	435
8.2.4 Interface IActualValue.....	436
8.2.5 Interface IModuleControl.....	436
8.2.6 Interface IOperationModeHandler.....	437
8.2.7 Interface IOutputData.....	437
8.3 Reglerverwaltung.....	437
8.3.1 Verbinden des Reglers mit der Reglerverwaltung.....	437
8.3.2 Zyklischer Ablauf der Reglerverwaltung.....	437
8.3.3 Zustandsmaschine Betriebszustand.....	438
8.3.4 Zyklischer Ablauf des Reglers.....	438
8.3.5 Betriebsartenhandling.....	439
8.3.6 Vordefinierte Generatoren der Betriebsarten.....	441
8.3.7 Diagnosehandling.....	442
9 ML_IPOImpactControl.library	443
9.1 Einführung und Übersicht.....	443
9.2 ML_IPOImpactControl.....	444
9.3 ML_IPOImpactControl-Methoden.....	445
9.3.1 mSetNewState.....	445
9.3.2 mGetNewState.....	445
9.3.3 mSetDirection.....	446
9.3.4 mGetDirection.....	446
9.3.5 mSetAcceleration.....	446
9.3.6 mGetAcceleration.....	447
9.3.7 mSetBreakBehaviour.....	447
9.3.8 mGetBreakBehaviour.....	447
9.3.9 mSetPosition.....	448
9.3.10 mGetPosition.....	448
9.3.11 mSetValidValues.....	448
9.3.12 mGetValidValues.....	449
9.3.13 mSetVelocity.....	449
9.3.14 mGetVelocity.....	449
9.3.15 mGetAxisControlledBy.....	449
9.4 Anwendung der Methoden.....	450
9.4.1 Einleitung.....	450
9.4.2 Beispiel 1: Zustand "Break" aktivieren	450
9.4.3 Beispiel 2: Zustand "Jump" aktivieren.....	451
10 ML_ParamBase.library.....	453
10.1 Verwendung.....	453

Inhaltsverzeichnis

	Seite
10.2 Funktionsbausteine.....	454
10.2.1 PARAM_Parametersystem.....	454
Initialisierung.....	454
Anmelden von Parametern.....	454
Funktionalitäten.....	454
Allgemeines.....	454
Sprache umschalten.....	454
Parameter lesen.....	455
Parameter schreiben.....	455
Parameter sichern.....	455
Parameter wiederherstellen.....	456
Defaultwerte laden.....	456
Parameterlisten erstellen.....	457
Parameterwerte synchronisieren.....	458
Zyklischer Schreibschutz.....	458
10.2.2 PARAM_SercosParameter	458
Allgemeines.....	458
Interfaces.....	458
Verwendung.....	458
Lese-Interface.....	458
Schreib-Interface.....	459
Kommando-Interface.....	459
Attribute.....	460
Zugriffsmethoden.....	461
Datenstatus lesen.....	461
IDN lesen.....	461
Name lesen.....	461
Attribut lesen.....	461
Minimalwert lesen.....	461
Maximalwert lesen.....	461
Datum lesen.....	462
Datum schreiben.....	462
Administration.....	462
Defaultwerte laden.....	462
Parametertyp.....	463
Zyklischer Schreibschutz.....	463
10.2.3 PARAM_ExtSercosParameter.....	463
Allgemeines.....	463
Attribute.....	463
Zugriffsmethoden.....	464
Datum lesen.....	464
Datum schreiben.....	465
Administration.....	465
Defaultwerte laden.....	465
Parametertyp.....	465
10.2.4 PARAM_CmdExecuter.....	465

Inhaltsverzeichnis

	Seite
Kurzbeschreibung.....	465
Methoden.....	466
10.3 Kommandohandling.....	467
10.4 Datentypen.....	468
10.4.1 Sprache.....	468
10.4.2 Datenzugriff.....	468
10.5 Hilfsfunktionen.....	469
10.6 Konstanten.....	470
11 Service und Support.....	473
Index.....	475

1 Zu dieser Dokumentation

1.1 Gültigkeit der Dokumentation

Zielgruppe Diese Dokumentation richtet sich an Anwender, die sich über die Bibliotheken der Systeme IndraLogic XLC und IndraMotion MLC 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 beschreibt die Funktionsbausteine, Funktionen, Schnittstellen und Datentypen der Bibliotheken

- [RIL_CommonTypes, Seite 35](#), Datentypen zur allgemeinen Nutzung in allen IndraWorks-Systemen,
- [ML_Base, Seite 79](#), Datentypen des Systems IndraLogic XLC und IndraMotion MLC für Sercos-basierte Achsen,
- *PLCopen-Funktionsbausteine, Funktionen und Datentypen für*
 - Sercos-basierte Achsen, [ML_PLCopen, Seite 169](#)
 - Profibus-basierte Antriebe, [RMB_PLCopenFieldbus, Seite 359](#),
- Die Bibliothek [ML_ControllerBase, Seite 433](#), bietet die Möglichkeit auf der Basis vorgefertigter Module einen Regler in der SPS zu implementieren.
- Die Bibliothek [ML_IPOImpactControl, Seite 443](#), enthält einen Funktionsbaustein für eine direkte Interpolationsbeeinflussung in einer XLC/MLC.
- Die Bibliothek [ML_ParamBase.library, Seite 453](#), ermöglicht es, die Reglerachse mit einer erweiterten Parametrierbarkeit auszustatten.

Die Dokumentation enthält ferner

- [Informationen zu den Standard-Schnittstellen der Bausteine, Seite 21](#), und
- die [Fehlerreaktion der Bausteine, Seite 346](#).

1.2 Gebrauch der Sicherheitshinweise

1.2.1 Aufbau der Sicherheitshinweise

Die Sicherheitshinweise sind wie folgt aufgebaut:

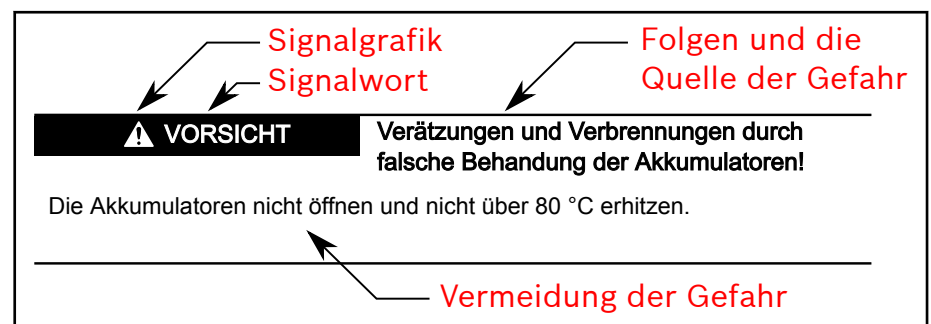


Abb. 1-1: Aufbau der Sicherheitshinweise

Zu dieser Dokumentation

1.2.2 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.

1.2.3 Verwendete Symbole

Fingerzeige werden wie folgt dargestellt:



Dies ist ein Hinweis.

Tipps werden wie folgt dargestellt:



Dies ist ein Tipp.

1.3 Erforderliche und ergänzende Dokumentationen

Dokumentationstitel mit Typenschlüssel und Materialnummer

IndraWorks		MLC	XLC
/36/	Rexroth IndraWorks 13VRS Software-Installation DOK-IWORKS-SOFTINS*V13-CORS-DE-P, R911336879 Die Dokumentation beschreibt den Installationsprozess von IndraWorks.	X	X
/5/	Rexroth IndraWorks 13VRS Engineering DOK-IWORKS-ENGINEE*V13-APRS-DE-P, R911336869 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.	X	X
/20/	Rexroth IndraMotion MLC 13VRS Funktionsbeschreibung DOK-MLC***-FUNC****V13-APRS-DE-P, R911336294 Die Dokumentation dient der Beschreibung der Wizards, Kontextmenüs, Dialoge, Steuerungsinbetriebnahme, Gerätekonfiguration, Funktionalitäten der IndraMotion MLC.	X	
/20/	Rexroth IndraLogic XLC 13VRS Funktionsbeschreibung DOK-XLC***-FUNC****V13-APRS-DE-P, R911336351 Die Dokumentation dient der Beschreibung der Wizards, Kontextmenüs, Dialoge, Steuerungsinbetriebnahme, Gerätekonfiguration, Funktionalitäten der IndraLogic XLC.		X
/7/	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 Kurvenscheibebearbeitung.	X	X
/37/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Automation Interface DOK-XLCMLC-AUT*INT*V13-APRS-DE-P, R911336355 Diese Dokumentation beschreibt den skriptbasierten Zugriff auf IndraWorks-Projektdateien über die Schnittstelle des Automation Interfaces.	X	X
/38/	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.	X	X

Zu dieser Dokumentation

IndraWorks		MLC	XLC
/29/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Projektkonvertierung DOK-XLCMLC-PROCONV*V13-APRS-DE-P, R911336365 Diese Dokumentation beschreibt die Projektkonvertierung von IndraLogic 04VRS und IndraMotion MLC04VRS auf IndraWorks Version 12 mit IndraLogic 2G. Es wird insbesondere auf die Änderungen im Bereich Motion und SPS eingegangen.	X	X
/28/	Rexroth IndraMotion MLC 13VRS Inbetriebnahme DOK-MLC***-STARTUP*V13-CORS-DE-P, R911336307 Das vorliegende Handbuch beschreibt die Inbetriebnahme- und Servicearbeitsschritte für das System IndraMotion MLC. Es beinhaltet Checklisten für häufig durchzuführende Arbeiten und eine ausführliche Beschreibung der Arbeitsschritte.	X	

Tab. 1-1: XLC/MLC-Dokumentationsübersicht

Motion		MLC	XLC
/23/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS PLCopen-Bibliotheken DOK-XLCMLC-FUNLIB**V13-LIRS-DE-P, R911336359 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.	X	X
/27/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Generic Application Template DOK-XLCMLC-TF*GAT**V13-APRS-DE-P, R911336369 Dieses Handbuch stellt dem IndraLogic SPS-Programmierer ein strukturiertes Template, welches zum Hinzufügen und Bearbeiten des SPS-Programmiercodes verwendet werden kann, zur Verfügung. Beschrieben werden das Template selbst, der Template Wizard und Beispielapplikationen.	X	X
/31/	Rexroth IndraMotion MLC 13VRS RCL-Programmieranleitung DOK-MLC***-RCL*PRO*V13-APRS-DE-P, R911336296 Das vorliegende Handbuch informiert über die Robot-Control V1. Die Programmiersprache RCL (Robot Control Language) steht dabei im Vordergrund. Beschrieben werden die Programmstruktur, Variablen, Funktionen, Bewegungsanweisungen sowie die notwendigen Parameter des Systems.	X	
/32/	Rexroth IndraMotion MLC 13VRS Robot-Control V2 DOK-MLC***-ROCO***V13-RE01-DE-P, R911341034 Die Dokumentation informiert über die Robot-Control V2. Die Programmierung mittels 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.	X	
/21/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Parameter DOK-XLCMLC-PARAM***V13-RERS-DE-P, R911336363 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.	X	X

Zu dieser Dokumentation

Motion		MLC	XLC
/10/	Rexroth IndraDrive Firmware für Antriebsregelgeräte MPH-, MPB-, MPD-, MPC-07 DOK-INDRV*-MP*-07VRS**-FKRS-DE-P, R911328669 Diese Dokumentation beschreibt alle funktionellen Eigenschaften der IndraDrive-Firmware in den Varianten MPH-07, MPB-07, MPD-07 und MPC-07.		
/11/	Rexroth IndraDrive MPx-17 Funktionen DOK-INDRV*-MP*-17VRS**-APRS-DE-P, R911331235 Diese Dokumentation beschreibt alle funktionellen Eigenschaften der IndraDrive-Firmware in den Varianten MPB-17, MPM-17, MPC-17 und MPE-17.		

Tab. 1-2: XLC/MLC-Dokumentationsübersicht

Feldbusse		MLC	XLC
/4/	Rexroth IndraWorks 13VRS Feldbusse DOK-IWORKS-FB*****V13-APRS-DE-P, R911336871 Diese Dokumentation dient der Beschreibung der unterstützten Feldbusse und ihrer Diagnose-Funktionsbausteine.	X	X
/4a/	Rexroth IndraWorks 13VRS Feldbus-Bibliotheken DOK-IWORKS-FB*LIB**V13-LIRS-DE-P, R911337856 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.	X	X
/63/	sercos Systemhandbuch für I/O-Geräte DOK-CTRL-ILS3*****-APRS-DE-P, R911333511 Diese Dokumentation beschreibt die Projektierung, Parametrierung, Inbetriebnahme und Diagnose von I/O-Geräten mit sercos Schnittstelle.	X	X

Tab. 1-3: XLC/MLC-Dokumentationsübersicht

HMI		MLC	XLC
/8/	Rexroth IndraWorks 13VRS HMI DOK-IWORKS-HMI*****V13-APRS-DE-P, R911336873 Diese Dokumentation beschreibt die Funktionen, die Konfiguration und die Bedienung der Bedienoberflächen IndraWorks HMI Engineering und IndraWorks HMI Operation.	X	X
/6/	Rexroth IndraWorks 13VRS WinStudio DOK-IWORKS-WINSTUD*V13-APRS-DE-P, R911336881 Diese Dokumentation beschreibt die Installation der Software, das Arbeiten mit WinStudio und das Erstellen und den Betrieb von Applikationen.	X	X
/50/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS HMI-Anbindung DOK-XLCMLC-HMI*****V13-APRS-DE-P, R911336361 Dieses Dokument beschreibt die von der IndraLogic XLC und IndraMotion MLC unterstützten Visualisierungssysteme sowie deren Anbindung.	X	X

Tab. 1-4: XLC/MLC-Dokumentationsübersicht

Zu dieser Dokumentation

SPS		MLC	XLC
/3/	Rexroth IndraWorks 13VRS IndraLogic 2G Programmieranleitung DOK-IWORKS-IL2GPRO*V13-APRS-DE-P, R911336875 Diese Dokumentation beschreibt das SPS-Programmierungswerkzeug IndraLogic 2G und seine Verwendung. Beschrieben werden die Basisverwendung, Erste Schritte, die Visualisierung, Menüpunkte und die Editoren.	X	X
/33/	Rexroth IndraWorks 13VRS Basis-Bibliotheken IndraLogic 2G DOK-IL*2G*-BASLIB**V13-LIRS-DE-P, R911336284 Das vorliegende Handbuch beschreibt die systemübergreifenden SPS-Bibliotheken.	X	X

Tab. 1-5: XLC/MLC-Dokumentationsübersicht

Technologie		MLC	XLC
/30/	Rexroth IndraMotion MLC 13VRS Technologie-Bibliotheken DOK-MLC***-TF*LIB**V13-LIRS-DE-P, R911336323 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.	X	
/60/	Rexroth IndraMotion MLC 13VRS RegisterControl (Bibliothek) DOK-MLC***-REGI*CO*V13-LIRS-DE-P, R911336305 Diese Dokumentation beschreibt die Ein- und Ausgänge der einzelnen Funktionsbausteine und gibt Hinweise für deren Verwendung.	X	
/62/	Rexroth IndraMotion MLC 13VRS RegisterControl (Anwendungsbeschreibung) DOK-MLC***-REGI*CO*V13-APRS-DE-P, R911336303 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.	X	
/49/	Rexroth IndraMotion MLC 13VRS Anwendung Wicklerfunktionen DOK-MLC***-TF*WIND*V13-APRS-DE-P, R911336325 Diese anwendungsbezogene Systemdokumentation beschreibt die Anwendung der Wickler-Technologiefunktionen.	X	

Tab. 1-6: XLC/MLC-Dokumentationsübersicht

Hardware		MLC	XLC
/1/	Rexroth IndraControl L45/L65/L85 Steuerung DOK-CONTRL-ICL45L65L85-PRRS-DE-P, R911332115 Diese Dokumentation dient der Beschreibung der Steuerungen IndraControl L45/L65/L85.	X	X
/2/	Rexroth IndraControl L25 DOK-CONTRL-IC*L25*****-PRRS-DE-P, R911328473 Diese Dokumentation dient der Beschreibung der Steuerungen IndraControl L25.	X	X
/24/	Rexroth IndraControl Lxx 13VRS Funktionsmodule DOK-CONTRL-FM*LXX**V13-APRS-DE-P, R911336282 Diese Dokumentation dient der Beschreibung aller Funktionsmodule der Lxx Steuerungen inkl. Engineering und Diagnose.	X	X

Zu dieser Dokumentation

Hardware		MLC	XLC
/12/	Rexroth IndraDrive Antriebsregelgeräte MPx-02 bis MPx-07 DOK-INDRV*-GEN-**VRS**-PARS-DE-P, R911297316 Diese Dokumentation enthält die Beschreibung aller, in der Firmware für Antriebsregelgeräte der IndraDrive-Familie implementierten Parameter. Sie unterstützt die Parametrierung der Antriebsregelgeräte.		
/13/	Rexroth IndraDrive MPx-02 bis MPx-07 und HMV DOK-INDRV*-GEN-**VRS**-WARS-DE-P, R911297318 Diese Dokumentation enthält die Beschreibung aller in den folgenden Firmwares implementierten Diagnosen: • Antriebsregelgerät-Firmwares MPx-02 bis MPx-08 und • Firmwares der Versorgungsgeräte des Typs "HMV". Sie unterstützt den Maschinenbediener und den Anlagenprogrammierer bei der Fehlerbehebung.		
/35/	Rexroth IndraDrive Antriebsregelgeräte Steuerteile CSB01, CSH01, CDB01 DOK-INDRV*-CSH*****-PRRS-DE-P, R911295011 Diese Dokumentation ist eine Projektierungsbeschreibung zur Auswahl und Verwendung der Steuerteile CSB01, CDB01 und CSH01 für Antriebsregelgeräte der Produktfamilien Rexroth IndraDrive M und Rexroth IndraDrive C.		

Tab. 1-7: XLC/MLC-Dokumentationsübersicht

Diagnose und Service		MLC	XLC
/26/	Rexroth IndraWorks 13VRS IndraMotion Service Tool DOK-IWORKS-IMST****V13-RERS-DE-P, R911337706 Die Dokumentation beschreibt das IndraMotion Service Tool (IMST). Es ist ein webbasiertes Diagnosewerkzeug, das zum Zugriff auf die Steuerungssysteme IndraMotion MLC L25, L45 oder L65 sowie MLP 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 der Internet Explorer Version 6 oder 7.	X	X
/22/	Rexroth IndraLogic XLC IndraMotion MLC 13VRS Diagnosen DOK-XLCMLC-DIAG****V13-RERS-DE-P, R911336357 Diese Dokumentation enthält alle in den Steuerungssystemen IndraLogic XLC und IndraMotion MLC implementierten Steuerungsdiagnosen.	X	X

Tab. 1-8: XLC/MLC-Dokumentationsübersicht

Systemübersicht		MLC	XLC
/48/	Rexroth IndraMotion for Printing 13VRS Systemübersicht DOK-IM*PR*-SYSTEM**V13-PRRS-DE-P, R911336288 Das vorliegende Handbuch beschreibt das Produkt IndraMotion for Printing. Es stellt die Steuerungs-, Antriebs- und IO-Systeme vor und gibt eine Einführung in die Inbetriebnahme und Programmierung.	X	
/48/	Rexroth IndraMotion for Packaging 13VRS Systemübersicht DOK-IM*PA*-SYSTEM**V13-PRRS-DE-P, R911336286 Das vorliegende Handbuch beschreibt das Produkt IndraMotion for Packaging. Es stellt die Steuerungs-, Antriebs- und IO-Systeme vor und gibt eine Einführung in die Inbetriebnahme und Programmierung.	X	

Zu dieser Dokumentation

Systemübersicht		MLC	XLC
/9/	Rexroth IndraMotion MLC 13VRS Systemübersicht DOK-MLC***-SYSTEM**V13-PRRS-DE-P, R911336309 Diese Dokumentation stellt eine Übersicht der möglichen Hardware-/Softwarekomponenten des Automationssystems IndraMotion MLC in der genannten Version dar. Die Dokumentation dient als Auswahlhilfe zur Zusammenstellung eines Systems.	X	
/9/	Rexroth IndraLogic XLC 13VRS Systemübersicht DOK-XLC***-SYSTEM**V13-PRRS-DE-P, R911336353 Diese Dokumentation stellt eine Übersicht der möglichen Hardware-/Softwarekomponenten des Automationssystems IndraLogic XLC in der genannten Version dar. Die Dokumentation dient als Auswahlhilfe zur Zusammenstellung eines Systems.		X

Tab. 1-9: XLC/MLC-Dokumentationsübersicht

Erste Schritte		MLC	XLC
/25/	Rexroth IndraMotion MLC 13VRS Die ersten Schritte DOK-MLC***-F*STEP**V13-CORS-DE-P, R911336292 Diese Dokumentation beschreibt die "ersten Schritte" mit der IndraMotion MLC und RobotControl. Beschrieben werden die Hard- und Softwarevoraussetzungen und das Erstellen eines Projekts.	X	
/25/	Rexroth IndraLogic XLC 13VRS Die ersten Schritte DOK-XLC***-F*STEP**V13-CORS-DE-P, R911336349 Diese Dokumentation beschreibt die "ersten Schritte" mit der IndraLogic XLC. Beschrieben werden die Hard- und Softwarevoraussetzungen und das Erstellen eines Projekts.		X

Tab. 1-10: XLC/MLC-Dokumentationsübersicht

2 Standardschnittstellen an Funktionsbausteinen

2.1 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.

2.2 Bausteintypen

Mit Funktionsbausteinen ist es möglich, komplexe Aufgaben wiederverwendbar zu kapseln und diese 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 er einen definierten Zustand erreicht hat, arbeitet er „zustandsgesteuert“.

Flankengesteuert Erfüllt ein Funktionsbaustein nach dem Einschalten seine Aufgabe genau einmal, so arbeitet er „flankengesteuert“.

Abschließende Bearbeitung Kann ein Baustein seinen Auftrag endgültig bearbeiten und seine Arbeit anschließend einstellen, so ist dies eine „abschließende Bearbeitung“.

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. Er 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 diese solange bis der Steuereingang (Enable) zurückgesetzt wird und diese 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 sie nicht mehr über die Schnittstellen dieses Bausteines deaktiviert werden.) z.B. MC_MoveVelocity

Tab. 2-1: Übersicht Bausteintypen

Standardschnittstellen an Funktionsbausteinen

2.3 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 Funktionsbausteins 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 gleiches Verhalten.

Standardisierte Bezeichner

Bezeichner	Beschreibung
Eingänge	
Enable	Aktivierungseingang zustandsgesteuerter FBs
Execute	Aktivierungseingang flankengesteuerter FBs. Andere FBs können diesen unterbrechen.
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 bleiben, 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. 2-2: Übersicht standardisierter Bezeichner

2.4 Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung

Steuerung / Bearbeitung	E/A	Variablenname	Beschreibung
Zustandsge- steuert / ab- schließend	E	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.
	A	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.
	A	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!
	A	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!
	A	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 gleichzeitig mit „CommandAborted“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.
	A	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 gleichzeitig mit „Error“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.

Tab. 2-3: E/A 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 Variablenamen 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).

Standardschnittstellen an Funktionsbausteinen

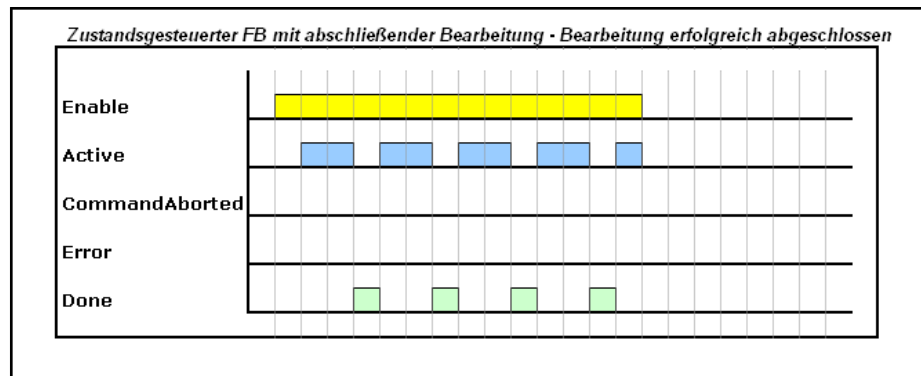


Abb. 2-1: E/A zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung ohne „Shutdown“. Bearbeitung erfolgreich abgeschlossen

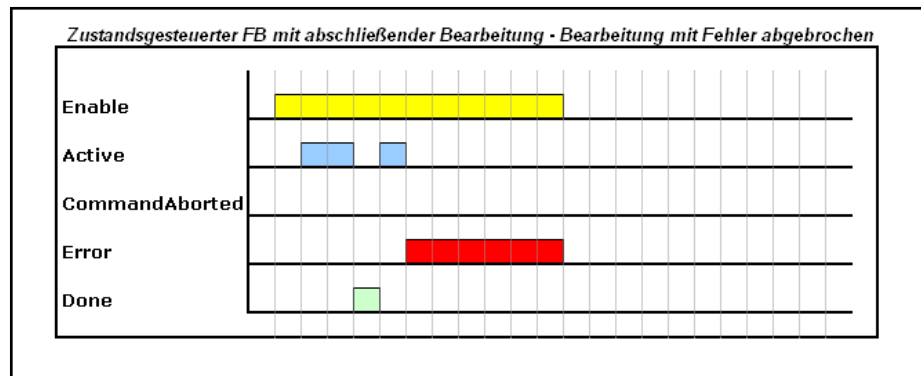


Abb. 2-2: E/A zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung ohne „Shutdown“. Bearbeitung mit Fehler abgeschlossen

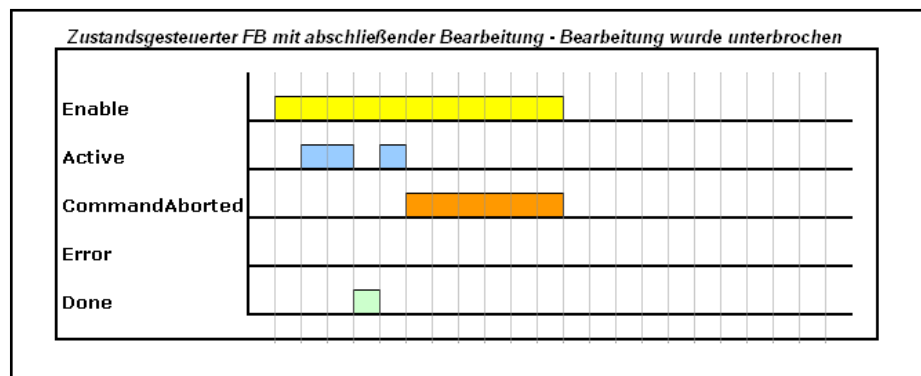


Abb. 2-3: E/A zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung ohne „Shutdown“. Bearbeitung unterbrochen

Standardschnittstellen an Funktionsbausteinen

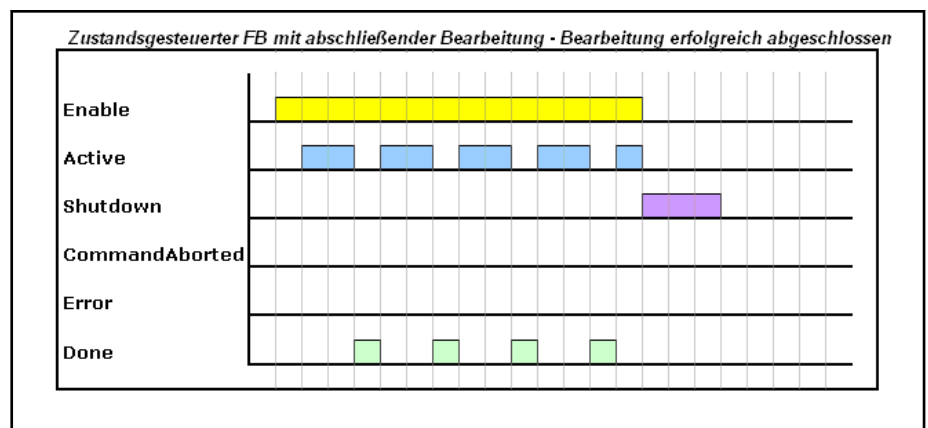


Abb. 2-4: E/A zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit „Shutdown“. Bearbeitung erfolgreich abgeschlossen.

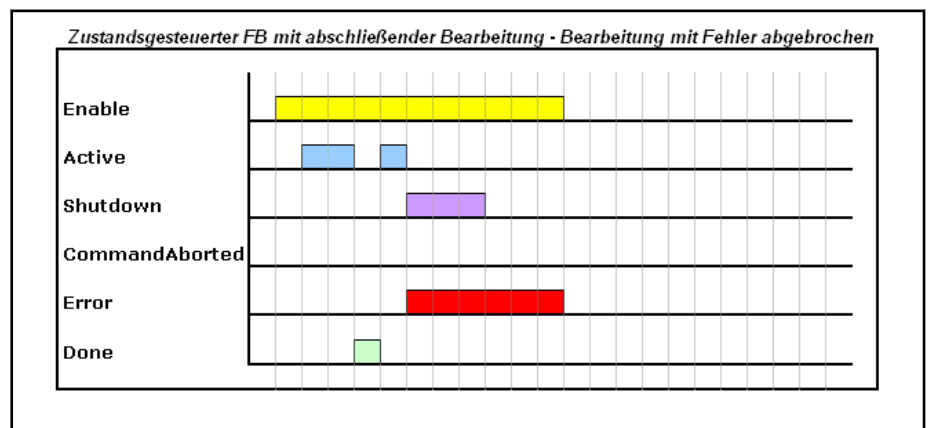


Abb. 2-5: E/A zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit „Shutdown“. Bearbeitung mit Fehler abgeschlossen.

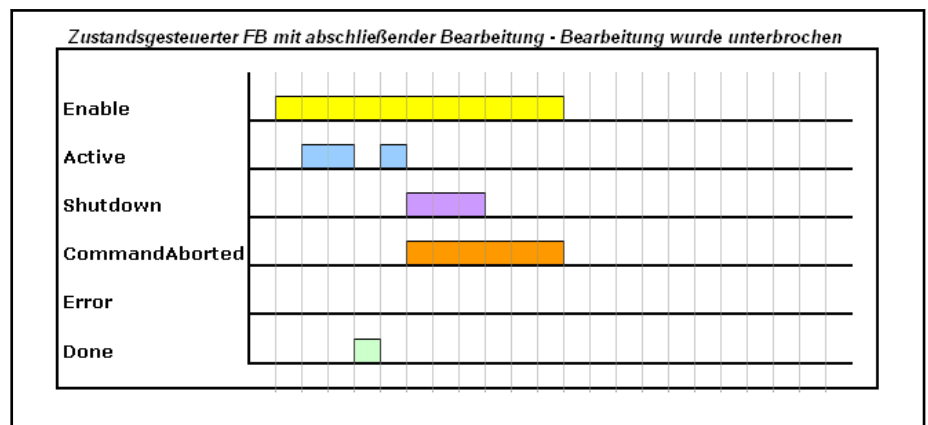


Abb. 2-6: E/A zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit „Shutdown“. Bearbeitung wurde unterbrochen.

Standardschnittstellen an Funktionsbausteinen

2.5 Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung

Steuerung / Bearbeitung	E/A	Name	Beschreibung
Flankengesteuert / abschließend	E	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 (Retriggung, Nachtriggern). Ist ein „Shutdown“- Ausgang vorhanden, wird Retriggen möglicherweise nicht unterstützt. Die Bausteindokumentation schafft hierüber Klarheit.
	A	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 gleichzeitig mit „Done“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.
	A	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!
	A	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!
	A	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 gleichzeitig mit „CommandAborted“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.
	A	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 gleichzeitig mit „Error“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.

Tab. 2-4: E/A zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung

Standardschnittstellen an Funktionsbausteinen



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).

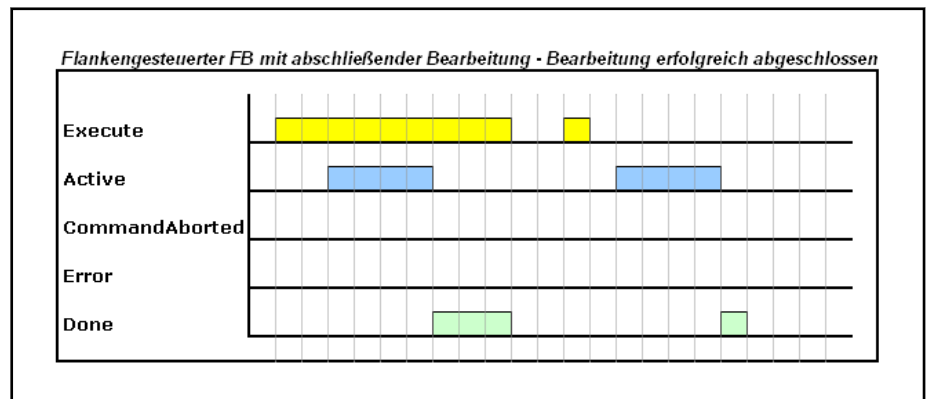


Abb. 2-7: E/A flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne „Shutdown“. Bearbeitung erfolgreich abgeschlossen

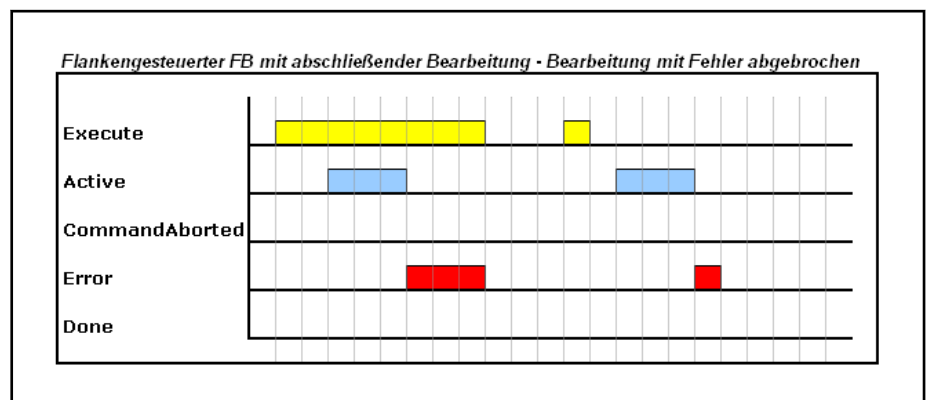


Abb. 2-8: E/A flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne „Shutdown“. Bearbeitung mit Fehler abgeschlossen

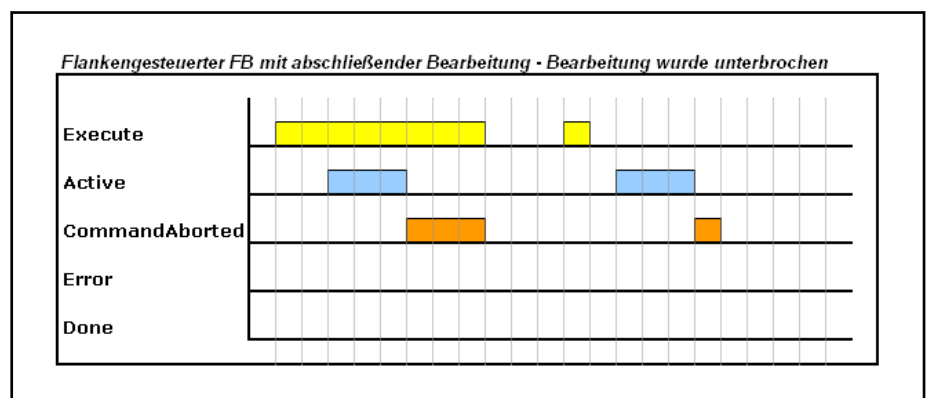


Abb. 2-9: E/A flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne „Shutdown“. Bearbeitung unterbrochen

Standardschnittstellen an Funktionsbausteinen

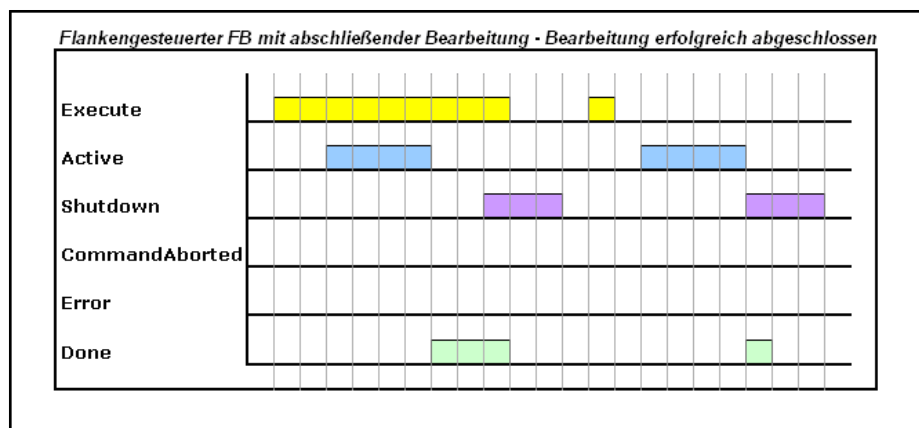


Abb. 2-10: E/A flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit „Shutdown“. Bearbeitung erfolgreich abgeschlossen

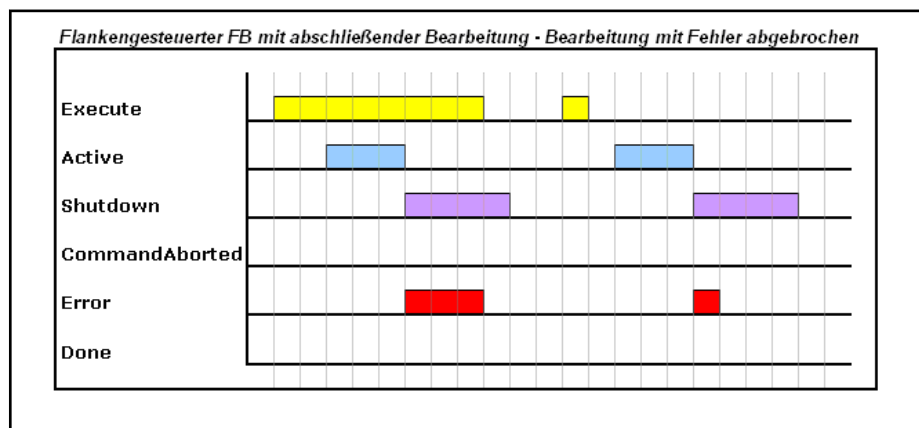


Abb. 2-11: E/A flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit „Shutdown“. Bearbeitung mit Fehler abgeschlossen

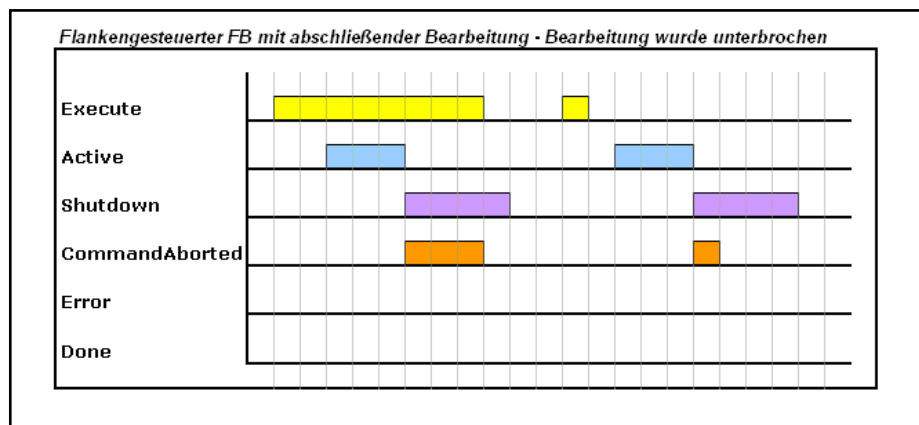


Abb. 2-12: E/A flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit „Shutdown“. Bearbeitung unterbrochen

2.6 Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung

Steuerung / Bearbeitung	E/A	Variablenname	Beschreibung
Zustandsge- steuert / per- manent	E	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.
	A	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.
	A	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!
	A	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!
	A	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 gleichzeitig mit „CommandAborted“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.
	A	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 gleichzeitig mit „Error“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.

Tab. 2-5: E/A 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 Variablenamen kenntlich zu machen!

Standardschnittstellen an Funktionsbausteinen



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).

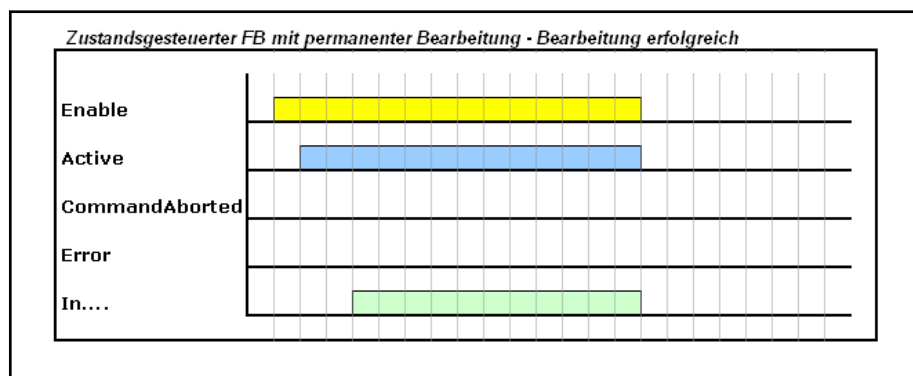


Abb. 2-13: E/A zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne „Shutdown“. Bearbeitung erfolgreich

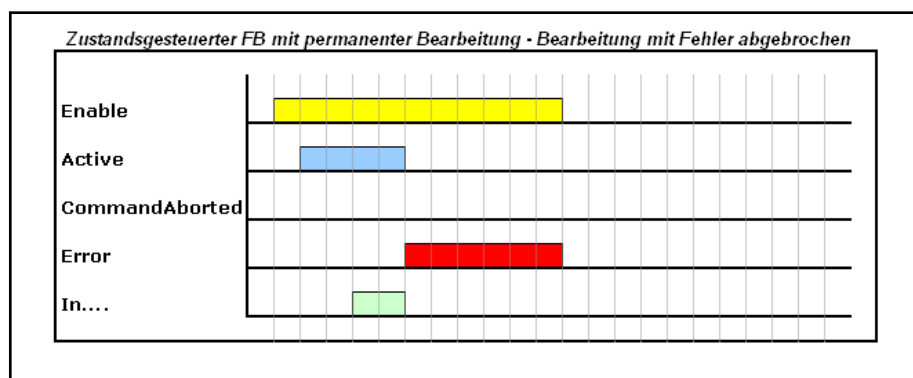


Abb. 2-14: E/A zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne „Shutdown“. Bearbeitung mit Fehler abgebrochen

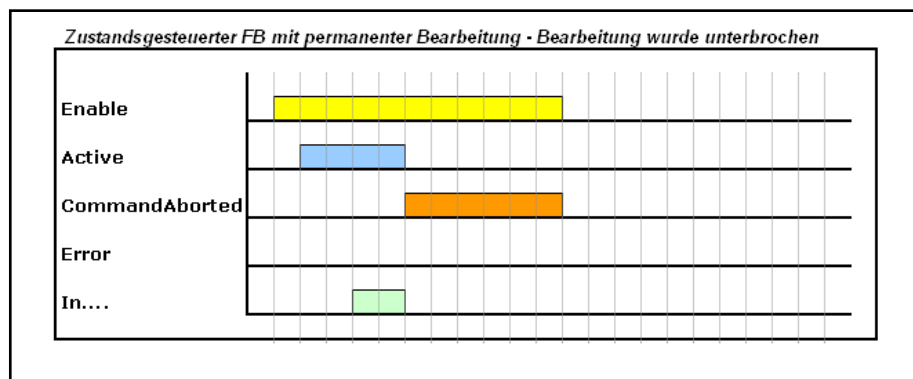


Abb. 2-15: E/A zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne „Shutdown“. Bearbeitung unterbrochen

Standardschnittstellen an Funktionsbausteinen

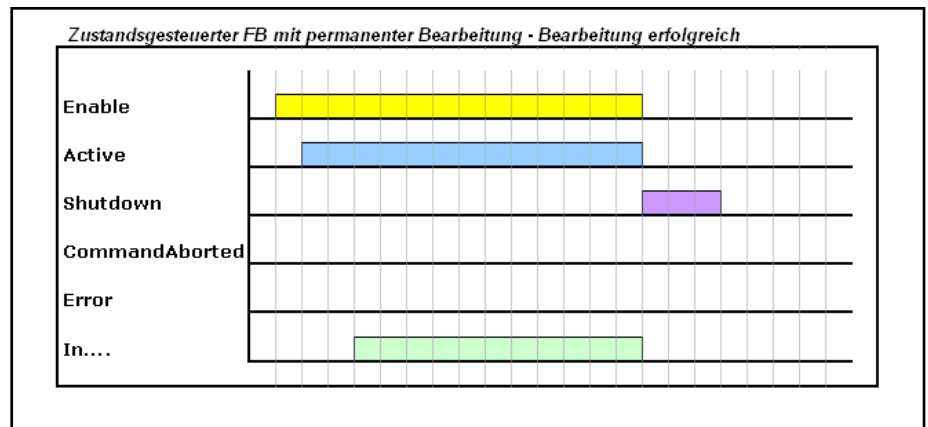


Abb. 2-16: E/A zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit „Shutdown“. Bearbeitung erfolgreich

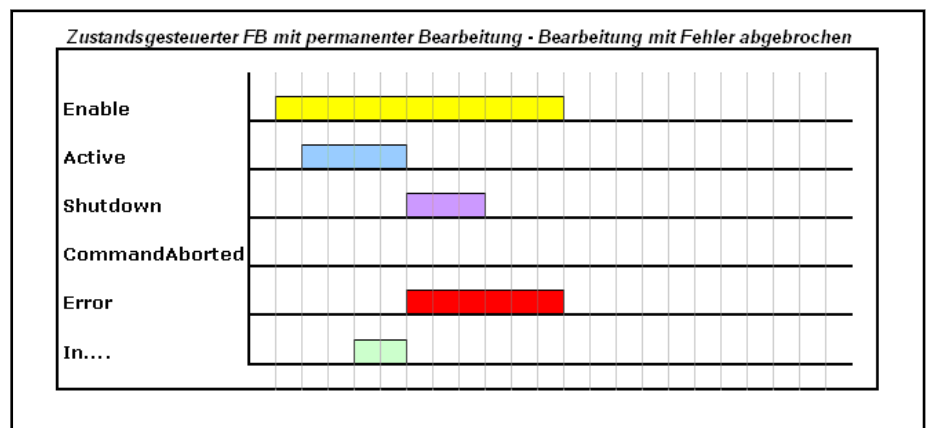


Abb. 2-17: E/A zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit „Shutdown“. Bearbeitung mit Fehler abgebrochen

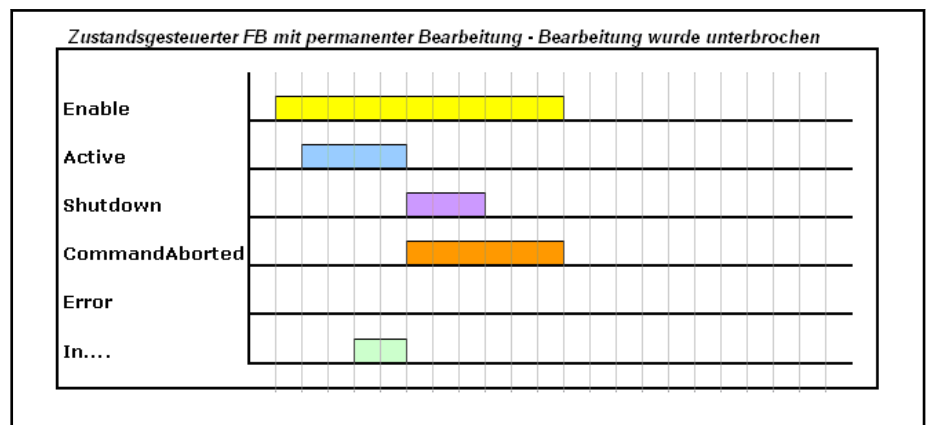


Abb. 2-18: E/A zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit „Shutdown“. Bearbeitung unterbrochen

Standardschnittstellen an Funktionsbausteinen

2.7 Ein- und Ausgänge flankengesteuerter Funktionsbausteine, permanente Bearbeitung

Steuerung / Bearbeitung	E/A	Variablenname	Beschreibung
Flankengesteuert / permanent	E	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 (Retriggierung, Nachtriggern). Hat der Baustein einen „Shutdown“-Ausgang, ist dieses Retriggern eventuell nicht erlaubt.
	A	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.
	A	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!
	A	Shutdown (optional)	Ist ein „Shutdown“-Ausgang vorhanden, benötigt der Baustein, sobald er 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!
	A	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 gleichzeitig mit „CommandAborted“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.
	A	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 gleichzeitig mit „Error“ TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis „Shutdown“ FALSE wird.

Tab. 2-6: E/A flankengesteuerter Funktionsbausteine, permanente Bearbeitung

Standardschnittstellen an Funktionsbausteinen



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).

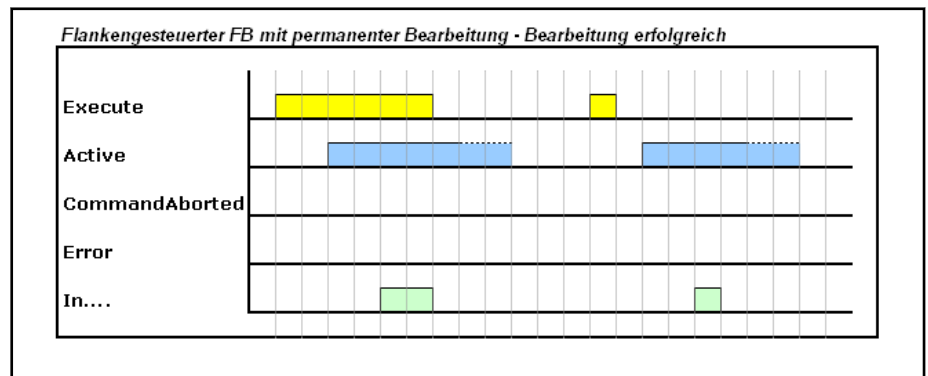


Abb. 2-19: E/A flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne „Shutdown“. Bearbeitung erfolgreich

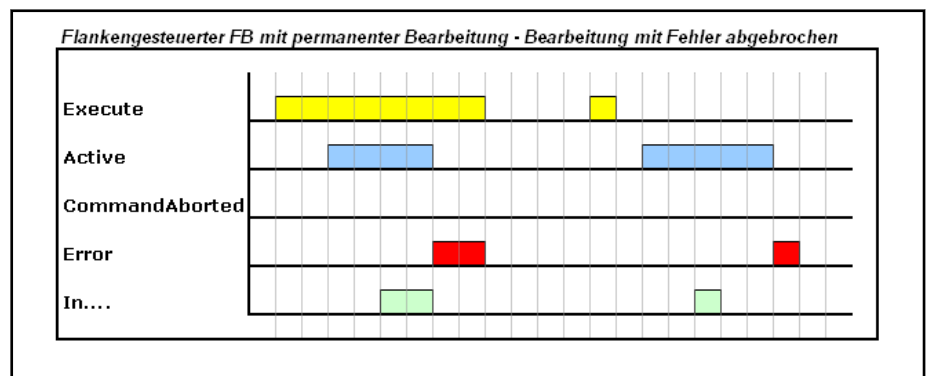


Abb. 2-20: E/A flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne „Shutdown“. Bearbeitung mit Fehler abgebrochen

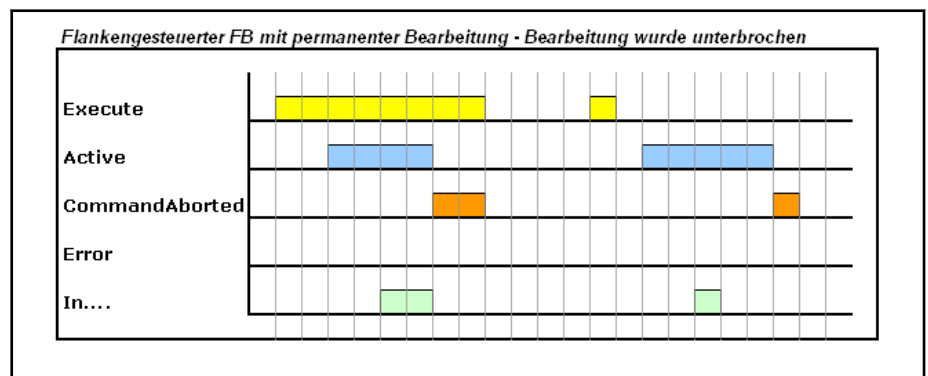


Abb. 2-21: E/A flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne „Shutdown“. Bearbeitung unterbrochen

Standardschnittstellen an Funktionsbausteinen

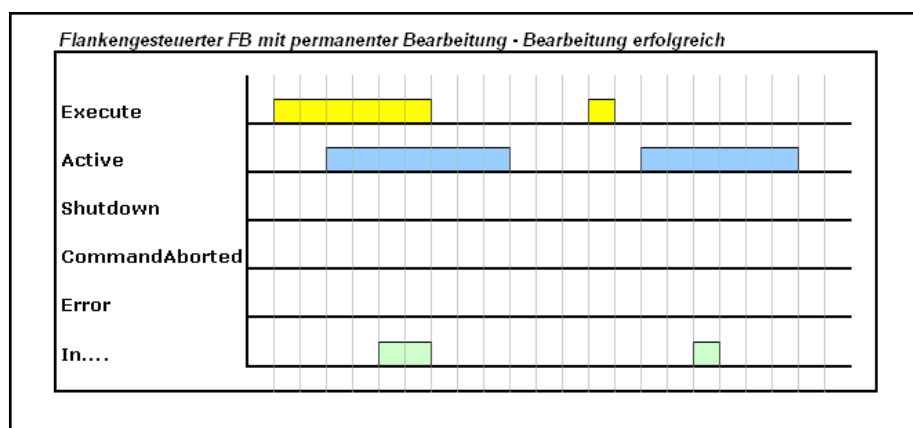


Abb. 2-22: E/A flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit „Shutdown“. Bearbeitung erfolgreich

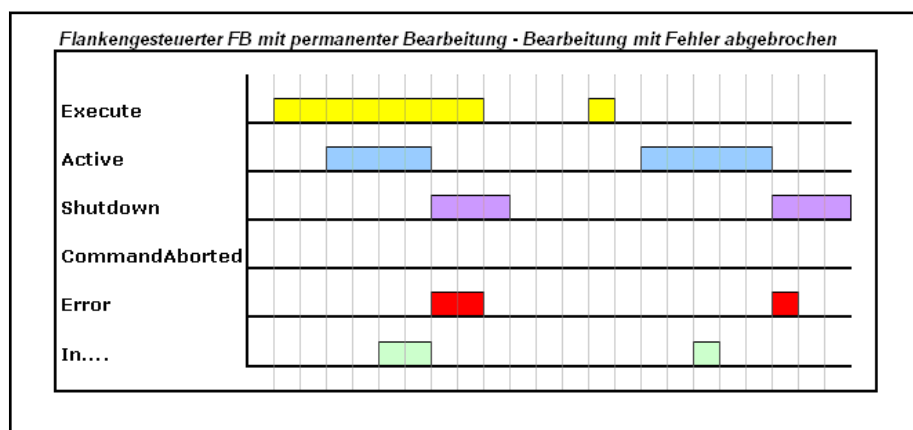


Abb. 2-23: E/A flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit „Shutdown“. Bearbeitung mit Fehler abgebrochen

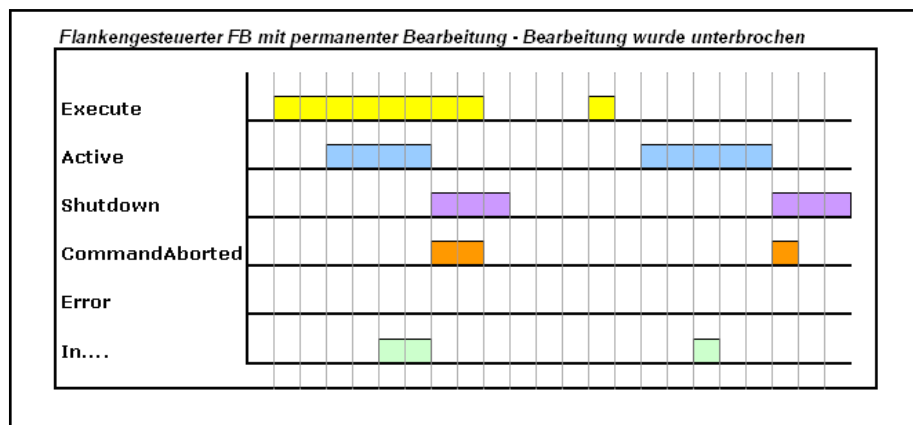


Abb. 2-24: E/A flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit „Shutdown“. Bearbeitung unterbrochen

3 RIL_CommonTypes.library

3.1 RIL_CommonTypes, Überblick

Die Bibliothek RIL_CommonTypes.library enthält die Datentypen, die Ein- und Ausgangsvariablen der IndraMotion-Bausteine der Firma Bosch Rexroth beschreiben.

Die Bibliothek weist folgende Struktur auf:

Globale_Variablen

Globale_Variablen

- [IL_NULL](#), Seite 36, NIL-Pointer.

SERCOSAttribute

- Die globale Variablenliste enthält die sercos Attribute, einschließlich eines Kommentars bezüglich der Verwendung.

Datentypen

Datatypes of logical addresses

- [CONTROLS](#), Seite 37, (ENUM),
- [KIN_OBJECTS](#), Seite 40, (ENUM),
- [OBJECTS](#), Seite 38, (ENUM),
- [TP_OBJECTS](#), Seite 41, (ENUM),

Datatypes of PLCopen

- [AXIS_GROUP_REF](#), Seite 39, (STRUCT),
- [AXIS_REF](#), Seite 37, (STRUCT),
- [MC_CAM_ID](#), Seite 42, (ENUM),
- [MC_START_MODE](#), Seite 66, (ENUM),
- [MC_STEP_MODE](#), Seite 42, (ENUM),
- [MC_SYNC_DIRECTION](#), Seite 68, (ENUM),
- [TOUCHPROBE_REF](#), Seite 40, (STRUCT),

Datatypes of POU diagnosis

- [ERROR_STRUCT](#), Seite 70, (STRUCT)

Includes

- [ERROR_CODE](#), Seite 69, (ENUM)
- [ERROR_TABLE](#), Seite 71, (ENUM)

Datatypes of times

- [EXT_SYS_TIME_DATE](#), Seite 72, (STRUCT)
- [SYS_TIME64](#), Seite 72, (STRUCT)
- [SYS_TIME_DATE](#), Seite 72, (STRUCT)

TouchProbe

- [MB_DIFF_EVAL](#), Seite 73, (ENUM)
- [MB_EXPECT_WINDOW_MODE](#), Seite 74, (ENUM)

RIL_CommonTypes.library

- [MB_INTERPOLATION_MODE](#), Seite 75, (ENUM)
- [MB_POSITION_EVAL](#), Seite 75, (ENUM)
- [MB_PROBE_DATA](#), Seite 76, (ENUM)
- [MB_PROBE_INPUT](#), Seite 76, (ENUM)
- [MB_PROBE_MODULE](#), Seite 75, (ENUM)
- [MB_PROBE_CONFIG](#), Seite 77, (ENUM)

Sie wurde vereinbart, um einen einheitlichen Zugriff auf unterschiedliche IndraMotion-Gerätetypen für unterschiedliche Steuerungssysteme zu definieren.

3.2 Globale_Variablen

3.2.1 IL_NULL

Kurzbeschreibung Definiert wurde ein NIL-Pointer als POINTER TO BYTE.

Bibliothek	Bereich
RIL_CommonTypes	Globale_Variablen

Tab. 3-1: Bibliothekszuordnung

Programm:

```
VAR_GLOBAL
  IL_NULL: POINTER TO BYTE := 0;
END_VAR
```

3.3 Datentypen zur Referenzierung von Objekten bei IndraMotion-Bausteinen

3.3.1 Referenzierung von Objekten, allgemeines

Zur Referenzierung von Objekten (Steuerungen, Achsen, Kinematiken/Achsgruppen, Messtastern usw.) werden für jeden Objekttyp eine REF-Struktur mit zugeordneten Aufzählungstypen angeboten.

Die Informationsübergabe erfolgt per VAR_IN_OUT an den jeweiligen Funktionsbaustein.

Die REF-Struktur enthält zwei 16-Bit-Adresselemente:

- die Steuerungsadresse (Element von [CONTROLS](#), Seite 37) und
- die Objektadresse, Element des Aufzählungstypes.

REF-Struktur	Aufzählungstyp
AXIS_REF , Seite 37, Achsen, z.Z. noch Nockenschaltwerke	OBJECTS , Seite 38
AXIS_GROUP_REF , Seite 39, Achsgruppen/ Kinematiken	KIN_OBJECTS , Seite 40
TOUCHPROBE_REF , Seite 40, Messtaster	TP_OBJECTS , Seite 41

Tab. 3-2: REF-Strukturen und ihre Aufzählungstypen



Die für den konkreten Einsatzfall verwendbaren Elemente der Aufzählungstypen sind abhängig von

- System,
- Steuerungstyp / Antriebtyp und
- Firmware-Version.

3.3.2 CONTROLS

Kurzbeschreibung

Die Aufzählung enthält alle Steuerungen, die im IndraMotion-Verbund über IndraMotion-Funktionsbausteine adressiert werden können:



Im Falle einer physikalischen Adressierung ist die "lokale Steuerung" über zwei Möglichkeiten erreichbar, Nummer "0", lokal, und über ihre eigene Adresse z.B. CTRL_3.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of logical addresses

Tab. 3-3: Bibliothekszuordnung

Typdefinition

```
// Enumeration of valid logical control addresses
TYPE CONTROLS :
(
    LOCAL_CNTRL := 0,      // Local control (default)
    CNTRL_1 := 1,          // Control 1
    CNTRL_2 := 2,          // Control 2
    // ...
    CNTRL_99 := 99         // Control 99
);
END_TYPE
```

Funktionsbeschreibung

Diese Aufzählung der angeschlossenen Steuerungen enthält alle Steuerungen, die im IndraControl-Verbund über IndraMotion-Funktionsbausteine adressiert werden können:

- Die Nummer 0 ist freigegeben für die lokale Steuerung (die Steuerung selbst),
- Der Zahlenbereich 1 .. 99 ist reserviert für alle verbundenen Steuerungen.

3.3.3 AXIS_REF

Kurzbeschreibung

Die Struktur enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt (Achse, Nockenschaltwerk, Oszilloskop...).

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of PLCopen

Tab. 3-4: Bibliothekszuordnung

Typdefinition

```
(* Structure of PLCopen element AXIS_REF. The AXIS_REF is a structure that contains *)
(* information on the corresponding object. *)
(* It is used as a VAR_IN_OUT in all Motion Control Function Blocks. *)
TYPE AXIS_REF :
STRUCT
    CntrlNo: CONTROLS := LOCAL_CNTRL; (* control reference number *)
    AxisNo: OBJECTS := NO_OBJECT; (* Steuerungsnummer, Init: lokale Steuerung *)
    (* object reference number *)
    (* Achsreferenznummer, Init: kein Objekt *)
END_STRUCT
```

RIL_CommonTypes.library

```
END_STRUCT
END_TYPE
```

Funktionsbeschreibung

Die Informationsübergabe erfolgt per VAR_IN_OUT an den jeweiligen PLCopen-Funktionsbaustein.

Die AXIS_REF-Struktur enthält zwei 16-Bit-Adresselemente:

- die Steuerungsadresse (Element von [CONTROLS](#), Seite 37) und
- die Objektadresse (Element von [OBJECTS](#), Seite 38).

Die Elemente können per logischer oder physischer Adresse angesprochen werden. Im Falle einer logischen Adresse zeigt das Element auf eine Parameterstruktur mit einer physikalischen Adresse (z.B. Ethernet-Adresse der Steuerung, SERCOS-Adresse des Antriebs o.ä.).

3.3.4 OBJECTS

Kurzbeschreibung

Die Aufzählung der verfügbaren Achsen enthält alle Objekte eines IndraMotion-Verbundes, auf die per "AXIS_REF "durch IndraMotion-Bausteine zugegriffen werden kann.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of logical addresses

Tab. 3-5: Bibliothekszuordnung

Typdefinition

```
// Enumeration of valid logical object addresses
TYPE OBJECTS :
(
  // not specified object
  NO_OBJECT := 0,           // no object specified
  // drives 0001..0999
  AXIS_1 := 1,              // drive 1
  // ...
  AXIS_99 := 99,            // drive 99
  AXIS_100 := 100,          // drive 100
  // ...
  AXIS_512 := 512,          // drive 512

  RMA_1 := 1001,             (* real master axis 1 *)
  (* RMA_2 := 1002,          (* real master axis 2 *)

  VMA_1 := 2001,             (* virtual master axis 1 *)
  VMA_2 := 2002,             (* virtual master axis 2 *)
  VMA_INT := 2003,           (* internal virtual master axis *)
  VMA_EXT := 2004,           (* external virtual master axis *)
  MA_LINK_1 := 2005,         (* link axis 1 *)
  MA_LINK_2 := 2006,        (* link axis 2 *) *)

  // ELS Group Local Master 3001..3999
  GLMA_1 := 3001,           // ELS Group Local Master Axis 1
  // ...
  GLMA_8 := 3008,           // ELS Group Local Master Axis 8

  // ELS Groups 4001..4008
  ELSG_1 := 4001,           // ELS Group 1
  // ...
  ELSG_8 := 4008,           // ELS Group 8

  // ELS System Masters 5001..5006
  ELSM_1 := 5001,           // ELS System Master 1
  // ...
  ELSM_6 := 5006,           // ELS System Master 6

  // FieldBus Drives 6001..6099
  FieldBusAxis_1 := 6001,   //Fieldbus Drive 1
  // ...
  FieldBusAxis_99 := 6099,  //Fieldbus Drive 99

  // Program Limit Switch 8001..8999
  PLS_1 := 8001,            //Program Limit Switch unit 1
  // ...
  PLS_4 := 8004,            //Program Limit Switch unit 4
```

```

// Oscilloscope 9001..9999
OSCI_CHANNEL_1 :=9001, // Oscilloscope Channel 1
// ...
OSCI_CHANNEL_128 :=9128 // Oscilloscope Channel 128

// Sercos address 10001..10512
SERCOS_ADDRESS_1 :=10001, // Sercos address 1
// ...
SERCOS_ADDRESS_512 :=10512, // Sercos address 512

// Sercos topology 11001..11512
SERCOS_TOPOLOGY_1 :=11001, // Sercos topology 1
// ...
SERCOS_TOPOLOGY_512 :=11512, // Sercos topology 512

// Logical drive 12001..12512 with native operation data *)
AXIS_NATIVE_1 := 12001, // Logical drive 1 with native operation data
// ...
AXIS_NATIVE_512 := 12512 // Logical drive 512 with native operation data
);
END_TYPE

```



Die Nummer "00", "NO_OBJECT" (es wurde kein Objekt angegeben) wird für deaktivierte Achsen verwendet.



Die Vergabe des Namens und der Nummern erfolgt beim Anlegen der jeweiligen Achse/ des Objektes in der IndraWorks-Suite (IndraLogic, GlobaleMlcVariablen).

3.3.5 AXIS_GROUP_REF

Kurzbeschreibung

Die Struktur enthält Informationen zur gewünschten Steuerung und zur korrespondierenden Achsgruppe.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of PLCopen

Tab. 3-6: Bibliothekszuordnung
Typdefinition

```

// Structure of PLCopen element AXIS_GROUP_REF.
// The AXIS_GROUP_REF is a structure that contains information of the axis-group.
// It is used as a VAR_IN_OUT in Motion Control Function Blocks.
TYPE AXIS_GROUP_REF:
STRUCT
  CntrlNo: CONTROLS := LOCAL_CNTRL; // control reference number
                                     // Steuerungsnummer, Init: lokale Steuerung
  GroupNo: KIN_OBJECTS := NO_KINEMATICS; // object reference number
                                     // Achsgruppenreferenznummer, Init: NO_KINEMATICS
END_STRUCT
END_TYPE

```

Funktionsbeschreibung

Die Informationsübergabe erfolgt per VAR_IN_OUT an den jeweiligen in der "ML_PLCopen.library" definierten PLCopen-Funktionsbaustein, der sich auf eine Gruppe von Achsen bezieht.

Die AXIS_GROUP_REF-Struktur enthält zwei 16-Bit-Adresselemente:

- die Steuerungsadresse (Element von [CONTROLS](#), Seite 37) und
- die Objektadresse (Element von [KIN_OBJECTS](#), Seite 40).

Die Elemente können per logischer oder physischer Adresse angesprochen werden. Im Falle einer logischen Adresse zeigt das Element auf eine Parameterstruktur mit einer physikalischen Adresse (z.B. Ethernet-Adresse der Steuerung).

RIL_CommonTypes.library

3.3.6 KIN_OBJECTS

Kurzbeschreibung Die Aufzählung der verfügbaren Kinematiken (Achsgruppen) enthält Objekte eines IndraMotion-Verbundes, auf die durch Coordinated Motion-Bausteine zugegriffen werden kann, siehe auch ([AXIS_GROUP_REF](#), Seite 39).

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of logical addresses

Tab. 3-7: Bibliothekszuordnung
Typdefinition

```

TYPE KIN_OBJECTS :
(
(* not specified object                                *)
NO_KINEMATICS := 0,                                (* no object specified *)
(* axis groups 1..16 *)
KINEMATICS_1 := 01, (* synchronize axis group 1 *)
KINEMATICS_2 := 02, (* synchronize axis group 2 *)
KINEMATICS_3 := 03, (* synchronize axis group 3 *)
KINEMATICS_4 := 04, (* synchronize axis group 4 *)
KINEMATICS_5 := 05, (* synchronize axis group 5 *)
KINEMATICS_6 := 06, (* synchronize axis group 6 *)
KINEMATICS_7 := 07, (* synchronize axis group 7 *)
KINEMATICS_8 := 08, (* synchronize axis group 8 *)
KINEMATICS_9 := 09, (* synchronize axis group 9 *)
KINEMATICS_10 := 10, (* synchronize axis group 10 *)
KINEMATICS_11 := 11, (* synchronize axis group 11 *)
KINEMATICS_12 := 12, (* synchronize axis group 12 *)
KINEMATICS_13 := 13, (* synchronize axis group 13 *)
KINEMATICS_14 := 14, (* synchronize axis group 14 *)
KINEMATICS_15 := 15, (* synchronize axis group 15 *)
KINEMATICS_16 := 16, (* synchronize axis group 16 *)
);
END_TYPE

```



Die Vergabe des Namens und der Nummern erfolgt beim Anlegen des Objektes in der IndraWorks-Suite (IndraLogic, GlobalMlcVariablen).



Die Anzahl der realisierbaren Kinematiken hängt vom Steuerungstyp und der verwendeten Firmwareversion ab.

3.3.7 TOUCHPROBE_REF

Kurzbeschreibung Die Struktur enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt (Messtaster).

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of PLCopen

Tab. 3-8: Bibliothekszuordnung
Typdefinition

```

(* Structure of PLCopen element TOUCHPROBE_REF. The TOUCHPROBE_REF is a structure that *)
(* contains information on the corresponding object. *)

TYPE TOUCHPROBE_REF :
STRUCT
  CntrlNo:      CONTROLS    := LOCAL_CNTRL; (* control reference number *)
              (* Steuerungsnummer, Init: lokale Steuerung *)
  TouchProbeNo: TP_OBJECTS := NO_TOUCHPROBE (* object reference number *)
              (* Messtastereferenznummer, Init: kein Objekt *)
END STRUCT
END_TYPE

```

- Funktionsbeschreibung** Die Informationsübergabe erfolgt per VAR_IN_OUT an den jeweiligen PLCopen-Funktionsbaustein.
- Die TOUCHPROBE_REF-Struktur enthält zwei 16-Bit-Adresselemente:
- die Steuerungsadresse (Element von [CONTROLS](#), [Seite 37](#)) und
 - die Objektadresse (Element von [TP_OBJECTS](#), [Seite 41](#)).

3.3.8 TP_OBJECTS

- Kurzbeschreibung** Die Aufzählung der verfügbaren Messtaster (Touch Probe) enthält Objekte eines IndraMotion-Verbundes, auf die durch Messtaster-Bausteine zugegriffen werden kann, siehe auch ([TOUCHPROBE_REF](#), [Seite 40](#)).

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of logical addresses

Tab. 3-9: Bibliothekszuordnung
Typdefinition

```
(* Enumeration of valid logical object addresses *)
TYPE TP_OBJECTS :
(
  (* not specified object *)
  NO_TOUCHPROBE := 0,      (* no object specified *)

  (* Touch Probe 1..100, realized on controller *)
  TOUCHPROBE_1   := 01,    (* Touch Probe 1 *)
  (* ... *)
  TOUCHPROBE_100 := 100    (* Touch Probe 100 *)

  (* Touch Probe 1001..1002, realized on drive *)
  TOUCHPROBE_DRIVE_1 := 1001 (* Touch Probe Drive 1 *)
  TOUCHPROBE_DRIVE_2 := 1002 (* Touch Probe Drive 2 *)
);
END_TYPE
```



Die Vergabe des Namens und der Nummern erfolgt beim manuellen Anlegen des jeweiligen Messtasters in der IndraWorks-Suite (IndraLogic, GlobaleMlcVariablen).



Die Anzahl der realisierbaren Messtaster hängt vom Steuerungstyp und der verwendeten Firmwareversion ab.

3.4 Datentypen zur Unterstützung von Kurvenscheiben, Motion- und FlexProfilen

3.4.1 Datentyp zum Umschaltung zwischen Kurvenscheiben, allgemeines

In Abhängigkeit von der Leistungsfähigkeit der Steuerelektronik können folgende Kurventabellen hinterlegt werden:

- MPDx - 4 Tabellen á 1024 Punkte (Tabelle 1...4),
- MPBx - 4 Tabellen á 1024 Punkte (Tabelle 1...4),
- MPHx - 4 Tabellen á 1024 Punkte (Tabelle 1...4), 4 Tabellen á 128 Punkte (Tabelle 5...8).

Im laufenden Betrieb kann zwischen den Kurventabellen umgeschaltet werden. Auswahl und Umschaltung können über den Funktionsbaustein MC_CamIn (CamTableID: [MC_CAM_ID](#), [Seite 42](#)) oder über Parametersteuerung "A-0-2740, ELS, CamIn, Kurventabellenvorwahl" erfolgen.

RIL_CommonTypes.library

3.4.2 MC_CAM_ID

Kurzbeschreibung Der Aufzählungstyp unterstützt die Auswahl der gewünschten Kurventabelle ([Datentypen zur Unterstützung von Kurvenscheiben, Motion- und FlexProfilen, Seite 41](#)).

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of PLCopen

Tab. 3-10: Bibliothekszuordnung

Programm:

```

TYPE MC_CAM_ID :
(
  CAM_TABLE_1   := 1,   (* Kurventabelle 1   *)
  CAM_TABLE_2   := 2,   (* Kurventabelle 2   *)
  (* ...         *)
  CAM_TABLE_255 := 255  (* Kurventabelle 255 *)
);
END_TYPE

```

Funktionsbeschreibung Im laufenden Betrieb kann zwischen den Kurventabellen umgeschaltet werden. Auswahl und Umschaltung können über den Funktionsbaustein MC_CamIn, ML_PLCopen.library, oder über Parametersteuerung "A-0-2740, ELS, CamIn, Kurventabellenvorwahl", erfolgen.

3.4.3 MC_STEP_MODE, MotionProfile erweitert zu FlexProfile

Kurzbeschreibung Der Aufzählungsdatentyp MC_STEP_MODE beschreibt die Abhängigkeit der Folgeachseposition von der Leitachse während eines Bewegungsschrittes (MotionProfile erweitert zu FlexProfile).

Die Abhängigkeit kann gegeben sein als funktionale Abhängigkeit oder in Form einer Kurventabelle.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of PLCopen

Tab. 3-11: Bibliothekszuordnung

Typdefinition

```

TYPE MC_STEP_MODE :
(
  REST_IN_REST_INCLINEDSINE := 16#0000, (* MotionProfile, FlexProfile *)
  REST_IN_REST_POLY5        := 16#0100, (* MotionProfile, FlexProfile *)
  REST_IN_VELOCITY_POLY5    := 16#0200, (* MotionProfile, FlexProfile *)
  REST_IN_VELOCITY_POLY7    := 16#0300, (* FlexProfile *)
  VELOCITY_IN_REST_POLY5    := 16#0400, (* MotionProfile, FlexProfile *)
  VELOCITY_IN_REST_POLY7    := 16#0500, (* FlexProfile *)
  CONSTANT_VELOCITY         := 16#0600, (* MotionProfile, FlexProfile *)
  VELOCITY_IN_VELOCITY_POLY5 := 16#0700, (* MotionProfile, FlexProfile *)

  REST_IN_REST_POLY5_VLIM   := 16#1200, (* FlexProfile *)
  REST_IN_REST_PARABOLA    := 16#1300, (* FlexProfile *)
  REST_IN_REST_POLY8       := 16#1400, (* FlexProfile *)

  REST_IN_REST_LINEAR      := 16#0800, (* FlexProfile *)
  REST_IN_REST_POLY7       := 16#0900, (* FlexProfile *)
  REST_IN_REST_SINE        := 16#0A00, (* FlexProfile *)
  REST_IN_REST_GUTMANSINE  := 16#0B00, (* FlexProfile *)
  REST_IN_REST_SINEACC     := 16#0C00, (* FlexProfile *)
  REST_IN_REST_SINETORQUE  := 16#0D00, (* FlexProfile *)
  REST_IN_REST_MOD_TRAPEZE := 16#0E00, (* FlexProfile *)
  REST_IN_REST_MOD_SINE    := 16#0F00, (* FlexProfile *)

  VELOCITY_IN_VELOCITY_POLY7 := 16#1000, (* FlexProfile *)
  VELOCITY_IN_VELOCITY_MOD_SINE := 16#1100, (* FlexProfile *)

  MOTION_IN_MOTION_POLY5   := 16#2000, (* FlexProfile, *)
  (* auch Flex-Bewegungsschritt *)
  MOTION_IN_MOTION_POLY7   := 16#2100, (* FlexProfile, *)

```

```

COMMON_POLY5                := 16#4000, (* auch Flex-Bewegungsschritt *)
COMMON_POLY7                := 16#4100, (* FlexProfile *)
COMMON_POLY2                := 16#4200, (* FlexProfile *)
COMMON_POLY3                := 16#4300, (* FlexProfile *)
COMMON_POLY4                := 16#4400, (* FlexProfile *)
COMMON_POLY8                := 16#4500, (* FlexProfile *)

(* Erweiterte Bewegungsgesetze *)
X_VELOCITY_IN_VELOCITY_TRAPEZE_ALIM := 16#5000, (* FlexProfile *)
X_MOTION_IN_MOTION_POLY5_VLIM      := 16#6000, (* FlexProfile *)
X_MOTION_IN_MOTION_POLY5_SLIM      := 16#6100, (* FlexProfile, *)
                                   (* Auch Flex-Bewegungsschritt *)
X_FIT_VEL_TRAPEZE_ALIM             := 16#7000, (* FlexProfile *)
X_FIT_SINE_TRAPEZE_ALIM            := 16#7100, (* FlexProfile *)
X_FIT_ACC_TRAPEZE_JLIM             := 16#7200, (* FlexProfile *)

X_MOTION_IN_MOTION_ACAM            := 16#8000, (* FlexProfile, *)
X_MOTION_IN_MOTION_VCAM            := 16#8100, (* FlexProfile, *)
X_MOTION_IN_MOTION_VCAM2_A         := 16#8200, (* FlexProfile, *)
X_MOTION_IN_MOTION_VCAM2_B         := 16#8300, (* FlexProfile, *)

(* Kurventabellen *)
CAMTABLE_1                     := 1,          (* Kurventabelle 1, C-0-2001 *)
CAMTABLE_2                     := 2,          (* Kurventabelle 2, C-0-2002 *)
CAMTABLE_3                     := 3,          (* Kurventabelle 3, C-0-2003 *)
(* ..... *)
CAMTABLE_98                    := 98,         (* Kurventabelle 98, C-0-2098 *)
CAMTABLE_99                    := 99,         (* Kurventabelle 99, C-0-2099 *)
);
END_TYPE

```

Funktionsbeschreibung



Nachstehende Funktionsbeschreibung enthält die Details für Motion- und FlexProfile.

Als Modus eines Schrittes können verschiedene Normprofile oder eine Kurventabelle definiert werden.

Als Modi stehen zur Verfügung:

- R-R: Rast in Rast, Polynom 5. Ordnung,
- R-R: Rast in Rast, geneigte Sinuslinie,
- R-G: Rast in Geschwindigkeit, Polynom 5. Ordnung,
- G-R: Geschwindigkeit in Rast, Polynom 5. Ordnung,
- G: konstante Geschwindigkeit,
- G-G: Geschwindigkeit in Geschwindigkeit, Polynom 5. Ordnung,
- Kurvenscheibentabelle.

R-R: Rast in Rast

Für das Profil "Rast in Rast" stehen zehn Profile zur Verfügung. Es kann zwischen den normierten Bewegungsgesetzen "Polynom 5. Ordnung" und "Geneigte Sinuslinie" ... gewählt werden.

Diese Profile sind durch den Hub und Leitachsbereich definiert. Die Endgeschwindigkeit des Schrittes wird vom Folgeschritt bestimmt.

Das Profil für die "**Polynom 5. Ordnung**" ist durch nachstehende Funktion bestimmt:

$$f(z) = 6z^5 - 15z^4 + 10z^3$$

Abb. 3-1: Rast in Rast, Polynom 5. Ordnung

Die möglichen Randwerte für das Bewegungsgesetz sind:

RIL_CommonTypes.library

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-12: Randwerte für das Bewegungsgesetz - Polynom 5. Ordnung

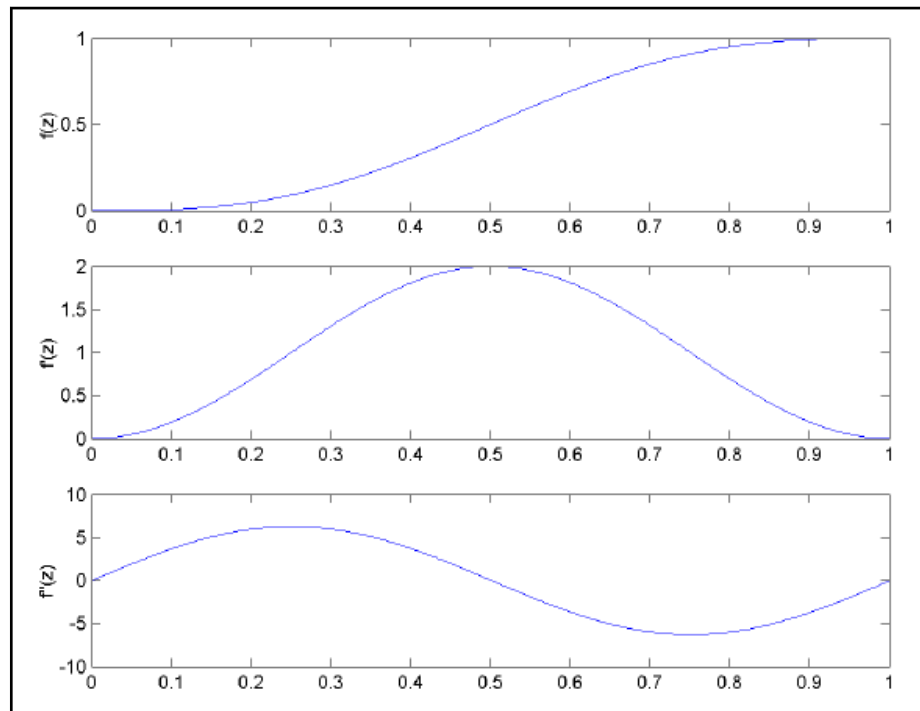


Abb. 3-2: Rast in Rast, Polynom 5. Ordnung

Für das "Polynom 7. Ordnung" gilt die Formel :

$$f(z) = -20z^7 + 70z^6 - 84z^5 - 35z^4$$

Abb. 3-3: Rast in Rast, Polynom 7. Ordnung

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	0	0

Tab. 3-13: Randwerte für das Bewegungsgesetz - Polynom 7. Ordnung

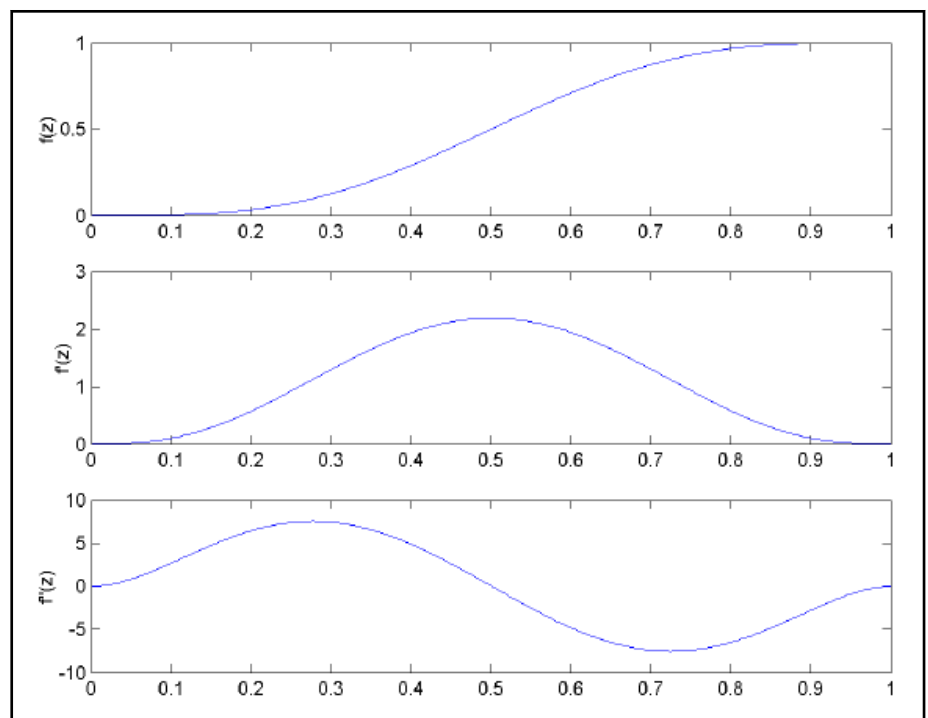


Abb. 3-4: Rast in Rast, Polynom 7. Ordnung

Das Profil für die **"Geneigte Sinuslinie"** ist durch nachstehende Funktion bestimmt:

$$f(z) = z - \frac{1}{2\pi} \sin(2\pi z)$$

Abb. 3-5: Rast in Rast, Geneigte Sinuslinie

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-14: Randwerte für Bewegungsgesetz - geneigte Sinuslinie

RIL_CommonTypes.library

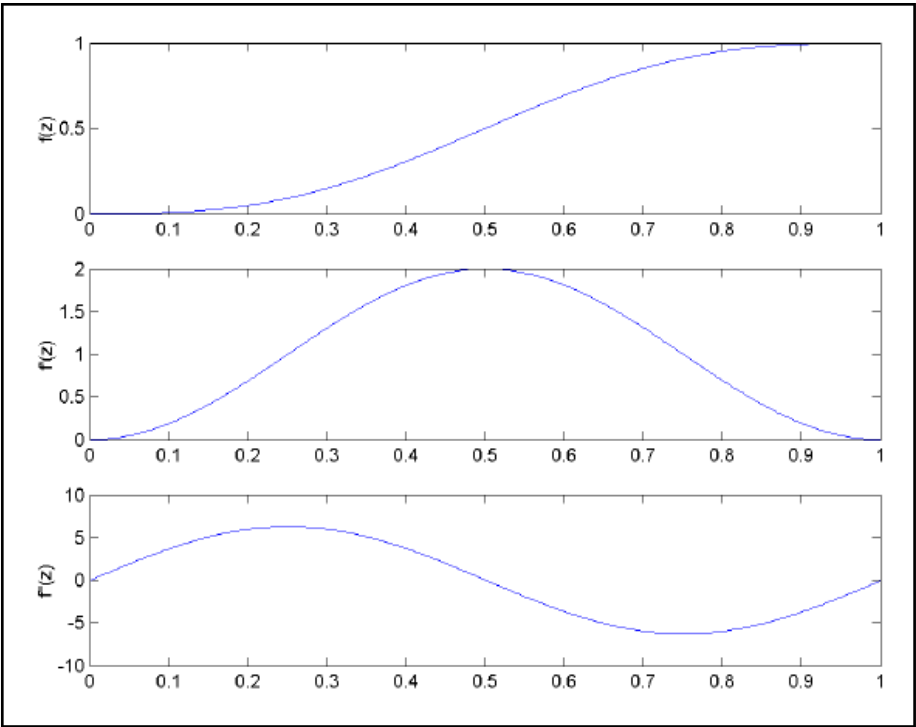


Abb. 3-6: Rast in Rast, Geneigte Sinuslinie (BESTEHORN)
Das Profil für die "Sinuslinie (einfache Sinoide)" ist durch nachstehende Funktion bestimmt:

$$f(z) = \frac{1}{2} [1 - \cos(\pi z)]$$

Abb. 3-7: Rast in Rast, Sinuslinie
Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	automatisch	automatisch
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-15: Randwerte für das Bewegungsgesetz Sinuslinie

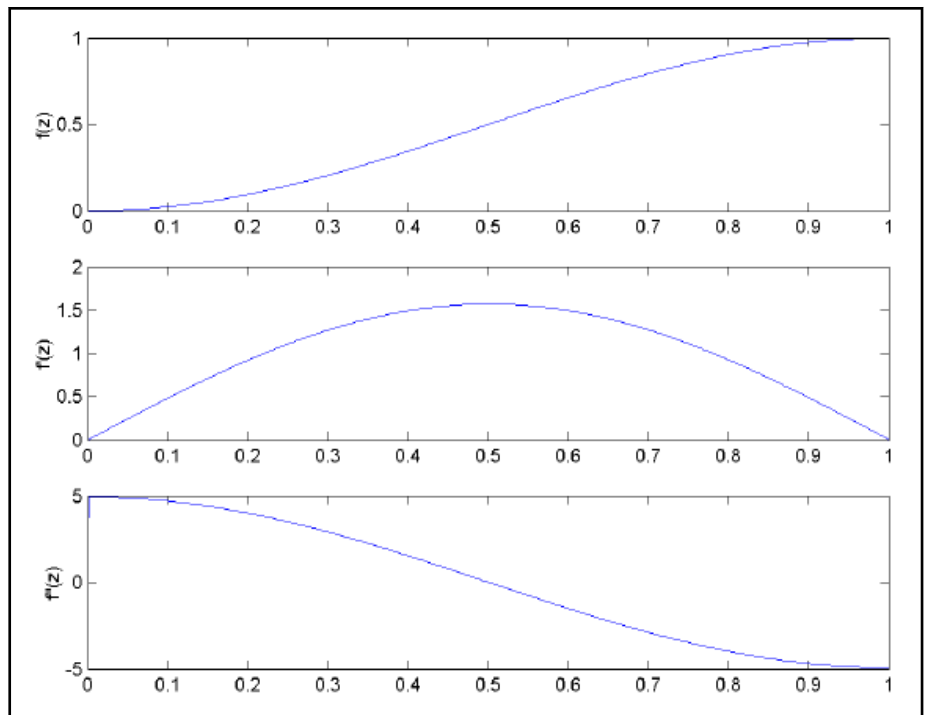


Abb. 3-8: Rast in Rast, Sinuslinie (einfache Sinoide)

Das Profil für die **"Lineare Rast"** ist durch nachstehende Funktion bestimmt:

$$f(z) = 0$$

Abb. 3-9: Lineare Rast

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	0	0

Tab. 3-16: Randwerte für Bewegungsgesetze - Lineare Rast

RIL_CommonTypes.library

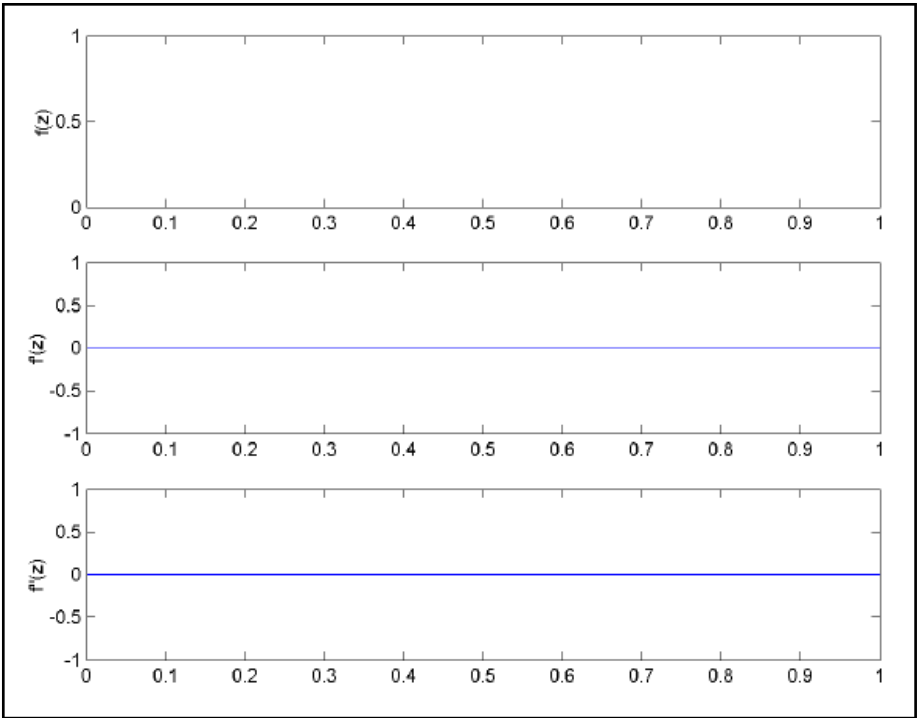


Abb. 3-10: Lineare Rast

Das Profil für die **"Sinoide von Gutman"** ist durch nachstehende Funktion bestimmt:

$$f(z) = z - \frac{15}{32\pi} \sin(2\pi z) - \frac{1}{96\pi} \sin(6\pi z)$$

Abb. 3-11: Sinoide nach Gutman

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-17: Randwerte für das Bewegungsgesetz - Sinoide von Gutman

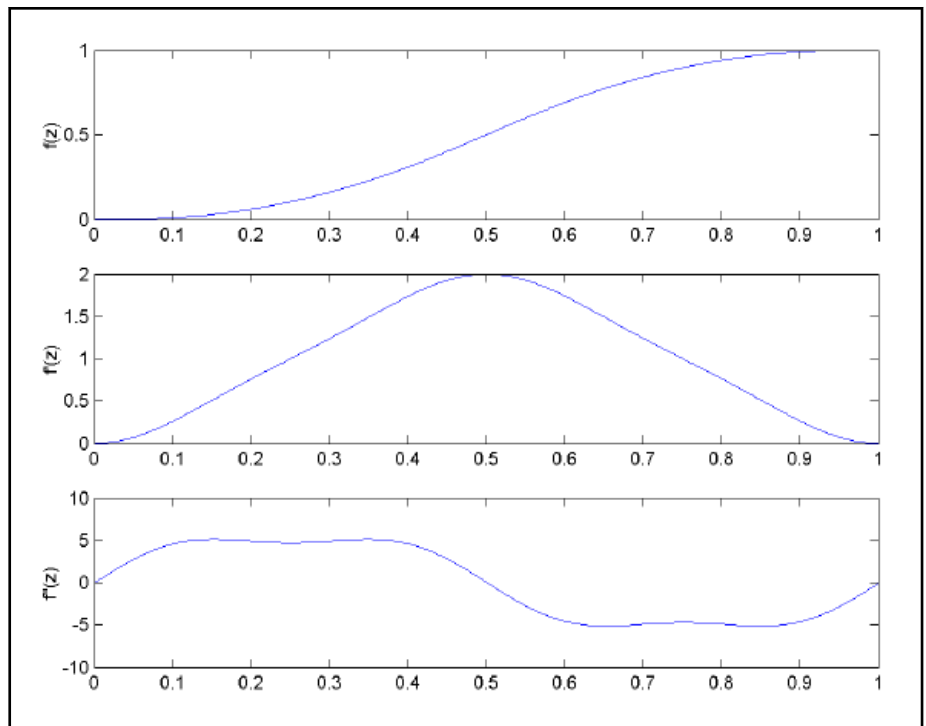


Abb. 3-12: Rast in Rast, Sinoide von Gutman

Das Profil für die **"Beschleunigungsoptimale geneigte Sinuslinie"** ist durch nachstehende Funktion bestimmt:

$$f(z) = z - \frac{1}{2\pi} \sin(2\pi z)$$

$$g(z) = z - \frac{0.41}{2\pi} \sin(2\pi z)$$

Abb. 3-13: Beschleunigungsoptimale geneigte Sinuslinie

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-18: Randwerte für das Bewegungsgesetz - Beschleunigungsoptimale geneigte Sinuslinie

RIL_CommonTypes.library

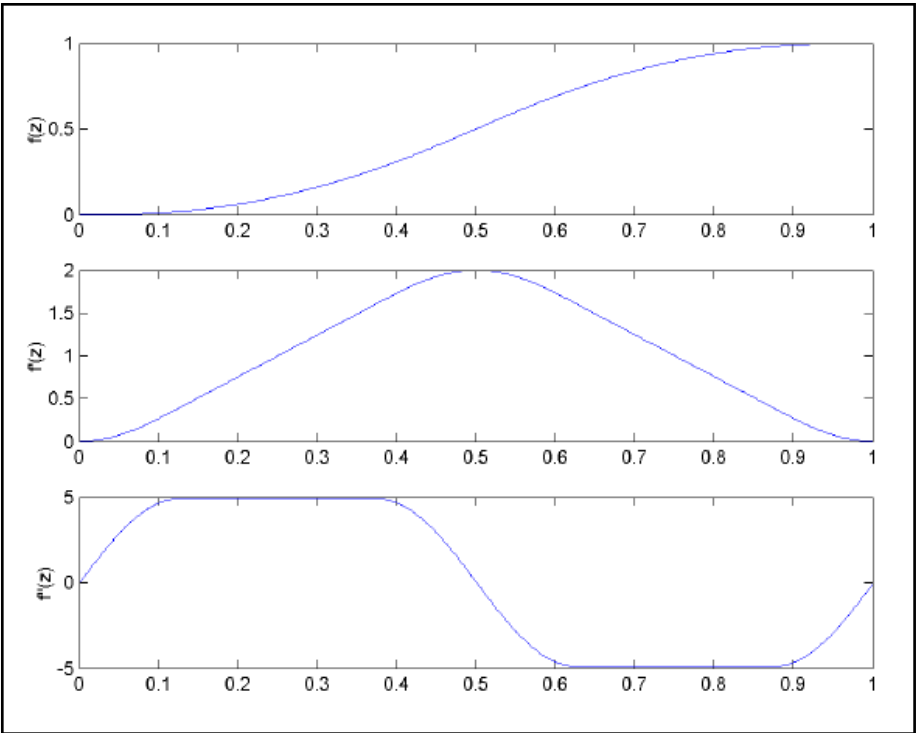


Abb. 3-14: *Rast in Rast, Beschleunigungsoptimale geneigte Sinuslinie*
Das Profil für die **"Momentenoptimale geneigte Sinuslinie"** ist durch nachstehende Funktion bestimmt:

$$f(z) = z - \frac{1}{2\pi} \sin(2\pi z)$$
$$g(z) = z - \frac{0.134}{2\pi} \sin(2\pi z)$$

Abb. 3-15: *Momentenoptimale geneigte Sinuslinie*
Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-19: *Randwerte für das Bewegungsgesetz - Momentenoptimale geneigte Sinuslinie*

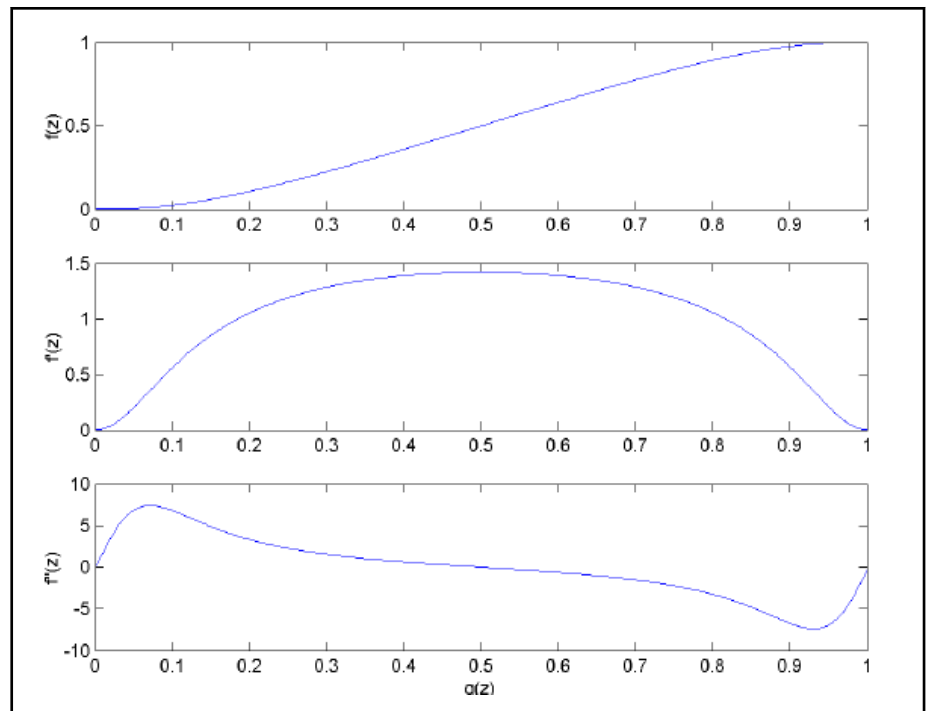


Abb. 3-16: Rast in Rast, Momentenoptimale geneigte Sinuslinie

Das Profil für die **"Modifizierte Sinoide"** ist durch nachstehende Funktion bestimmt:

$$\begin{aligned}
 0 \leq z \leq \frac{1}{8}: f(z) &= \frac{1}{4+\pi} \left[\pi z - \frac{1}{4} \sin(4\pi z) \right] \\
 \frac{1}{8} \leq z \leq \frac{7}{8}: f(z) &= \frac{1}{4+\pi} \left[2 + \pi z - \frac{4}{9} \sin\left(\frac{4\pi}{3}z + \frac{\pi}{3}\right) \right] \\
 \frac{7}{8} \leq z \leq 1: f(z) &= \frac{1}{4+\pi} \left[4 + \pi z - \frac{1}{4} \sin(4\pi z) \right]
 \end{aligned}$$

Abb. 3-17: Modifizierte Sinoide

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	$z=0$	$z=1$
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-20: Randwerte für das Bewegungsgesetz - Modifizierte Sinoide

RIL_CommonTypes.library

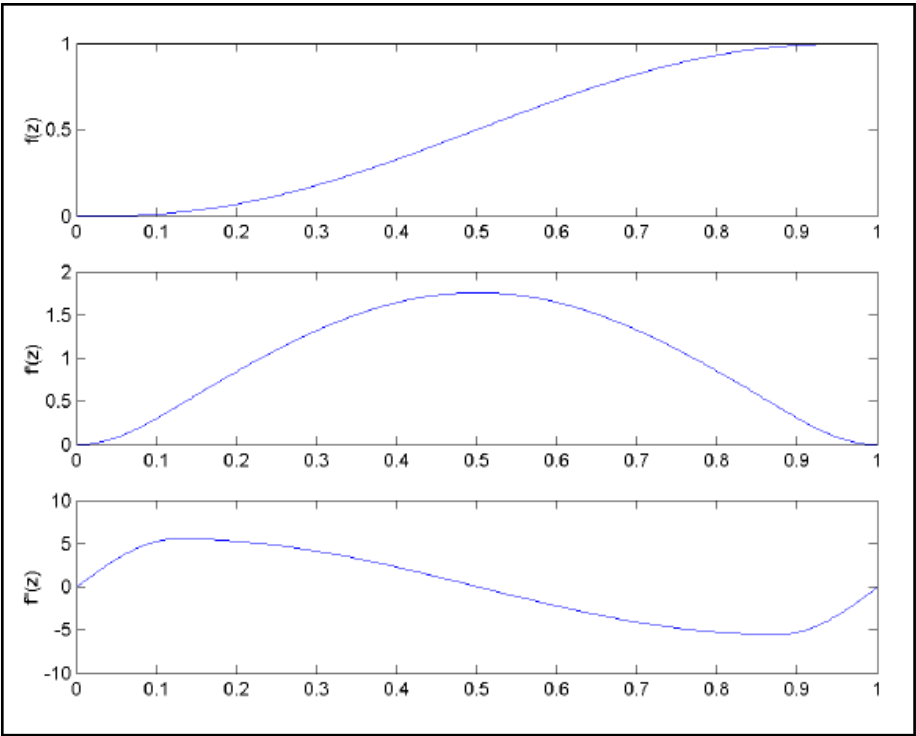


Abb. 3-18: Rast in Rast, Modifizierte Sinoide

Das Profil für die **"Modifiziertes Beschleunigungstrapez"** ist durch nachstehende Funktion bestimmt:

$$0 \leq z \leq \frac{1}{8} : f(z) = \frac{1}{2 + \pi} \left[2z - \frac{1}{2\pi} \sin(4\pi z) \right]$$
$$\frac{1}{8} \leq z \leq \frac{3}{8} : f(z) = \frac{1}{2 + \pi} \left[-\frac{1}{2\pi} + 2z + 4\pi \left(z + \frac{1}{8} \right)^2 \right]$$
$$\frac{3}{8} \leq z \leq \frac{1}{2} : f(z) = \frac{1}{2 + \pi} \left[-\frac{\pi}{2} + 2(1 + \pi)z - \frac{1}{2\pi} \sin(4\pi z - \pi) \right]$$
$$\frac{1}{2} \leq z \leq 1 : f(z) = 1 - f(1 - z)$$

Abb. 3-19: Modifiziertes Beschleunigungstrapez

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	0
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-21: Randwerte für das Bewegungsgesetz - Modifiziertes Beschleunigungstrapez

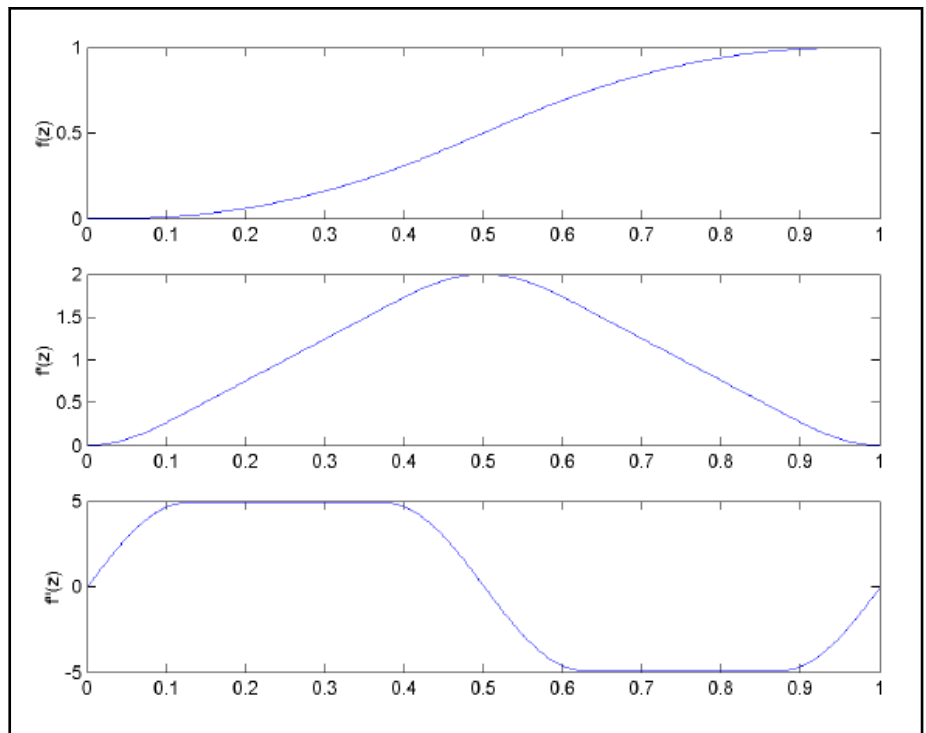


Abb. 3-20: Rast in Rast, Modifiziertes Beschleunigungstrapez

R-G: Rast in Geschwindigkeit

Dieses Profil dient zum Übergang aus dem Stillstand in eine bestimmte Geschwindigkeit.

Das Profil wird mit einem "Polynom 5. Ordnung" berechnet :

$$f(z) = A_5 z^5 + A_4 z^4 + A_3 z^3$$

Abb. 3-21: Polynom 5. Ordnung

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	wählbar
2. Ordnung (Beschleunigung)	0	wählbar
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-22: Randwerte für das Bewegungsgesetz Polynom 5. Ordnung

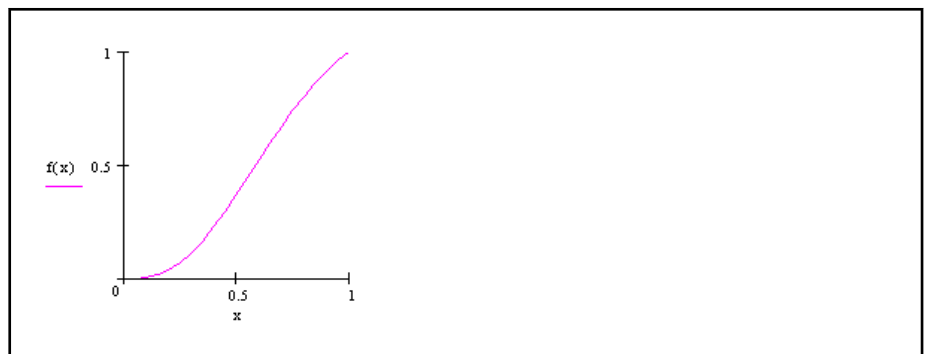


Abb. 3-22: Rast in Geschwindigkeit, Polynom 5. Ordnung

Dieses Profil dient zum Übergang aus dem Stillstand in eine bestimmte Geschwindigkeit.

RIL_CommonTypes.library

Das Profil wird mit einem **"Polynom 7. Ordnung"** berechnet :

$$f(z) = A_7z^7 + A_6z^6 + A_5z^5 + A_4z^4 + A_3z^3$$

Abb. 3-23: Polynom 7. Ordnung

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	0	wählbar
2. Ordnung (Beschleunigung)	0	wählbar
3. Ordnung (Ruck)	automatisch	wählbar

Tab. 3-23: Randwerte für das Bewegungsgesetz - Polynom 7. Ordnung

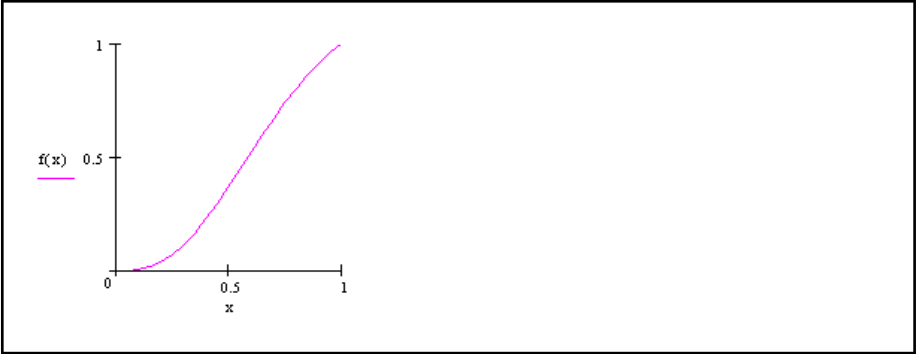


Abb. 3-24: Rast in Geschwindigkeit, Polynom 7 Ordnung

G-R: Geschwindigkeit in Rast

Das Profil wird zum Übergang aus einer definierten Geschwindigkeit in den Stillstand verwendet.

Bei diesem Bewegungsprofil wird ein Weg verfahren, der durch den Hub für diesen Bewegungsschritt festgelegt ist.

Das Profil wird mit einem **"Polynom 5. Ordnung"**berechnet :

$$f(z) = A_5z^5 + A_4z^4 + A_3z^3 + A_1z$$

Abb. 3-25: Geschwindigkeit in Rast, Polynom 5. Ordnung

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	wählbar	0
2. Ordnung (Beschleunigung)	wählbar	0
3. Ordnung (Ruck)	automatisch	0

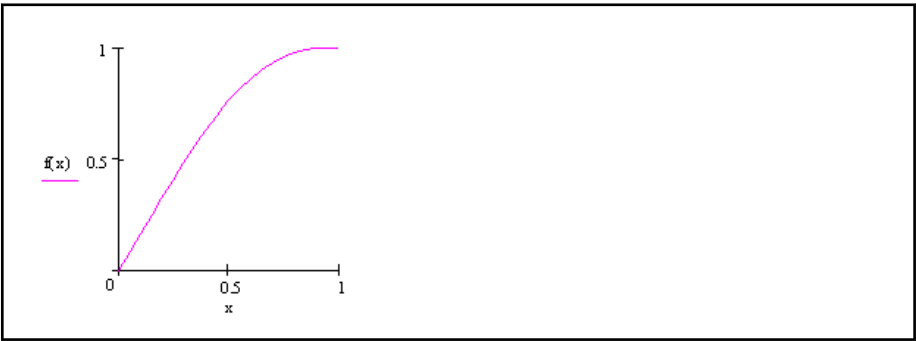


Abb. 3-26: Geschwindigkeit in Rast, Polynom 5. Ordnung

Das Profil wird zum Übergang aus einer definierten Geschwindigkeit in den Stillstand verwendet.

Das Profil wird mit einem **"Polynom 7. Ordnung"** berechnet :

$$f(z) = A_7 z^7 + A_6 z^6 + A_5 z^5 + A_4 z^4 + A_3 z^3 + A_2 z^2 + A_1 z$$

Abb. 3-27: Rast in Geschwindigkeit, Polynom 7. Ordnung

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	wählbar	0
2. Ordnung (Beschleunigung)	wählbar	0
3. Ordnung (Ruck)	wählbar	0

Tab. 3-24: Randwerte für das Bewegungsgesetz - Polynom 7. Ordnung

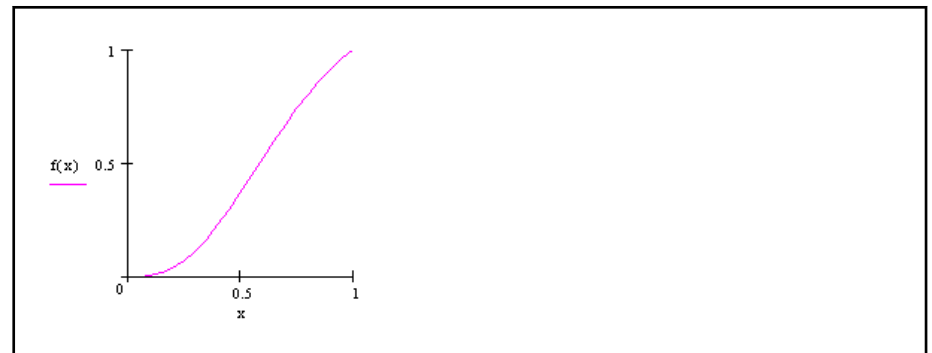


Abb. 3-28: Rast in Geschwindigkeit, Polynom 7. Ordnung

G: konstante Geschwindigkeit

Das Profil dieses Bewegungsschrittes ist eine **"Gerade"**. Die Geschwindigkeit innerhalb dieses Schrittes ist konstant. Es wird ein Weg verfahren, der durch den Hub für diesen Bewegungsschritt festgelegt ist.

Die Formel für das Profil lautet in normierter Form:

$$f(z) = z.$$

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	automatisch	automatisch
2. Ordnung (Beschleunigung)	automatisch	automatisch
3. Ordnung (Ruck)	automatisch	automatisch

RIL_CommonTypes.library

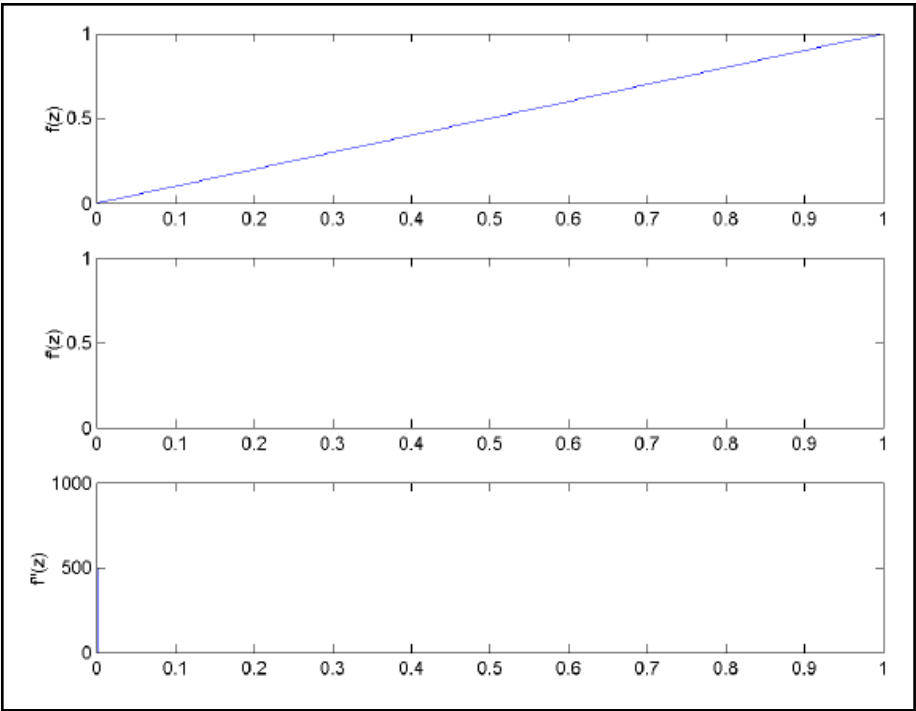


Abb. 3-29: Konstante Geschwindigkeit

Die Folgeachs-Geschwindigkeit wird aus den für diesen Bewegungsschritt vorgegebenen Werten bestimmt:

- Leitachsbereich
- Hub
- Leitachsgeschwindigkeit.

Die Eingabe eines Wertes für die Folgeachs-Geschwindigkeit ist nicht erforderlich.

G-G: Geschwindigkeit in Geschwindigkeit

Das Profil für die "Polynom 5. Ordnung" ist durch nachstehende Funktion bestimmt:

$$f(z) = A_5z^5 + A_4z^4 + A_3z^3 + A_2z^2 + A_1z$$

Abb. 3-30: Polynom 5. Ordnung

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	wählbar	wählbar
2. Ordnung (Beschleunigung)	wählbar	wählbar
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-25: Randwerte für das Bewegungsgesetz - Polynom 5. Ordnung

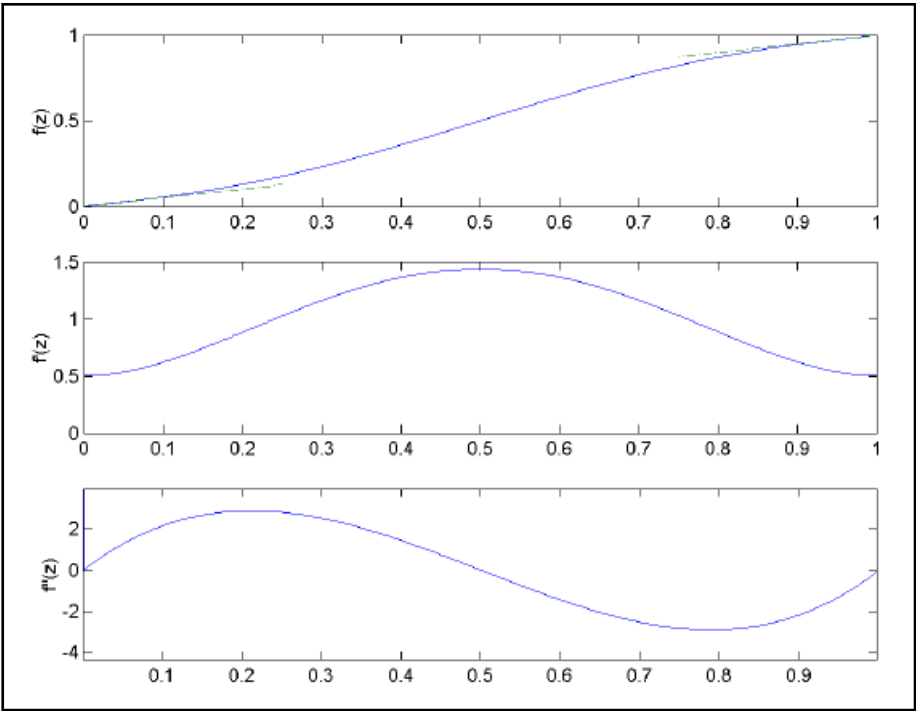


Abb. 3-31: Geschwindigkeit in Geschwindigkeit, Polynom 5. Ordnung
Das Profil für die "Polynom 7. Ordnung" ist durch nachstehende Funktion bestimmt:

$$f(z) = A_7 z^7 + A_6 z^6 + A_5 z^5 + A_4 z^4 + A_3 z^3 + A_2 z^2 + A_1 z$$

Abb. 3-32: Polynom 7. Ordnung
Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	wählbar	wählbar
2. Ordnung (Beschleunigung)	wählbar	wählbar
3. Ordnung (Ruck)	wählbar	wählbar

Tab. 3-26: Randwerte für das Bewegungsgesetz - Polynom 7. Ordnung

RIL_CommonTypes.library

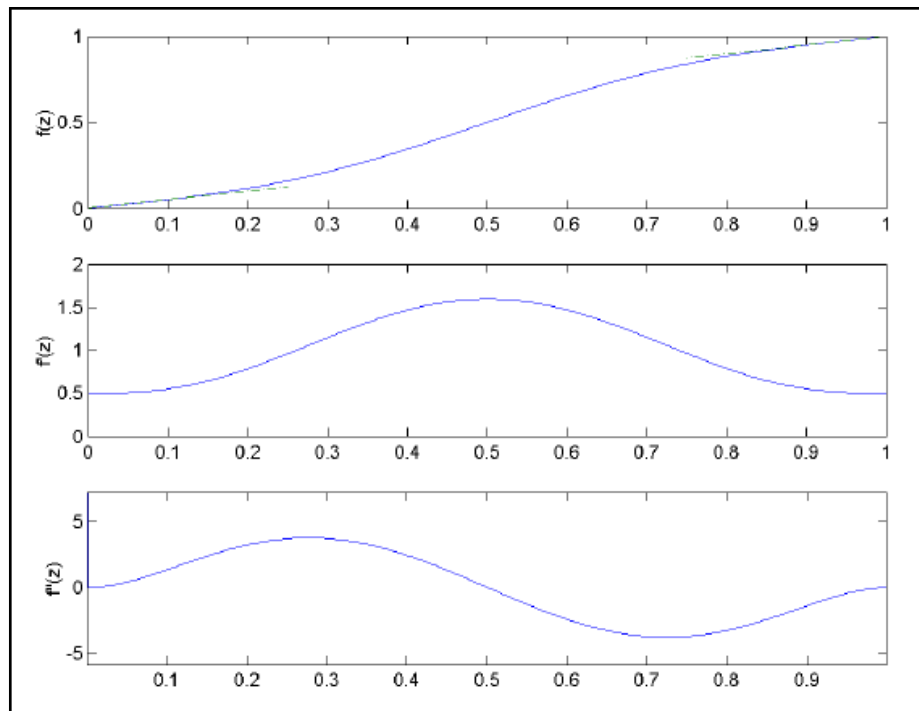


Abb. 3-33: Geschwindigkeit in Geschwindigkeit, Polynom 7. Ordnung

Das Profil für die **"Modifizierte Sinuslinie"** ist durch nachstehende Funktion bestimmt:

$$\begin{aligned}
 0 \leq z \leq \frac{\lambda}{8}: f'(z) &= C_a \frac{\lambda}{2\pi} \left(z - \frac{\lambda}{2\pi} \sin \left(2\pi \frac{z}{\lambda} \right) \right) + f'(0)z \\
 \frac{\lambda}{8} \leq z \leq \lambda: f'(z) &= C_a \frac{\lambda}{2\pi} \left(z - \frac{9\lambda}{2\pi} \sin \left(2\pi \frac{z + \frac{\lambda}{2}}{3\lambda} \right) + \frac{4\lambda}{\pi} \right) + f'(0)z \\
 \lambda \leq z \leq 1 - \frac{1-\lambda}{4}: f'(z) &= C_b \frac{1-\lambda}{2\pi} \left(z - \frac{9(1-\lambda)}{2\pi} \sin \left(2\pi \frac{z + \frac{3-5\lambda}{3(1-\lambda)}}{3\lambda} \right) + \frac{4\lambda}{\pi} \right) \\
 &\quad + f'(1) + (f'(0) - f'(1)) \left(\frac{3}{4} - \frac{1}{\pi} \right) + \lambda \\
 1 - \frac{1-\lambda}{4} \leq z \leq 1: f'(z) &= -C_b \frac{1-\lambda}{2\pi} \left(z - \frac{1-\lambda}{2\pi} \sin \left(2\pi \frac{z + \lambda}{1-\lambda} \right) - 1 \right) \\
 &\quad + 1 + f'(1) + (z - 1)
 \end{aligned}$$

Abb. 3-34: Modifizierte Sinuslinie

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	wählbar	wählbar
2. Ordnung (Beschleunigung)	0	0
3. Ordnung (Ruck)	0	0

Tab. 3-27: Randwerte für das Bewegungsgesetz - Modifizierte Sinuslinie

Erweiterte Bewegungsgesetze

Bei den erweiterten Bewegungsgesetzen handelt es sich um spezielle Bewegungsgesetze. Sie unterscheiden sich von den bekannten Bewegungsgesetzen aus den VDI Richtlinien durch neue Freiheitsgrade. Während ein Standard-Bewegungsgesetz durch die Angabe seiner linken und rechten Randwerte eindeutig definiert ist, erlauben die hier dargestellten Bewegungsgesetze noch weitere unterschiedliche Einstellungsmöglichkeiten, wie z.B. eine geforderte maximale Geschwindigkeit. In dem Fall ist der Leitachsabschnitt des Bewegungsschrittes nicht anzugeben. Stattdessen ergibt sich dieser aus der Parametrierung des Bewegungsgesetzes und der Festlegung von Kennwerten. So erlaubt beispielsweise die beschleunigungsbegrenzte Bewegung die Angabe von Hub, Anfangs- und Endgeschwindigkeit, nicht aber die Vorgabe eines Leitachsabschnitts. Dieser, und damit die Dauer des Bewegungsschrittes, ergibt sich aus den ebenfalls festlegbaren Werten einer maximalen Beschleunigung und einer maximalen Geschwindigkeit.

	Hub	Leitachsabschnitt	$v_{\max}$	$a_{\max}$	$j_{\max}$
Überschwingungsfreies Polynom 5. Ordnung	wählbar	wählbar	wählbar	automatisch	automatisch
Geschwindigkeitsbegrenztes Polynom 5. Ordnung	wählbar	wählbar	wählbar	automatisch	automatisch
Beschleunigungsbegrenzte Bewegung	wählbar	automatisch	wählbar	wählbar	automatisch
Beschleunigungsbegrenztes Sinoidenprofil	wählbar	automatisch	wählbar	wählbar	automatisch
Ruckbegrenzte Bewegung	wählbar	automatisch	wählbar	wählbar	wählbar

Tab. 3-28: Erweiterte Bewegungsgesetze

Das Profil für **"Geschwindigkeitsbegrenztes Polynom 5. Ordnung"** kann wie folgt werden:

Das aus der VDI bekannte Polynom 5. ermöglicht keine Kontrolle der maximalen Verfahrgeschwindigkeit. Zur Lösung dieses Problems wird das Bewegungsgesetz in drei Sektionen aufgeteilt. Die mittlere Sektion entspricht einer Geraden mit der geforderten Geschwindigkeit. Umgeben wird diese von zwei Polynomen 5. Grades. Diese sorgen für einen glatten Übergang von der Anfangs- bzw. Endgeschwindigkeit auf die Verfahrgeschwindigkeit. Zu ermitteln sind die Schnittpunkte zwischen der ersten und zweiten sowie der zweiten und dritten Sektion. Die Punkte werden so gewählt, dass sich für die beiden Polynome kein Wendepunkt einstellt

RIL_CommonTypes.library

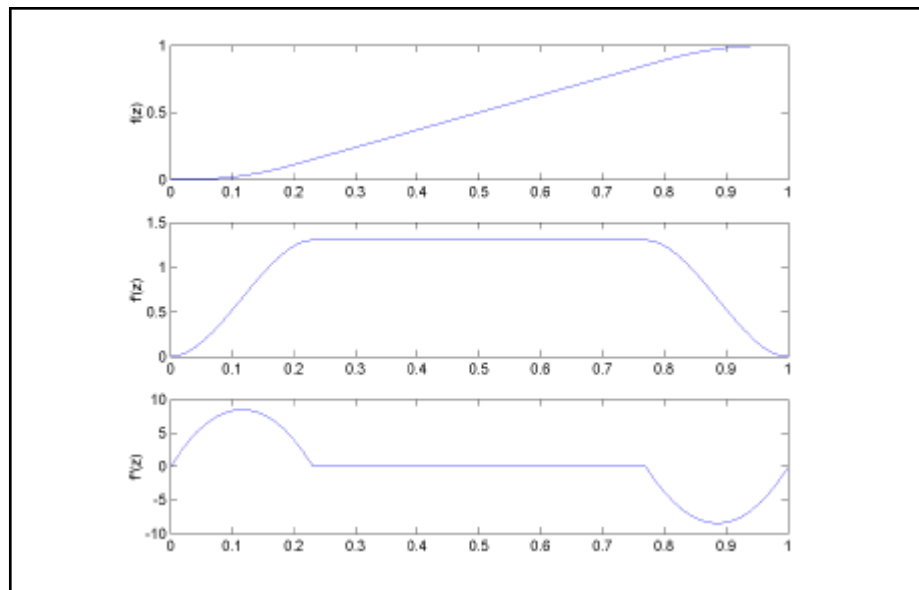


Abb. 3-35: Geschwindigkeitsbegrenztes Polynom 5. Ordnung

Das Profil für **"Überschwingungsfreies Polynom 5. Ordnung"** kann wie folgt beschrieben werden:

Das hier gezeigte Bewegungsgesetz erkennt und verhindert die mögliche Polynom-Überschwingung. Wie diese verändert wird, hängt dabei von der Kombination der normierten Geschwindigkeitsrandwerte ab. Es ergeben sich vier Fälle. Für Kombinationen in denen die normierte Anfangsgeschwindigkeit "v0" und die normierte Endgeschwindigkeit "v1" unterschiedliche Vorzeichen haben, kann dieses Bewegungsgesetz nicht verwendet werden. Dies würde bedeuten, dass die Bewegungsbahn in einem Randpunkt einen Knick hätte, wenn sie den normierten Bereich von $0 \leq z \leq 1$ und $0 \leq f(z) \leq 1$ nicht verlassen dürfte.

- Fall 1:
 $V_0 \leq 1$ und $V_1 \leq 1$

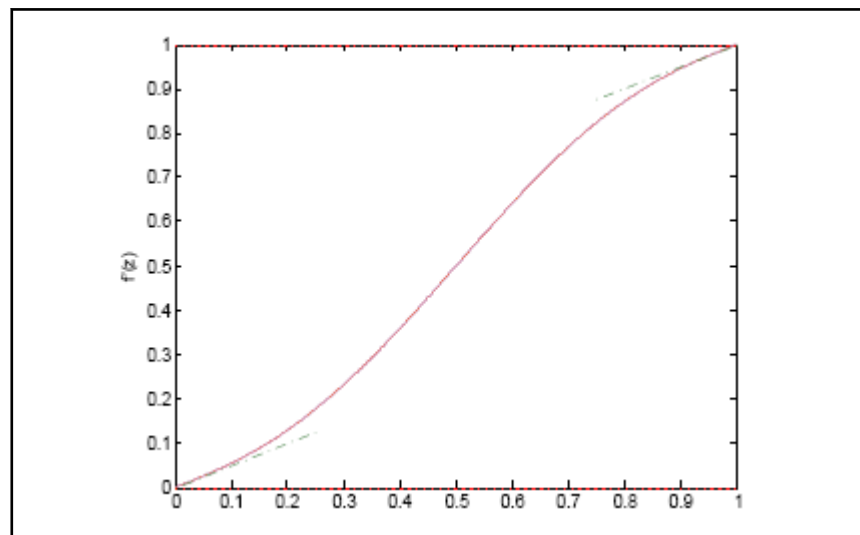


Abb. 3-36: Fall 1

- Fall 2:
 $V_0 \leq 1$ und $V_1 > 1$

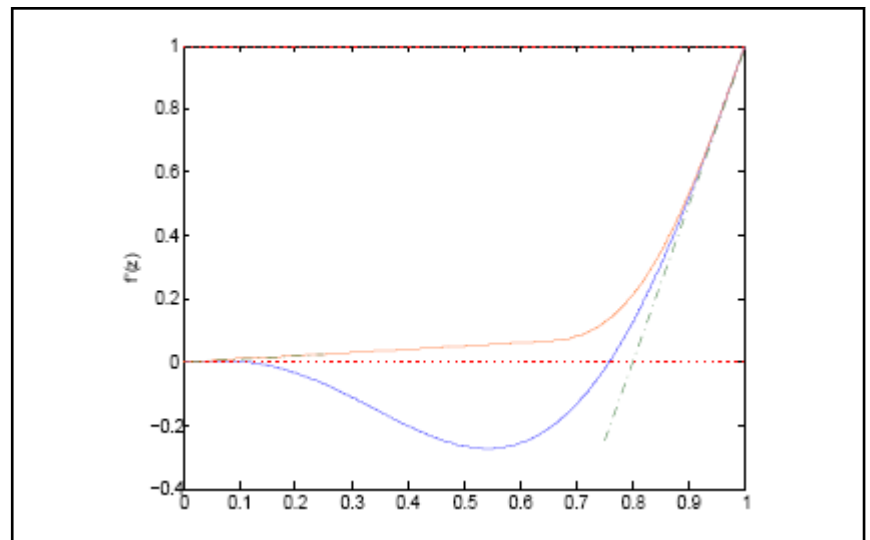


Abb. 3-37: Fall 2

- Fall 3:
 $V_0 > 1$ und $V_1 \leq 1$

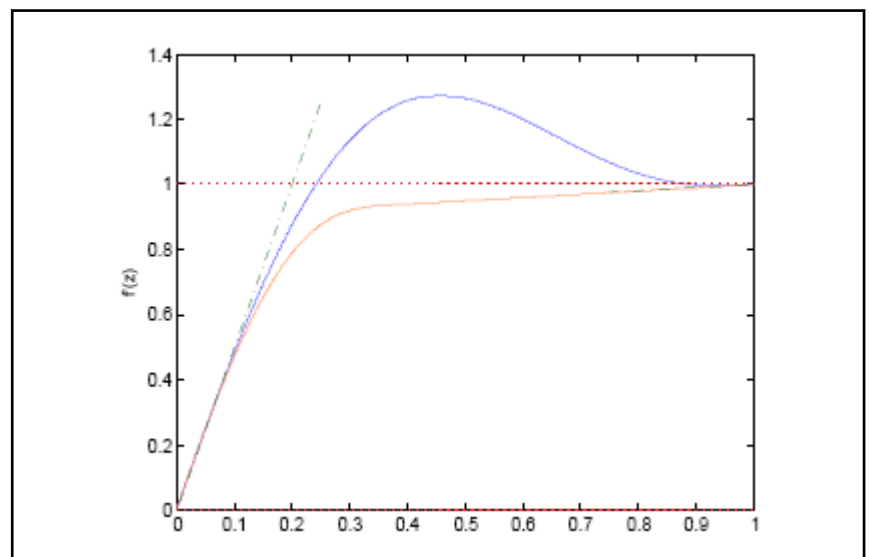


Abb. 3-38: Fall 3

- Fall 4:
 $V_0 > 1$ und $V_1 > 1$

RIL_CommonTypes.library

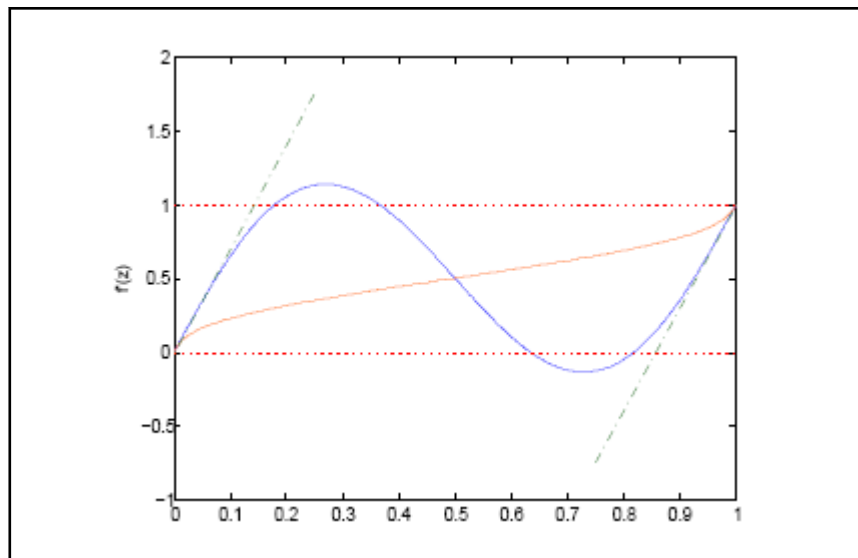


Abb. 3-39: Fall 4

Das Profil für die **"Beschleunigungsbegrenzte Bewegung"** kann wie folgt beschrieben werden:

Als einfachste Art der Bahninterpolation kann das Trapezprofil, oft auch Rampenprofil genannt, gesehen werden. Soll eine Wegstrecke "s" verfahren werden, so lässt sich hierbei der Bahnverlauf in drei Phasen aufteilen. In eine erste Phase, in der mit einem vorgegebenen Beschleunigungswert auf eine vorgegebene Fahrergeschwindigkeit beschleunigt wird, in eine zweite Phase konstanter Geschwindigkeit und schließlich in eine Bremsphase in der ebenfalls mit einer angegebenen Verzögerung wieder abgebremst wird. Aufgabe der Bahninterpolation ist es, aus diesen vorgegebenen Beschleunigungswerten a_1 und a_3 sowie der gewünschten Geschwindigkeit v_m und der zurückzulegenden Wegstrecke "s", die Start- und Endzeitpunkte t_1 , t_2 und t_3 bzw. die einzelnen Fahrzeiten T_1 , T_2 und T_3 zu berechnen. In der hier vorgestellten Variante wird außerdem auch eine geforderte Startgeschwindigkeit v_0 sowie Endgeschwindigkeit v_3 berücksichtigt.

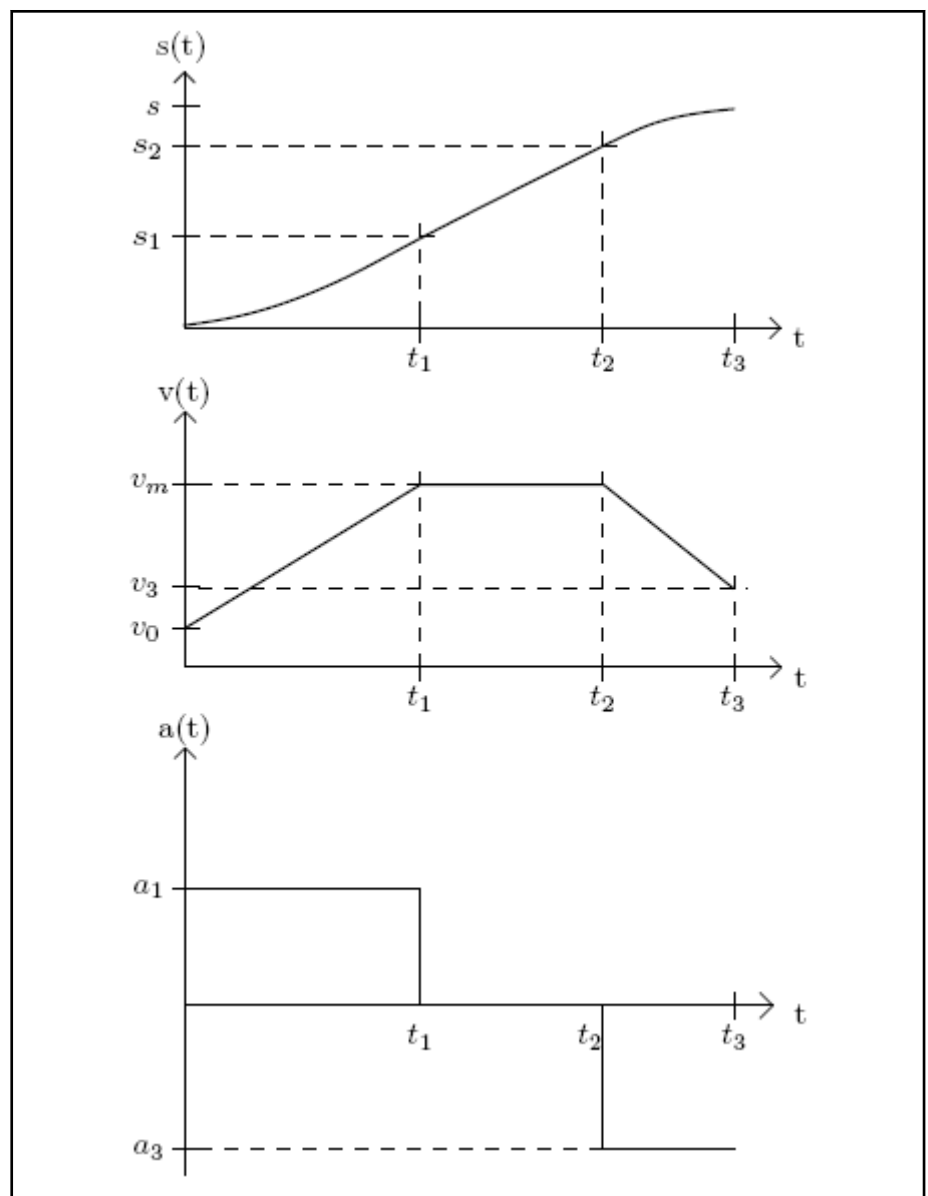


Abb. 3-40: Beschleunigungsbegrenzte Bewegung

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	wählbar	wählbar
2. Ordnung (Beschleunigung)	automatisch	automatisch
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-29: Randwerte für Bewegungsgesetz - Beschleunigungsbegrenzte Bewegung

Das Profil für die **"Beschleunigungsbegrenzte Bewegung mit Sinoidenprofil"** kann wie folgt beschrieben werden:

Beim Sinoidenprofil kann die Start- und Endgeschwindigkeiten ungleich 0 sein, sowie unterschiedliche Beschleunigungsgrößen a_1 und a_3 für die Beschleunigungs- und Bremsphase haben. Weiterhin werden auch die Sonderfälle berücksichtigt, in denen die Startgeschwindigkeit v_0 bereits größer als

RIL_CommonTypes.library

die gewünschte Verfahrensgeschwindigkeit v_m ist. Gleiches gilt auch für die Endgeschwindigkeit v_3 .

Die möglichen Randwerte für das Bewegungsgesetz sind:

Randwert	z=0	z=1
1. Ordnung (Geschwindigkeit)	wählbar	wählbar
2. Ordnung (Beschleunigung)	automatisch	automatisch
3. Ordnung (Ruck)	automatisch	automatisch

Tab. 3-30: Randwerte für Bewegungsgesetz - Beschleunigungsbegrenzte Bewegung mit Sinoidenprofil

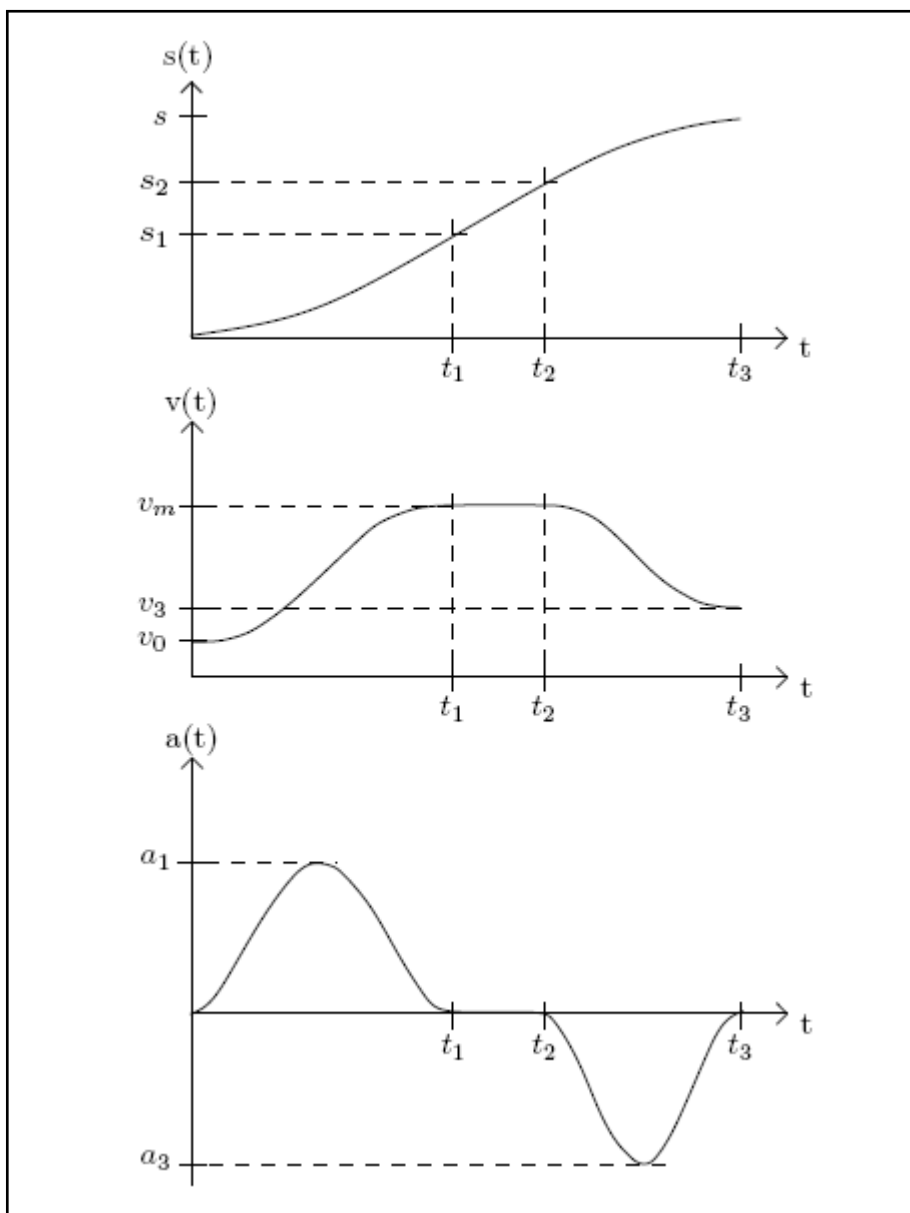


Abb. 3-41: Beschleunigungsbegrenzte Bewegung mit Sinoidenprofil

Das Profil für die **"Ruckbegrenzte Bewegung"** kann wie folgt beschrieben werden:

RIL_CommonTypes.library

Für viele anspruchsvollere Bewegungsaufgaben gilt die Einhaltung einer Beschleunigung mit Ruckbegrenzung. Beispielsweise für den Transport von Flüssigkeiten oder aber auch bei leicht schwingungsfähigen Maschinen.

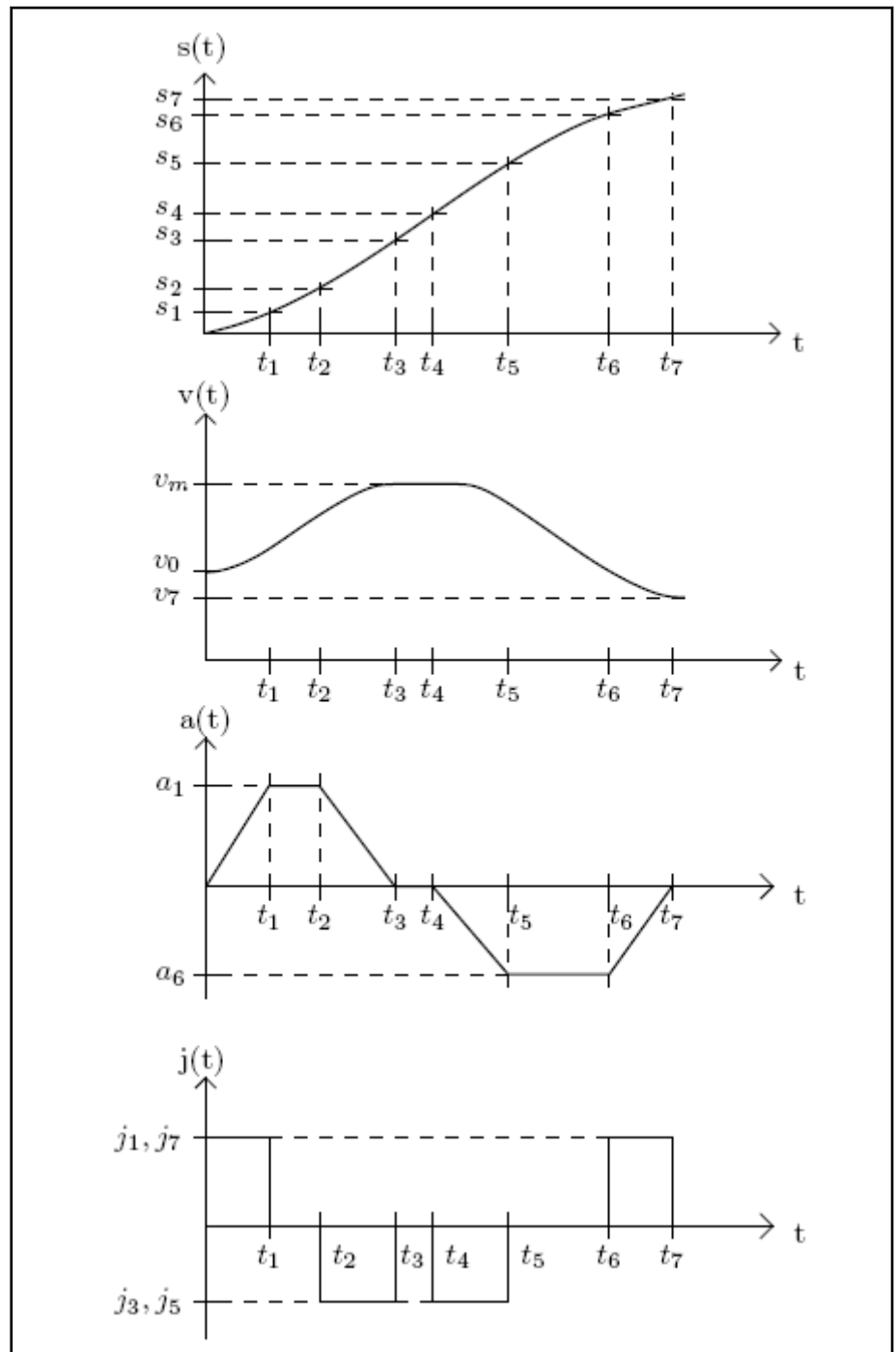


Abb. 3-42: Ruckbegrenzte Bewegung

RIL_CommonTypes.library

3.5 Datentypen zum Aufsynchronisieren in den lagegeregelten Synchron-Betriebsarten

3.5.1 Allgemeines

Die im folgenden beschriebenen Funktionen sind in den Betriebsarten Winkelsynchronisation und Elektronische Kurvenscheibe wirksam. Generell wird das Aufsynchronisieren nach dem Zuschalten der Synchron-Betriebsart durchgeführt. Dieser Aufsynchronisiervorgang ist in zwei Schritte unterteilt:

1. Zunächst beschleunigt bzw. verzögert der Folgeantrieb mit der "Aufsynchronisierbeschleunigung" auf die synchrone Geschwindigkeit.
2. Im zweiten Schritt wird die Positions Differenz abgebaut. Dazu wird der synchronen Geschwindigkeit eine "Aufsynchronisierungsgeschwindigkeit" überlagert.

Bezüglich der Lageanpassung im zweiten Schritt wird zwischen relativer und absoluter Synchronisation ([MC_START_MODE](#), Seite 66) unterschieden:

- Bei relativer Synchronisation ist das Aufsynchronisieren mit dem Erreichen der synchronen Geschwindigkeit abgeschlossen.
- Nur bei absoluter Synchronisation wird nach dem Erreichen der synchronen Geschwindigkeit der absolute Winkelbezug hergestellt (Aufsynchronisierungsrichtung, [MC_SYNC_DIRECTION](#), Seite 68).

3.5.2 MC_START_MODE

Kurzbeschreibung

Dieser Parameter gibt die Art der automatischen dynamischen Synchronisation bei Phasensynchronisation an.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of PLCopen

Tab. 3-31: Bibliothekszuordnung

Typdefinition

```

TYPE MC_START_MODE :
(
  ABSOLUTE      := 0, (* Aufsynchronisieren mit absolute Winkelbezug, *)
                  (* nur bei erster Ausführung *)
  RELATIVE      := 1, (* Aufsynchronisieren auf die synchrone Geschwindigkeit, *)
                  (* nur bei erster Ausführung *)
  ABSOLUTE_RAMPIN := 2, (* Aufsynchronisieren mit absolute Winkelbezug, *)
                  (* bei jeder Ausführung *)
  RELATIVE_RAMPIN := 3, (* Aufsynchronisieren auf die synchrone Geschwindigkeit, *)
                  (* bei jeder Ausführung *)
);
END_TYPE

```

Absolute Synchronisation

Die Folgeachseposition beim Nulldurchgang der Leitachseposition ist eindeutig definiert und ändert sich nicht bei jedem Aufruf von MB_MotionProfile/MC_CamIn.

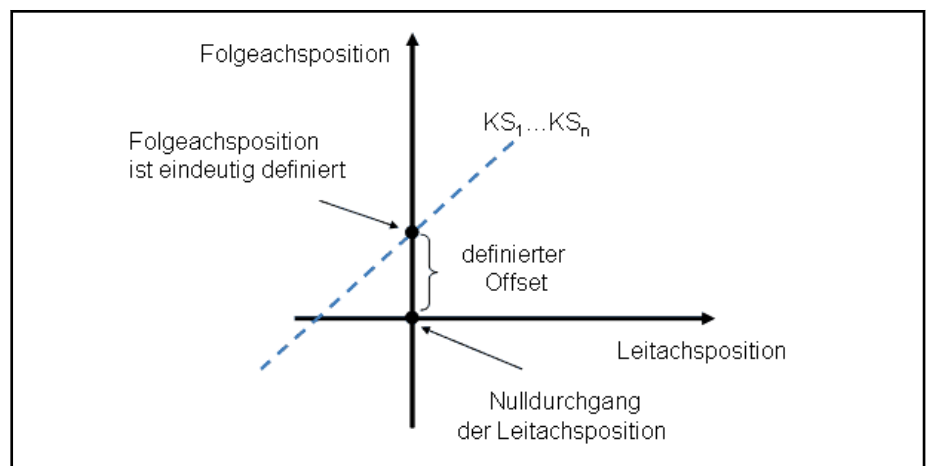


Abb. 3-43: Absolute Synchronisation

Relative Synchronisation

Die Folgeachseposition beim Nulldurchgang der Leitachseposition ergibt sich zufällig und ändert sich bei jedem Aufruf von MB_MotionProfile/ MC_CamIn.

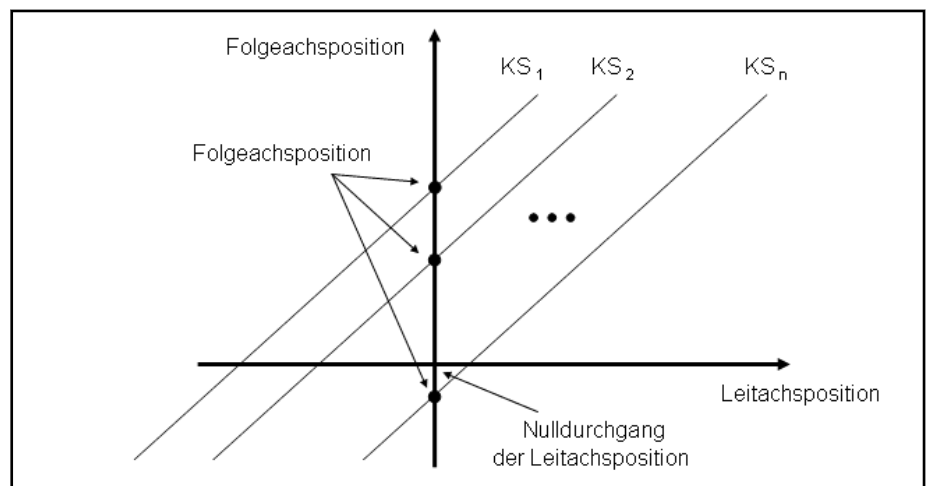


Abb. 3-44: Relative Synchronisation

"Harte" Synchronisation

- StartMode= 0: absolute Synchronisation, "hartes Umschalten" bei Profilwechsel,
- StartMode= 1: relative Synchronisation, "hartes Umschalten" bei Profilwechsel.

Die Umschaltung erfolgt nach Überfahren der Leitachs-Umschaltposition (A-0-2900) durch die Leitachse.

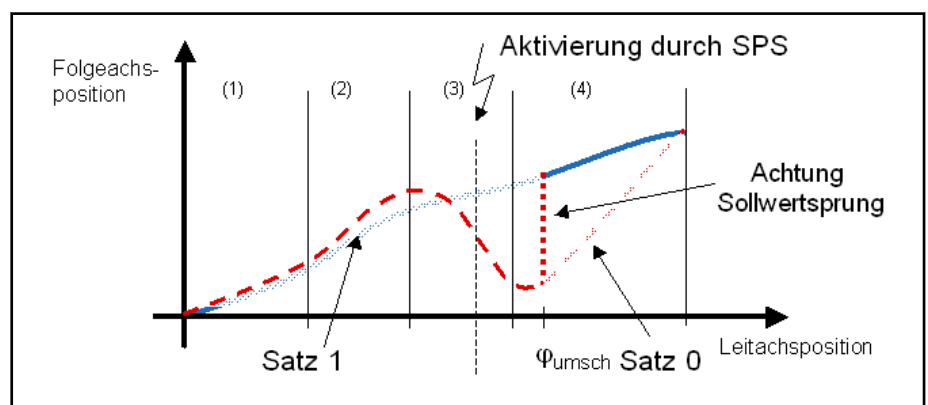


Abb. 3-45: Hartes Umschalten

RIL_CommonTypes.library

- "Weiche" Synchronisation
- StartMode= 2: absolute Synchronisation, "weiches Umschalten" bei Profilwechsel,
 - StartMode= 3: relative Synchronisation, "weiches Umschalten" bei Profilwechsel.

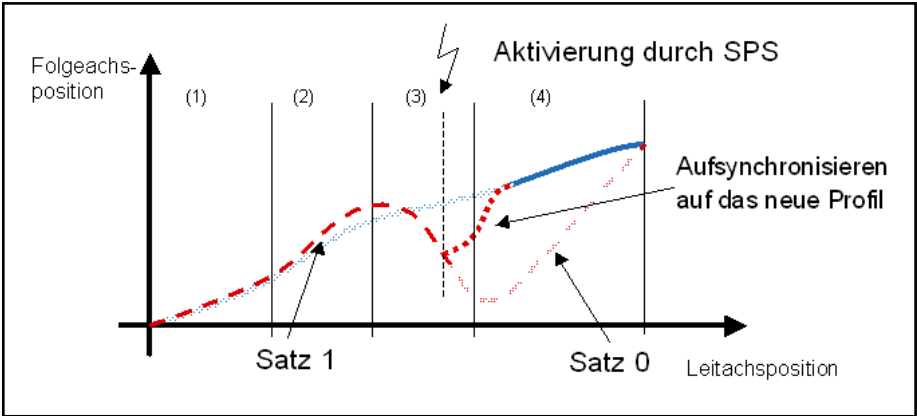


Abb. 3-46: Weiches Umschalten
Die Umschaltung erfolgt sofort nach Starten des FBs, es wird nicht auf das Überfahren der Leitachs-Umschaltposition (A-0-2900) durch die Leitachse gewartet.

Das "Weiche Aufsynchronisieren" setzt die Berechnung auf der Steuerung voraus.

3.5.3 MC_SYNC_DIRECTION

Kurzbeschreibung

Bei Modulo-Achsen kann die Bewegung der Achse in positiver oder negativer Richtung erfolgen.
Durch die Aufsynchronisierrichtung wird festgelegt, in welche Richtung sich die Achse bewegen soll.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of PLCopen

Tab. 3-32: Bibliothekszuordnung
Typdefinition

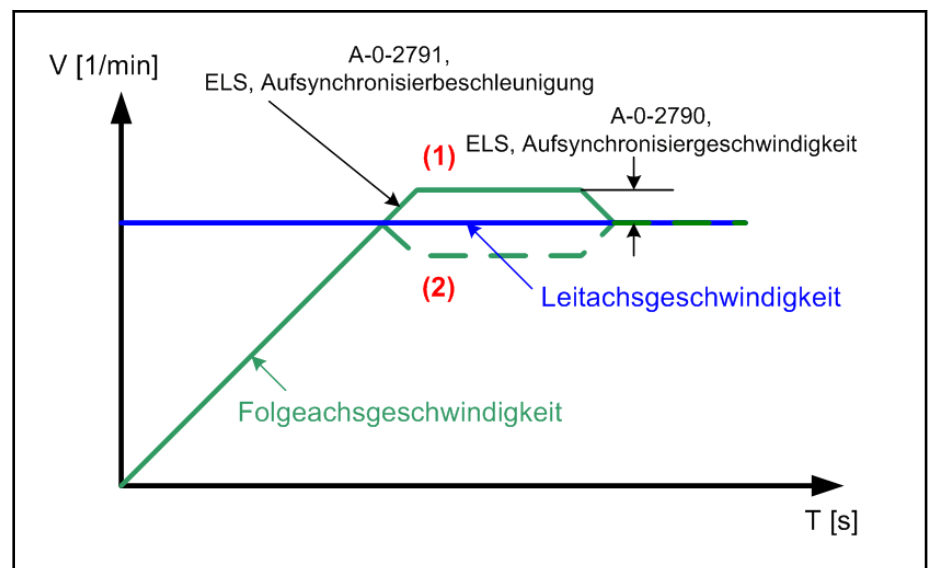
```

TYPE MC_SYNC_DIRECTION :
(
    SHORTEST_WAY      := 0,  (* kürzester Weg      *)
    CATCH_UP          := 1,  (* immer vorwärts *)
    SLOW_DOWN         := 2
);
END_TYPE

```

Ist der kürzeste Weg zur absoluten Synchronisation jedoch kleiner als das "Aufsynchronisierfenster", bei Modulo-Format, wird der kürzeste Weg verfahren und die vorgegebene Aufsynchronisierrichtung ignoriert.

Beispiel MLC:



① Folgeachse läuft Leitachse nach

② Leitachse läuft Folgeachse nach

Abb. 3-47: Übergang geschwindigkeitssynchron nach lagesynchron

Im Bild beschleunigt zunächst die Folgeachse auf die gleiche Geschwindigkeit wie die Leitachse. Danach muss die Winkeldifferenz zwischen Leit- und Folgeachse abgebaut werden, je nachdem, wer vor dem anderen läuft.

Beim Aufsynchonisieren wirken die beiden Parameter "A-0-2790, ELS, Aufsynchonisiergeschwindigkeit" und "A-0-2791, ELS, Aufsynchonisierbeschleunigung".

Der Vorgang ist abgeschlossen, wenn beide geschwindigkeits- und lagesynchron laufen.

3.6 Datentypen zur Fehlerdiagnose bei IndraMotion-Bausteinen

3.6.1 ERROR_CODE

Kurzbeschreibung Die Aufzählung enthält die Fehlerarten, die einen Fehler am Funktionsbaustein charakterisieren.

Der Errorcode soll dem Anwender einen schnellen Hinweis auf die Fehlerursache geben.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of POU diagnosis -> Includes

Tab. 3-33: Bibliothekszuordnung

Typdefinition

```
(* Enumeration of valid types of diagnosis/ Aufzählung gültiger Fehlerarten *)
TYPE ERROR_CODE :
(
  NONE_ERROR           := 16#0000, (* No error code available *)
  INPUT_INVALID_ERROR  := 16#0001, (* kein Fehlercode verfügbar, FB fehlerfrei *)
  COMMUNICATION_ERROR  := 16#0002, (* Invalid input *)
  RESOURCE_ERROR       := 16#0003, (* Fehler auf Grund ungültiger Eingabewerte *)
  ACCESS_ERROR         := 16#0004, (* Error while communication *)
                          (* Fehler beim Aufbau einer Kommunikation *)
                          (* bzw.während der Kommunikation *)
                          (* Resource error *)
                          (* Ressource (z.B.Parameter) ist nicht vorhanden *)
                          (* Access error *)
                          (* Ressource ist zwar da, aber man kann nicht *)
                          (* auf sie zugreifen. *)
                          (* Beispiel: Man möchte Parameter in Phase 4 be- *)
)
```

RIL_CommonTypes.library

```

STATE_MACHINE_ERROR    := 16#0005, (* schreiben, der nur in Phase 2 beschreibbar ist*)
                          (* Invalid state State - Machine *)
                          (* Der aktuelle Zustand der State-Machine *)
                          (* verhindert die weitere Abarbeitung *)
INPUT_RANGE_ERROR      := 16#0006, (* Error because Input out of range *)
                          (* Während der Verarbeitung wird festgestellt, *)
                          (* dass durch eine oder mehrere Eingangsgrößen *)
                          (* undefinierte bzw. zu große oder zu kleine *)
                          (* Sollwerte generiert werden. *)
                          (* Error while calculation *)
CALCULATION_ERROR      := 16#0007, (* Ein Berechnungsfehler ist aufgetreten, *)
                          (* z.B. Division durch 0 *)
                          (* Device Error occurred *)
DEVICE_ERROR           := 16#0008, (* Übergeordneter Gerätefehler ist aufgetreten, *)
                          (* z.B. IndraDrive: *)
                          (* Kurvenscheibenbetriebsart F2005, *)
                          (* Ungültige Kurvenscheibe *)
OTHER_ERROR            := 16#7FFE, (* Undefined Error *)
                          (* allgemeiner (undefinierter) Fehler *)
SYSTEM_ERROR           := 16#7FFF (* System error occured *)
                          (* z.B. Betriebssystemfehler *)
);
END_TYPE

```

3.6.2 ERROR_STRUCT

Kurzbeschreibung Die Fehlerstruktur beinhaltet alle Detailinformationen zu einem konkreten Fehler.

Die Information ist an der jeweiligen Instanz eines PLCopen-Funktionsbausteines direkt auswertbar und ablesbar. Die Interpretation der Bestandteile muss wie nachstehend beschrieben erfolgen:

1. Ermitteln der Tabellenummer ([ERROR_TABLE](#), [Seite 71](#)). Damit steht die Herkunft des Fehlers fest.
2. Mit dieser Kenntnis wird "Additional1" ausgewertet. Der Inhalt ist die Diagnosenummer, die in den jeweiligen Bausteinbeschreibungen zu finden ist (siehe auch Hinweise zur Störungsbeseitigung, MLC). Kontrolle, ob "Additional2" = 0 ist, wenn ja, ist die Diagnose komplett, wenn nein, gehe weiter.
3. Auswertung von "Additional2" (bei MLC_Table bisher nicht genutzt).

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of POU diagnosis

Tab. 3-34: Bibliothekszuordnung
Typdefinition

```

(* Structure for description of diagnosis of function blocks. *)
(* The ERROR_STRUCT is a structure that contains information on the *)
(* corresponding diagnose, which is market by the PLCopen output 'Error' *)

TYPE ERROR_STRUCT :
STRUCT
  Table      : ERROR_TABLE; (* Additional diagnosis table *)
                          (* weist auf die Liste, die zum Identifizieren *)
                          (* des Fehler vom "Additional1" benötigt wird *)
  Additional1 : DWORD;      (* Additional diagnosis number1 *)
                          (* Enthält konkrete Fehlernummer entsprechend *)
                          (* der vorgegebenen Tabelle. *)
  Additional2 : DWORD;      (* Additional diagnosis number2 *)
                          (* Kann Ergänzung zu "Additional1" enthalten. *)
END_STRUCT
END_TYPE

```



In **Additional1** werden die in den "Rexroth IndraMotion MLC, Diagnosen - Hinweise zur Störungsbeseitigung" beschriebenen Meldungen, Warnungen und Fehlermeldungen angezeigt.

3.6.3 ERROR_TABLE

Kurzbeschreibung Die ERROR_TABLE liefert das erste Element der [ERROR_STRUCT](#), Seite 70. Anhand der Nummer ist die Tabelle auszuwählen, in der der "Additional1"- Fehlercode gesucht werden muß.

Beispiel:

Für MLC-Anwendungen sollte dies z.B. die Tabelle "MLC_TABLE", 16#0030, sein.

Die in der jeweiligen Baustein-Beschreibung enthaltenen Fehlernummern unter "Additional1" beziehen ihre Informationen aus dieser Tabelle.

Bei Fehlern im Zugriff auf einen Parameter können Bausteine die Tabelle SERCOS_TABLE, 16#0010, verwenden.

Bei Fehlern in der Kommando-Abarbeitung können Bausteine die Tabellen F_RELATED_TABLE, 16#0170, oder INDRV_TABLE, 16#0070, verwenden.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of POU diagnosis - includes

Tab. 3-35: Bibliothekszuordnung
Typdefinition

```
(* Enumeration of valid tables in additional information *)
TYPE ERROR_TABLE:
(
  NO_TABLE_USED      := 16#0000, (* reserved *)
  SERCOS_TABLE       := 16#0010, (* Sercos Error: Addtitionall = Sercos Code *)
  MLD_TABLE          := 16#0020, (* Drive-based Motion - Logic *)
  MLC_TABLE           := 16#0030, (* Controller - based Motion - Logic *)
  MTX_TABLE          := 16#0040, (* CNC *)
  MLP_TABLE          := 16#0050, (* PC - based Motion Logic *)
  PLC_TABLE          := 16#0060, (* PLC *)
  INDRV_TABLE        := 16#0070, (* IndraDrive *)
  DIAx_TABLE         := 16#0080, (* DIAx *)
  ECO_TABLE          := 16#0090, (* EcoDrive *)
  PB_DP_TABLE        := 16#0130, (* Profibus *)
  DEVICENET_TABLE    := 16#0140, (* DeviceNet *)
  ETHERNET_TABLE     := 16#0150, (* EtherNet *)
  ETHERNET_IP_TABLE  := 16#0151, (* EtherNet/IP ODVA *)
  INTERBUS_TABLE     := 16#0160, (* Interbus *)
  F_RELATED_TABLE    := 16#0170, (* function related *)
  CAN_TABLE          := 16#0180, (* CAN bus *)
  INLINEIO_TABLE     := 16#0190, (* Inline IOs *)
  PROFINETIO_TABLE   := 16#0200, (* ProfinetIO, ProfinetIO Device *)

  USER1_TABLE        := 16#1000, (* free User Table *)
  ...
  USER10_TABLE       := 16#1009 (* free User Table *)
);
END_TYPE
```

3.7 Datentypen zur Wiedergabe des Datum und absoluter Zeitangaben bei IndraMotion-Bausteinen

3.7.1 Wiedergabe des Datum und absoluter Zeitangaben, allgemeines

Zur Wiedergabe des Datum und anderer absoluter Zeitangaben existieren drei Strukturen:

- [SYS_TIME64](#), Seite 72, Systemzeit in Mikrosekunden seit 01.01.1970, 00:00 Uhr.
- [EXT_SYS_TIME_DATE](#), Seite 72, Systemzeit in Mikrosekunden seit 01.01.1970, 00:00 Uhr, Zusatzdaten Jahr, Monat, Tag, Stunden, Minuten, Sekunden, Millisekunden und Wochentag werden aus der Systemzeit berechnet.

RIL_CommonTypes.library

- [SYS_TIME_DATE](#), Seite 72, Jahr, Monat, Tag, Stunden, Minuten, Sekunden, Millisekunden und Wochentag, berechnet auf der Basis der Systemzeit in Mikrosekunden seit 01.01.1970, 00:00 Uhr.

3.7.2 SYS_TIME64

Kurzbeschreibung Struktur zur Wiedergabe der Systemzeit in Mikrosekunden seit 01.01.1970, 00:00 Uhr.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of time

Tab. 3-36: Bibliothekszuordnung

Typdefinition

```
(* Structure of system time - Microseconds since 1970 *)
TYPE SYS_TIME64 :
STRUCT
  LowValue:      DWORD;      (* Low value [us] *)
  HighValue:     DWORD;      (* High value [us] *)
END_STRUCT
END_TYPE
```

3.7.3 EXT_SYS_TIME_DATE

Kurzbeschreibung Struktur zur Wiedergabe der Systemzeit in Mikrosekunden seit 01.01.1970, 00:00 Uhr und der Zusatzdaten Jahr, Monat, Tag, Stunden, Minuten, Sekunden, Millisekunden und Wochentag, die aus der Systemzeit berechnet werden.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of time

Tab. 3-37: Bibliothekszuordnung

Typdefinition

```
(* Extended structure of system time *)
TYPE EXT_SYS_TIME_DATE :
STRUCT
  LowValue: DWORD;      (* Low value [us] *)
  HighValue: DWORD;     (* High value [us] *)
  Year : UINT;          (* Year *)
  Month : UINT;         (* Month *)
  Day : UINT;           (* Day *)
  Hour : UINT;          (* Hour *)
  Minute : UINT;        (* Minute *)
  Second : UINT;        (* Second *)
  Milliseconds : UINT;  (* Milliseconds *)
  DayOfWeek: UINT;     (* Day of week *)
END_STRUCT
END_TYPE
```

3.7.4 SYS_TIME_DATE

Kurzbeschreibung Struktur zur Wiedergabe der Daten Jahr, Monat, Tag, Stunden, Minuten, Sekunden, Millisekunden und Wochentag, die aus einer Systemzeit in Mikrosekunden seit 01.01.1970, 00:00 Uhr berechnet werden.

Bibliothek	Bereich
RIL_CommonTypes	Datatypes of time

Tab. 3-38: Bibliothekszuordnung

Typdefinition

```

(* Structure of system time *)
TYPE SYS_TIME_DATE :
STRUCT
    Year : UINT;          (* Year *)
    Month : UINT;         (* Month *)
    Day : UINT;           (* Day *)
    Hour : UINT;          (* Hour *)
    Minute : UINT;        (* Minute *)
    Second : UINT;        (* Second *)
    Milliseconds : UINT;  (* Milliseconds *)
    DayOfWeek: UINT;     (* Day of week *)
END_STRUCT
END_TYPE

```

3.8 Datentypen zur Unterstützung von Messtaster-Funktionen

3.8.1 Datentypen zur Unterstützung von Messtaster-Funktionen, allgemeines

Der Abschnitt enthält Datentypen, die die Messtaster-Funktionen sowohl bei Abarbeitung im Antrieb als auch in der Steuerung unterstützen:

- [MB_DIFF_EVAL](#), Seite 73, Einstellung der Methoden der Differenzmessung
- [MB_EXPECT_WINDOW_MODE](#), Seite 74, Konfiguration und Aktivierung des Erwartungsfensters
- [MB_INTERPOLATION_MODE](#), Seite 75, Aktivierung der Interpolation
- [MB_POSITION_EVAL](#), Seite 75, Auswahl und Aktivierung des Triggersignals
- [MB_PROBE_MODULE](#), Seite 75, Auswahl der Triggerquelle für den Messtaster
- [MB_PROBE_INPUT](#), Seite 76, Auswahl des gewünschten Einganges auf dem ausgewählten Modul
- [MB_PROBE_DATA](#), Seite 76, Auswahl der Messtaster-Bezugssignalquelle

Die zu einer Messtaster-Funktion gehörenden Daten sind in nachstehender Struktur zusammengestellt:

- [MB_PROBE_CONFIG](#), Seite 77

3.8.2 MB_DIFF_EVAL

Kurzbeschreibung Der Aufzählungstyp "MB_DIFF_EVAL" unterstützt die Einstellung der Methoden der Differenzmessung.

Bibliothek	Bereich
RIL_CommonTypes	TouchProbe

Tab. 3-39: Bibliothekszuordnung

Typdefinition

```

TYPE MB_DIFF_EVAL :
(
    DIFF_DISABLED      := 0, (* diff.measurement is disabled *)
    DIFF_STANDARD_MODE := 1, (* diff. last two measured values*)
    DIFF_NEG_TO_NEG_EDGE:= 2, (* neg. edge to neg. edge diff. *)
    DIFF_POS_TO_NEG_EDGE:= 3, (* pos. edge to neg. edge diff. *)
    DIFF_NEG_TO_POS_EDGE:= 4, (* neg. edge to pos. edge diff. *)
    DIFF_POS_TO_POS_EDGE:= 5 (* pos. edge to pos. edge diff. *)
);
END_TYPE

```

- Funktionsbeschreibung
- Folgende Einstellungen werden unterstützt:
- 0 - Differenzmessung ist abgeschaltet, Differenz-Signale für ML_Touch-Probe bzw. ML_TouchProbe_Continuous werden nicht berechnet.
 - 1 - Differenz zwischen den letzten zwei Messwerten, je nach aktivierter Flanke.

Beispiel: beide Flanken sind aktiviert, erfasst wird Differenz pos/neg Flanke, weiter neg/pos Flanke, weiter pos/neg Flanke.
 - Im nachstehenden Bild sind die erweiterten Methoden 2..5 der Differenzmessung dargestellt:

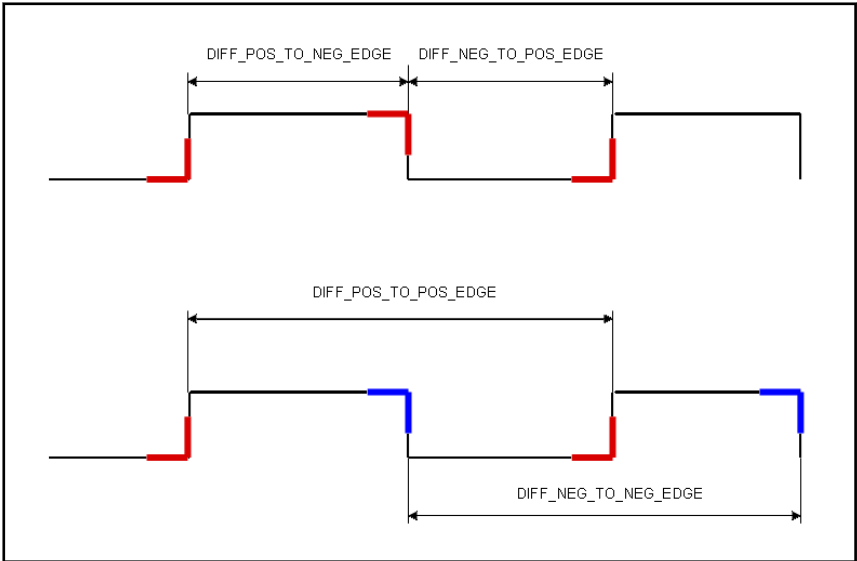


Abb. 3-48: Methoden der Differenzmessung

3.8.3 MB_EXPECT_WINDOW_MODE

Kurzbeschreibung

Der Aufzählungstyp "MB_EXPECT_WINDOW_MODE" unterstützt die Konfiguration und Aktivierung des Erwartungsfensters.

Bibliothek	Bereich
RIL_CommonTypes	TouchProbe

Tab. 3-40: Bibliothekszuordnung
Typdefinition

```
TYPE MB_EXPECT_WINDOW_MODE :  
(  
  EXPECT_WINDOW_DISABLED:= 0, (* expectation window disabled *)  
  EXPECT_WINDOW_ABS      := 1, (* expectation window absolute *)  
  EXPECT_WINDOW_REL      := 2, (* expectation window relative *)  
  (* to "ExpectWindowSetPoint" *)  
);  
END_TYPE
```

- Funktionsbeschreibung
- Der Anwender kann sich getrennt für die positive und die negative Flanke des Messtasterwertes für ein Erwartungsfenster entscheiden.
- Entscheidet sich der Anwender für die Nutzung eines Erwartungsfensters um , so kann er dessen Grenzen auf zwei unterschiedliche Arten definieren:
- Erwartungsfenster deaktiviert:

Die jeweilige Flanke des Messtastersignals wird ungefiltert verwendet.

2. Erwartungsfenster absolut definiert:

Die Startposition und die Endposition des Erwartungsfensters sind absolut anzugeben. Der Mittelpunktwert ist in diesem Fall bedeutungslos.

3. Erwartungsfenster relativ definiert:

Der Mittelpunktwert des Erwartungsfensters sind absolut anzugeben. Die Startposition und die Endposition geben die Entfernung zum Mittelpunktswert an.

3.8.4 MB_INTERPOLATION_MODE

Kurzbeschreibung Der Aufzählungstyp "MB_INTERPOLATION_MODE" unterstützt die Aktivierung der Interpolation.

Bibliothek	Bereich
RIL_CommonTypes	TouchProbe

Tab. 3-41: Bibliothekszuordnung
Typdefinition

```

TYPE MB_INTERPOLATION_MODE:
(
  NO_INTERPOLATION      := 0,  (* no interpolation      *)
  LINEAR_INTERPOLATION  := 1,  (* linear interpolation  *)
  CUBIC_3_INTERPOLATION := 2,  (* cubic interpolation with 3 setpoints *)
  CUBIC_4_INTERPOLATION := 3,  (* cubic interpolation with 4 setpoints *)
);
END_TYPE

```

3.8.5 MB_POSITION_EVAL

Kurzbeschreibung Der Aufzählungstyp "MB_POSITION_EVAL" unterstützt die Auswahl und Aktivierung des Triggersignals.

Mit diesen Enumerationswerten kann ausgewählt, ob auf die positive oder die negative Flanke bzw. auf beiden Flanken getriggert werden soll.



Der Defaultwert "Null" blockiert die Triggierung!

Bibliothek	Bereich
RIL_CommonTypes	TouchProbe

Tab. 3-42: Bibliothekszuordnung
Typdefinition

```

TYPE MB_POSITION_EVAL :
(
  POSITION_EVAL_DISABLED:= 0, (* disabled *)
  POSITION_POS_EDGE     := 1, (* positive edge is evaluated *)
  POSITION_NEG_EDGE     := 2, (* negative edge is evaluated *)
  POSITION_POS_NEG_EDGE := 3  (* pos.and neg. edge are evaluated *)
);
END_TYPE

```

3.8.6 MB_PROBE_MODULE

Kurzbeschreibung Der Aufzählungstyp "MB_PROBE_MODULE" unterstützt die Auswahl der Triggerquelle für den Messtaster.

Bibliothek	Bereich
RIL_CommonTypes	TouchProbe

Tab. 3-43: Bibliothekszuordnung

RIL_CommonTypes.library

Typdefinition

```

TYPE MB_PROBE_MODULE :
(
  ONBOARD_MODULE      := 0, (* Onboard module MLC,      Inputs 1...8          *)

  FUNCTION_MODULE_1:= 1, (* Function module 1 MLC, Inputs 1..16 (CFL-01.1-E2, Fast IO) *)
  FUNCTION_MODULE_2:= 2, (* Function module 2 MLC, Inputs 1..16 (CFL-01.1-E2, Fast IO) *)
  FUNCTION_MODULE_3:= 3, (* Function module 3 MLC, Inputs 1..16 (CFL-01.1-E2, Fast IO) *)
  FUNCTION_MODULE_4:= 4, (* Function module 4 MLC, Inputs 1..16 (CFL-01.1-E2, Fast IO) *)
);
END_TYPE

```

3.8.7 MB_PROBE_INPUT

Kurzbeschreibung Der Aufzählungstyp "MB_PROBE_INPUT" unterstützt die Auswahl des gewünschten Einganges auf dem ausgewählten Modul.



Für das OnBoard -Modul stehen die Eingänge 1..8 zur Verfügung.
Die Fast IO- Module haben die Eingänge 1..16 als Triggerquelle.

Bibliothek	Bereich
RIL_CommonTypes	TouchProbe

Tab. 3-44: Bibliothekszuordnung
Typdefinition

```

TYPE MB_PROBE_INPUT :
(
  D_INPUT_1      := 1, (* digital input 1 *)
  D_INPUT_2      := 2, (* digital input 2 *)
  D_INPUT_3      := 3, (* digital input 3 *)
  D_INPUT_4      := 4, (* digital input 4 *)
  D_INPUT_5      := 5, (* digital input 5 *)
  D_INPUT_6      := 6, (* digital input 6 *)
  D_INPUT_7      := 7, (* digital input 7 *)
  D_INPUT_8      := 8, (* digital input 8 *)
  D_INPUT_9      := 9, (* digital input 9 *)
  D_INPUT_10     := 10, (* digital input 10 *)
  D_INPUT_11     := 11, (* digital input 11 *)
  D_INPUT_12     := 12, (* digital input 12 *)
  D_INPUT_13     := 13, (* digital input 13 *)
  D_INPUT_14     := 14, (* digital input 14 *)
  D_INPUT_15     := 15, (* digital input 15 *)
  D_INPUT_16     := 16, (* digital input 16 *)
);
END_TYPE

```

3.8.8 MB_PROBE_DATA

Kurzbeschreibung Der Aufzählungstyp "MB_PROBE_DATA" unterstützt die Auswahl der Mess-taster-Bezugssignalquelle sowohl für die Messtaster der XLC/MLC (A-Parameter), als auch für die Messtaster des Antriebs (S/P-Parameter).



Bis zur Version 13V04 der XLC/MLC wurde von den Messtastern der Steuerung nur der Parameter A-0-0100, "Positionswert der Achse, Geber 1" unterstützt.

Bibliothek	Bereich
RIL_CommonTypes	TouchProbe

Tab. 3-45: Bibliothekszuordnung
Typdefinition

```

TYPE MB_PROBE_DATA :
(
  TP_ACT_POS_FEEDBACK1 := 0, // S-0-0051 / A-0-0100, Lage-Istwert Geber 1
  TP_ACT_POS_FEEDBACK2 := 1, // S-0-0053 / -----, Lage-Istwert Geber 2
);

```

```

TP_ACT_POSIN_ACT_CYCLE      := 2, // P-0-0753 / -----, Lageistwert im Istwertzyklus
TP_SYNC_CMD_POS            := 3, // P-0-0778 / A-0-2653, Synchroner Lagesollwert
TP_PROBE_TIME              := 4, // ----- / -----, Zeitmessung
TP_ACT_MEASURING_ENC_POS   := 5, // P-0-0052 / -----, Lageistwert Messgeber
TP_RES_MASTER_AXIS_POS     := 6, // P-0-0775 / A-0-2650, Resultierende Leitachspannung
TP_EFF_MASTER_AXIS_POS     := 7, // P-0-0776 / A-0-2651, Wirksame Leitachspannung
TP_CAM_SHAFT_PROF_ACCESS_ANGLE := 8, // P-0-0227 / A-0-2656, Kurvenscheibentabelle, Zugriffswinkel
TP_ACT_MASTER_AXIS_POS     := 9, // noch nicht unterstützt!!!
                             // P-0-0053 / A-0-0300, Leitachspannung
TP_ELS_CMD_POS             := 10, // ----- / A-0-2655, ELS, Positionssollwert
TP_USER_DATA_CONTAINER_A   := 11, // ----- / A-0-0521, Container des benutzerdefinierten Istwerts A
TP_USER_DATA_CONTAINER_B   := 12, // ----- / A-0-0523, Container des benutzerdefinierten Istwerts B
TP_USER_DATA_CONTAINER_C   := 13, // ----- / A-0-0525, Container des benutzerdefinierten Istwerts C
TP_USER_DATA_CONTAINER_D   := 14, // ----- / A-0-0527, Container des benutzerdefinierten Istwerts D
TP_USER_DATA_CONTAINER_E   := 15, // ----- / A-0-0529, Container des benutzerdefinierten Istwerts E
TP_USER_DATA_CONTAINER_F   := 16, // ----- / A-0-0531, Container des benutzerdefinierten Istwerts F
);
END_TYPE

```

3.8.9 MB_PROBE_CONFIG

Kurzbeschreibung Die Struktur "MB_PROBE_CONFIG" dient der Aufnahme der Daten einer Messtaster-Funktion.

Bibliothek	Bereich
RIL_CommonTypes	TouchProbe

Tab. 3-46: Bibliotheksuzuordnung
Typdefinition

```

TYPE MB_PROBE_CONFIG :
STRUCT
ModuloValueDrive      :REAL;          (* Modulo-Wert *)
ProcessingFormat       :BOOL;         (* Wichtung Lage, abs./modulo *)
ProbePositionEval     :MB_POSITION_EVAL; (* Aktivierung Triggersignal, def.: Aus *)
                             (* pos., neg., beide Flanken *)
DifferenzMeasurment   :MB_DIFF_EVAL;  (* Methode der Differenzmessung, def.: Aus *)
ExpectWindowPosMode   :MB_EXPECT_WINDOW_MODE; (* Mode des Erwartungsfensters pos. Flanke, *)
                             (* def.: Aus, absolut / relativ zum Setpoint *)
ExpectWindowNegMode   :MB_EXPECT_WINDOW_MODE; (* Mode des Erwartungsfensters neg. Flanke, *)
                             (* def.: Aus, absolut / relativ zum Setpoint *)
ExpectWindowPosStart   :REAL;          (* Startpos. Erwartungsfensters pos. Flanke *)
ExpectWindowPosEnd     :REAL;          (* Endpos. Erwartungsfensters pos. Flanke *)
ExpectWindowNegStart   :REAL;          (* Startpos. Erwartungsfensters neg. Flanke *)
ExpectWindowNegEnd     :REAL;          (* Endpos. Erwartungsfensters neg. Flanke *)
END_STRUCT
END_TYPE

```

3.8.10 MB_PROBE_CONFIG, Besonderheiten bei SERCOS-basierten Achsen

Funktionsbeschreibung: Nachstehende Tabelle spiegelt die Verbindung zum Parametersystem der Steuerung wider:

Strukturelement	Datentyp	Parameter
ModuloValueDrive	REAL	A-0-0045, Modulowert
ProcessingFormat	BOOL	A-0-0059.7, Wichtungsart für Positionsdaten, Bit 7 absolut/modulo
ProbePositionEval	MB_POSITION_EVAL, Seite 75	Aktivierung positive /negative Flanke, M-0-0007, Steuerwort
DifferenzMeasurment	MB_DIFF_EVAL, Seite 73	Differenzmessung, M-0-0007, Steuerwort
ExpectWindowPosMode	MB_EXPECT_WINDOW_MODE, Seite 74	Aktivierung und Konfiguration des Erwartungsfensters der positiven Flanke M-0-0007, Steuerwort

RIL_CommonTypes.library

Strukturelement	Datentyp	Parameter
ExpectWindowNegMode	MB_EXPECT_WINDOW_MODE , Seite 74	Aktivierung und Konfiguration des Erwartungsfensters der negativen Flanke M-0-0007, Steuerwort
ExpectWindowPosStart	REAL	M-0-0202, Startposition Erwartungsfenster, positive Flanke
ExpectWindowPosEnd	REAL	M-0-0203, Endposition Erwartungsfenster, positive Flanke
ExpectWindowNegStart	REAL	M-0-0204, Startposition Erwartungsfenster, negative Flanke
ExpectWindowNegEnd	REAL	M-0-0205, Endposition Erwartungsfenster, negative Flanke

Tab. 3-47: Parameterzuordnung Messtaster-Funktion

4 ML_Base.library

4.1 ML_Base, Überblick

Die ML_Base-Bibliothek enthält die für die Realisierung der Systeme XLC/MLC wesentlichen Datentypen, die im Unterschied zur RIL_CommonTypes-Bibliothek nur von nachstehenden Systemen verwendet werden.

- IndraLogic XLC mit und ohne sercos Interface und
- IndraMotion MLC

Diese Datentypen wurden ergänzt um Funktionen, Funktionsbausteine und Methoden zum Zugriff auf die konkrete Steuerungshardware.

Der Name der Bibliothek "ML_Base" wird aus diesem Grund um den konkreten Bezug erweitert, also

"ML_Base_XLC_L45": IndraLogic XLC, Steuerung CML45 oder

"ML_Base_MLC_L25": IndraMotion MLC, Steuerung CML25.



Die Verwendung von Elementen, außer der hier beschriebenen, ist zu unterlassen, da Nebenwirkungen, einschließlich Störungen im Bewegungsablauf der Achsen, nicht ausgeschlossen werden können!

Die Bibliothek weist folgende Struktur auf:

Datentypen

- **Command Types (interne Verwendung)**
- **Common Types**
 - [MB_IDN auf Seite 81](#), definiert die Länge eines Parameters (systemabhängig)
 - übrige: interne Verwendung
- **Flex Structs**
 - [MB_CHANGESTEP_MODE auf Seite 82](#), (ENUM), zugelassenen Änderungsmöglichkeiten innerhalb eines FlexProfiles
 - [MB_EVENT_ACTION auf Seite 83](#), (ENUM), Aktionen, die durch ein Event ausgelöst werden sollen
 - [MB_EVENT_TRIGGER auf Seite 84](#), (ENUM), beschreibt den Auslöser (Trigger), der bei einem Event zu einem Ausführen einer gewünschten Aktion führen soll
 - [MB_EXECUTION_MODE auf Seite 84](#), (ENUM), einmalige oder zyklische Abarbeitung des FlexProfile
 - [MB_FLEX_EVENT auf Seite 85](#), (STRUCT), Konfiguration von Bedingungen, die an einem vorgegebenen Schritte eines FlexProfils eingebunden werden können
 - [MB_MASTER_TYPE auf Seite 85](#), (ENUM), Abhängigkeit des Bewegungsschrittes, aktuelle Masterachse (MASTER_AXIS0) oder Zeit (MASTER_TIME)
 - [MB_PROFILE_START_POINT auf Seite 86](#), (ENUM), beschreibt, wie bei einem Profilstart oder Profilwechsel der Ursprung des Profils zu interpretieren ist; er kann sowohl für die Leit- als auch Folgeachse auf die aktuelle Position oder den Nullpunkt bezogen werden

ML_Base.library

- [MB_PROFILE_STEP](#) auf Seite 87, (STRUCT), Konfiguration eines Bewegungsschritts in einem FlexProfile
- [MB_STEP_TYPE](#) auf Seite 88, (ENUM), Eigenschaft des Bewegungsschrittes, starrer (STEP_FIX) Bewegungsschritt oder Flex-Bewegungsschritt
- [MB_SYNC_TYPE](#) auf Seite 88, (ENUM), Aufsynchronisier-Richtung der Achse und Art des Umschaltens
- **Read Write Parameter Types**
 - [MB_SERCOS_DATA_STATUS](#) auf Seite 88, (ENUM), Darstellung des Status eines sercos Parameterzugriffs
 - übrige: interne Verwendung
- **Shared Memory Types**
 - [ML_AXISDATA_SM](#) auf Seite 117, (STRUCT), Achsdaten über Shared Memory
 - [ML_CONTROLDATA_SM](#) auf Seite 119, (STRUCT), Steuerungsdaten über Shared Memory
 - [ML_LINK](#) auf Seite 121, Gültigkeitsbit der Steuerung im MLC-Verbund
 - [ML_PLSDATA_SM](#) auf Seite 120, (STRUCT), Nockenschaltwerk über Shared Memory
 - [ML_PROBEDATA_SM](#) auf Seite 120, (STRUCT), Messtasterdaten über Shared Memory
 - [ML_SYNC](#) auf Seite 121, enthält die vier Eingangsbits der Steuerung, die im MLC-Verbund ausgetauscht werden
 - [SM_EXT_TYPES](#) auf Seite 122, (STRUCT), zur typrichtigen Interpretation von Daten
 - [SM_TYPES](#) auf Seite 122, (STRUCT), zur typrichtigen Interpretation von Daten

Error-/ Diagnostics Memory

- [Error-/ Diagnostics Memory](#) - Unterstützung von Anwenderdiagnosen und Logbuch-Zugriffen auf Seite 123

Global Variables List

- [GVL_Base](#) auf Seite 129
- [GVL_BaseConstant](#) auf Seite 130
- **Global Variable Index**

Vereinbarungsdateien für Globale Konstanten C2C-Steuerungsverbund

- [GVL_A_GlobalVarIdx](#) auf Seite 131
- **Parameter ID**
Vereinbarungsdateien für Globale Konstanten
 - [GVL_A_Param_ID](#) auf Seite 134
 - [GVL_C_Param_ID](#) auf Seite 134
 - [GVL_M_Param_ID](#) auf Seite 135
 - [GVL_N_Param_ID](#) auf Seite 135
 - [GVL_O_Param_ID](#) auf Seite 136
 - [GVL_P_Param_ID](#) auf Seite 136
 - [GVL_S_Param_ID](#) auf Seite 137

Interface

- [Interface auf Seite 137](#), (Zugriff auf sercos Parameter)

IP-Settings

- [IP-Konfiguration auf Seite 143](#)

Projekinformation

- Bibliotheks-Info und Zugriffsfunktionen, diese Bestandteile der Bibliothek werden automatisch für die jeweilige Bibliothek erzeugt und enthalten keine spezifischen Anteile

SystemTime

- [Systemzeit auf Seite 151](#)

Target

- [Zielsystem auf Seite 156](#)

ML_Base.library wurde vereinbart, um einen einheitlichen Zugriff auf unterschiedliche IndraMotion-Gerätetypen zu definieren.

Weitere (systemunabhängige) Typdefinitionen sind in der Bibliothek RIL_CommonTypes.library enthalten.

4.2 Data Types - Datentypen

4.2.1 Data Types/ Common Types - Allgemeine Typen

MB_IDN

- Kurzbeschreibung** Der Datentyp erlaubt die Definition einer systemspezifischen Parameterlänge.
- Für die Parameter der IndraLogic XLC bzw. IndraMotion MLC sind dies 64 Bit.
- S- und P-Parameter werden automatisch auf 64 Bit aufgefüllt.

Bibliothek	Bereich
ML_Base	CommonTypes

Tab. 4-1: Bibliothekszuordnung

Programm:

```
TYPE
  MB_IDN : LINT ;
END_TYPE
```

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
M-0-0000	193273528320	16#2D 0000 0000

ML_Base.library

Parametergruppe	dezimal	hexadezimal
N-0-0000	197568495616	16#2E 0000 0000
O-0-0000	201863462912	16#2F 0000 0000

Tab. 4-2: Parameter-Offsets

4.2.2 Data Types/ Flex Structs

Datentypen zur Unterstützung von Motion- und FlexProfile-Funktionen, Allgemeines

Der Abschnitt enthält Datentypen, die die Motion- und FlexProfile-Funktionen bei Abarbeitung in der Steuerung unterstützen:

- [MB_CHANGESTEP_MODE](#), Seite 82, zugelassenen Änderungsmöglichkeiten innerhalb eines FlexProfiles,
- [MB_EVENT_ACTION](#), Seite 83, Aktionen, die durch ein Event ausgelöst werden sollen,
- [MB_EVENT_TRIGGER](#), Seite 84, beschreibt den Auslöser (Trigger), der bei einem Event zu einem Ausführen einer gewünschten Aktion führen soll.
- [MB_EXECUTION_MODE](#), Seite 84, einmalige oder zyklische Abarbeitung des FlexProfile,
- [MB_FLEX_EVENT](#), Seite 85, Konfiguration von Bedingungen, die an einem vorgegebenen Schritte eines FlexProfiles eingebunden werden können,
- [MB_MASTER_TYPE](#), Seite 85, Abhängigkeit des Bewegungsschrittes, aktuelle Masterachse (MASTER_AXIS0) oder Zeit (MASTER_TIME),
- [MB_PROFILE_START_POINT](#), Seite 86, beschreibt, wie bei einem Profilstart oder Profilwechsel der Ursprung des Profils zu interpretieren ist; er kann sowohl für die Leit- als auch Folgeachse auf die aktuelle Position oder den Nullpunkt bezogen werden,
- [MB_PROFILE_STEP](#), Seite 87, Konfiguration eines Bewegungsschritts in einem FlexProfile.
- [MB_STEP_TYPE](#), Seite 88, Eigenschaft des Bewegungsschrittes, starrer (STEP_FIX) Bewegungsschritt oder Flex-Bewegungsschritt.
- [MB_SYNC_TYPE](#), Seite 88, Aufsynchonisier-Richtung der Achse und Art des Umschaltens.

MB_CHANGESTEP_MODE

Kurzbeschreibung Der Aufzählungsdatentyp MB_CHANGESTEP_MODE beschreibt die zugelassenen Änderungsmöglichkeiten innerhalb des FlexProfils.

Bibliothek	Bereich
ML_Base.library	FlexTypes

Tab. 4-3: Bibliothekszuordnung

Funktionsbeschreibung

Programm:

```

TYPE MB_CHANGESTEP_MODE :
(
  CHANGESTEP_EDIT      := 0, (* Bewegungsschritt bearbeiten*)

```

```

CHANGESTEP_REMOVE      := 1, (* Bewegungsschritt entfernen*)
CHANGESTEP_APPEND      := 2, (* Bewegungsschritt anfügen *)
CHANGESTEP_INSERT_BEFORE := 3, (* Bewegungsschritt Einfügen bevor*)
CHANGESTEP_INSERT_AFTER  := 4 (* Bewegungsschritt Einfügen danach*)
);
END_TYPE

```

MB_EVENT_ACTION

Kurzbeschreibung Der Aufzählungsdatentyp MB_EVENT_ACTION beschreibt die gewünschte Aktionen die durch ein Event ausgelöst werden soll.

Bibliothek	Bereich
ML_Base.library	FlexType

Tab. 4-4: Bibliothekszuordnung

Funktionsbeschreibung

Programm:

```

TYPE MB_EVENT_ACTION :
(
  ACTION_STATUSBIT0_INSTANT      := 16#0000, (* Bit0 sofort setzen *)
  ACTION_STATUSBIT1_INSTANT      := 16#0001, (* Bit1 sofort setzen *)
  ACTION_STATUSBIT2_INSTANT      := 16#0002, (* Bit2 sofort setzen *)
  ACTION_STATUSBIT3_INSTANT      := 16#0003, (* Bit3 sofort setzen *)
  ACTION_STATUSBIT4_INSTANT      := 16#0004, (* Bit4 sofort setzen *)
  ACTION_STATUSBIT5_INSTANT      := 16#0005, (* Bit5 sofort setzen *)
  ACTION_STATUSBIT6_INSTANT      := 16#0006, (* Bit6 sofort setzen *)
  ACTION_STATUSBIT7_INSTANT      := 16#0007, (* Bit7 sofort setzen *)
  ACTION_STATUSBIT8_INSTANT      := 16#0008, (* Bit8 sofort setzen *)
  ACTION_STATUSBIT9_INSTANT      := 16#0009, (* Bit9 sofort setzen *)
  ACTION_STATUSBIT10_INSTANT     := 16#000A, (* Bit10 sofort setzen *)
  ACTION_STATUSBIT11_INSTANT     := 16#000B, (* Bit11 sofort setzen *)
  ACTION_STATUSBIT12_INSTANT     := 16#000C, (* Bit12 sofort setzen *)
  ACTION_STATUSBIT13_INSTANT     := 16#000D, (* Bit13 sofort setzen *)
  ACTION_STATUSBIT14_INSTANT     := 16#000E, (* Bit14 sofort setzen *)
  ACTION_STATUSBIT15_INSTANT     := 16#000F, (* Bit15 sofort setzen *)

  ACTION_STATUSBIT0_AFTERSTEP    := 16#0100, (* Bit0 am Ende des Schritts setzen *)
  ACTION_STATUSBIT1_AFTERSTEP    := 16#0101, (* Bit1 am Ende des Schritts setzen *)
  ACTION_STATUSBIT2_AFTERSTEP    := 16#0102, (* Bit2 am Ende des Schritts setzen *)
  ACTION_STATUSBIT3_AFTERSTEP    := 16#0103, (* Bit3 am Ende des Schritts setzen *)
  ACTION_STATUSBIT4_AFTERSTEP    := 16#0104, (* Bit4 am Ende des Schritts setzen *)
  ACTION_STATUSBIT5_AFTERSTEP    := 16#0105, (* Bit5 am Ende des Schritts setzen *)
  ACTION_STATUSBIT6_AFTERSTEP    := 16#0106, (* Bit6 am Ende des Schritts setzen *)
  ACTION_STATUSBIT7_AFTERSTEP    := 16#0107, (* Bit7 am Ende des Schritts setzen *)
  ACTION_STATUSBIT8_AFTERSTEP    := 16#0108, (* Bit8 am Ende des Schritts setzen *)
  ACTION_STATUSBIT9_AFTERSTEP    := 16#0109, (* Bit9 am Ende des Schritts setzen *)
  ACTION_STATUSBIT10_AFTERSTEP   := 16#010A, (* Bit10 am Ende des Schritts setzen *)
  ACTION_STATUSBIT11_AFTERSTEP   := 16#010B, (* Bit11 am Ende des Schritts setzen *)
  ACTION_STATUSBIT12_AFTERSTEP   := 16#010C, (* Bit12 am Ende des Schritts setzen *)
  ACTION_STATUSBIT13_AFTERSTEP   := 16#010D, (* Bit13 am Ende des Schritts setzen *)
  ACTION_STATUSBIT14_AFTERSTEP   := 16#010E, (* Bit14 am Ende des Schritts setzen *)
  ACTION_STATUSBIT15_AFTERSTEP   := 16#010F, (* Bit15 am Ende des Schritts setzen *)

  ACTION_STATUSBIT0_AFTERPROFILE := 16#0200, (* Bit0 am Ende des Profils setzen *)
  ACTION_STATUSBIT1_AFTERPROFILE := 16#0201, (* Bit1 am Ende des Profils setzen *)
  ACTION_STATUSBIT2_AFTERPROFILE := 16#0202, (* Bit2 am Ende des Profils setzen *)
  ACTION_STATUSBIT3_AFTERPROFILE := 16#0203, (* Bit3 am Ende des Profils setzen *)
  ACTION_STATUSBIT4_AFTERPROFILE := 16#0204, (* Bit4 am Ende des Profils setzen *)
  ACTION_STATUSBIT5_AFTERPROFILE := 16#0205, (* Bit5 am Ende des Profils setzen *)
  ACTION_STATUSBIT6_AFTERPROFILE := 16#0206, (* Bit6 am Ende des Profils setzen *)
  ACTION_STATUSBIT7_AFTERPROFILE := 16#0207, (* Bit7 am Ende des Profils setzen *)
  ACTION_STATUSBIT8_AFTERPROFILE := 16#0208, (* Bit8 am Ende des Profils setzen *)
  ACTION_STATUSBIT9_AFTERPROFILE := 16#0209, (* Bit9 am Ende des Profils setzen *)
  ACTION_STATUSBIT10_AFTERPROFILE := 16#020A, (* Bit10 am Ende des Profils setzen *)
  ACTION_STATUSBIT11_AFTERPROFILE := 16#020B, (* Bit11 am Ende des Profils setzen *)
  ACTION_STATUSBIT12_AFTERPROFILE := 16#020C, (* Bit12 am Ende des Profils setzen *)
  ACTION_STATUSBIT13_AFTERPROFILE := 16#020D, (* Bit13 am Ende des Profils setzen *)
  ACTION_STATUSBIT14_AFTERPROFILE := 16#020E, (* Bit14 am Ende des Profils setzen *)
  ACTION_STATUSBIT15_AFTERPROFILE := 16#020F, (* Bit15 am Ende des Profils setzen *)

  ACTION_STATUSBIT0_INSTANT_NEXTSTEP := 16#0300, (* Bit0 sofort setzen und *)
                                           (* sofort nächsten Schritt anschliessen *)
  ACTION_STATUSBIT1_INSTANT_NEXTSTEP := 16#0301, (* Bit1 sofort setzen und *)

```

ML_Base.library

```

(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT2_INSTANT_NEXTSTEP := 16#0302, (* Bit2 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT3_INSTANT_NEXTSTEP := 16#0303, (* Bit3 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT4_INSTANT_NEXTSTEP := 16#0304, (* Bit4 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT5_INSTANT_NEXTSTEP := 16#0305, (* Bit5 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT6_INSTANT_NEXTSTEP := 16#0306, (* Bit6 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT7_INSTANT_NEXTSTEP := 16#0307, (* Bit7 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT8_INSTANT_NEXTSTEP := 16#0308, (* Bit8 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT9_INSTANT_NEXTSTEP := 16#0309, (* Bit9 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT10_INSTANT_NEXTSTEP := 16#030A, (* Bit10 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT11_INSTANT_NEXTSTEP := 16#030B, (* Bit11 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT12_INSTANT_NEXTSTEP := 16#030C, (* Bit12 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT13_INSTANT_NEXTSTEP := 16#030D, (* Bit13 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT14_INSTANT_NEXTSTEP := 16#030E, (* Bit14 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)
ACTION_STATUSBIT15_INSTANT_NEXTSTEP := 16#030F, (* Bit15 sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)

(* Welches Bit gesetzt wird ergibt sich in diesem Fall automatisch durch die EventNr: *)
ACTION_STATUSBIT_INSTANT := 16#00FF, (* Bit sofort setzen *)
ACTION_STATUSBIT_AFTERSTEP := 16#01FF, (* Bit am Ende des Schritts setzen *)
ACTION_STATUSBIT_AFTERPROFILE := 16#02FF, (* Bit am Ende des Profils setzen *)
ACTION_STATUSBIT_INSTANT_NEXTSTEP := 16#03FF (* Bit sofort setzen und *)
(* sofort nächsten Schritt anschliessen *)

);
END_TYPE
```

MB_EVENT_TRIGGER

Kurzbeschreibung

Der Aufzählungsdatentyp MB_EVENT_TRIGGER beschreibt den Auslöser (Trigger), der bei einem Event zu einem Ausführen einer gewünschten Aktion (MB_EVENT_ACTION, Seite 83,) führen soll.

Bibliothek	Bereich
ML_Base.library	FlexTypes

Tab. 4-5: Bibliothekszuordnung

Funktionsbeschreibung

.

Programm:

```

TYPE MB_EVENT_TRIGGER :
(
    TRIGGER_CONTROLBIT_ONLY := 0, (* Trigger nur auf SPS Signal *)
    TRIGGER_ABS_SLAVE_REACHED := 1, (* Trigger auf Folgeachsposition erreicht*)
    TRIGGER_REL_SLAVE_REACHED := 2, (* Trigger auf Folgeachsweg verfahren *)
    TRIGGER_ABS_MASTER_REACHED := 3, (* Trigger auf Masterposition erreicht *)
    TRIGGER_REL_MASTER_REACHED := 4, (* Trigger auf Masterweg verfahren *)
    TRIGGER_REL_TIME_MASTER_REACHED := 5 (* Trigger auf Zeit verstrichen *)
);
END_TYPE
```

MB_EXECUTION_MODE

Kurzbeschreibung

Der Aufzählungsdatentyp MB_EXECUTION_MODE beschreibt, ob die Abarbeitung des FlexProfiles einmalig oder zyklisch durchgeführt werden soll.

Bibliothek	Bereich
ML_Base.library	FlexType

Tab. 4-6: Bibliothekszuordnung

Funktionsbeschreibung*Programm:*

```

TYPE MB_EXECUTION_MODE:

    EXECUTE_CYCLIC := 0, (* Zyklische Abarbeitung *)
    EXECUTE_SINGLE := 1 (* Einmalige Profil Abarbeitung *)

);
END_TYPE

```

MB_FLEX_EVENT**Kurzbeschreibung**

Der Aufzählungsdatentyp MB_FLEX_EVENT beschreibt die Konfiguration von Bedingungen, die an einem vorgegebenen Schritte eines FlexProfils eingebunden werden können

Bibliothek	Bereich
ML_Base.library	FlexType

Tab. 4-7: Bibliothekszuordnung

Funktionsbeschreibung*Programm:*

```

TYPE MB_FLEX_EVENT :
STRUCT
    StepNumber: INT; (* Bewegungsschritt *)
    TriggerMode: MB_EVENT_TRIGGER; (* Wahl des Auslöser-Modus *)
    TriggerValue: LREAL; (* Abhängig nach Modus; zu erreichende Position *)
                        (* oder Verfahrenweg *)
    ActionMode: MB_EVENT_ACTION; (* Wahl des Aktions-Modus *)
END_STRUCT
END_TYPE

```

MB_MASTER_TYPE**Kurzbeschreibung**

Der Aufzählungsdatentyp MB_MASTER_TYPE beschreibt die Abhängigkeit des Bewegungsschrittes. Dies kann entweder die aktuelle Masterachse (MASTER_AXIS0) oder aber auch die Zeit (MASTER_TIME) sein.

Bewegungsschritte, die an einen Zeit-Master gekoppelt sind, laufen unabhängig von der Leitachse und immer in der gewünschten Zeitdauer ab.

Wird als Master die Zeit gewählt, wird der Parameter für den Leitachsabschnitt nicht in Grad sondern als Zeitangabe in Millisekunden interpretiert.

Bibliothek	Bereich
ML_Base.library	FlexTypes

Tab. 4-8: Bibliothekszuordnung

Funktionsbeschreibung*Programm:*

```

TYPE MB_MASTER_TYPE :
(
    MASTER_TIME := 0, (* Bewegungsschritt mit Zeit-Master *)
    MASTER_AXIS0 := 1 (* Bewegungsschritt mit Achse als Master *)
)

```

ML_Base.library

```
);  
END_TYPE
```

MB_PROFILE_START_POINT

Kurzbeschreibung Der Aufzählungsdatentyp MB_PROFILE_START_POINT beschreibt, wie bei einem Profilstart oder Profilwechsel der Ursprung des Profils zu interpretieren ist. Dieser kann sowohl für die Leit- als auch Folgeachse auf die aktuelle Position oder den Nullpunkt bezogen werden.

Bibliothek	Bereich
ML_Base.library	FlexType

Tab. 4-9: Bibliothekszuordnung

Funktionsbeschreibung

Programm:

```
TYPE MB_PROFILE_START_POINT:  
(  
  SLAVE_ORIGIN_MASTER_ORIGIN:  = 0, (* Absoluter Einstieg in das Profil *)  
  SLAVE_CURRENT_MASTER_CURRENT: = 1, (* Profil startet an der aktuellen Leit- und *)  
                                  (* Folgeachsposition *)  
  SLAVE_ORIGIN_MASTER_CURRENT:  = 2, (* Profil startet an der aktuellen Leitachs- *)  
                                  (* position und an der absoluten Folgeachsposition *)  
  SLAVE_CURRENT_MASTER_ORIGIN:  = 3, (* Profil startet an der aktuellen Folgeachs- *)  
                                  (* position und an der absoluten Leitachsposition*)  
  SLAVE_RELATIVE_MASTER_ORIGIN: = 4, (* Profil startet an der aktuellen Leitachs- *)  
                                  (* position. Die Folgeachsposition wird so *)  
                                  (* bestimmt, so dass die Achse relativ verfahren wird.*)  
)  
END_TYPE
```

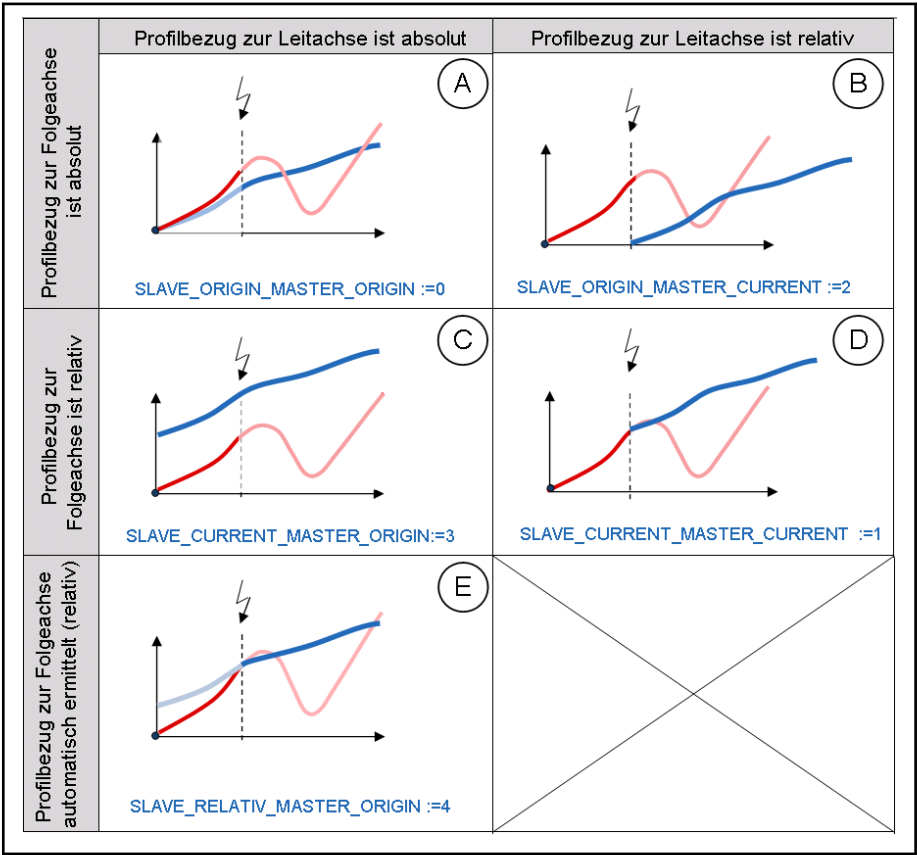


Abb. 4-1: Funktionsbeschreibung



Beschreibt das Bewegungsprofil einen festen Positionsbezug zwischen Leitachse und Folgeachse, so eignet sich zum Aufsynchronisieren am besten der Modus A (SLAVE_ORIGIN_MASTER_ORIGIN). Spielt ein Phasenversatz der Folgeachse keine Rolle und soll die Folgeachse beim Aufsynchronisieren mit stehender Leitachse keine Bewegung ausführen, dann eignet sich hierfür Modus E (SLAVE_RELATIVE_MASTER_ORIGIN). Der Profileinstieg erfolgt dann relativ.



Bei den Modi B (SLAVE_ORIGIN_MASTER_CURRENT) und D (SLAVE_CURRENT_MASTER_CURRENT) wird das Bewegungsprofil immer von dessen Anfang aus abgearbeitet. Dies ist hilfreich, wenn ein fester Leitachsbezug nicht nötig ist. Auf diese Weise lassen sich beispielsweise Vorschübe oder Relativbewegungen von der aktuellen Achsposition aus realisieren.



Am Baustein ML_FlexProfile lässt sich über die Eingänge SlaveOffset und MasterOffset die Lage und der Start des Bewegungsprofils weiter beeinflussen. Über den SlaveOffset wird ein fixer Phasenoffset auf die Folgeachsposition addiert (Verschiebung des Profils an der Y-Achse). Mit Hilfe des MasterOffset kann der Profileinstieg an einem späteren Punkt im Profil erfolgen (Verschiebung des Profils an der X-Achse nach links).

MB_PROFILE_STEP

Kurzbeschreibung

Die Struktur MB_PROFILE_STEP beschreibt die Konfiguration eines Bewegungsschritts in einem FlexProfile.

Mehrere Bewegungsschritte hintereinander ergeben einen Bewegungssatz.

Diese können als FlexProfile-Bewegungssatz mit Hilfe des Bausteins MB_ChangeFlexProfileSet auf die Achse geschrieben und schließlich mit dem Motion-Baustein ML_FlexProfile gestartet werden.

Bibliothek	Bereich
ML_Base.library	FlexTypes

Tab. 4-10: Bibliothekszuordnung

Funktionsbeschreibung

Programm:

```

TYPE MB_PROFILE_STEP :
STRUCT
    Master:MB_MASTER_TYPE := MASTER_AXIS0; (* Auswahl der Leitachsquellen: *)
    (* Zeit oder Position*)
    Hub:LREAL; (* Hub *)
    Range:LREAL; (* Definition des Leitachsabschnitts*)
    MotionLaw:MC_STEP_MODE; (* Definition des Bewegungsgesetzes *)
    StepType:MB_STEP_TYPE := STEP_FIX; (* Schrittyp: Standard(Fix) oder Flex *)
    StartVel: LREAL := 0.0; (* Folgeachsanfangsgeschwindigkeit*)
    StartAcc: LREAL := 0.0; (* Folgeachseanfangsbeschleunigung*)
    StartJrk: LREAL := 0.0; (* Folgeachseanfangsruck*)
    EndVel: LREAL := 0.0; (* Folgeachsendgeschwindigkeit*)
    EndAcc: LREAL := 0.0; (* Folgeachsendbeschleunigung*)
    EndJrk: LREAL := 0.0; (* Folgeachsendruck*)
    TravelVel:LREAL := 0.0; (* Folgeachsverfahrensgeschwindigkeit,*)
    (* nicht von allen Bewegungsgesetzen *)
    (* berücksichtigt *)
    LimitAcc: LREAL := 0.0; (* Folgeachsverfahrensbeschleunigung, *)
    (* nicht von allen Bewegungsgesetzen *)
    (* berücksichtigt *)

```

ML_Base.library

```

LimitJrk: LREAL      := 0.0;      (* Folgeachsverfahrruck nicht von allen *)
                                (* Bewegungsgesetzen berücksichtigt*)
TurningPointDisplacement: LREAL := 0.5; (* Wendepunkteverschiebung für symmetrische *)
                                (* Bewegungsgesetze *)

```

```

END_STRUCT
END_TYPE

```

MB_STEP_TYPE

Kurzbeschreibung Der Aufzählungsdatentyp MB_STEP_TYPE beschreibt die Eigenschaft des Bewegungsschrittes. Grundsätzlich lässt sich zwischen starren (STEP_FIX) Bewegungsschritten und Flex-Bewegungsschritten unterscheiden. Die Flex-Bewegungsschritte lassen sich weiter hinsichtlich ihrer Leit- und Folgeachs-bezüge in vier Untertypen klassifizieren .

Bibliothek	Bereich
ML_Base.library	FlexTypes

Tab. 4-11: Bibliothekszuordnung

Typdefinition

```

TYPE MB_STEP_TYPE :
(
STEP_FIX      := 0, (*Standardschritt*)
STEP_FLEX_REL_REL := 1, (*Flex-Schritt mit relativem Hub und relativem Leitachsabschnitt*)
STEP_FLEX_REL_ABS := 2, (*Flex-Schritt mit relativem Hub und absoluter Leitachsposition*)
STEP_FLEX_ABS_REL := 3, (*Flex-Schritt mit absoluter Folgeachsposition und relativem Leitachsabschnitt*)
STEP_FLEX_ABS_ABS := 4, (*Flex-Schritt mit absoluter Folgeachsposition und absoluter Leitachsposition*)
);
END_TYPE

```

Ein fixer Bewegungsschritt entspricht dem aus anderen Kurvenscheibenlösungen bekannten Bewegungsschritt, der durch seinen Hub und Leitachsabschnitt in seinen Abmessungen fest definiert ist. Weiterhin werden seine linken und rechten Randwerte für Geschwindigkeit, Beschleunigung und ggf. Ruck fest vorgegeben.

MB_SYNC_TYPE

Kurzbeschreibung Der Aufzählungsdatentyp MB_SYNC_TYPE legt sowohl die Aufsynchronisierungsrichtung der Achse als auch die Art des Umschaltens fest.

Bibliothek	Bereich
ML_Base.library	FlexTypes

Tab. 4-12: Bibliothekszuordnung

Funktionsbeschreibung

.

Programm:

```

TYPE MB_SYNC_TYPE:
(
SYNC_RAMPIN_SHORTEST_WAY := 0, (* weiches Umschalten kürzeste Weg *)
SYNC_RAMPIN_CATCH_UP     := 1, (* weiches Umschalten positive Richtung*)
SYNC_RAMPIN_SLOW_DOWN    := 2, (* weiches Umschalten negative Richtung*)
SYNC_DIRECT              := 3, (* hartes Umschalten *)
);
END_TYPE

```

4.2.3 Data Types/ Read Write Parameter - Parameterzugriff

MB_SERCOS_DATA_STATUS

Kurzbeschreibung Der Status eines vom Anwender gewählten Parameters wird durch den Funktionsbaustein MB_ReadSercosDataStatus, ML_Plcopen.library, mittels ENUM-Typ "MB_SERCOS_DATA_STATUS" als Klartext lesbar angezeigt.

Bibliothek	Bereich
ML_Base.library	RDWR_Parameter

Tab. 4-13: Bibliothekszuordnung

Programm:

```

TYPE MB_SERCOS_DATA_STATUS:
(* Status vom Parameter *)
(
  MB_SERCOS_DATA_NOT_SET      := 0,      (* Not set and not enabled *)
  MB_SERCOS_DATA_SETTING     := 1,      (* Set and enabled *)
  MB_SERCOS_DATA_END_OK      := 3,      (* Command correctly executed *)
  MB_SERCOS_DATA_INTERRUPTED := 5,      (* Command execution interrupted *)
  MB_SERCOS_DATA_IN_PROCESS  := 7,      (* In process *)
  MB_SERCOS_DATA_BREAK_ERROR := 15,     (* Error, command execution impossible *)
  MB_SERCOS_DATA_DATUM_FAULT := 256     (* Data invalid *)
);
END_TYPE

```



Der Datenstatus "MB_SERCOS_DATA_DATUM_FAULT" besagt, dass der Wert des Parameters ausserhalb seiner Gültigkeitsgrenzen liegt, d.h. kleiner als sein Minimalwert oder größer als sein zulässiger Maximalwert ist.

4.2.4 Data Types/ Shared Memory - Direktzugriff

Datentypen zur Unterstützung des Shared Memory zwischen Motion und SPS

- [ML_AXISDATA_SM](#), Seite 117, (STRUCT), Achsdaten über Shared Memory.
- [ML_CONTROLDATA_SM](#), Seite 119, (STRUCT), Steuerungsdaten über Shared Memory.
- [ML_LINK](#), Seite 121, , Gültigkeitsbit der Steuerung im MLC-Verbund.
- [ML_PLSDATA_SM](#), Seite 120, (STRUCT), Nockenschaltwerk über Shared Memory.
- [ML_PROBEDATA_SM](#), Seite 120, (STRUCT), Messtasterdaten über Shared Memory.
- [ML_SYNC](#), Seite 121, enthält die vier Eingangsbits der Steuerung, die im MLC-Verbund ausgetauscht werden.
- [SM_EXT_TYPES](#), Seite 122, (STRUCT), zur typrichtigen Interpretation von Daten.
- [SM_TYPES](#), Seite 122, (STRUCT), zur typrichtigen Interpretation von Daten.

Achsdaten über Shared Memory und Funktionsvariablenzugriff

Achsdaten über Shared Memory und Funktionsvariablenzugriff, Allgemeines

Auf die in den A-Parametern gespeicherten Achsdaten kann von der SPS aus entweder über eine vereinfachte, blockierende Anweisung "Funktionsvariablenzugriff" (alias "Direktvariable") oder über Shared Memory zugegriffen werden.

Achsdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Die Istwerte der Achsdaten im Shared Memory werden stets am Ende der Bearbeitung des Motion-Kernels MK1 aufgefrischt, die Sollwerte werden stets am Ende der Bearbeitung des Motion-Kernels MK2 übertragen.

ML_Base.library

Achsdaten über Funktionsvariablenzugriff werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.

Ein Teil der Achsdaten stehen sowohl im Shared Memory als auch über Funktionsvariablenzugriff zur Verfügung. Da Achsdaten im Shared Memory unabhängig von der Nutzung zyklisch ausgetauscht werden, sollten weitere redundante Zugriffe über Funktionsvariablenzugriff 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 puffern

Achsdaten 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 der SPS-Tasks.

Der vereinfachte, blockierende Zugriff auf eine Funktionsvariable ist ein impliziter Funktionsaufruf, welche 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.

Achsdaten können ferner über einen C2C-Verbund zwischen mehreren Steuerungen ausgetauscht werden.

Auf die im C2C-Verbund synchronisierten Betriebsdaten dieser Parameter kann über eine vereinfachte, blockierende Anweisung "Funktionsvariablenzugriff" (alias "Direktvariable") zugegriffen werden.

Shared Memory

Achsdaten im Shared Memory sind als globales Array „AxisData[]“ der Datenstruktur [ML_AXISDATA_SM](#), [Seite 91](#), deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Die Achsen sind durchnummeriert, der Achsname der einzelnen Achse ist für diese Zugriffsart bedeutungslos.¹⁾

Achsdaten im Shared Memory:

```
VAR_GLOBAL
```

```
...
AxisData AT %MBaaa: ARRAY[MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_AXISDATA_SM;
...
END_VAR
```

¹⁾ MB_MAX_AXIS_NUMBER, maximal zulässige Achsanzahl, z.Z. 64, steuerungs-
typabhängig.



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.

xxx: mlc | xlc

yyy: l65 | l45 | l25

Funktionsvariablenzugriff

Funktionsvariablen der Achsdaten sind als globales Array „DV_Axis[]“ des Funktionsbausteins "ML_DirectVarAxis" deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“, siehe [DV_Axis\[\]](#), Seite 94.

Funktionsvariablen der Achsdaten:

```
VAR_GLOBAL
```

```
...
  DV_Axis: ARRAY [MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_DirectVarAxis;
...
END_VAR
```



Steht für einen Parameter der Funktionsvariablenzugriff zur Verfügung, so ist dies bei der entsprechenden Parameterbeschreibung vermerkt (Kopf der Parameterbeschreibung, Zeile Direktzugriff, Bsp. A-0-0000).

Funktionsvariablenzugriff synchron im C2C Verbund

Funktionsvariablen synchron im C2C Verbund der Achsdaten sind als globales Array „GV_LinkAxis[]“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“, siehe [GV_LinkAxis\[\]](#), Seite 96.

Funktionsvariablen synchron im C2C Verbund der Achsdaten:

```
VAR_GLOBAL
```

```
...
  GV_LinkAxis: ARRAY [MB_MIN_GLOBAL_CONTROL_NUMBER..MB_MAX_GLOBAL_CONTROL_NUMBER,
                     MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_GlobalVarAxis;
...
END_VAR
```



Steht für einen Parameter der Funktionsvariablenzugriff synchron im C2C Verbund zur Verfügung, so ist dies bei der entsprechenden Parameterbeschreibung vermerkt (Kopf der Parameterbeschreibung, Zeile C2C-Zugriff, Bsp. A-0-0021).

AxisData[] - Achsdaten im Shared Memory

Kurzbeschreibung

Beim Erzeugen einer neuen Achse in IndraWorks werden Geräte- und Antriebsadresse abgefragt, weiterhin wird der Achse automatisch eine Achsnummer zugeordnet.

Mit Hilfe der automatisch erzeugten Achsnummer kann indiziert auf Achsdaten

- über Funktionsvariablenzugriff („DV_Axis[]“),
 - wenn ihre Steuerung die Achse für einen C2C-Verbund freigegeben hat, über Funktionsvariablenzugriff („GV_LinkAxis[]“) und
 - Shared Memory („AxisData[]“)
- zugegriffen werden.

ML_Base.library



Achsdaten im Shared Memory werden im Motion Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Achsdaten als Direktvariablen werden unabhängig vom Motion Modus zum Zeitpunkt des Zugriffs aktualisiert.

Achsdaten über Shared Memory und Funktionsvariablen (alias "Direktvariablen") können sich, abhängig von den Taskeigenschaften der POU, über mehrere Zyklen der Motion (Sollwertgenerierung, Istwertbereitstellung) verteilen und sind somit nicht zwingend konsistent zueinander.

Achsdaten im Shared Memory (Merkerbereich) sind als globales Array „AxisData[]“ der Datenstruktur „ML_AXISDATA_SM“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Definition des globalen Array "AxisData[]"

```
VAR_GLOBAL
```

```
...
AxisData AT %MBaaa: ARRAY[MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_AXISDATA_SM;
...
END_VAR
```



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.

xxx: mlc | xlc

yyy: l65 | l45 | l25

Siehe auch

- [ML_AXISDATA_SM, Seite 117.](#)

SPS-Zugriff auf Parameterwerte über Shared Memory

Der lesende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Lesender Zugriff auf Daten im Shared Memory:

```
MyVar:= AxisData[MyAxisName.AxisNo].FlexProfil_Active;
MyVar:= AxisData[Index].FlexProfil_Active;
MyVar:= AxisData[1].FlexProfil_Active;
```

Der schreibende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Schreibende Zugriff auf Daten im Shared Memory:

```
AxisData[MyAxisName.AxisNo].dwFlexProfileEventControlBits_q:= MyVar;
AxisData[Index].dwFlexProfileEventControlBits_q:= MyVar;
AxisData[1].dwFlexProfileEventControlBits_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.

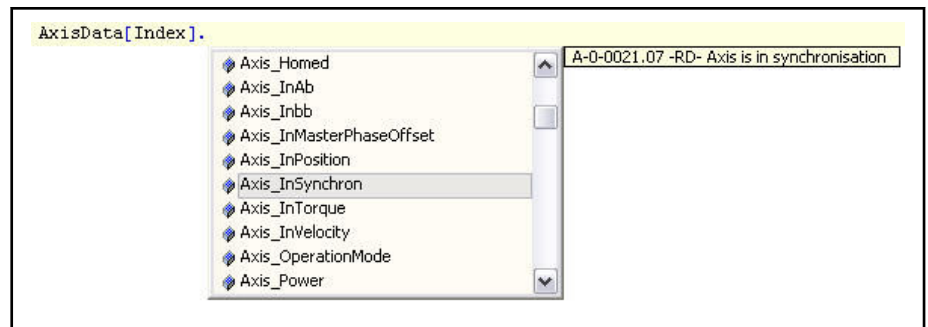


Abb. 4-2: IntelliSense einer Variablen im Shared Memory



Auf Variablen einer Achse im Shared Memory kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Benutzerdefinierte Bezeichnung durch Referenzierung:

```
VAR
  MyAxisName: REFERENCE TO ML_AXISDATA_SM:= AxisData[MyAxisName.AxisNo];
END_VAR
```

```
MyVar:= MyAxisName.FlexProfil_Active;
MyAxisName.dwFlexProfileEventControlBits_q:= MyVar;
```

Ausgewählte Bereiche der Datenstruktur ML_AXISDATA_SM:

```
dwUserCmdDataA_q :SM_TYPES; (* A-0-0501 -RW- Container user defined commanded data A *)
dwUserCmdDataB_q :SM_TYPES; (* A-0-0503 -RW- Container user defined commanded data B *)
dwUserCmdDataC_q :SM_TYPES; (* A-0-0505 -RW- Container user defined commanded data C *)
dwUserCmdDataD_q :SM_TYPES; (* A-0-0507 -RW- Container user defined commanded data D *)
dwUserCmdDataE_q :SM_TYPES; (* A-0-0509 -RW- Container user defined commanded data E *)

dwUserActualDataA_i :SM_TYPES; (* A-0-0521 -RD- Container user defined actual data A *)
dwUserActualDataB_i :SM_TYPES; (* A-0-0523 -RD- Container user defined actual data B *)
dwUserActualDataC_i :SM_TYPES; (* A-0-0525 -RD- Container user defined actual data C *)
dwUserActualDataD_i :SM_TYPES; (* A-0-0527 -RD- Container user defined actual data D *)
dwUserActualDataE_i :SM_TYPES; (* A-0-0529 -RD- Container user defined actual data E *)
dwUserActualDataF_i :SM_TYPES; (* A-0-0531 -RD- Container user defined actual data F *)
```

Die Datencontainer dwUserCmdData*_q und dwUserActualData*_i enthalten Informationen des "Zyklischen SERCOS-Kanals".

Abhängig von der Konfiguration können die Container Werte mit unterschiedlichen Datentypen enthalten.



Die Datentypen zu übertragender A-Parameter sind in der Parameterbeschreibung (Attribute, SPS-Datentyp) enthalten.

Für S- und P-Parameter sind die SPS-Datentypen aus den Attributen des jeweiligen Parameters zu ermitteln:

- Nachkommastellen ≠ 0, Typ "REAL"
- Nachkommastellen = 0 und FLOAT, Typ "REAL"
- Nachkommastellen = 0 und
 - 4 Byte mit VZ, "DINT"
 - 2 Byte mit VZ, "INT"
 - 4 Byte ohne VZ, "UDINT"
 - 2 Byte ohne VZ, "UINT"

ML_Base.library

Auf den entsprechenden Datentyp kann über den Punktoperator zugegriffen werden.

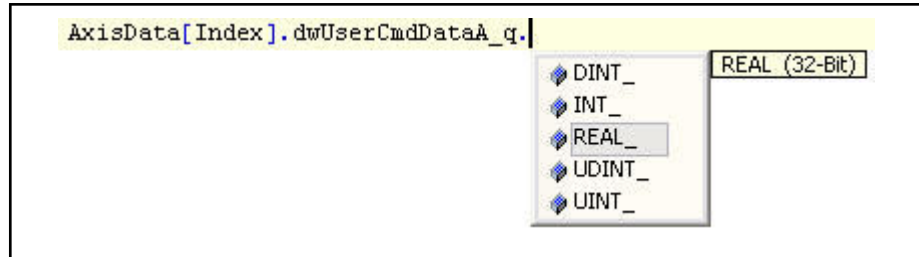


Abb. 4-3: Vorgabe des Datentyps über Punktoperator

Siehe auch

- [SM_TYPES](#), Seite 122,
- [ML_AXISDATA_SM](#), Seite 117.

DV_Axis[] - Achsdaten über Funktionsvariablenzugriff

Kurzbeschreibung

Beim Erzeugen einer neuen Achse in IndraWorks werden Gerätenamen sowie Antriebsadresse abgefragt, weiterhin wird der Achse automatisch eine Achsnummer `MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER` ([GVL_Base-Constant](#), Seite 130) zugeordnet.

Mit Hilfe der automatisch erzeugten Achsnummer kann indiziert auf Achsdaten

- über Funktionsvariablenzugriff („DV_Axis[]“),
- wenn ihre Steuerung die Achse für einen C2C-Verbund freigegeben hat, über Funktionsvariablenzugriff („GV_LinkAxis[]“) und
- Shared Memory („AxisData[]“)

zugegriffen werden.



Achsdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt.

Achsdaten über Funktionsvariablenzugriff werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.

Achsdaten 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.

Achsdaten über Funktionsvariablenzugriff sind als globales Array „DV_Axis[]“ des Funktionsbausteins „ML_DirectVarAxis“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Auf die Achsdaten (Betriebsdatum des Parameters) wird über eine vereinfachte, blockierende Anweisung "Funktionsvariablenzugriff" (alias "Direktvariable") zugegriffen werden. Ein Funktionsvariablenaufruf erfolgt blockierend und kann je nach Ausprägung, beispielsweise des Achstyps, unterschiedliche Zeiträume benötigen.

Der Funktionsbaustein „ML_DirectVarAxis“ 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 Achs-typ und Motion-Modus und erzeugt im negativen Fall keine Fehlermeldung.



Der Funktionsbaustein „ML_DirectVarAxis“ ist nicht für die direkte Verwendung im Anwenderprogramm vorgesehen.

Auszug aus GVL_Base

```
VAR_GLOBAL
...
  DV_Axis: ARRAY [MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_DirectVarAxis;
...
END_VAR
```

SPS-Zugriff auf Parameterwerte mit Funktionsvariablenzugriff

Der lesende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine GET Property (READ) verfügt.

Lesender Zugriff auf eine Funktionsvariable

```
MyVar:= DV_Axis[MyAxisName.AxisNo].A_0_0004;
MyVar:= DV_Axis[Index].A_0_0004;
MyVar:= DV_Axis[1].A_0_0004;
```

Der schreibende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine SET Property (WRITE) verfügt.

Schreibender Zugriff auf eine Funktionsvariable

```
DV_Axis[MyAxisName.AxisNo].A_0_0216:= MyVar;
DV_Axis[Index].A_0_0216:= MyVar;
DV_Axis[1].A_0_0216:= MyVar;
```

Der IntelliSense einer Funktionsvariablen enthält die Parameteridentnummer, die Zugriffsmöglichkeiten (READ bzw. READ/WRITE), den Parameternamen sowie die Methode zur Ermittlung des aktuellen Zugriffsstatus.

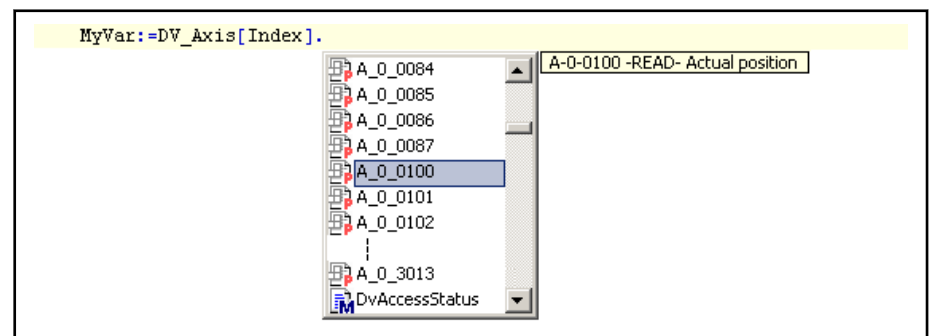


Abb. 4-4: Funktionsvariablenzugriff unter Nutzung der IntelliSense-Funktionalität



Auf Funktionsvariablen einer Achse kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Zugriff über REFERENCE_TO

```
VAR
  MyAxisName: REFERENCE TO ML_DirectVarAxis:= DV_Axis[MyAxisName.AxisNo];
END_VAR
```

ML_Base.library

```
MyVar:= MyAxisName.A_0_0004;
MyAxisName.A_0_0216:= MyVar;
```

Status bei Funktionsvariablenzugriff

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

```
A_Status:= DV_Axis[MyAxisName.AxisNo].DvAccessStatus(FP_A_0_0000);
C_Status:= DV_Control[0].DvAccessStatus(FP_C_0_0023);
M_Status:= DV_Probe[MB_MIN_PROBE_NUMBER].DvAccessStatus(FP_M_0_0021);
N_Status:= DV_Pls[Index].DvAccessStatus(FP_N_0_0303);
O_Status:= DV_Osci[Index].DvAccessStatus(FP_O_0_0021);
```

Vereinbarungsdateien für Globale Konstanten

- [GVL_A_Param_ID, Seite 134,](#)
- [GVL_C_Param_ID, Seite 134,](#)
- [GVL_M_Param_ID, Seite 135,](#)
- [GVL_N_Param_ID, Seite 135,](#)
- [GVL_O_Param_ID, Seite 136,](#)

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).

GV_LinkAxis[] - Achsdaten über Funktionsvariablenzugriff synchron im C2C-Verbund**Kurzbeschreibung**

Beim Erzeugen einer neuen Achse in IndraWorks werden Geräte- und Achsenadresse abgefragt, weiterhin wird der Achse automatisch eine Achsnummer zugeordnet.

Arbeitet die Steuerung in einem C2C-Verbund kann über "Funktionsvariablenzugriff synchron im C2C-Verbund" mit Hilfe der automatisch erzeugten Achsnummer indiziert auf Achsdaten („GV_LinkAxis[]“) der eigenen und fremder Steuerungen zugegriffen werden.

Achsdaten über "Funktionsvariablenzugriff synchron im C2C-Verbund" sind als globales Array „GV_LinkAxis[]“ des Funktionsbausteins „ML_GlobalVarAxis“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Auf die im C2C-Verbund synchronisierten Achsdaten (Betriebsdatum des jeweiligen Parameters) kann über eine vereinfachte, blockierende Anweisung "Funktionsvariablenzugriff" (alias "Direktvariable") zugegriffen werden. Ein

Funktionsvariablenaufruf erfolgt blockierend und kann je nach Ausprägung unterschiedliche Zeiträume benötigen.

Der Funktionsbaustein „ML_GlobalVarAxis“ 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 erzeugt im negativen Fall keine Fehlermeldung.



Der Funktionsbaustein „ML_GlobalVarAxis“ ist nicht für die direkte Verwendung im Anwenderprogramm vorgesehen.

Auszug aus GVL_Base

```
VAR_GLOBAL
...
GV_LinkAxis: ARRAY[MB_MIN_GLOBAL_CONTROL_NUMBER..MB_MAX_GLOBAL_CONTROL_NUMBER,
                  MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_GlobalVarAxis;
...
END_VAR
```

SPS-Zugriff auf Parameterwerte mit Funktionsvariablenzugriff syn- chron im C2C-Verbund

Der lesende Zugriff auf eine Funktionsvariable synchron im C2C-Verbund kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine GET Property (READ) verfügt.

Programm:

```
MyVar:= GV_LinkAxis[CtrlNo, AxisNo].A_0_0021;
MyVar:= GV_LinkAxis[0, 1].A_0_0021;
```

Der indizierte, blockierende Zugriff mit CtrlNo gleich LOCAL_CTRL greift auf die im C2C Verbund synchronisierte Betriebsdaten der lokalen Steuerung zu. Dieser Zugriff ist äquivalent zum indizierten, blockierenden Zugriff mit CtrlNo gleich C-0-0031.

Der schreibende Zugriff auf eine Funktionsvariable synchron im C2C-Verbund wird nicht unterstützt.

Der IntelliSense einer Funktionsvariablen synchron im C2C-Verbund enthält die Parameteridentnummer, die Zugriffsmöglichkeiten (READ bzw. READ/ WRITE), den Parameternamen sowie die Methode zur Ermittlung des aktuellen Zugriffsstatus.

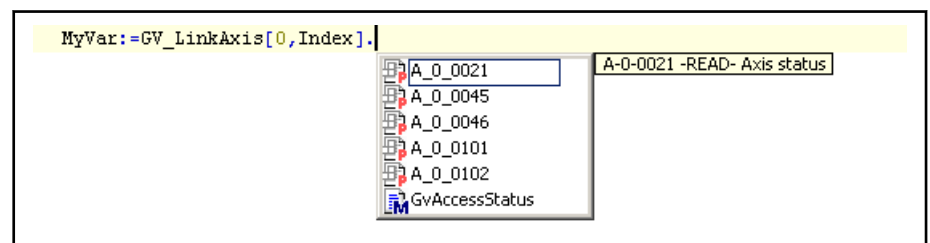


Abb. 4-5: Funktionsvariablenzugriff synchron im C2C-Verbund unter Nutzung der IntelliSense-Funktionalität

ML_Base.library



Auf Funktionsvariablen einer Achse kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Zugriff über REFERENCE_TO

```
VAR
  MyAxisName: REFERENCE TO ML_GlobalVarAxis:= GV_LinkAxis[CtrlNo, AxisNo];
END_VAR

MyVar:= MyAxisName.A_0_0021;
```

Status bei Funktionsvariablenzugriff synchron im C2C-Verbund

Die Funktionsbausteine für "Funktionsvariablenzugriff synchron im C2C-Verbund" bieten neben den Properties zum Lesen (und Schreiben) von Betriebsdaten auch die Möglichkeit, den **aktuellen Zugriffstatus** eines Parameters über die Methode **GvAccessStatus** 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 synchron im C2C-Verbund

```
Status:= GV_LinkAxis[CtrlNo, AxisNo].GvAccessStatus(GV_A_0_0021);
Status:= GV_LinkAxis[0, 1].GvAccessStatus(GV_A_0_0021);
```

GV_A_0_xxxx sind Konstanten der Global Variable List / Global Variable Index / [GVL_A_GlobalVarIdx, Seite 131](#).

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).

Steuerungsdaten über Shared Memory und Funktionsvariablenzugriff

Steuerungsdaten über Shared Memory und Funktionsvariablenzugriff, Allgemeines

Auf die in den C-Parametern gespeicherten Steuerungsdaten kann von der SPS aus entweder über eine vereinfachte, blockierende Anweisung "Funktionsvariablenzugriff" (alias "Direktvariable") oder über Shared Memory zugegriffen werden.

Steuerungsdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Die Istwerte der Steuerungsdaten im Shared Memory werden stets am Ende der Bearbeitung des Motion-Kernels MK1 aufgefrischt, die Sollwerte werden stets am Ende der Bearbeitung des Motion-Kernels MK2 übertragen.

Steuerungsdaten über Funktionsvariablenzugriff werden unabhängig vom MotionModus zum Zeitpunkt des Zugriffs aktualisiert.

Ein Teil der Steuerungsdaten stehen sowohl im Shared Memory als auch über Funktionsvariablenzugriff zur Verfügung. Da Steuerungsdaten im Shared Memory unabhängig von der Nutzung zyklisch ausgetauscht werden, sollten weitere redundante Zugriffe über Direktvariablen aus Performancegründen vermieden werden.



Der vereinfachte, blockierende Zugriff auf eine Funktionsvariable ist ein impliziter Funktionsaufruf und benötigt mehr Laufzeiterourcen 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 puffern

Steuerungsdaten 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 der SPS-Tasks.

Der vereinfachte, blockierende Zugriff auf eine Funktionsvariable ist ein impliziter Funktionsaufruf, welche 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.

Shared Memory

Steuerungsdaten im Shared Memory sind als globales Array „ControlData[]“ der Datenstruktur [ML_CONTROLDATA_SM, Seite 100](#), deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Der Steuerungsname ist für diese Zugriffsart bedeutungslos. Es werden stets die Daten der lokalen Steuerung bereit gestellt.

Steuerungsdaten im Shared Memory

```
VAR_GLOBAL
...
ControlData AT %MBbbb:ARRAY[MB_MIN_CONTROL_NUMBER..MB_MAX_CONTROL_NUMBER]
                                     OF ML_CONTROLDATA_SM;
...
END_VAR
```



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.

xxx: mlc | xlc

yyy: l65 | l45 | l25

Funktionsvariablenzugriff

Funktionsvariablen der Steuerungsdaten sind als globales Array „DV_Control[]“ des Funktionsbausteins „ML_DirectVarControl“ deklariert. Die

ML_Base.library

Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“, siehe [DV_Control\[\]](#), [Seite 101](#).

Funktionsvariablen der Steuerungsdaten

```
VAR_GLOBAL
```

```
...
DV_Control: ARRAY[MB_MIN_CONTROL_NUMBER..MB_MAX_CONTROL_NUMBER] OF ML_DirectVarControl;
...
END_VAR
```



Steht für einen Parameter der Funktionsvariablenzugriff zur Verfügung, so ist dies bei der entsprechenden Parameterbeschreibung vermerkt (Parameterattribute, Direktzugriff, Bsp. C-0-0000).

ControlData[] - Steuerungsdaten im Shared Memory

Kurzbeschreibung

Beim Anlegen eines neuen Projektes in IndraWorks werden automatisch Steuerungsdaten bereitgestellt. Mit Hilfe des fest definierten Index MB_MIN_CONTROL_NUMBER bzw. MB_MAX_CONTROL_NUMBER kann auf Steuerungsdaten über

- Funktionsvariablenzugriff („DV_Control[]“) und
- Shared Memory („ControlData[]“)

zugegriffen werden.



Steuerungsdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Steuerungsdaten als Direktvariablen werden unabhängig vom Motion Modus zum Zeitpunkt des Zugriffs aktualisiert.

Steuerungsdaten im Shared Memory und Funktionsvariablenzugriff können sich, abhängig von den Taskeigenschaften der POU, über mehrere Zyklen der Motion (Sollwertgenerierung, Istwertbereitstellung) verteilen und sind somit nicht zwingend konsistent zueinander.

Steuerungsdaten im Shared Memory sind als globales Array „ControlData[]“ der Datenstruktur „ML_CONTROLDATA_SM“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Definition des globalen Array "ControlData[]"

```
VAR_GLOBAL
```

```
...
ControlData AT %MBbbb:ARRAY[MB_MIN_CONTROL_NUMBER..MB_MAX_CONTROL_NUMBER]
...
END_VAR
```



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.

xxx: mlc | xlc

yyy: l65 | l45 | l25

Funktionsbeschreibung Der lesende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Lesender Zugriff auf Daten

```
MyVar:= ControlData[MB_MIN_CONTROL_NUMBER].LINK_ESTABLISHED;
MyVar:= ControlData[Index].LINK_ESTABLISHED;
MyVar:= ControlData[0].LINK_ESTABLISHED;
```

Der schreibende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Schreibender Zugriff auf Daten

```
ControlData[Index].LINK_REBUILD_RING:= MyVar;
ControlData[1].LINK_REBUILD_RING:= 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.

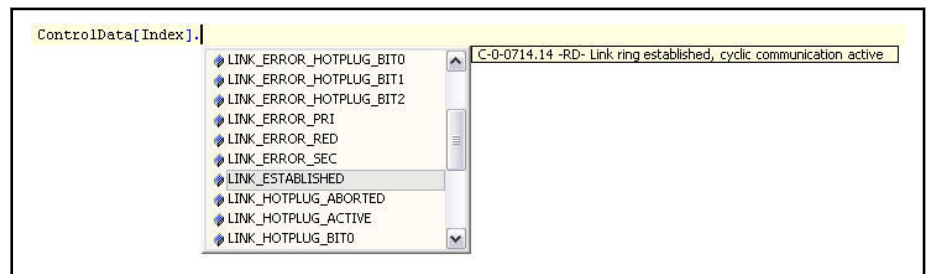


Abb. 4-6: IntelliSense einer Variablen im Shared Memory



Auf Variablen der Steuerung im Shared Memory kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Programm:

```
VAR
    MyControlName: REFERENCE TO ML_CONTROLDATA_SM:= ControlData[MB_MIN_CONTROL_NUMBER];
END_VAR

MyVar:= MyControlName.LINK_ESTABLISHED;
MyControlName.LINK_REBUILD_RING:= MyVar;
```

Siehe auch:

- [ML_CONTROLDATA_SM](#), Seite 119.

DV_Control[] - Steuerungsdaten über Funktionsvariablenzugriff

Kurzbeschreibung Beim Anlegen eines neuen Projektes in IndraWorks werden automatisch Steuerungsdaten bereitgestellt. Mit Hilfe des fest definierten Index MB_MIN_CONTROL_NUMBER .. MB_MAX_CONTROL_NUMBER ([GVL_BaseConstant](#), Seite 130) kann auf Steuerungsdaten über Funktionsvariablenzugriff („DV_Control[]“) und Shared Memory („ControlData[]“) zugegriffen werden.

ML_Base.library



Steuerungsdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt.

Steuerungsdaten über Funktionsvariablenzugriff (alias "Direktvariablen") werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.

Steuerungsdaten 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.

Steuerungsdaten über Funktionsvariablenzugriff sind als globales Array „DV_Control[]“ des Funktionsbausteins „ML_DirectVarControl“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Der Funktionsbaustein „ML_DirectVarControl“ 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 abhängig vom Motion Modus und erzeugt im negativen Fall keine Fehlermeldung.



Der Funktionsbaustein „ML_DirectVarControl“ ist nicht für die direkte Verwendung im Anwenderprogramm vorgesehen.

Definition des globalen Array "DV_Control[]"

```
VAR_GLOBAL
```

```
...
DV_Control: ARRAY[MB_MIN_CONTROL_NUMBER..MB_MAX_CONTROL_NUMBER] OF ML_DirectVarControl;
...
END_VAR
```

Funktionsbeschreibung

Der lesende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine GET Property (READ) verfügt.

Lesender Zugriff auf Daten

```
MyVar:= DV_Control[MB_MIN_CONTROL_NUMBER].C_0_0023;
MyVar:= DV_Control[1].C_0_0023;
```

Der schreibende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine SET Property (WRITE) verfügt.

Schreibender Zugriff auf Daten

```
DV_Control[MB_MIN_CONTROL_NUMBER].C_0_0450:= MyVar;
DV_Control[1].C_0_0450:= MyVar;
```

Der IntelliSense einer Funktionsvariablen enthält die Parameteridentnummer, die Zugriffsmöglichkeiten (READ bzw. READ/WRITE), den Parameternamen sowie die Methode zur Ermittlung des aktuellen Zugriffsstatus.

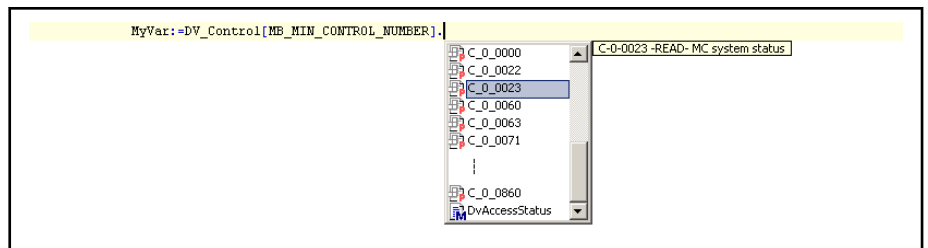


Abb. 4-7: Funktionsvariablenzugriff unter Nutzung der IntelliSense-Funktionalität



Auf Funktionsvariablen einer Steuerung kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Benutzerdefinierte Bezeichnung durch Referenzieren

```
VAR
  MyControlName: REFERENCE TO ML_DirectVarControl:= DV_Control[MB_MIN_CONTROL_NUMBER];
END_VAR

MyVar:= MyControlName.C_0_0023;
MyControlName.C_0_0450:= MyVar;
```

Status bei Funktionsvariablenzugriff

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 **DvAccessStatus** zu ermitteln.

Der Methode wird die entsprechende Parameterkonstante übergeben.

Die Methode gibt den Rückgabewert des letzten Zugriffes (== aktueller Zugriffstatus) zurück.

Das Lesen des aktuellen Zugriffstatus einer Funktionsvariablen erfolgt mittels folgender Anweisung (ST):

Lesen des aktuellen Zugriffstatus einer Funktionsvariablen

```
A_Status:= DV_Axis[MyAxisName.AxisNo].DvAccessStatus(FP_A_0_0000);
C_Status:= DV_Control[0].DvAccessStatus(FP_C_0_0023);
M_Status:= DV_Probe[MB_MIN_PROBE_NUMBER].DvAccessStatus(FP_M_0_0021);
N_Status:= DV_Pls[Index].DvAccessStatus(FP_N_0_0303);
O_Status:= DV_Osci[Index].DvAccessStatus(FP_O_0_0021);
```

Vereinbarungsdateien für Globale Konstanten

- GVL_A_Param_ID, Seite 134,
- GVL_C_Param_ID, Seite 134,
- GVL_M_Param_ID, Seite 135,
- GVL_N_Param_ID, Seite 135,
- GVL_O_Param_ID, Seite 136,

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).

ML_Base.library

Nocksaltwerkdaten über Shared Memory und Funktionsvariablenzugriff

Nocksaltwerkdaten über Shared Memory und Funktionsvariablenzugriff, Allgemeines



N-Parameter werden durch die IndraLogic XLC nicht unterstützt.

Auf die in N-Parametern gespeicherten Nocksaltwerkdaten kann von der SPS aus entweder über eine vereinfachte, blockierende Anweisung "Funktionsvariablenzugriff" (alias "Direktvariable") oder über Shared Memory zugegriffen werden.

Nocksaltwerkdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Die Istwerte der Nocksaltwerkdaten im Shared Memory werden stets am Ende der Bearbeitung des Motion-Kernels MK1 aufgefrischt, die Sollwerte werden stets am Ende der Bearbeitung des Motion-Kernels MK2 übertragen.

Nocksaltwerkdaten über Funktionsvariablenzugriff werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.

Ein Teil der Nocksaltwerkdaten stehen sowohl im Shared Memory als auch für den Zugriff über Direktvariablen zur Verfügung. Da Nocksaltwerkdaten 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 Laufzeiteresourcen 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 puffern

Nocksaltwerkdaten 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 der SPS-Tasks.

Der vereinfachte, blockierende Zugriff auf eine Funktionsvariable ist ein impliziter Funktionsaufruf, welche 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.

Shared Memory

Nocksaltwerkdaten im Shared Memory sind als globales Array „PlsData[]“ der Datenstruktur [ML_PLSDATA_SM](#), [Seite 105](#), deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Nocksaltwerkdaten im Shared Memory

```
VAR_GLOBAL
...
PlsData AT %MBccc: ARRAY [MB_MIN_PLS_NUMBER..MB_MAX_PLS_NUMBER] OF ML_PLSDATA_SM;
...
END_VAR
```



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.

xxx: mlc | xlc

yyy: l65 | l45 | l25

Funktionsvariablenzugriff

Funktionsvariablen der Daten der Nocksaltwerke sind als globales Array „DV_Pls[]“ des Funktionsbausteins „ML_DirectVarPls“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“, siehe auch [DV_Pls\[\]](#), Seite 107.

Funktionsvariablen der Daten der Nocksaltwerke

```
VAR_GLOBAL
...
DV_Pls: ARRAY [MB_MIN_PLS_NUMBER..MB_MAX_PLS_NUMBER] OF ML_DirectVarPls;
...
END_VAR
```



Steht für einen Parameter der Funktionsvariablenzugriff zur Verfügung, so ist dies bei der entsprechenden Parameterbeschreibung vermerkt, (Parameterattribute, Direktzugriff, Bsp. N-0-0000).

PlsData[] - Nockensaltwerkdaten im Shared Memory

Kurzbeschreibung

Beim Anlegen eines neuen Projektes in IndraWorks werden automatisch Nockensaltwerkdaten bereitgestellt. Mit Hilfe der fest definierten Indizes MB_MIN_PLS_NUMBER .. MB_MAX_PLS_NUMBER kann auf Nockensaltwerkdaten über

- Funktionsvariablenzugriff („DV_Pls[]“) und
- Shared Memory („PlsData[]“)

zugegriffen werden.



Nockensaltwerkdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Nockensaltwerkdaten als Direktvariablen werden unabhängig vom Motion Modus zum Zeitpunkt des Zugriffs aktualisiert.

Nockensaltwerkdaten im Shared Memory und Funktionsvariablenzugriff können sich, abhängig von den Taskeigenschaften der POU, über mehrere Zyklen der Motion (Sollwertgenerierung, Istwertbereitstellung) verteilen und sind somit nicht zwingend konsistent zueinander.

Nocksaltwerkdaten im Shared Memory sind als globales Array „PlsData[]“ der Datenstruktur „ML_PLSDATA_SM“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

ML_Base.library

Definition des globalen Array "PlsData[]"

VAR_GLOBAL

```
...
PlsData AT %MBccc: ARRAY [MB_MIN_PLS_NUMBER..MB_MAX_PLS_NUMBER] OF ML_PLSDATA_SM;
...
```

END_VAR



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.

xxx: mlc | xlc

yyy: l65 | l45 | l25

Siehe auch

- [ML_PLSDATA_SM, Seite 120.](#)

Funktionsbeschreibung

Der lesende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Lesender Zugriff auf Daten

```
MyVar:= PlsData[MB_MAX_PLS_NUMBER].Register_0_i;
MyVar:= PlsData[Index].Register_0_i;
MyVar:= PlsData[1]. Register_0_i;
```

Der schreibende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Schreibender Zugriff auf Daten

```
PlsData[MB_MAX_PLS_NUMBER].Force_0_q:= MyVar;
PlsData[Index].Force_0_q:= MyVar;
PlsData[1].Force_0_q:= MyVar;
```

Der IntelliSense einer Variablen im Shared Memory enthält die Parameteridentnummer, die Zugriffsmöglichkeiten (RD bzw. RW) sowie den Parameternamen.

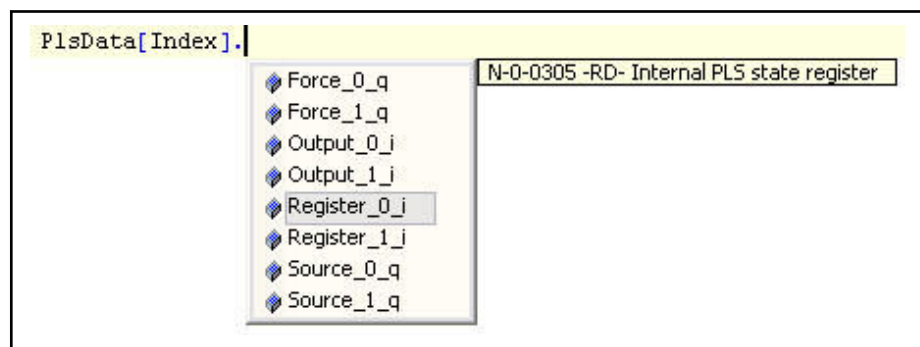


Abb. 4-8: *IntelliSense einer Variablen im Shared Memory*



Auf Variablen eines Nockenschaltwerkes im Shared Memory kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Benutzerdefinierte Bezeichnung durch Referenzieren

```

VAR
  MyPlsName: REFERENCE TO ML_PLSDATA_SM:= PlsData[MB_MAX_PLS_NUMBER];
END_VAR

```

```

MyVar:= MyPlsName.Register_0_i;
MyPlsName.Force_0_q:= MyVar;

```

Siehe auch:

- [ML_PLSDATA_SM, Seite 120.](#)

DV_Pls[] - Nockenschaltwerkdaten über Funktionsvariablenzugriff**Kurzbeschreibung**

Beim Anlegen eines neuen Projektes in IndraWorks werden automatisch Nockenschaltwerkdaten bereitgestellt. Mit Hilfe der fest definierten Indizes MB_MIN_PLS_NUMBER .. MB_MAX_PLS_NUMBER ([GVL_BaseConstant, Seite 130](#)) kann auf Nockenschaltwerkdaten über Funktionsvariablenzugriff („DV_Pls[]“) und Shared Memory („PlsData[]“) zugegriffen werden.



Nockenschaltwerkdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt.

Nockenschaltwerkdaten über Funktionsvariablenzugriff (alias "Direktvariablen") werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.

Nockenschaltwerkdaten als Funktionsvariablen können sich, abhängig von den Taskeigenschaften der POU, über mehrere Zyklen der Motion (Sollwertgenerierung, Istwertbereitstellung) verteilen und sind somit nicht zwingend konsistent zueinander.

Nockenschaltwerkdaten über Funktionsvariablenzugriff sind als globales Array „DV_Pls[]“ des Funktionsbausteins „ML_DirectVarPls“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Der Funktionsbaustein „ML_DirectVarPls“ stellt den Funktionsvariablenzugriff als Methoden bereit, die Existenz der Properties GET und SET der jeweiligen Methode definiert die Zugriffsmöglichkeit. Die Ausführung des Schreibzugriffs SET auf eine Funktionsvariable ist abhängig vom Motion Modus und erzeugt im negativen Fall keine Fehlermeldung.



Der Funktionsbaustein "ML_DirectVarPls" ist nicht für die direkte Verwendung im Anwenderprogramm vorgesehen.

Definition des globalen Array "DV_Pls[]"

```

VAR_GLOBAL
  ...
  DV_Pls: ARRAY [MB_MIN_PLS_NUMBER..MB_MAX_PLS_NUMBER] OF ML_DirectVarPls;
  ...
END_VAR

```

Funktionsbeschreibung

Der lesende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine GET Property (READ) verfügt.

ML_Base.library

Lesender Zugriff auf Daten

```
MyVar:= DV_Pls[Index].N_0_0303;
MyVar:= DV_Pls[1].N_0_0303;
```

Der schreibende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine SET Property (WRITE) verfügt.

Schreibender Zugriff auf Daten

```
DV_Pls[Index].N_0_0317:= MyVar;
DV_Pls[1].N_0_0317:= MyVar;
```

Der IntelliSense einer Funktionsvariablen enthält die Parameteridentnummer, die Zugriffsmöglichkeiten (READ bzw. READ/WRITE), den Parameternamen sowie die Methode zur Ermittlung des aktuellen Zugriffsstatus.

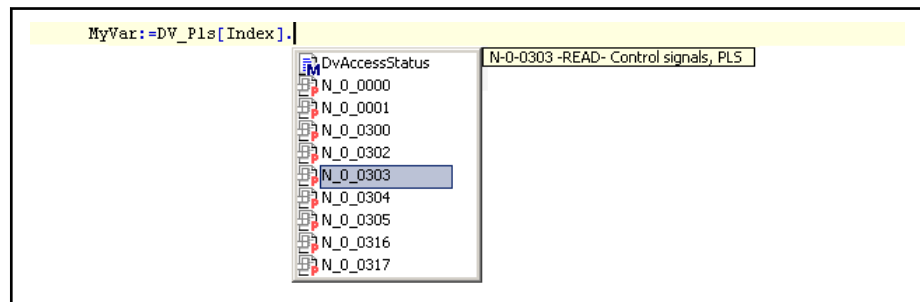


Abb. 4-9: Funktionsvariablenzugriff unter Nutzung der IntelliSense-Funktionalität



Auf Funktionsvariablen eines Nockenschaltwerks kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Benutzerdefinierte Bezeichnung durch Referenzieren

```
VAR
MyPlsName: REFERENCE TO ML_DirectVarPls:= DV_Pls[MB_MAX_PLS_NUMBER];
END_VAR

MyVar:= MyPlsName.N_0_0303;
MyPlsName.N_0_0317:= MyVar;
```

Status bei Funktionsvariablenzugriff

Die Funktionsbausteine für Funktionsvariablenzugriff bieten neben den Properties zum Lesen und Schreiben von Betriebsdaten auch die Möglichkeit, den **aktuellen Zugriffsstatus** eines Parameters über die Methode **DvAccessStatus** zu ermitteln.

Der Methode wird die entsprechende Parameterkonstante übergeben.

Die Methode gibt den Rückgabewert des letzten Zugriffs (== aktueller Zugriffsstatus) zurück.

Das Lesen des aktuellen Zugriffsstatus einer Funktionsvariablen erfolgt mittels folgender Anweisung (ST):

Lesen des aktuellen Zugriffsstatus einer Funktionsvariablen

```
A_Status:= DV_Axis[MyAxisName.AxisNo].DvAccessStatus(FP_A_0_0000);
C_Status:= DV_Control[0].DvAccessStatus(FP_C_0_0023);
M_Status:= DV_Probe[MB_MIN_PROBE_NUMBER].DvAccessStatus(FP_M_0_0021);
N_Status:= DV_Pls[Index].DvAccessStatus(FP_N_0_0303);
O_Status:= DV_Osci[Index].DvAccessStatus(FP_O_0_0021);
```

Vereinbarungsdateien für Globale Konstanten

- [GVL_A_Param_ID, Seite 134,](#)
- [GVL_C_Param_ID, Seite 134,](#)
- [GVL_M_Param_ID, Seite 135,](#)
- [GVL_N_Param_ID, Seite 135,](#)
- [GVL_O_Param_ID, Seite 136,](#)

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).

Messtasterdaten über Shared Memory und Funktionsvariablenzugriff**Messtasterdaten über Shared Memory und Funktionsvariablenzugriff, Allgemeines**

Auf die in den M-Parametern gespeicherten Messtaster-Daten kann von der SPS aus entweder über eine vereinfachte, blockierende Anweisung "Funktionsvariablenzugriff" (alias "Direktvariable") oder über Shared Memory zugegriffen werden.

Messtasterdaten im Shared Memory werden im Motion Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Die Aktualisierung der Messtasterdaten im Shared Memory kann zur Steigerung der Performance vollständig deaktiviert werden (C-0-0036). Die Istwerte der Messtasterdaten im Shared Memory werden stets am Ende der Bearbeitung des Motion-Kernels MK1 aufgefrischt.

Messtasterdaten über Funktionsvariablenzugriff werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.

Ein Teil der Messtasterdaten stehen sowohl im Shared Memory als auch für den Zugriff über Direktvariablen zur Verfügung. Da Messtasterdaten im Shared Memory unabhängig von der Nutzung zyklisch ausgetauscht werden, sollten weitere redundante Zugriffe über Direktvariablen 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 puffern

Messtasterdaten 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.

ML_Base.library



Eine Funktionsvariable ist kein Bestandteil des Prozessabbildes der SPS-Tasks.

Der vereinfachte, blockierende Zugriff auf eine Funktionsvariable ist ein impliziter Funktionsaufruf, welche 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.

Shared Memory

Messtasterdaten im Shared Memory sind als globales Array „ProbeData[]“ der Datenstruktur [ML_PROBEDATA_SM](#), [Seite 110](#) deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Die Messtaster sind durchnummeriert, der Messtaster-Name ist für diese Zugriffsart bedeutungslos.²⁾

Messtasterdaten im Shared Memory

```
VAR_GLOBAL
...
ProbeData AT %MBddd: ARRAY [MB_MIN_PROBE_NUMBER..MB_MAX_PROBE_NUMBER] OF ML_PROBEDATA_SM;
...
END_VAR
```



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.

xxx: mlc | xlc

yyy: l65 | l45 | l25

Funktionsvariablenzugriff

Funktionsvariablen der Messtasterdaten sind als globales Array „DV_Probe[]“ des Funktionsbausteins „ML_DirectVarProbe“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“, siehe auch [DV_Probe\[\]](#), [Seite 112](#).

Funktionsvariablen der Messtasterdaten

```
VAR_GLOBAL
...
DV_Probe: ARRAY [MB_MIN_PROBE_NUMBER..MB_MAX_PROBE_NUMBER] OF ML_DirectVarProbe;
...
END_VAR
```



Steht für einen Parameter der Funktionsvariablenzugriff zur Verfügung, so ist dies bei der entsprechenden Parameterbeschreibung vermerkt, (Parameterattribute, Direktzugriff, Bsp. M-0-0000).

ProbeData[] - Messtasterdaten im Shared Memory**Kurzbeschreibung**

Beim Anlegen eines neuen Projektes in IndraWorks wird automatisch eine definierte Anzahl an Messtastern erzeugt.

Mit Hilfe der entsprechenden Messtasternummer kann indiziert auf Messtasterdaten über

²⁾ MB_MAX_PROBE_NUMBER, maximal zulässige Messtaster-Anzahl, z.Z. 100, steuerungstypabhängig.

- Funktionsvariablenzugriff („DV_Probe[]“) und
- Shared Memory („ProbeData[]“)

zugegriffen werden.



Messtasterdaten im Shared Memory werden im Motion Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Die Aktualisierung der Messtasterdaten im Shared Memory kann zur Steigerung der Performance vollständig deaktiviert werden (C-0-0036). Messtasterdaten als Direktvariablen werden unabhängig vom Motion Modus zum Zeitpunkt des Zugriffs aktualisiert.

Messtasterdaten im 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.

Messtasterdaten im Shared Memory sind als globales Array „ProbeData[]“ der Datenstruktur „ML_PROBEDATA_SM“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Definition des globalen Array "ProbeData[]"

VAR_GLOBAL

```
...
ProbeData AT %MBddd: ARRAY [MB_MIN_PROBE_NUMBER..MB_MAX_PROBE_NUMBER] OF ML_PROBEDATA_SM;
...
END_VAR
```



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.

xxx: mlc | xlc

yyy: l65 | l45 | l25

Siehe auch:

- [ML_PROBEDATA_SM, Seite 120.](#)

Funktionsbeschreibung

Der lesende Zugriff auf Daten im Shared Memory kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen.

Lesender Zugriff auf Daten im Shared Memory:

```
MyVar:= ProbeData[TouchProbe???.TouchProbeNo].FlexProfil_Active;
MyVar:= ProbeData[Index].FlexProfil_Active;
MyVar:= ProbeData[1].FlexProfil_Active;
```

Der schreibende Zugriff auf Messtasterdaten im Shared Memory ist nicht möglich.

Der IntelliSense einer Variablen im Shared Memory enthält die Parameteridentnummer sowie den Parameternamen.

ML_Base.library

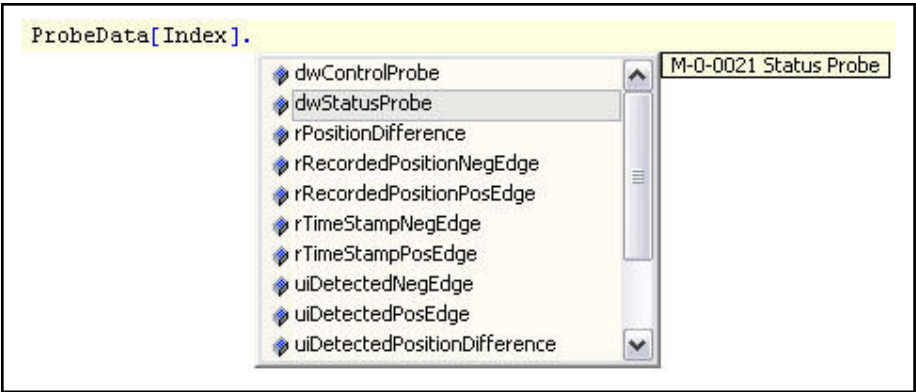


Abb. 4-10: IntelliSense einer Variablen im Shared Memory

 Auf Variablen eines Messtasters im Shared Memory kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Benutzerdefinierte Bezeichnung durch Referenzieren:

```
VAR
  MyProbeName:REFERENCE TO ML_PROBEDATA_SM:= ProbeData[TouchProbe???.TouchProbeNo];
END_VAR

MyVar:= MyProbeName.dwStatusProbe;
```

Ausgewählte Bereiche der Datenstruktur ML_POBEDATA_SM

rRecordedPositionPosEdge	:SM_EXT_TYPES;	(* M-0-0130 Value Positive Edge Probe*)
rRecordedPositionNegEdge	:SM_EXT_TYPES;	(* M-0-0131 Value Negative Edge Probe*)
rPositionDifference	:SM_EXT_TYPES;	(* M-0-0132 Difference Value Probe*)

Die Datencontainer rRecordedPositionPosEdge, rRecordedPositionNegEdge und rPositionDifference enthalten konfigurierbare Echtzeitdaten. Abhängig von der Konfiguration können die Container Werte mit unterschiedlichen Datentypen enthalten. Auf den entsprechenden Datentyp kann über den Punktoperator zugegriffen werden.

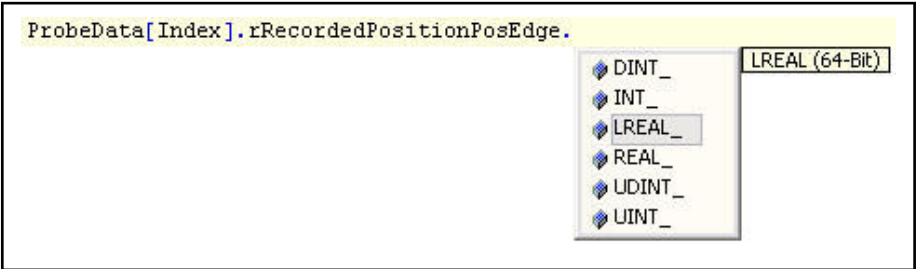


Abb. 4-11: Vorgabe des Datentyps über Punktoperator

Siehe auch

- [SM_EXT_TYPES](#), Seite 122,
- [ML_PROBEDATA_SM](#), Seite 120.

DV_Probe[] - Messtasterdaten über Funktionsvariablenzugriff

Kurzbeschreibung

Beim Anlegen eines neuen Projektes in IndraWorks wird automatisch eine definierte Anzahl an Messtastern erzeugt. Mit Hilfe der entsprechenden

Messtasternummer MB_MIN_PROBE_NUMBER..MB_MAX_PROBE_NUMBER ([GVL_BaseConstant](#), [Seite 130](#)) kann indiziert auf Messtasterdaten über Funktionsvariablenzugriff („DV_Probe[]“) und Shared Memory („Probe-Data[]“) zugegriffen werden.



Messtasterdaten im Shared Memory werden im Motion-Modus „Betriebsmodus“ zyklisch konsistent zueinander aktualisiert und sind aus Gründen der notwendigen Aktualisierungszeit im Umfang eingeschränkt. Die Aktualisierung der Messtasterdaten im Shared Memory kann zur Steigerung der Performance vollständig deaktiviert werden (C-0-0036).

Messtasterdaten über Funktionsvariablenzugriff (alias "Direktvariablen") werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.

Messtasterdaten als Funktionsvariablen können sich, abhängig von den Taskeigenschaften der POU, über mehrere Zyklen der Motion (Sollwertgenerierung, Istwertbereitstellung) verteilen und sind somit nicht zwingend konsistent zueinander.

Messtasterdaten über Funktionsvariablenzugriff sind als globales Array „DV_Probe[]“ des Funktionsbausteins „ML_DirectVarProbe“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“.

Der Funktionsbaustein „ML_DirectVarProbe“ stellt den Funktionsvariablenzugriff als Methoden bereit, die Existenz der Properties GET und SET der jeweiligen Methode definiert die Zugriffsmöglichkeit. Die Ausführung des Schreibzugriffs SET auf eine Direktvariable ist abhängig vom Motion-Modus und erzeugt im negativen Fall keine Fehlermeldung.



Der Funktionsbaustein „ML_DirectVarProbe“ ist nicht für die direkte Verwendung im Anwenderprogramm vorgesehen.

Definition des globalen Array "DV_Probe[]"

```
VAR_GLOBAL
```

```
...
DV_Probe:ARRAY[MB_MIN_PROBE_NUMBER..MB_MAX_PROBE_NUMBER] OF ML_DirectVarProbe;
...
END_VAR
```

Funktionsbeschreibung

Der lesende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine GET Property (READ) verfügt.

Lesender Zugriff auf Daten

```
MyVar:= DV_Probe[TouchProbe???. TouchProbeNo].M_0_0130;
MyVar:= DV_Probe[Index].M_0_0130;
MyVar:= DV_Probe[1].M_0_0130;
```

Der schreibende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine SET Property (WRITE) verfügt.

Schreibender Zugriff auf Daten

```
DV_Probe[TouchProbe???. TouchProbeNo].M_0_0205:= MyVar;
```


ML_Base.library

```
DV_Probe[Index].A_0_0205:= MyVar;
DV_Probe[1].A_0_0205:= MyVar;
```

Der IntelliSense einer Funktionsvariablen enthält die Parameteridentnummer, die Zugriffsmöglichkeiten (READ bzw. READ/WRITE), den Parameternamen sowie die Methode zur Ermittlung des aktuellen Zugriffsstatus.

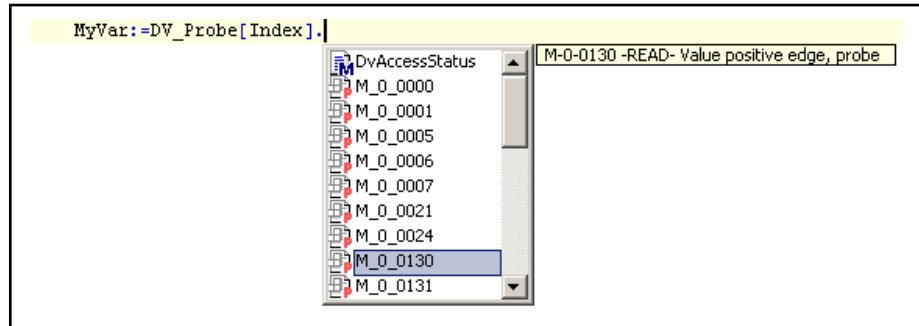


Abb. 4-12: Funktionsvariablenzugriff unter Nutzung der IntelliSense-Funktionalität



Auf Funktionsvariablen eines Messtasters kann durch Verwendung von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Benutzerdefinierte Bezeichnung durch Referenzieren

```
VAR
  MyProbeName: REFERENCE TO ML_DirectVarProbe:= DV_Probe[TouchProbe???. TouchProbeNo];
END_VAR

MyVar:= MyProbeName.M_0_0130;
MyProbeName.M_0_0205:= MyVar;
```

Status bei Funktionsvariablenzugriff

Die Funktionsbausteine für Funktionsvariablenzugriff bieten neben den Properties zum Lesen und Schreiben von Betriebsdaten auch die Möglichkeit, den **aktuellen Zugriffsstatus** eines Parameters über die Methode **DvAccessStatus** zu ermitteln.

Der Methode wird die entsprechende Parameterkonstante übergeben.

Die Methode gibt den Rückgabewert des letzten Zugriffs (== aktueller Zugriffsstatus) zurück.

Das Lesen des aktuellen Zugriffsstatus einer Funktionsvariablen erfolgt mittels folgender Anweisung (ST):

Lesen des aktuellen Zugriffsstatus einer Funktionsvariablen

```
A_Status:= DV_Axis[MyAxisName.AxisNo].DvAccessStatus(FP_A_0_0000);
C_Status:= DV_Control[0].DvAccessStatus(FP_C_0_0023);
M_Status:= DV_Probe[MB_MIN_PROBE_NUMBER].DvAccessStatus(FP_M_0_0021);
N_Status:= DV_Pls[Index].DvAccessStatus(FP_N_0_0303);
O_Status:= DV_Osci[Index].DvAccessStatus(FP_O_0_0021);
```

Vereinbarungsdateien für Globale Konstanten

- [GVL_A_Param_ID](#), Seite 134,
- [GVL_C_Param_ID](#), Seite 134,
- [GVL_M_Param_ID](#), Seite 135,
- [GVL_N_Param_ID](#), Seite 135,
- [GVL_O_Param_ID](#), Seite 136,

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).

Oszilloskopdaten über Funktionsvariablenzugriff

Oszilloskopdaten über Funktionsvariablenzugriff, Allgemeines

Auf die in O-Parametern gespeicherten Oszilloskopdaten kann von der SPS über Funktionsvariablenzugriff zugegriffen werden. Oszilloskopdaten stehen nicht im Shared Memory zur Verfügung.

Oszilloskopdaten über Funktionsvariablenzugriff (alias "Direktvariablen") werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert.



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 puffern

Oszilloskopdaten 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.

Shared Memory Funktionsvariablenzugriff

Oszilloskopdaten stehen nicht im Shared Memory zur Verfügung.

Funktionsvariablen der Oszilloskopdaten sind als globales Array „DV_Osc[]“ des Funktionsbausteins „ML_DirectVarOsci“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base_xxx_yyy“, siehe auch [DV_Osc\[\]](#), [Seite 115](#).

Funktionsvariablen der Oszilloskopdaten

```
VAR_GLOBAL
```

```
...
DV_Osc: ARRAY [MB_MIN_OSCI_NUMBER..MB_MAX_OSCI_NUMBER] OF ML_DirectVarOsci;
...
END_VAR
```



Steht für einen Parameter der Funktionsvariablenzugriff zur Verfügung, so ist dies bei der entsprechenden Parameterbeschreibung vermerkt, (Parameterattribute, Direktzugriff, Bsp. O-0-0000).

DV_Osc[] - Oszilloskopdaten über Funktionsvariablenzugriff

Kurzbeschreibung

Beim Anlegen eines neuen Projektes in IndraWorks werden automatisch Oszilloskopdaten bereitgestellt. Mit Hilfe der fest definierten Indizes MB_MIN_OSCI_NUMBER .. MB_MAX_OSCI_NUMBER ([GVL_BaseConstant](#), [Seite 130](#)) kann auf Oszilloskopdaten über Direktvariablen („DV_Osc[]“) zugegriffen werden.

ML_Base.library



Oszilloskopdaten über Funktionsvariablenzugriff (alias "Direktvariablen") werden unabhängig vom Motion-Modus zum Zeitpunkt des Zugriffs aktualisiert. Oszilloskopdaten können sich, abhängig von den Taskeigenschaften der POU, über mehrere Zyklen der Motion (Sollwertgenierung, Istwertbereitstellung) verteilen und sind somit nicht zwingend konsistent zueinander.

Oszilloskopdaten über Funktionsvariablenzugriff sind als globales Array „DV_Osc[]“ des Funktionsbausteins „ML_DirectVarOsci“ deklariert. Die Definition ist Bestandteil der globalen Variablenliste „GVL_Base“ der Bibliothek „ML_Base“.

Der Funktionsbaustein „ML_DirectVarOsci“ stellt den Funktionsvariablenzugriff als Methoden bereit, die Existenz der Properties GET und SET der jeweiligen Methode definiert die Zugriffsmöglichkeit. Die Ausführung des Schreibzugriffs SET auf eine Funktionsvariable ist abhängig vom Motion Modus und erzeugt im negativen Fall keine Fehlermeldung.



Der Funktionsbausteins „ML_DirectVarOsci“ ist nicht für die direkte Verwendung im Anwenderprogramm vorgesehen.

Definition des globalen Array "DV_Osc[]"

```
VAR_GLOBAL
...
DV_Osc: ARRAY[MB_MIN_OSCI_NUMBER..MB_MAX_OSCI_NUMBER] OF ML_DirectVarOsci;
...
END_VAR
```

Funktionsbeschreibung

Der lesende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine GET Property (READ) verfügt.

Lesender Zugriff auf Daten

```
MyVar:= DV_Osc[Index].O_0_0021;
MyVar:= DV_Osc[1].O_0_0021;
```

Der schreibende Zugriff auf eine Funktionsvariable kann mittels folgenden äquivalenten Anweisungen (ST) erfolgen, sofern die jeweilige Methode über eine SET Property (WRITE) verfügt.

Schreibender Zugriff auf Daten

```
DV_Osc[Index].O_0_0000:= MyVar;
DV_Osc[1].O_0_0000:= MyVar;
```

Der IntelliSense einer Funktionsvariablen enthält die Parameteridentnummer, die Zugriffsmöglichkeiten (READ bzw. READ/WRITE), den Parameternamen sowie die Methode zur Ermittlung des aktuellen Zugriffsstatus.

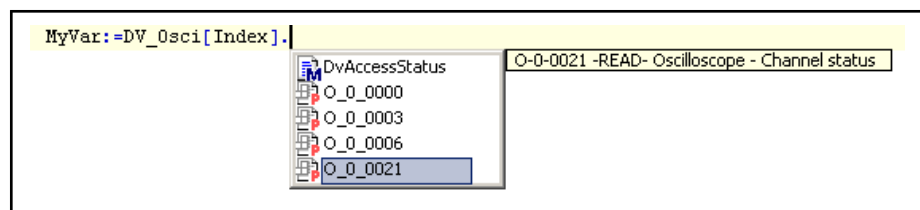


Abb. 4-13: Funktionsvariablenzugriff unter Nutzung der IntelliSense-Funktionalität



Auf Funktionsvariablen eines Oszilloskopes kann durch Verwenden von Referenzen auch über eine benutzerdefinierte Bezeichnung zugegriffen werden.

Benutzerdefinierte Bezeichnung durch Referenzierung

```
VAR
  MyOsciName: REFERENCE TO ML_DirectVarOsci:= DV_Osci[MB_MAX_OSCI_NUMBER];
END_VAR

MyVar:= MyOsciName.O_0_0021;
MyOsciName.O_0_0000:= MyVar;
```

Status bei Funktionsvariablenzugriff

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 **DvAccessStatus** 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

```
A_Status:= DV_Axis[MyAxisName.AxisNo].DvAccessStatus(FP_A_0_0000);
C_Status:= DV_Control[0].DvAccessStatus(FP_C_0_0023);
M_Status:= DV_Probe[MB_MIN_PROBE_NUMBER].DvAccessStatus(FP_M_0_0021);
N_Status:= DV_Pls[Index].DvAccessStatus(FP_N_0_0303);
O_Status:= DV_Osci[Index].DvAccessStatus(FP_O_0_0021);
```

Vereinbarungsdateien für Globale Konstanten

- [GVL_A_Param_ID, Seite 134,](#)
- [GVL_C_Param_ID, Seite 134,](#)
- [GVL_M_Param_ID, Seite 135,](#)
- [GVL_N_Param_ID, Seite 135,](#)
- [GVL_O_Param_ID, Seite 136,](#)

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).

ML_AXISDATA_SM

Kurzbeschreibung

Eine Instanz der Struktur ML_AXISDATA_SM enthält die Informationen der korrespondierenden Achse, auf die per **Shared Memory** zugegriffen werden kann.

ML_Base.library

Bibliothek	Bereich
ML_Base	Data Types/ Shared Memory

Tab. 4-14: Bibliothekszuordnung

Aufbau der Datenstruktur ML_AXISDATA_SM:

```
TYPE ML_AXISDATA_SM :
```

```
STRUCT
```

```

wUserCmdDataBOOLA_q :WORD; // A-0-0542 -RW- User defined commanded data BOOL number A
wUserCmdDataBOOLB_q :WORD; // A-0-0545 -RW- User defined commanded data BOOL number B
wUserCmdDataBOOLC_q :WORD; // A-0-0548 -RW- User defined commanded data BOOL number C
wUserCmdDataBOOLD_q :WORD; // A-0-0551 -RW- User defined commanded data BOOL number D

dwUserCmdDataA_q :SM_TYPES; // A-0-0501 -RW- Container user defined commanded data A
dwUserCmdDataB_q :SM_TYPES; // A-0-0503 -RW- Container user defined commanded data B
dwUserCmdDataC_q :SM_TYPES; // A-0-0505 -RW- Container user defined commanded data C
dwUserCmdDataD_q :SM_TYPES; // A-0-0507 -RW- Container user defined commanded data D
dwUserCmdDataE_q :SM_TYPES; // A-0-0509 -RW- Container user defined commanded data E

dwMotionProfileStatusBOOLs_i :DWORD; // A-0-2931 -RD- Motion profile status word
dwFlexProfileEventControlBOOLs_q :DWORD; // A-0-3011 -RW- Condition control variable, FP
FlexProfile_Sync :BIT; // A-0-3010.08 -RD- FlexProfile is in synchronisation
FlexProfile_Active :BIT; // A-0-3010.09 -RD- FlexProfile is active
FlexProfile_WaitForSwitching :BIT; // A-0-3010.10 -RD- FP is waiting for switching position
FlexProfile_ExeSingleFinished :BIT; // A-0-3010.11 -RD- FP EXECUTE SINGLE is finished
FlexProfile_Set0_Checked :BIT; // A-0-3010.16 -RD- FlexProfile Set0 is checked
FlexProfile_Set0_Ok :BIT; // A-0-3010.17 -RD- FlexProfile Set0 is ok
FlexProfile_Set1_Checked :BIT; // A-0-3010.18 -RD- FlexProfile Set0 is checked
FlexProfile_Set1_Ok :BIT; // A-0-3010.19 -RD- FlexProfile Set0 is ok
FlexProfile_Set2_Checked :BIT; // A-0-3010.20 -RD- FlexProfile Set0 is checked
FlexProfile_Set2_Ok :BIT; // A-0-3010.21 -RD- FlexProfile Set0 is ok
FlexProfile_Set3_Checked :BIT; // A-0-3010.22 -RD- FlexProfile Set0 is checked
FlexProfile_Set3_Ok :BIT; // A-0-3010.23 -RD- FlexProfile Set0 is ok
FlexProfile_EvtSet0_Checked :BIT; // A-0-3010.24 -RD- FlexProfile Event Set0 is checked
FlexProfile_EvtSet0_Ok :BIT; // A-0-3010.25 -RD- FlexProfile Event Set0 is ok
FlexProfile_EvtSet1_Checked :BIT; // A-0-3010.26 -RD- FlexProfile Event Set0 is checked
FlexProfile_EvtSet1_Ok :BIT; // A-0-3010.27 -RD- FlexProfile Event Set0 is ok
FlexProfile_EvtSet2_Checked :BIT; // A-0-3010.28 -RD- FlexProfile Event Set0 is checked
FlexProfile_EvtSet2_Ok :BIT; // A-0-3010.29 -RD- FlexProfile Event Set0 is ok
FlexProfile_EvtSet3_Checked :BIT; // A-0-3010.30 -RD- FlexProfile Event Set0 is checked
FlexProfile_EvtSet3_Ok :BIT; // A-0-3010.31 -RD- FlexProfile Event Set0 is ok
dwFlexProfileEventStatusBits_i :DWORD; // A-0-3012 -RD- Condition status variable, FP

Axis_CamTab_0 :BIT; // A-0-0021.00 -RD- Cam shaft table BIT 0
Axis_CamTab_1 :BIT; // A-0-0021.01 -RD- Cam shaft table BIT 1
Axis_CamTab_2 :BIT; // A-0-0021.02 -RD- Cam shaft table BIT 2 (reserved)
Axis_CamSwitching :BIT; // A-0-0021.03 -RD- Status cam shaft distance switching
Axis_InVelocity :BIT; // A-0-0021.04 -RD- Axis cmd velocity == actual velocity
Axis_Standstill :BIT; // A-0-0021.05 -RD- Axis velocity <= A-0-0222 (standstill)
Axis_InPosition :BIT; // A-0-0021.06 -RD- Axis cmd position == actual position
Axis_InSynchron :BIT; // A-0-0021.07 -RD- Axis is in synchronisation
Axis_Warning :BIT; // A-0-0021.08 -RD- Axis reports warning
Axis_Error :BIT; // A-0-0021.09 -RD- Axis reports error
Axis_Homed :BIT; // A-0-0021.10 -RD- Axis is relative / in homing
Axis_InTorque :BIT; // A-0-0021.11 -RD- Axis cmd torque / force == actual torque
Axis_OperationMode :BIT; // A-0-0021.12 -RD- Axis is in operation mode (P4)
Axis_Inbb :BIT; // A-0-0021.13 -RD- Axis is ready for power to the amplif. (bb)
Axis_InAb :BIT; // A-0-0021.14 -RD- Axis is ready for power to the engine (Ab)
Axis_Power :BIT; // A-0-0021.15 -RD- Axis has power on (AH_AF)
Axis_CmdActive :BIT; // A-0-0021.16 -RD- Command active
Axis_ErrInProgress :BIT; // A-0-0021.17 -RD- Error reaction in progress
Axis_CyclicPosChannelOpen :BIT; // A-0-0021.18 -RD- Cyclic Position Channel is open for Axis
Axis_InMasterPhaseOffset :BIT; // A-0-0021.19 -RD- Master Phase offset achieved
Axis_CyclicTrqChannelOpen :BIT; // A-0-0021.20 -RD- Cyclic Position Channel is open for Axis
Axis_CyclicVelChannelOpen :BIT; // A-0-0021.21 -RD- Cyclic Velocity Channel is open for Axis
Axis_CyclicAnaChannelOpen :BIT; // A-0-0021.22 -RD- Cyclic Analoge Channel is open for Axis
Axis_NRT :BIT; // A-0-0021.23 -RD- Connected Axis is connected with NRT channel
// Parameter access takes place over NRT channel
Axis_Interrupted :BIT; // A-0-0021.24 -RD- Axis is not following commanded values
Axis_PLCCControlled :BIT; // A-0-0021.27 -RD- Interpolation influenced by PLC

ErrorStop :BIT; // A-0-0022.00 -RD- Axis is in "ErrorStop"
Stopping :BIT; // A-0-0022.01 -RD- Axis is in "Stopping"
Homing :BIT; // A-0-0022.02 -RD- Command homing is active
PowerOn :BIT; // A-0-0022.03 -RD- Axis got power
DiscreteMotion :BIT; // A-0-0022.04 -RD- Axis is in "Discrete Motion"
ContinuousMotion :BIT; // A-0-0022.05 -RD- Axis is in "Continuous Motion"
SynchronizedMotion :BIT; // A-0-0022.06 -RD- Axis is in "Synchronized Motion"
Standstill :BIT; // A-0-0022.07 -RD- Axis is stopped
CoordinatedMotion :BIT; // A-0-0022.08 -RD- Axis is in "CoordinatedMotion"

```

```

ControlledMotion          :BIT; // A-0-0022.09 -RD- Axis is in "ControlledMotion"

dwAxisCondition_i         :DWORD; // A-0-0024 -RD- Axis condition
rActualPosition_i         :REAL; // A-0-0100 -RD- Actual position
rActualVelocity_i         :REAL; // A-0-0102 -RD- Actual velocity
rActualTorqueForce_i       :REAL; // A-0-0110 -RD- Actual torque/force

wUserActualDataBitA_i     :WORD; // A-0-0562 -RD- User defined actual data bit number A
wUserActualDataBitB_i     :WORD; // A-0-0565 -RD- User defined actual data bit number B
wUserActualDataBitC_i     :WORD; // A-0-0568 -RD- User defined actual data bit number C
wUserActualDataBitD_i     :WORD; // A-0-0571 -RD- User defined actual data bit number D

dwUserActualDataA_i       :SM_TYPES; // A-0-0521 -RD- Container user defined actual data A
dwUserActualDataB_i       :SM_TYPES; // A-0-0523 -RD- Container user defined actual data B
dwUserActualDataC_i       :SM_TYPES; // A-0-0525 -RD- Container user defined actual data C
dwUserActualDataD_i       :SM_TYPES; // A-0-0527 -RD- Container user defined actual data D
dwUserActualDataE_i       :SM_TYPES; // A-0-0529 -RD- Container user defined actual data E
dwUserActualDataF_i       :SM_TYPES; // A-0-0531 -RD- Container user defined actual data F

END_STRUCT
END_TYPE

```

Siehe auch:

- [AxisData\[\] - Achsdaten im Shared Memory, Seite 91,](#)
- [SM_TYPES, Seite 122.](#)

ML_CONTROLDATA_SM

Kurzbeschreibung

Eine Instanz der Struktur ML_CONTROLDATA_SM enthält die Informationen der korrespondierenden Steuerung, auf die per **Shared Memory** zugegriffen werden kann.

Bibliothek	Bereich
ML_Base	Data Types/ Shared Memory

Tab. 4-15: Bibliothekszuordnung

Aufbau der Datenstruktur ML_CONTROLDATA_SM:

```

TYPE ML_CONTROLDATA_SM :
STRUCT
  LINK_SYNC1          :BIT; // C-0-0704.00 -RW- Link synchronization bit 0
  LINK_SYNC2          :BIT; // C-0-0704.01 -RW- Link synchronization bit 1
  LINK_SYNC3          :BIT; // C-0-0704.02 -RW- Link synchronization bit 2
  LINK_SYNC4          :BIT; // C-0-0704.03 -RW- Link synchronization bit 3
  LINK_REBUILD_RING   :BIT; // C-0-0704.04 -RW- Rebuild link ring
  LINK_DISABLE_HOTPLUG :BIT; // C-0-0704.05 -RW- Disable hotplug within C2C master

  arLinkSyncBits_i    :ARRAY[0..64] OF ML_SYNC; // C-0-0713 -RD- Link synchr. bits of ctrls
  arLinkValidBits_i    :ARRAY[0..64] OF ML_LINK; // C-0-0712 -RD- Link valid. status bits of ctrls

  wCurrentMotionMode   :WORD; // C-0-0452 - RD - Current motion mode

  LINK_ERROR           :BIT; // C-0-0714.00 -RD- Link error
  LINK_ERROR_PRI       :BIT; // C-0-0714.01 -RD- Link error primary optical ring
  LINK_ERROR_SEC       :BIT; // C-0-0714.02 -RD- Link error secondary optical ring
  LINK_ERROR_RED       :BIT; // C-0-0714.03 -RD- Redundancy loss
  LINK_ERROR_HOTPLUG   :BIT; // C-0-0714.04 -RD- Hotplug error pending (see index 0..2)
  LINK_ERROR_HOTPLUG_BIT0 :BIT; // C-0-0714.05 -RD- Hotplug error; index 0
  LINK_ERROR_HOTPLUG_BIT1 :BIT; // C-0-0714.06 -RD- Hotplug error; index 1
  LINK_ERROR_HOTPLUG_BIT2 :BIT; // C-0-0714.07 -RD- Hotplug error; index 2
  LINK_PHASESWITCH_ACTIVE :BIT; // C-0-0714.08 -RD- Phase switch of link ring active
  LINK_HOTPLUG_ACTIVE  :BIT; // C-0-0714.09 -RD- Hotplug of new device active
  LINK_HOTPLUG_ABORTED :BIT; // C-0-0714.10 -RD- Hotplug of new device aborted
  LINK_HOTPLUG_BIT0     :BIT; // C-0-0714.11 -RD- Hotplug status; status 0
  LINK_HOTPLUG_BIT1     :BIT; // C-0-0714.12 -RD- Hotplug status; status 1
  LINK_HOTPLUG_BIT2     :BIT; // C-0-0714.13 -RD- Hotplug status; status 2
  LINK_MASTER_ACTIVE   :BIT; // C-0-0714.14 -RD- Link master active, cyclic commun. active
  LINK_PLL_LOCKED      :BIT; // C-0-0714.15 -RD- Link device synchron. to link master
  LINK_ACYC_PARA_UPDATE :BIT; // C-0-0714.16 -RD- Exchange of non-cyclic transmitted parameters is active

  arSercos             :ARRAY[0..0] OF ML_SERCOS; // C-0-0514 -RD- Diag of sercos interface

```

ML_Base.library

END_STRUCT
END_TYPE

Siehe auch:

- [ML_LINK](#), Seite 121,
- [ML_SERCOS](#), Seite 121
- [ML_SYNC](#), Seite 121,

ML_PLSDATA_SM

Kurzbeschreibung

Eine Instanz der Struktur ML_PLSDATA_SM enthält die Informationen des korrespondierenden Nockenschaltwerk, auf das per **Shared Memory** zugegriffen werden kann.

Bibliothek	Bereich
ML_Base	Data Types/ Shared Memory

Tab. 4-16: Bibliothekszuordnung

Aufbau der Datenstruktur ML_PLSDATA_SM:

```

TYPE ML_PLSDATA_SM :
STRUCT
  Source_0_q :BYTE; (* N-0-0302 -RW- Source control reg. (0/1 = force reg./PLS reg.) *)
  Source_1_q :BYTE; (* N-0-0302 -RW- Source control reg. (0/1 = force reg./PLS reg.) *)
  Force_0_q :BYTE; (* N-0-0303 -RW- Force reg. for output 'X201' *)
  Force_1_q :BYTE; (* N-0-0303 -RW- Force reg. for output 'X202' *)

  Output_0_i :BYTE; (* N-0-0304 -RD- Physical output 'X201' *)
  Output_1_i :BYTE; (* N-0-0304 -RD- Physical output 'X202' *)
  Register_0_i :BYTE; (* N-0-0305 -RD- Internal PLS state register *)
  Register_1_i :BYTE; (* N-0-0305 -RD- Internal PLS state register *)

  MasterPosOffset: ARRAY[1..8] OF REAL; (* PLS master position offset *)
END_STRUCT
END_TYPE

```

ML_PROBEDATA_SM

Kurzbeschreibung

Eine Instanz der Struktur ML_PROBEDATA_SM enthält die Informationen des korrespondierenden Messtasters, auf das per **Shared Memory** zugegriffen werden kann.

Bibliothek	Bereich
ML_Base	Data Types/ Shared Memory

Tab. 4-17: Bibliothekszuordnung

Aufbau der Datenstruktur ML_PROBEDATA_SM:

```

TYPE ML_PROBEDATA_SM :
STRUCT
  dwControlProbe :DWORD; (* M-0-0007 Control Probe*)
  dwStatusProbe :DWORD; (* M-0-0021 Status Probe*)
  rRecordedPositionPosEdge :SM_EXT_TYPES; (* M-0-0130 Value pos. Edge Probe*)
  rRecordedPositionNegEdge :SM_EXT_TYPES; (* M-0-0131 Value neg. Edge Probe*)
  rPositionDifference :SM_EXT_TYPES; (* M-0-0132 Difference Value Probe*)
  rTimeStampPosEdge :SYS_TIME64; (* M-0-0140 Timestamp pos. Edge Probe*)
  rTimeStampNegEdge :SYS_TIME64; (* M-0-0141 Timestamp neg. Edge Probe*)
  uiMissingMarksPosEdge :UINT; (* M-0-0224 Number of pos. Marker Failures Probe*)
  uiMissingMarksNegEdge :UINT; (* M-0-0225 Number of neg. Marker Failures Probe*)
  uiStatusInputProbe :UINT; (* M-0-0401 Status Input Probe *)
  wEnablingProbe :WORD; (* M-0-0405 Enabling Probe*)
  uiDetectedPosEdge :UINT; (* M-0-0409 Counter Values pos. Edge Probe*)

```



```

uiDetectedNegEdge           :UINT;           (* M-0-0410 Counter Values neg. Edge Probe*)
uiDetectedPositionDifference :UINT;           (* M-0-0411 Counter Difference Values Probe*)
END_STRUCT
END_TYPE

```

Siehe auch:

- [SM_EXT_TYPES](#), Seite 122.

ML_LINK

Kurzbeschreibung Der Datentyp enthält das Gültigkeitsbit der Steuerung. Dieses zeigt an, ob die Daten der Steuerung im MLC-Verbund gültig sind.

Bibliothek	Bereich
ML_Base	Shared Memory Types

Tab. 4-18: Bibliothekszuordnung

Typdefinition

```

TYPE ML_LINK :
STRUCT
  iLinkBit0 :BIT; // C-0-0712.00 -RD- Link validation status bit 0 of control
END_STRUCT
END_TYPE

```

ML_SERCOS

Kurzbeschreibung Der Datentyp enthält die Diagnosebits des Sercos-Interfaces der Steuerung.

Bibliothek	Bereich
ML_Base	Shared Memory Types/Types

Tab. 4-19: Bibliothekszuordnung

Typdefinition

```

TYPE ML_SERCOS :
STRUCT
  ERROR_PORT1 :BIT; // C-0-0514.01 -RD- Link error Port1
  ERROR_PORT2 :BIT; // C-0-0514.02 -RD- Link error Port2
  ERROR_RED   :BIT; // C-0-0514.03 -RD- Bus error
  REBUILD_RING :BIT; // C-0-0514.04 -RD- Rebuild Ring Flag

  PHASE_SWITCH_ACTIVE :BIT; // C-0-0514.05 -RD- PHASE_SWITCH_ACTIVE

  ACTUAL_PHASE_BIT0 :BIT; // C-0-0514.27 -RD- ACTUAL_PHASE_BIT0
  ACTUAL_PHASE_BIT1 :BIT; // C-0-0514.28 -RD- ACTUAL_PHASE_BIT1
  ACTUAL_PHASE_BIT2 :BIT; // C-0-0514.29 -RD- ACTUAL_PHASE_BIT2
  ACTUAL_PHASE_BIT3 :BIT; // C-0-0514.30 -RD- ACTUAL_PHASE_BIT3
  ACTUAL_PHASE_BIT4 :BIT; // C-0-0514.31 -RD- ACTUAL_PHASE_BIT4
END_STRUCT
END_TYPE

```

ML_SYNC

Kurzbeschreibung Der Datentyp enthält die vier Eingangsbits der Steuerung, die im MLC-Verbund ausgetauscht werden.

ML_Base.library

Bibliothek	Bereich
ML_Base	Shared Memory Types

Tab. 4-20: Bibliothekszuordnung

Typdefinition

```
TYPE ML_SYNC :
STRUCT
  iSyncBit0 :BIT; // C-0-0713.00 -RD- Link synchronization bit 0 of control
  iSyncBit1 :BIT; // C-0-0713.01 -RD- Link synchronization bit 1 of control
  iSyncBit2 :BIT; // C-0-0713.02 -RD- Link synchronization bit 2 of control
  iSyncBit3 :BIT; // C-0-0713.03 -RD- Link synchronization bit 3 of control
END_STRUCT
END_TYPE
```

SM_TYPES

Kurzbeschreibung Die Struktur wird z.B. verwendet, um Informationen eines 32-Bit-Datencontainers (AT oder MDT) dem Anwender typgerecht zur Verfügung zu stellen.
Das gewünschte Strukturelement ist durch "." getrennt anzuschließen.

Bibliothek	Bereich
ML_Base	Data Types/ Shared Memory Types

Tab. 4-21: Bibliothekszuordnung

Deklaration des Datentyps:

```
TYPE SM_TYPES:
STRUCT
  REAL_ :REAL; // REAL (32-Bit)
  DINT_ :DINT; // DINT (32-Bit)
  UDINT_ :UDINT; // UDINT (32-Bit)
  UINT_ :UINT; // UINT (16-Bit)
  INT_ :INT ; // INT (16-Bit)
END_STRUCT
END_TYPE
```

 Die 16-Bit-Werte entstehen durch Abschneiden der oberen 16 Bits des Containerwertes.

Beispiel:
AxisData[1].dwUserActualDataB_i.REAL_ bedeutet Achse 1, AT-Datencontainer B, als REAL interpretiert.

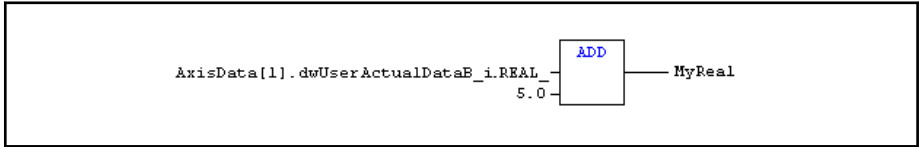


Abb. 4-14: Nutzung des zyklischen SERCOS-Datenkanals zum Lesen

SM_EXT_TYPES

Kurzbeschreibung Die Struktur wird z.B. verwendet, um Informationen eines Parameters dem Anwender typgerecht zur Verfügung zu stellen.
Das gewünschte Strukturelement ist durch "." getrennt anzuschließen.

Bibliothek	Bereich
ML_Base	Data Types/ Shared Memory Types

Tab. 4-22: Bibliothekszuordnung

Typdefinition SM_EXT_TYPES

```

TYPE SM_EXT_TYPES:
STRUCT
    LREAL_      :LREAL; (* LREAL (64-Bit) *)
    REAL_       :REAL;  (* REAL (32-Bit) *)
    DINT_       :DINT;   (* DINT (32-Bit) *)
    UDINT_      :UDINT;  (* UDINT (32-Bit) *)
    UINT_       :UINT;   (* UINT (16-Bit) *)
    INT_        :INT;    (* INT (16-Bit) *)
END_STRUCT
END_TYPE

```

Beispiel:

M-0-0130, Messwert positive Flanke, Messtaster, ist ein Datum der Länge 4Byte/Float.

ProbeData[3].rValuePositiveEdgeProbe.REAL_

bedeutet, dass Messwert, positive Flanke, Messtaster 3, als REAL-Wert zur Verfügung gestellt wird.

Will der Anwender diesen Wert jedoch als LREAL-Wert weiter verarbeiten, muss er nachstehende Interpretation wählen:

ProbeData[3].rValuePositiveEdgeProbe.LREAL_

4.3 Error-/ Diagnostics Memory - Unterstützung von Anwenderdiagnosen und Logbuch-Zugriffen

4.3.1 ML_ReadDiagLog

Kurzbeschreibung Der Funktionsbaustein ML_ReadDiagLog liest Einträge aus dem Diagnose-Logbuch.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Error-/Diagnostics Memory

Tab. 4-23: Bibliothekszuordnung

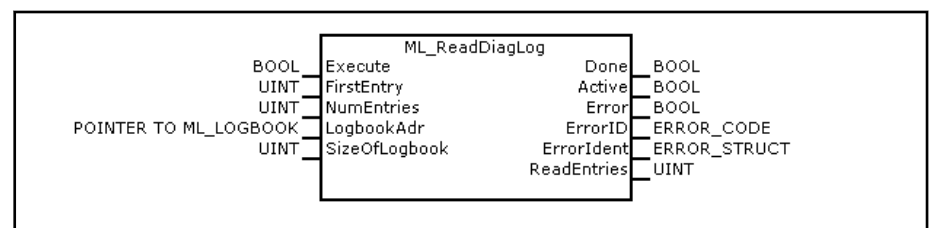
Schnittstellenbeschreibung

Abb. 4-15: Funktionsbaustein ML_ReadDiagLog

	Name	Typ	Kommentar
VAR_INPUT	Execute	UDINT	Bearbeitungsfreigabe bei steigender Flanke
	FirstEntry	UINT	Startindex ab dem das Diagnose-Logbuch gespeichert werden soll
	NumEntries	UINT	Maximale Anzahl der zu lesenden Einträge

ML_Base.library

	Name	Typ	Kommentar
	LogbookAdr	POINTER TO ML_LOGBOOK	Adresse auf dem die Einträge des Diagnose-Logbuchs gespeichert werden sollen siehe Kap. 4.3.7 "ML_LOGBOOK" auf Seite 129
	SizeOfLogbook	UINT	Größe des in LogbookAdr angelegten Datenbereichs in Byte
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler Siehe "Fehlerbehandlung" auf Seite 125
	ReadEntries	UINT	Anzahl der gelesenen Logbucheinträge

Tab. 4-24: Schnittstelle von ML_ReadDiagLog

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
FirstEntry	UINT	0	999	0	steigende Flanke an "Execute"
NumEntries	UINT	1	1000	10	steigende Flanke an "Execute"
LogbookAdr	POINTER TO ML_LOGBOOK	>0		0	steigende Flanke an "Execute"
SizeOfLogbook	UINT	SIZEOF (ML_LOGBOOK)		0	steigende Flanke an "Execute"

Tab. 4-25: Eingangsverhalten ML_ReadDiagLog

Funktionsbeschreibung

Der Funktionsbaustein ML_ReadDiagLog liest aus dem Steuerungslogbuch ab dem in "FirstEntry" angegebenen Eintrag "NumEntries" Einträge aus und überträgt sie in das mit "LogbookAdr" vorgegebene Array.

Die Eingänge "FirstEntry" und "NumEntries" legen den zu lesenden Bereich des Logbuchs fest. Der neuste Eintrag hat dabei die Nummer 0. Das Logbuch kann bis zu 1000 Einträge speichern.

Am Eingang "LogbookAdr" muss die Adresse eines Arrays des Datentyps ML_LOGBOOK zugewiesen werden. Die Adresse kann mit dem ADR()-Operator ermittelt werden. Die Größe des Arrays in Byte wird am Eingang "SizeOfLogbook" übergeben. Diese Größe kann mit dem SIZEOF()-Operator ermittelt werden.

Bei steigender Flanke an "Execute" werden die Eingänge "FirstEntry", "NumEntries", "LogbookAdr" und "SizeOfLogbook" übernommen und die Abarbeitung startet. Während der Abarbeitung ist der Ausgang "Active" gesetzt. Nach erfolgreicher Abarbeitung ist der Ausgang "Done" gesetzt. Die Daten im am Eingang "LogbookAdr" übergebenen Array sind gültig. "ReadEntries" enthält die Anzahl der in das Array übertragenen Logbuch-Einträge.

Tritt während der Abarbeitung ein Fehler auf, so wird der Ausgang "Error" gesetzt. "ErrorID" und "ErrorIdent" enthalten dann weitere Informationen über den Fehler.

Fehlerbehandlung Der Funktionsbaustein ML_ReadDiagLog gibt die folgenden Fehlercodes der F_RELATED_TABLE, ERROR_TABLE, Tabellennummer 16#0170, zurück:

ErrorID	Additional1	Additional2	Beschreibung
INPUT_INVALID_ERROR	16#00002800	16#00000000	Eingang LogbookAdr ist 0
INPUT_INVALID_ERROR	16#00002801	16#00000000	Eingang SizeOfLogbook ist 0
INPUT_INVALID_ERROR	16#00002801	16#00000001	SizeOfLogbook ist zu klein für die angeforderte Anzahl von Logbucheinträgen
INPUT_INVALID_ERROR	16#00002801	16#00000002	SizeOfLogbook ist kein Vielfaches der Struktur ML_LOG-BOOK
INPUT_INVALID_ERROR	16#00002802	16#00000000	Eingang FirstEntry ist größer als 999
INPUT_INVALID_ERROR	16#00002803	16#00000000	Eingang NumEntries ist größer als 1000
INPUT_INVALID_ERROR	16#00002803	16#00000001	Eingang NumEntries ist kleiner als 1

Tab. 4-26: Fehlercodes des Funktionsbausteins ML_ReadDiagLog

4.3.2 ML_SetAxisDiagEntry

Kurzbeschreibung Die Funktion bewirkt einen Eintrag in das Logbuch der Steuerung. Gleichzeitig kommt die Diagnose auf dem Display der Steuerung zur Anzeige.

In Abhängigkeit von der Fehlerklasse wird die über den Eingang "Axis" referenzierte Achse (Antrieb) beeinflusst.



Die Funktion darf **nicht** zyklisch im SPS-Programm aufgerufen werden, da mit jedem Aufruf ein Eintrag in das Logbuch erfolgt und dieses überläuft!

Bibliothek	Bereich
ML_Base_xxx_yyyy	Error-/Diagnostics Memory

Tab. 4-27: Bibliothekszuordnung

Schnittstellenbeschreibung

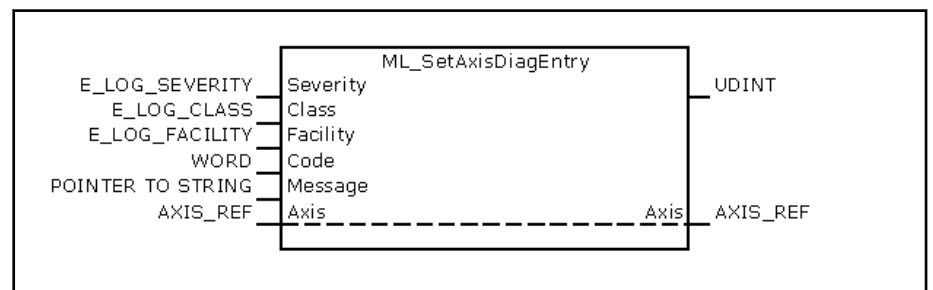


Abb. 4-16: Funktion ML_SetAxisDiagEntry

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Achse
VAR_INPUT	Severity	E_LOG_SEVERITY	Typ des Eintrags: MESSAGE: 16#A, WARNING: 16#E, ERROR: 16#F
	Class	E_LOG_CLASS	Class 0..9, bestimmt die Fehlerreaktion der Steuerung
	Facility	E_LOG_FACILITY	Facility für anwenderdefinierte Einträge zulässig: 16#003B0000

ML_Base.library

	Name	Typ	Kommentar
	Code	WORD	Anwendercode 16#0000 16#FFFF
	Message	POINTER TO STRING	Anwenderdefinierter Texteintrag für das Logbuch. Bitte nicht mehr als 59 Zeichen....
RETURN		UDINT	Kurzhinweis auf die Fehlerursache

Tab. 4-28: Schnittstelle von FUN ML_SetAxisDiagEntry

Funktionsbeschreibung

Die Funktion bewirkt einen Eintrag in das Logbuch der Steuerung. Gleichzeitig kommt die Diagnose auf dem Display der Steuerung zur Anzeige.

Die Diagnose kann eine Meldung oder Warnung sein, beide Arten unabhängig von der Schwere (Severity) ohne Fehlerreaktion.

Bei Fehlern wird durch das System zusätzlich eine Fehlerreaktion entsprechend der Fehlerklasse ausgelöst.

Bei Fehlern wird durch das System in Abhängigkeit von der Fehlerklasse die über den Eingang "Axis" referenzierte Achse (Antrieb) wie folgt beeinflusst:

- F03B....: Nicht fataler Fehler, keine Fehlerreaktion,
- F13B....: Achsgruppenfehler, Stillsetzung der Achse und aller übrigen zur Achsgruppe gehörenden Achsen,
- F23B..../F33B..../F43B....: Achsfehler, die referenzierte Achse wird bestmöglich still gesetzt,
- F53B..../F63B..../F73B....: Steuerungsfehler, **alle** Achsen werden bestmöglich still gesetzt,
- F83B....: Fataler Steuerungsfehler, **alle** Achsen werden bestmöglich still gesetzt.

Beispiel F23B4711, Falsche Drehrichtung, RA1

- **F:** Severity Fehler
- **2:** Class 2, Fehlerreaktion des Systems
 - Logbucheintrag wird generiert, Meldung wird am Display angezeigt.
 - Achse (oder Antrieb) wird bestmöglich stillgesetzt (Geschwindigkeit zu Null), alle anderen Achsen sind davon nicht betroffen.
- **4711:** Nummer entsprechend Anwendervorgabe.
- **Falsche Drehrichtung:** Text in der Anwendertextliste, referenziert über ADR / POINTER TO STRING
- **RA1:** Axis, Auszug aus **MlcVarGlobal** der Steuerung:
(*reale Achse "RA1" mit Achsnummer 1*)
RA1: AXIS_REF :=(CntrlNo:=LOCAL_CNTRL,AxisNo:=AXIS_1);

4.3.3 ML_SetDiagEntry

Kurzbeschreibung

Die Funktion bewirkt einen Eintrag in das Logbuch der Steuerung. Gleichzeitig kommt die Diagnose auf dem Display der Steuerung zur Anzeige.



Die Funktion darf **nicht** zyklisch im SPS-Programm aufgerufen werden, da mit jedem Aufruf ein Eintrag in das Logbuch erfolgt und dieses überläuft!

Bibliothek	Bereich
ML_Base_xxx_yyyy	Error-/Diagnostics Memory

Tab. 4-29: Bibliothekszuordnung

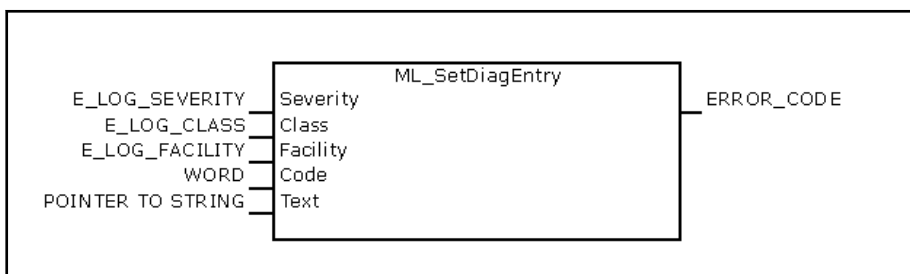
Schnittstellenbeschreibung

Abb. 4-17: Funktion ML_SetDiagEntry

	Name	Typ	Kommentar
VAR_INPUT	Severity	E_LOG_SEVERITY	Typ des Eintrags: MESSAGE: 16#A, WARNING: 16#E, ERROR: 16#F
	Class	E_LOG_CLASS	Class 0..9, bestimmt die Fehlerreaktion der Steuerung
	Facility	E_LOG_FACILITY	Facility für anwenderdefinierte Einträge zulässig: 16#003B0000
	Code	WORD	Anwendercode 16#0000 16#FFFF
	Text	POINTER TO STRING	Anwenderdefinierter Texteintrag für das Logbuch. Bitte nicht mehr als 59 Zeichen....
RETURN		ERROR_CODE	Kurzhinweis auf die Fehlerursache

Tab. 4-30: Schnittstelle von FUN ML_SetDiagEntry

Funktionsbeschreibung

Die Funktion bewirkt einen Eintrag in das Logbuch der Steuerung. Gleichzeitig kommt die Diagnose auf dem Display der Steuerung zur Anzeige.

Die Diagnose kann eine Meldung oder Warnung sein, beide Arten unabhängig von der Schwere (Severity) ohne Fehlerreaktion.

Bei Fehlern wird durch das System zusätzlich eine Fehlerreaktion entsprechend der Fehlerklasse ausgelöst.

Sinnvolle Fehlerklassen

- F03B....: Nicht fataler Fehler, keine Fehlerreaktion,
- F53B..../F63B..../F73B....: Steuerungsfehler, **alle** Achsen werden bestmöglich still gesetzt,
- F83B....: Fataler Steuerungsfehler, **alle** Achsen werden bestmöglich still gesetzt.

Beispiel*F53B0815, Synchronitätsverlust*

- **F**: Severity Fehler
- **5**: Class 5, Fehlerreaktion des Systems
 - Logbucheintrag wird generiert, Meldung wird am Display angezeigt.
 - **Alle** Achsen werden bestmöglich stillgesetzt.
- **0815**: Nummer entsprechend Anwendervorgabe.
- **Synchronitätsverlust**: Text in der Anwendertextliste, referenziert über ADR / POINTER TO STRING

ML_Base.library

4.3.4 E_LOG_CLASS

Kurzbeschreibung Der Aufzählungsdatentyp E_LOG_CLASS beschreibt die Klassen (Fehlerreaktionen) des Diagnosesystems der IndraMotion MLC, sowie der IndraLogic XLC.

Bibliothek	Bereich
ML_Base_xxx_yyy	Error-/Diagnostics Memory

Tab. 4-31: Bibliothekszuordnung

E_LOG_CLASS (ENUM)

```

TYPE E_LOG_CLASS:
(
  CLASS_0      := 16#0, // no error reaction
  CLASS_1      := 16#1, // see online help for error reaction
  CLASS_2      := 16#2, // see online help for error reaction
  CLASS_3      := 16#3, // see online help for error reaction
  CLASS_4      := 16#4, // see online help for error reaction
  CLASS_5      := 16#5, // see online help for error reaction
  CLASS_6      := 16#6, // see online help for error reaction
  CLASS_7      := 16#7, // see online help for error reaction
  CLASS_8      := 16#8  // see online help for error reaction
) DWORD;
END_TYPE

```



Die Klasse "9" steht dem Anwender nicht zur Verfügung.

4.3.5 E_LOG_FACILITY

Kurzbeschreibung Der Aufzählungsdatentyp E_LOG_FACILITY beschreibt die Diagnosegruppen des Diagnosesystems der IndraMotion MLC, sowie der IndraLogic XLC, die für Anwenderdiagnosen freigegeben sind.

Bibliothek	Bereich
ML_Base_xxx_yyy	Error-/Diagnostics Memory

Tab. 4-32: Bibliothekszuordnung

E_LOG_FACILITY (ENUM)

```

TYPE E_LOG_FACILITY:
(
  USER      := 16#003B0000 // PLC user message
) DWORD;
END_TYPE

```

4.3.6 E_LOG_SEVERITY

Kurzbeschreibung Der Aufzählungsdatentyp E_LOG_SEVERITY beschreibt die Kategorien des Diagnosesystems der IndraMotion MLC, sowie der IndraLogic XLC, die für Anwenderdiagnosen freigegeben sind.

Bibliothek	Bereich
ML_Base_xxx_yyy	Error-/Diagnostics Memory

Tab. 4-33: Bibliothekszuordnung

E_LOG_SEVERITY (ENUM)

```

TYPE E_LOG_SEVERITY:
(
  MESSAGE := 16#A, // Diagnostic identifier for Message "A"
  WARNING := 16#E, // Diagnostic identifier for Warning "E"
  ERROR   := 16#F  // Diagnostic identifier for Error "F"
)

```

```
) DWORD;
END_TYPE
```



Kommando-Fehler (Command Error) können vom Anwender nicht ausgegeben werden.

4.3.7 ML_LOGBOOK

Kurzbeschreibung Diese Struktur enthält einen Eintrag des Diagnose-Logbuchs. Sie wird vom Funktionsbaustein ML_ReadDiagLog verwendet.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Data Types/Info

Tab. 4-34: Bibliothekszuordnung

Strukturelement	Typ	Beschreibung
ErrorNumber	DWORD	Fehlernummer
State	UINT	Status des Diagnoseeintrags
LogTime	SYS_TIME_DATE	Zeitpunkt der Diagnose
DiagText	STRING(123)	Diagnosetext

Tab. 4-35: Datenstruktur ML_LOGBOOK

4.4 Globale Variables List - Globale Variablen-Liste

4.4.1 GVL_Base

Kurzbeschreibung Die Globale Variablenliste **GVL_Base** enthält den Datenbestand, der einer IndraLogic XLC / IndraMotion MLC zu zuordnen ist.

Nicht enthalten sind Daten, die sich auf Achsgruppen (Robot Control) beziehen (siehe Bibliothek ML_Robot.library).

Die in **GVL_Base** enthaltenen Konstanten sind in der Globalen Liste **GVL_BaseConstant**, Seite 130, vereinbart.

Bibliothek	Bereich
ML_Base	Global Variables Lists

Tab. 4-36: Bibliothekszuordnung

Deklaration der Globalen Variablenliste GVL_Base

```
VAR GLOBAL
  NoAxis:      AXIS_REF:=      (CntlNo:= LOCAL_CNTRL, AxisNo:= NO_OBJECT);
  NoTouchProbe: TOUCHPROBE_REF:= (CntlNo:= LOCAL_CNTRL, TouchProbeNo:= NO_TOUCHPROBE);

  AxisData      AT %MBaaa: ARRAY[MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_AXISDATA_SM;
  ControlData    AT %MBbbb: ARRAY[MB_MIN_CONTROL_NUMBER..MB_MAX_CONTROL_NUMBER] OF ML_CONTROLDATA_SM;
  PlsData        AT %MBccc: ARRAY[MB_MIN_PLS_NUMBER..MB_MAX_PLS_NUMBER] OF ML_PLSDATA_SM;
  ProbeData      AT %MBddd: ARRAY[MB_MIN_PROBE_NUMBER..MB_MAX_PROBE_NUMBER] OF ML_PROBEDATA_SM;

  DV_Osci:       ARRAY[MB_MIN_OSCI_NUMBER..MB_MAX_OSCI_NUMBER] OF ML_DirectVarOsci;
  DV_Probe:      ARRAY[MB_MIN_PROBE_NUMBER..MB_MAX_PROBE_NUMBER] OF ML_DirectVarProbe;
  DV_Pls:        ARRAY[MB_MIN_PLS_NUMBER..MB_MAX_PLS_NUMBER] OF ML_DirectVarPls;
  DV_Control:    ARRAY[MB_MIN_CONTROL_NUMBER..MB_MAX_CONTROL_NUMBER] OF ML_DirectVarControl;
  DV_Axis:       ARRAY[MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_DirectVarAxis;

  GV_LinkAxis:   ARRAY[MB_MIN_GLOBAL_CONTROL_NUMBER..MB_MAX_GLOBAL_CONTROL_NUMBER,
                      MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF ML_GlobalVarAxis;

  GlobalVarDirectVarData: ML_GlobalVarDirectVarData;
```


ML_Base.library

```
BaseDeviceSetup: BaseDeviceSetup;  
END_VAR
```



Die Adressen von "AxisData", "ControlData", "PlsData" und "ProbeData" entnehmen Sie bitte dem Bibliotheksverwalter der ml_Base_xxx_yyy, GVL_Base, Dokumentation.
xxx: mlc | xlc
yyy: l65 | l45 | l25

NoAxis	Standard-Initialwert
NoTouchProbe	Standard-Initialwert
GV_LinkAxis	GV_LinkAxis[] - Achsdaten über Funktionsvariablenzugriff synchron im C2C-Verbund , Seite 96
GlobalVarDirectVarData	Interne Verwendung
Achsdaten über Shared Memory und Funktionsvariablenzugriff, Seite 89	
AxisData	AxisData[] - Achsdaten im Shared Memory, Seite 91
DV_Axis	DV_Axis[] - Achsdaten über Funktionsvariablenzugriff, Seite 94
Steuerungsdaten über Shared Memory und Funktionsvariablenzugriff, Seite 98	
ControlData	ControlData[] - Steuerungsdaten im Shared Memory, Seite 100
DV_Control	ControlData[] - Steuerungsdaten im Shared Memory, Seite 100
Nockenschaltwerkdaten über Shared Memory und Funktionsvariablenzugriff, Seite 104	
PlsData	PlsData[] - Nockenschaltwerkdaten im Shared Memory, Seite 105
DV_Pls	DV_Pls[] - Nockenschaltwerkdaten über Funktionsvariablenzugriff, Seite 107
Messtasterdaten über Shared Memory und Funktionsvariablenzugriff, Seite 109	
ProbeData	ProbeData[] - Messtasterdaten im Shared Memory , Seite 110
DV_Probe	DV_Probe[] - Messtasterdaten über Funktionsvariablenzugriff, Seite 112
Oszilloskopdaten über Funktionsvariablenzugriff, Seite 115	
DV_Osci	DV_Osci[] - Oszilloskopdaten über Funktionsvariablenzugriff, Seite 115

Tab. 4-37: Komponenten der GVL_Base

4.4.2 GVL_BaseConstant

- Kurzbeschreibung
- Die Globale Liste **GVL_BaseConstant** enthält die wesentlichen globalen Konstanten einer
 - IndraLogic XLC mit und ohne sercos Interface und
 - IndraMotion MLC.

Bibliothek	Bereich
ML_Base_xxx_yyy ³⁾	Global Variable Lists

Tab. 4-38: Bibliothekszuordnung

3) xxx: XLC oder MLC; yyy: L25, L45, L65 oder Vxx

VAR_GLOBAL_CONSTANT	Type	Comment	XLC			MLC		
			L25	L45	L65	L25	L45	L65
IL_TARGET_TYPE	UINT	XLC // MLC	1	1	1	1	1	1
IL_TARGET_VERSION	UINT	XLC 13 VRS // MLC 13VRS	13	13	13	13	13	13
IL_SYSTEM_MLC // IL_SYSTEM_XLC	UINT	System XLC // System MLC	1	1	1	1	1	1
MB_MIN_OSCI_NUMBER	INT	min. number of oscilloscopes	1	1	1	1	1	1
MB_MAX_OSCI_NUMBER	INT	max. number of oscilloscopes	16	16	16	16	16	16
MB_NUMBER_OF_OSCI	INT		16	16	16	16	16	16
MB_MIN_PROBE_NUMBER	INT	min. number of touch probes	1	1	1	1	1	1
MB_MAX_PROBE_NUMBER	INT	max. number of touch probes	100	100	100	100	100	100
MB_NUMBER_OF_PROBE	INT		100	100	100	100	100	100
MB_MIN_PLS_NUMBER	INT	min. number of PLS FMs	1	1	1	1	1	1
MB_MAX_PLS_NUMBER	INT	max. number of PLS FMs	2	4	4	2	4	4
MB_NUMBER_OF_PLS	INT		2	4	4	2	4	4
MB_MIN_CONTROL_NUMBER	INT	min. number of controls	0	0	0	0	0	0
MB_MAX_CONTROL_NUMBER	INT	max. number of controls	0	0	0	0	0	0
MB_NUMBER_OF_CONTROL	INT		1	1	1	1	1	1
MB_MIN_AXIS_NUMBER	INT	min. number of axis	1	1	1	1	1	1
MB_MAX_AXIS_NUMBER	INT	max. number of axis	16	32	64	16	32	64
MB_NUMBER_OF_AXIS	INT		16	32	64	16	32	64
MB_MIN_GLOBAL_CONTROL_NUMBER	INT	min. number of global controls	0	0	0	0	0	0
MB_MAX_GLOBAL_CONTROL_NUMBER	INT	max. number of global controls	0	0	0	64	64	64
MB_NUMBER_OF_GLOBAL_CONTROL	INT		0	0	0	65	65	65
MB_MIN_GLOBAL_AXIS_NUMBER	INT	min. number of global axis	0	0	0	1	1	1
MB_MAX_GLOBAL_AXIS_NUMBER	INT	max. number of global axis	0	0	0	64	64	64
MB_NUMBER_OF_GLOBAL_AXIS	INT		0	0	0	64	64	64
ML_MAX_NUMBER_OF_APPLICATIONS	UINT	max. number of plc applications	1	1	1	1	1	1
ML_MIN_IDX_ARRAY_EXT_ATTRIBUTE	UDINT		0	0	0	0	0	0
ML_MAX_IDX_ARRAY_EXT_ATTRIBUTE	UDINT		3	3	3	3	3	3
ML_LOG_ID_PLC_BASE	UDINT		1	1	1	1	1	1
ML_LOG_ID_PLC_POU	UDINT		2	2	2	2	2	2
ML_LOCAL_SERCOS_IFC ¹⁾	INT	onboard sercos interface	0	0	0	0	0	0
ML_MIN_SERCOS_IFC ¹⁾	INT	min. number of sercos interfaces	0	0	0	0	0	0
ML_MAX_SERCOS_IFC ¹⁾	INT	max. number of sercos interfaces	0	0	0	0	0	0
ML_NUMBER_OF_SERCOS_IFC ¹⁾	INT	one sercos interface available	1	1	1	1	1	1
END_VAR								

¹⁾ In Entwicklung

Abb. 4-18: Deklaration der Globalen Liste GVL_BaseConstant, Initialwerte der Systeme

4.4.3 Global Variables Lists/ Global Variable Index

GVL_A_GlobalVarIdx

Kurzbeschreibung

Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter in der Verwendung "Status bei Funktionsvariablenzugriff synchron im C2C-Verbund".

Der Parameter-Konstantenname ergibt sich zu

A-0-0021 ⇒ GV_A_0_0021 usw..

ML_Base.library

Nummer und Parameter-Konstantenname können alternativ verwendet werden.

Bibliothek	Bereich
ML_Base	Global Variables Lists/ Global Variable Index

Tab. 4-39: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_A_GlobalVarIdx

```

VAR GLOBAL CONSTANT
  GV_A_0_0021: UDINT:= 0 ; (* Axis status *)
  GV_A_0_0045: UDINT:= 1 ; (* Modulo value *)
  GV_A_0_0046: UDINT:= 2 ; (* Feed distance of the slave drive *)
  GV_A_0_0101: UDINT:= 3 ; (* Actual position value, 8 bytes *)
  GV_A_0_0102: UDINT:= 4 ; (* Actual velocity value *)
  GV_A_0_2121: UDINT:= 5 ; (* Axis status *)
  GV_A_0_2800: UDINT:= 6 ; (* Master axis encoder, configuration *)

  GV_A_MIN_IDX: UDINT:= 0 ;
  GV_A_MAX_IDX: UDINT:= 6 ;
  GV_A_MAX_CNT: UDINT:= 7 ;
END_VAR

```

4.4.4 Parameter-Konstanten

Zugriff auf Daten einer Achse, Steuerung oder eines anderen Objektes

Aufbau und Zugriff auf Parameter aus dem SPS-Programm

Die Daten einer Achse, einer Steuerung oder eines Nockenschaltwerkes, sind über Parameterzugriff lesend und teilweise auch schreibend erreichbar. Folgende Parameterarten werden bei der IndraMotion MLC unterschieden:

- C-Parameter (control parameters), die sich auf eine konkrete Steuerung, die die Achsen beeinflusst, beziehen.
- A-Parameter (axis parameters), die sich auf jeweils eine Achse beziehen.
- K-Parameter (kinematics parameters), die sich auf eine Kinematik (Achsgruppe) beziehen, siehe ML_Robot.library.
- M-Parameter (touch probe parameters), die sich auf jeweils einen Mess-taster beziehen. Sie sind nur über PLCopen-Funktionsbausteine zu erreichen.
- N-Parameter (pls parameters), die sich auf jeweils ein Nockenschaltwerk beziehen.
- O-Parameter (oscilloscope parameters), die sich auf jeweils eine Oszilloskop-Funktion beziehen.

Zusätzlich zu diesen Parametern existieren weitere Gruppen, die auf Geräte an einem bestimmten Bus (S-Parameter, SERCOS-Interface) oder auf ein bestimmtes Produkt (P-Parameter) beziehen.

Alle IndraMotion-Parameter werden nach dem gleichen Bildungsgesetz aufgebaut:

Bsp.: A-0-0001 , mit "A" 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.

Dem Anwender stehen dazu für A-, ..., O-, P- und S-Parameter Vereinbarungsdateien für Globale Konstanten zur Verfügung.

Vereinbarungsdateien für Globale Konstanten

- GVL_A_Param_ID, Seite 134,
- GVL_C_Param_ID, Seite 134,
- GVL_M_Param_ID, Seite 135,
- GVL_N_Param_ID, Seite 135,
- GVL_O_Param_ID, Seite 136,
- GVL_P_Param_ID, Seite 136,
- GVL_S_Param_ID, Seite 137.

Bei Kenntnis des Namens des Parameters:

A-Parameter-Adresskonstanten

```
VAR_GLOBAL CONSTANT
```

```
  FP_A_0_0000: MB_IDN:= 4294967296 ; (* Dummy A parameter *)
  FP_A_0_0001: MB_IDN:= 4294967297 ; (* Logical axis number *)
  FP_A_0_0002: MB_IDN:= 4294967298 ; (* Axis name *)
  (* ..... *)
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
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. 4-40: *Parameter-Offsets*

Mit der Steuerung können mehrere Achsen angesteuert werden. Aus diesem Grund besitzen die Bausteine einen Adresseingang (VAR_IN_OUT) vom Typ AXIS_REF, RIL_CommonTypes.library, der wie folgt zu belegen ist:

Zugriff auf A-Parameter:

- Der Adressbezeichner wird durch IndraWorks ML* automatisch generiert (Achsenname/Achsennummer), Typ AXIS_REF.

Zugriff auf N-Parameter:

- Der Adressbezeichner wird durch IndraWorks ML* automatisch generiert oder kann ersatzweise durch den Anwender definiert werden (Objektname/ Objektnummer), Typ AXIS_REF.

Zugriff auf O-Parameter:

- Der Adressbezeichner wird durch IndraWorks ML* automatisch generiert oder kann ersatzweise durch den Anwender definiert werden (Objektname/ Objektnummer), Typ AXIS_REF.

ML_Base.library

Zugriff auf C-Parameter:

- Verwendung des Adressbezeichners einer beliebigen Achse dieser Steuerung oder Deklaration mit den Werten.
 - ContrlNo:= LOCAL_CNTRL bzw. CNTRL_1...
 - AxisNo:= NO_OBJECT.

Referenzen am Axis-Eingang, Bsp.: MB_ReadParameter

```
VA1:AXIS_REF:=(CntrlNo:=LOCAL_CNTRL,AxisNo:=AXIS_1)
(*A parameters, axis VA1, axis number 1*);

MLC_A: AXIS_REF:=(CntrlNo:=LOCAL_CNTRL,AxisNo:=NO_OBJECT)
(* C parameters, control MLC_A *);

PLS_A: AXIS_REF:=(CntrlNo:=LOCAL_CNTRL,AxisNo:=PLS_1)
(*N parameters, programmable limit switch PLS_A, obj.no.1*);

OSZ1: AXIS_REF:=(CntrlNo:=LOCAL_CNTRL,AxisNo:=OSCI_CHANNEL_1)
(*O parameters, oscilloscope OSZ1, obj.no.1*);
```

GVL_A_Param_ID

Kurzbeschreibung Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.

Der Parameter-Konstantenname ergibt sich zu
A-0-0001 ⇒ FP_A_0_0001 usw..

Nummer und Parameter-Konstantenname können alternativ verwendet werden.

Bibliothek	Bereich
ML_Base	Global Variables Lists/ Parameter ID

Tab. 4-41: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_A_Param_ID

```
VAR GLOBAL CONSTANT
  FP_A_0_0000: MB_IDN:= 4294967296 ; (* Dummy A parameter *)
  FP_A_0_0001: MB_IDN:= 4294967297 ; (* Logical axis number *)
  FP_A_0_0002: MB_IDN:= 4294967298 ; (* Axis name *)
  (* ..... *)
END_VAR
```

Siehe auch:

- [MB_IDN, Seite 81.](#)

GVL_C_Param_ID

Kurzbeschreibung Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.

Der Parameter-Konstantenname ergibt sich zu
C-0-0001 ⇒ FP_C_0_0001 usw..

Nummer und Parameter-Konstantenname können alternativ verwendet werden.

Bibliothek	Bereich
ML_Base	Global Variables Lists/ Parameter ID

Tab. 4-42: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_C_Param_ID

```

VAR_GLOBAL CONSTANT
  FP_C_0_0000: MB_IDN:= 8589934592 ; (* Dummy C parameter *)
  FP_C_0_0001: MB_IDN:= 8589934593 ; (* Language selection *)
  FP_C_0_0002: MB_IDN:= 8589934594 ; (* Name of parameter file *)
  (* ..... *)
END_VAR

```

Siehe auch:

- [MB_IDN, Seite 81.](#)

GVL_M_Param_ID

Kurzbeschreibung Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.

Der Parameter-Konstantenname ergibt sich zu
M-0-0001 ⇒ FP_M_0_0001 usw..

Nummer und Parameter-Konstantenname können alternativ verwendet werden.

Bibliothek	Bereich
ML_Base	Global Variables Lists/ Parameter ID

Tab. 4-43: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_M_Param_ID

```

VAR_GLOBAL CONSTANT
  FP_M_0_0000: MB_IDN:= 193273528320 ; (* Dummy M parameter *)
  FP_M_0_0001: MB_IDN:= 193273528321 ; (* Logical probe number *)
  FP_M_0_0002: MB_IDN:= 193273528322 ; (* Probe name *)
  (* ..... *)
END_VAR

```

Siehe auch:

- [MB_IDN, Seite 81.](#)

GVL_N_Param_ID

Kurzbeschreibung Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.

Der Parameter-Konstantenname ergibt sich zu
N-0-0001 ⇒ FP_N_0_0001 usw..

Nummer und Parameter-Konstantenname können alternativ verwendet werden.

ML_Base.library

Bibliothek	Bereich
ML_Base	Global Variables Lists/ Parameter ID

Tab. 4-44: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_N_Param_ID

```
VAR_GLOBAL CONSTANT
  FP_N_0_0000: MB_IDN:= 197568495616 ; (* Dummy N parameter *)
  FP_N_0_0001: MB_IDN:= 197568495617 ; (* Logical number, PLS *)
  FP_N_0_0002: MB_IDN:= 197568495618 ; (* Name, PLS *)
  (* ..... *)
END_VAR
```

Siehe auch:

- [MB_IDN, Seite 81.](#)

GVL_O_Param_ID

Kurzbeschreibung Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.
Der Parameter-Konstantenname ergibt sich zu
O-0-0001 ⇒ FP_O_0_0001 usw..
Nummer und Parameter-Konstantenname können alternativ verwendet werden.

Bibliothek	Bereich
ML_Base	Global Variables Lists/ Parameter ID

Tab. 4-45: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_O_Param_ID

```
VAR_GLOBAL CONSTANT
  FP_O_0_0000: MB_IDN:= 201863462912 ; (* Dummy O parameter *)
  FP_O_0_0001: MB_IDN:= 201863462913 ; (* Logical channel number, Oscilloscope *)
  FP_O_0_0002: MB_IDN:= 201863462914 ; (* Channel name, Oscilloscope *)
  (* ..... *)
END_VAR
```

Siehe auch:

- [MB_IDN, Seite 81.](#)

GVL_P_Param_ID

Kurzbeschreibung Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.
Der Parameter-Konstantenname ergibt sich zu
P-0-0001 ⇒ FP_P_0_0001 usw..
Nummer und Parameter-Konstantenname können alternativ verwendet werden.

Bibliothek	Bereich
ML_Base	Global Variables Lists/ Parameter ID

Tab. 4-46: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_P_Param_ID

```

VAR_GLOBAL CONSTANT
  FP_P_0_0001: MB_IDN:= 32769 ; (* Switching frequency of the power output stage *)
  FP_P_0_0002: MB_IDN:= 32770 ; (* SPI flash aging counter *)
  FP_P_0_0003: MB_IDN:= 32771 ; (* Status of parameter buffering *)
  FP_P_0_0004: MB_IDN:= 32772 ; (* Velocity loop smoothing time constant *)
  (* ..... *)
END_VAR

```

Siehe auch:

- [MB_IDN, Seite 81.](#)

GVL_S_Param_ID**Kurzbeschreibung**

Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.

Der Parameter-Konstantenname ergibt sich zu

S-0-0001 ⇒ FP_S_0_0001 usw..

Nummer und Parameter-Konstantenname können alternativ verwendet werden.

Bibliothek	Bereich
ML_Base	Global Variables Lists/ Parameter ID

Tab. 4-47: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_S_Param_ID

```

VAR_GLOBAL CONSTANT
  FP_S_0_0000: MB_IDN:= 0 ; (* Dummy S parameter *)
  FP_S_0_0001: MB_IDN:= 1 ; (* NC cycle time (TNcyc) *)
  FP_S_0_0002: MB_IDN:= 2 ; (* SERCOS cycle time (TScyc) *)
  (* ..... *)
END_VAR

```

Siehe auch:

- [MB_IDN, Seite 81.](#)

4.5 Interfaces, Funktionen und Funktionsbausteine

4.5.1 IL_SercosAttribute

Kurzbeschreibung

Der Funktionsbaustein IL_SercosAttribute dient der Aufschlüsselung der Elemente des Sercos-Attributes.

ML_Base.library

Schnittstellenbeschreibung

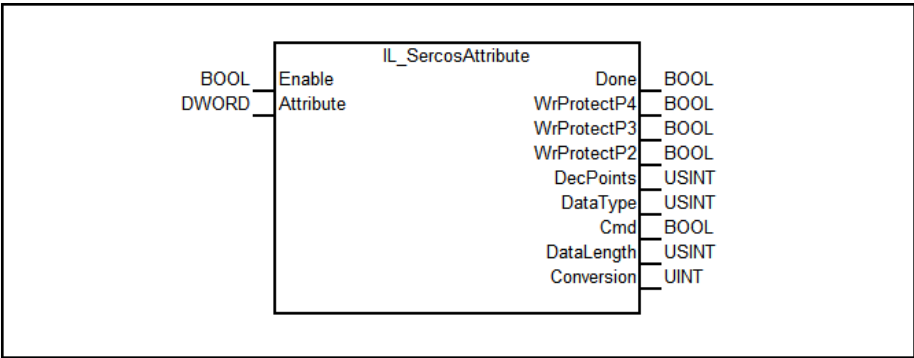


Abb. 4-19: Funktionsbaustein IL_SercosAttribute

	Name	Datentyp	Beschreibung
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe der Funktion (zyklisch, zustandsgesteuert)
	Attribute	DWORD	Sercos-Attribut
VAR_OUPUT	Done	BOOL	Aufschlüsselung des Sercos Attributs abgeschlossen, Ausgangsvariablen gültig
	WrProtectP4	BOOL	Betriebsdatum des Parameters im Betriebsmodus (Sercos Phase 4 - P4) schreibgeschützt
	WrProtectP3	BOOL	Betriebsdatum des Parameters in Sercos Phase 3 (P3) schreibgeschützt
	WrProtectP2	BOOL	Betriebsdatum des Parameters im Parametriermodus (Sercos Phase 2 - P2) schreibgeschützt
	DecPoints	USINT	Anzahl der Nachkommastellen des Betriebsdatums des Parameters
	DataType	USINT	Datentyp und Anzeigeformat des Betriebsdatums des Parameters
	Cmd	BOOL	Betriebsdatum ruft Kommandofunktion auf
	DataLength	USINT	Datenlänge des Betriebsdatums des Parameters
	Conversion	UINT	Bewertungsfaktor des Betriebsdatums des Parameters

Tab. 4-48: Schnittstelle IL_SercosAttribute

Sercos-Attribut Die folgende Tabelle gibt den Aufbau des Sercos-Attributs wieder.

Bit	Schnittstellenvariable	Beschreibung
30:	WrProtectP4	Schreibschutz des Parameter-Betriebsdatums im Betriebsmodus (Sercos-Phase 4)
		0: Betriebsdatum ist beschreibbar 1: Betriebsdatum ist schreibgeschützt
29:	WrProtectP3	Schreibschutz des Betriebsdatums des Parameters (Sercos-Phase 3)
		0: Betriebsdatum ist beschreibbar 1: Betriebsdatum ist schreibgeschützt
28:	WrProtectP2	Schreibschutz des Parameter-Betriebsdatums im Parametriermodus (Sercos-Phase 2)
		0: Betriebsdatum ist beschreibbar 1: Betriebsdatum ist schreibgeschützt

Bit	Schnittstellenvariable	Beschreibung																		
27..24:	DecPoints	Die Nachkommastellen legen die Stellung des Kommas für die Anzeige und Eingabe des betreffenden Betriebsdatums fest.																		
		0000: Keine Nachkommastellen 1111: 15 Nachkommastellen (Maximum)																		
23:		Reserviert																		
22..20:	DataType	Datentyp und Anzeigeformat wird benutzt, um das Betriebsdatum sowie den minimalen - und maximalen Eingabewert in das richtige Anzeigeformat umzuwandeln.																		
		<table><tr><th>Datentyp</th><th>Anzeigeformat</th></tr><tr><td>000: Binärzahl</td><td>binär</td></tr><tr><td>001: Vorzeichenlose ganze Zahl</td><td>Dezimalzahl ohne Vorzeichen</td></tr><tr><td>010: Ganze Zahl</td><td>Dezimalzahl mit Vorzeichen</td></tr><tr><td>011: Vorzeichenlose ganze Zahl</td><td>hexadezimal</td></tr><tr><td>100: Erweiterter Zeichensatz</td><td>Text</td></tr><tr><td>101: Vorzeichenlose ganze Zahl</td><td>Parameter-Identnummerformat (Sercos-Norm, 2 Byte)</td></tr><tr><td>110: Gleitkommazahl</td><td>Dezimalzahl mit Vorzeichen</td></tr><tr><td>111: Vorzeichenlose ganze Zahl</td><td>Parameter-Identnummerformat (Sercos-Norm, 4 Byte)</td></tr></table>	Datentyp	Anzeigeformat	000: Binärzahl	binär	001: Vorzeichenlose ganze Zahl	Dezimalzahl ohne Vorzeichen	010: Ganze Zahl	Dezimalzahl mit Vorzeichen	011: Vorzeichenlose ganze Zahl	hexadezimal	100: Erweiterter Zeichensatz	Text	101: Vorzeichenlose ganze Zahl	Parameter-Identnummerformat (Sercos-Norm, 2 Byte)	110: Gleitkommazahl	Dezimalzahl mit Vorzeichen	111: Vorzeichenlose ganze Zahl	Parameter-Identnummerformat (Sercos-Norm, 4 Byte)
		Datentyp	Anzeigeformat																	
		000: Binärzahl	binär																	
		001: Vorzeichenlose ganze Zahl	Dezimalzahl ohne Vorzeichen																	
		010: Ganze Zahl	Dezimalzahl mit Vorzeichen																	
		011: Vorzeichenlose ganze Zahl	hexadezimal																	
		100: Erweiterter Zeichensatz	Text																	
		101: Vorzeichenlose ganze Zahl	Parameter-Identnummerformat (Sercos-Norm, 2 Byte)																	
		110: Gleitkommazahl	Dezimalzahl mit Vorzeichen																	
111: Vorzeichenlose ganze Zahl	Parameter-Identnummerformat (Sercos-Norm, 4 Byte)																			
19:	Cmd	Die Funktion des Betriebsdatums zeigt an, dass mit diesem Betriebsdatum eine Kommandofunktion im Antrieb aufgerufen wird.																		
		0: Betriebsdatum oder Parameter 1: Kommando																		
18..16:	DataLength	Die Datenlänge wird von der Steuerung für das ordnungsgemäße Beenden der Datenübertragung über den Servicekanal benötigt.																		
		000: reserviert 001: Betriebsdatum ist 2 Byte lang 010: Betriebsdatum ist 4 Byte lang 011: reserviert 100: Variable Länge mit 1-Byte-Daten 101: Variable Länge mit 2-Byte-Daten 110: Variable Länge mit 4-Byte-Daten 111: reserviert																		
15..0:	Conversion	Der Bewertungsfaktor ist eine vorzeichenlose ganze Zahl, die dazu dient numerische Daten in ein Anzeigeformat umzuwandeln. Der Bewertungsfaktor ist auf den Wert von 1 gesetzt, wenn er für die Datenanzeige nicht verwendet wird.																		

Tab. 4-49: Elemente des Sercos-Attributs

ML_Base.library

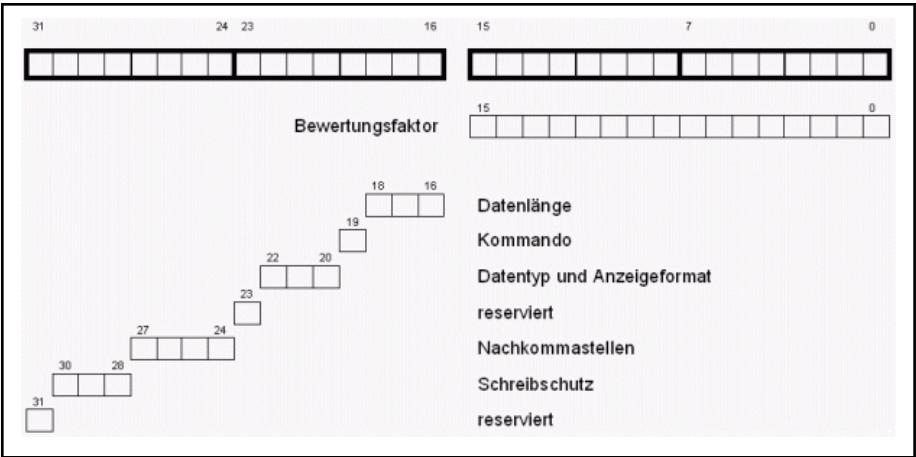


Abb. 4-20: Aufbau des Sercos-Attributs

Funktionsbeschreibung Der Funktionsbaustein IL_SercosAttribute interpretiert nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Enable" zyklisch das am Eingang "Attribute" angelegte Sercos-Attribut, die Gültigkeit der Elemente an den entsprechenden Ausgängen wird durch den Ausgang "Done" signalisiert.

Fehlerbehandlung Für diesen Funktionbaustein ist keine Fehlerbehandlung verfügbar.

4.5.2 IL_SercosElementsToParaNo

Kurzbeschreibung Die Funktion IL_SercosElementsToParaNo dient der Zusammensetzung der Einzelelemente eines SERCOS Parameterstrings und der gleichzeitigen Umwandlung zur SERCOS-Parameter-Identnummer.

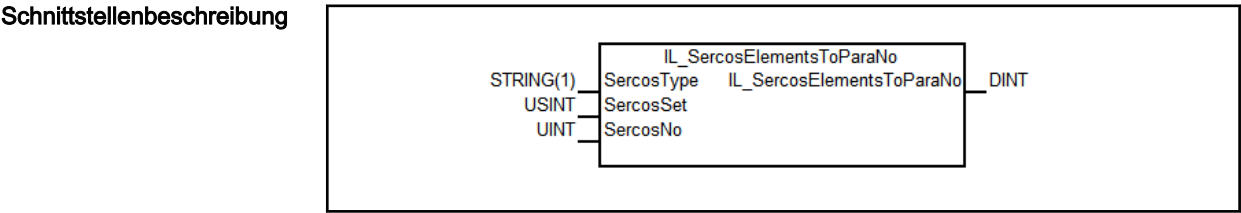


Abb. 4-21: Funktion IL_SercosElementsToParaNo

	Name	Datentyp	Beschreibung
VAR_INPUT	SercosType	STRING[1]	Parametertyp
	SercosSet	USINT	Parametersatz
	SercosNo	UINT	Parameternummer
RETURN	IL_SercosElements-ToParaNo	UDINT	Parameter-Identnummer

Tab. 4-50: Schnittstelle IL_SercosElementsToParaNo

Spezifikation Die Funktion IL_SercosElementsToParaNo erlaubt das Zusammen-
setzung folgender Einzelelemente eines SERCOS Parameterstrings:

- SercosType: "S", "P", "A", "C", "Y", "T", "s", "p", "a", "c", "y", "t"
- SercosSet: 0...7
- SercosNo: 0...4095

Funktionsbeschreibung Die Funktion IL_SercosElementsToParaNo setzt die angeschlossenen Ein-
gangelemente kontinuierlich zur entsprechenden SERCOS Parameter-
Identnummer zusammen.

Fehlerbehandlung

Die Funktion IL_SercosElementsToParaNo generiert zur Laufzeit keinen Fehler, bei ungültigen Einzelementen an den Eingängen (siehe Spezifikation) liefert die Funktion "-1" zurück.

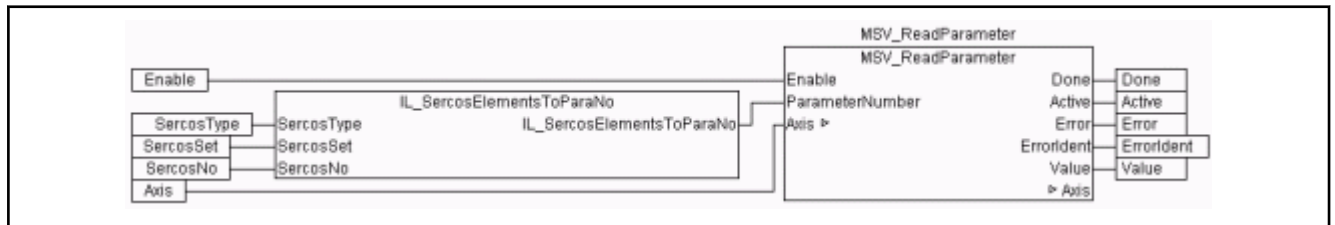
Anwendungsbeispiel

Abb. 4-22: Anwendung der Funktion (CFC): IL_SercosElementsToParaNo

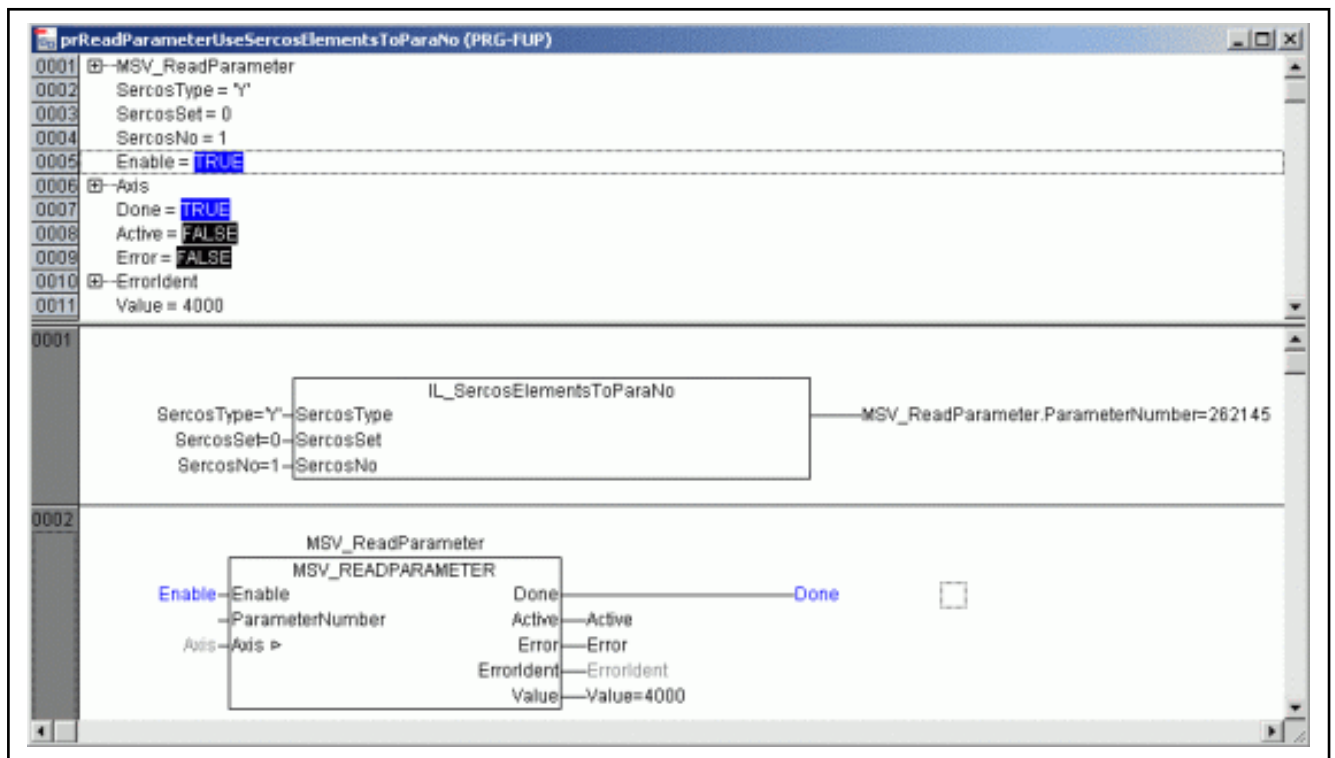


Abb. 4-23: Anwendung der Funktion (FUP): IL_SercosElementsToParaNo

4.5.3 IL_SercosStringToParaNo

Kurzbeschreibung

Die Funktion IL_SercosStringToParaNo dient der Umwandlung eines SERCOS Parameterstrings zur SERCOS Parameter-Identnummer.

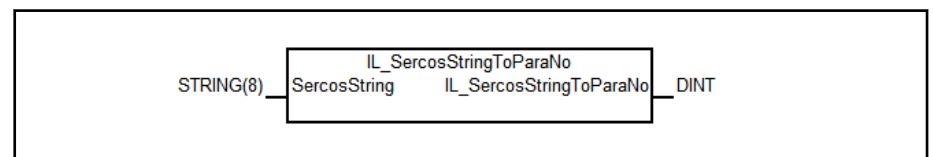
Schnittstellenbeschreibung

Abb. 4-24: Funktion IL_SercosStringToParaNo

ML_Base.library

	Name	Datentyp	Beschreibung
VAR_INPUT	SercosString	STRING[8]	SERCOS-Parameterstring
RETURN	IL_SercosStringToParaNo	UDINT	Parameter-Identnummer

Tab. 4-51: Schnittstelle IL_SercosStringToParaNo

- Spezifikation

Die Funktion IL_SercosStringToParaNo erlaubt die Umwandlung eines SERCOS-Parameterstrings mit folgenden Stringelementen (Formatierung: "X Y ZZZZ" bzw. "XZZZZ"):
- SercosType "X": "S", "P", "A", "C", "Y", "T", "s", "p", "a", "c", "y", "t"
 - SercosSet "Y": "0"... "7"
 - SercosNo "ZZZZ": "0"... "4095"
 - Seperator: "-"
- Funktionsbeschreibung

Die Funktion IL_SercosElementsToParaNo setzt die angeschlossenen Eingangselemente kontinuierlich zur entsprechenden SERCOS Parameter-Identnummer zusammen.
- Fehlerbehandlung

Die Funktion IL_SercosStringToParaNo generiert zur Laufzeit keinen Fehler, bei einem ungültigen SERCOS Parameterstring (siehe Spezifikation) liefert die Funktion "-1" zurück.

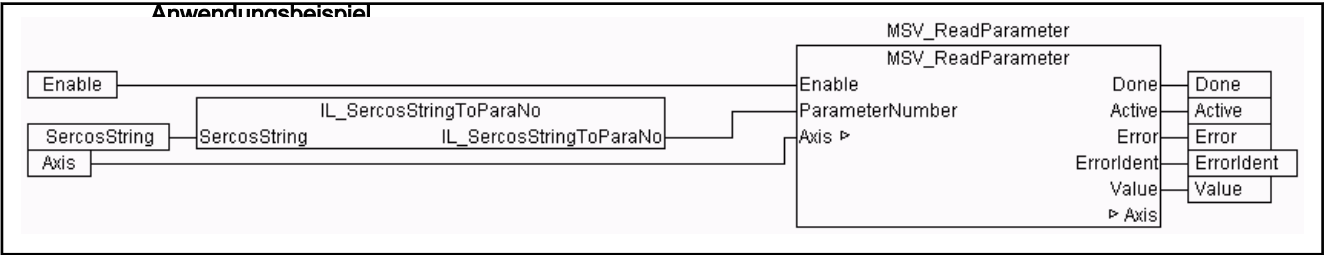


Abb. 4-25: Anwendung der Funktion (CFC): IL_SercosStringToParaNo

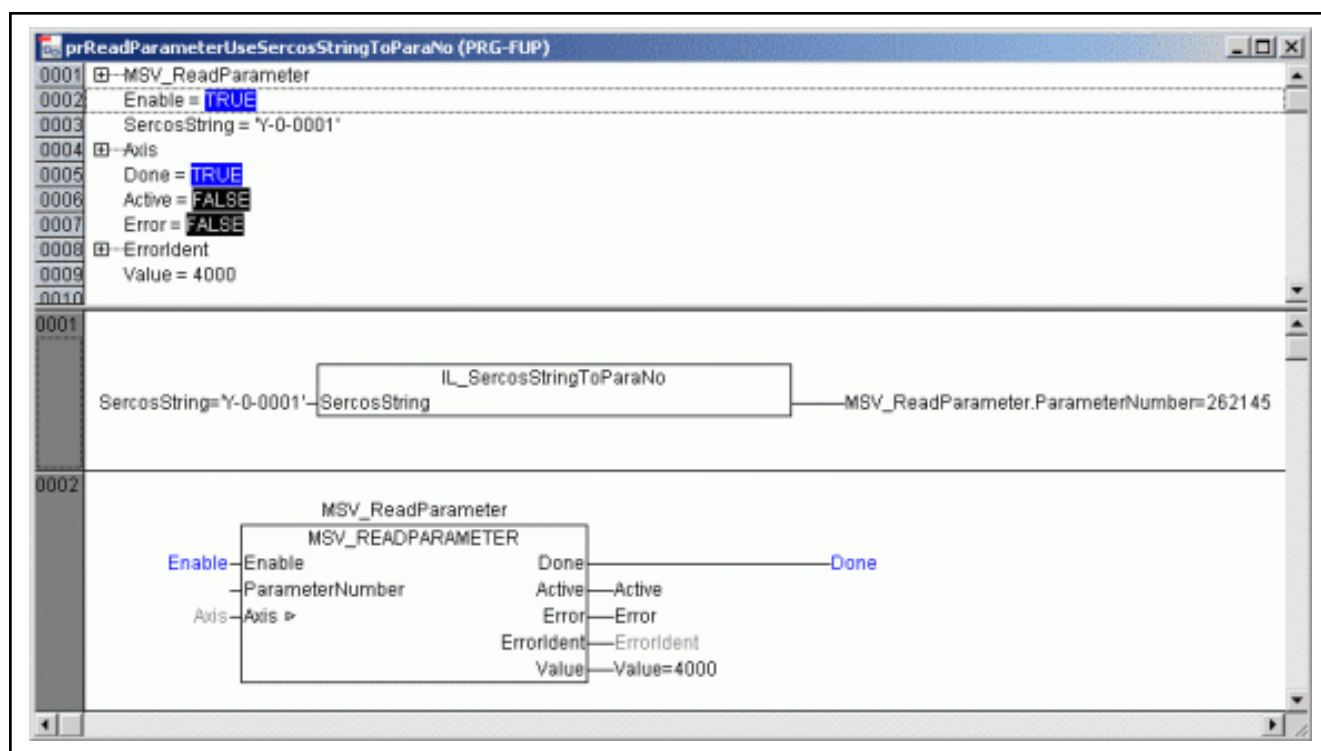


Abb. 4-26: Anwendung der Funktion (FUP): IL_SercosStringToParaNo

4.6 IP-Settings - IP-Konfiguration

4.6.1 ML_GetGateway

Kurzbeschreibung Dieser Funktionsbaustein liest die Gateway-Adresse der Steuerung. Das Gateway wird durch Auslesen des Parameters C-0-0202 ermittelt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	IP-Settings

Tab. 4-52: Bibliothekszuordnung

Schnittstellenbeschreibung

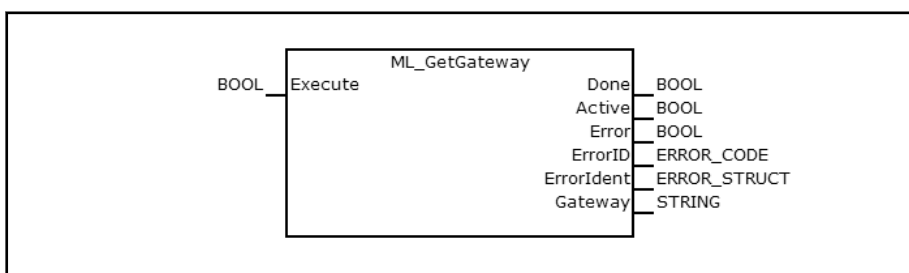


Abb. 4-27: Funktionsbaustein ML_GetGateway

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.

ML_Base.library

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Gateway	STRING(255)	Gelesene Gateway-Adresse

Tab. 4-53: Schnittstelle ML_GetGateway

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich

Tab. 4-54: Eingangsverhalten ML_GetGateway

Funktionsbeschreibung

Dieser Funktionsbaustein liest die Gateway-Adresse der Steuerung. Das Gateway wird durch Auslesen des Parameters C-0-0202 ermittelt.

Der Funktionsbaustein ML_GetGateway übermittelt nach der Bearbeitungs-freigabe des Funktionsbausteins mittels "Execute" die Gateway-Adresse der Steuerung. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt und der Ausgang "Gateway" ist gültig.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

Gateway

Der Ausgang "Gateway" gibt die Gateway-Adresse der Steuerung aus. Die Angabe erfolgt in der Regel als IP-Adresse. Diese besteht aus vier durch Punkte getrennte Dezimalzahlen zwischen 0 und 255.

Beispiel:

192.168.1.254

Fehlercodes

Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#).

4.6.2 ML_GetIpAddress

Kurzbeschreibung

Dieser Funktionsbaustein liest die IP-Adresse der Steuerung. Die IP-Adresse wird durch Auslesen des Parameters C-0-0200 ermittelt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	IP-Settings

Tab. 4-55: Bibliothekszuordnung

Schnittstellenbeschreibung

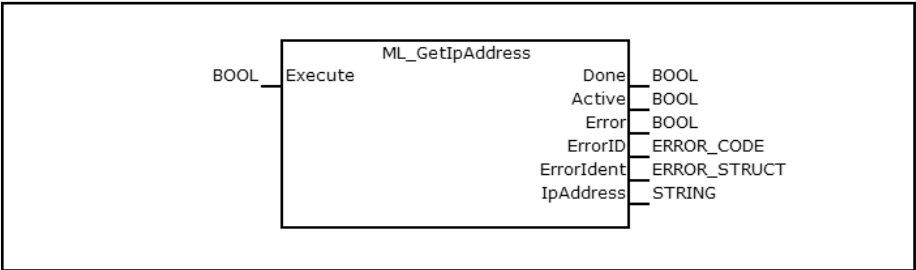


Abb. 4-28: Funktionsbaustein ML_GetIpAddress

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	IpAddress	STRING(255)	Gelesene IP-Adresse

Tab. 4-56: Schnittstelle ML_GetIpAddress

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich

Tab. 4-57: Eingangsverhalten ML_GetIpAddress

Funktionsbeschreibung

Dieser Funktionsbaustein liest die IP-Adresse der Steuerung. Die IP-Adresse wird durch Auslesen des Parameters C-0-0200 ermittelt.

Der Funktionsbaustein ML_GetIpAddress übermittelt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Execute" die IP-Adresse des Engineering-Ports der Steuerung. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt und der Ausgang "IpAddress" ist gültig.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

IpAddress

Der Ausgang "IpAddress" gibt die IP-Adresse der Steuerung aus. Die Angabe erfolgt in der Regel als aus vier durch Punkte getrennte Dezimalzahlen zwischen 0 und 255.

Beispiel:

192.168.1.1

Fehlercodes

Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#).

4.6.3 ML_GetMacAddress

Kurzbeschreibung

Dieser Funktionsbaustein liest die Mac-Adresse der Steuerung. Die Mac-Adresse wird durch Auslesen des Parameters C-0-0205 ermittelt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	IP-Settings

Tab. 4-58: Bibliothekszuordnung

ML_Base.library

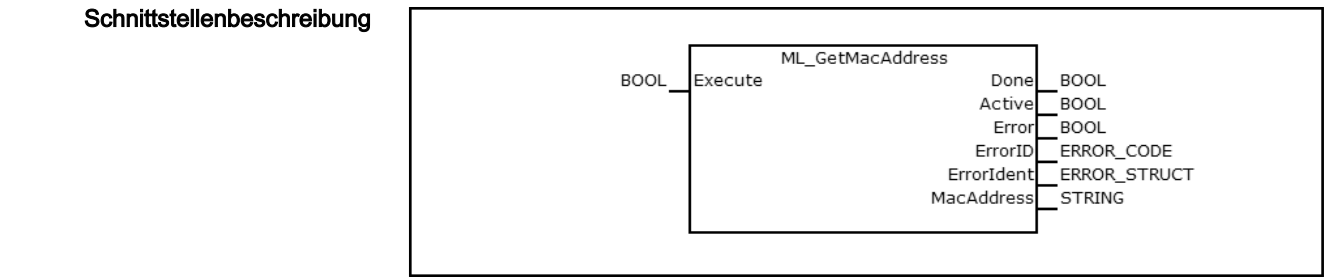


Abb. 4-29: Funktionsbaustein ML_GetMacAddress

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	MacAddress	STRING(255)	Gelesene Mac-Adresse

Tab. 4-59: Schnittstelle ML_GetMacAddress

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich

Tab. 4-60: Eingangsverhalten ML_GetMacAddress

Funktionsbeschreibung

Dieser Funktionsbaustein liest die Mac-Adresse der Steuerung. Die Mac-Adresse wird durch Auslesen des Parameters C-0-0205 ermittelt.

Der Funktionsbaustein ML_GetMacAddress übermittelt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Execute" die Mac-Adresse des Engineering-Ports der Steuerung. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt und der Ausgang "MacAddress" ist gültig.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

MacAddress

Der Ausgang "MacAddress" gibt die Mac-Adresse der Steuerung aus. Die Angabe erfolgt in der Regel als aus 6 durch Doppelpunkte getrennte Hexadezimalzahlen zwischen 00 und FF.

Beispiel:

00:30:D6:02:8D:FC

Fehlercodes

Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#).

4.6.4 ML_GetSubnetMask

Kurzbeschreibung

Dieser Funktionsbaustein liest die Subnet-Maske der Steuerung. Die Subnet-Maske wird durch Auslesen des Parameters C-0-0205 ermittelt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	IP-Settings

Tab. 4-61: Bibliothekszuordnung

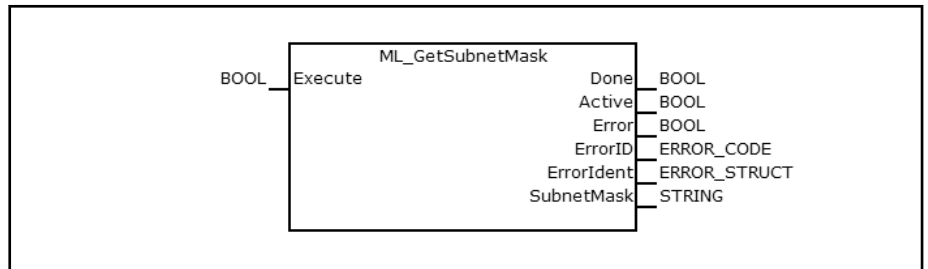
Schnittstellenbeschreibung

Abb. 4-30: Funktionsbaustein ML_GetSubnetMask

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	SubnetMask	STRING(255)	Gelesene Subnet-Maske

Tab. 4-62: Schnittstelle ML_GetSubnetMask

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich

Tab. 4-63: Eingangsverhalten ML_GetSubnetMask

Funktionsbeschreibung

Dieser Funktionsbaustein liest die Subnet-Maske der Steuerung. Die Subnet-Maske wird durch Auslesen des Parameters C-0-0205 ermittelt.

Der Funktionsbaustein ML_GetSubnetMask übermittelt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Execute" die Subnet-Maske des Engineering-Ports der Steuerung. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt und der Ausgang "SubnetMask" ist gültig.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

SubnetMask

Der Ausgang "SubnetMask" gibt die Subnet-Maske der Steuerung aus. Die Angabe erfolgt in der Regel als aus vier durch Punkte getrennte Dezimalzahlen zwischen 0 und 255.

Beispiel:

192.168.1.255

Fehlercodes

Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#).

ML_Base.library

4.6.5ML_SetGateway

Kurzbeschreibung Dieser Funktionsbaustein setzt die Gateway-Adresse der Steuerung. Das Gateway wird durch Schreiben des Parameters C-0-0202 gesetzt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	IP-Settings

Tab. 4-64: Bibliothekszuordnung

Schnittstellenbeschreibung

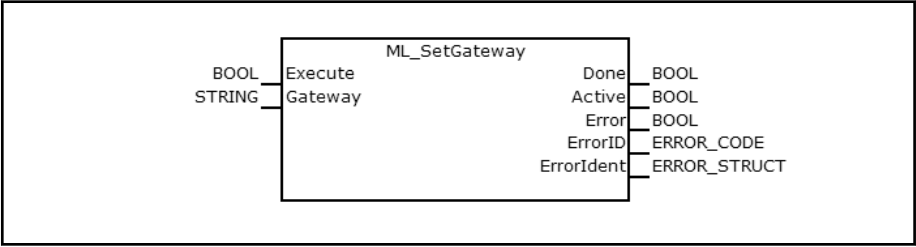


Abb. 4-31: Funktionsbaustein ML_SetGateway

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
	Gateway	STRING(255)	zu schreibende Gateway-Adresse
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 4-65: Schnittstelle ML_SetGateway

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
Gateway	STRING			"	steigende Flanke an "Execute"

Tab. 4-66: Eingangsverhalten ML_SetGateway

Funktionsbeschreibung Dieser Funktionsbaustein setzt die Gateway-Adresse der Steuerung. Das Gateway wird durch Schreiben des Parameters C-0-0202 gesetzt.

Der Funktionsbaustein ML_SetGateway setzt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Execute" die Gateway-Adresse der Steuerung. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

Gateway

Am Eingang "Gateway" wird die zu setzende Gateway-Adresse der Steuerung übergeben. Die Angabe erfolgt in der Regel als IP-Adresse. Diese besteht aus vier durch Punkte getrennte Dezimalzahlen zwischen 0 und 255.

Beispiel:

192.168.1.254

Fehlercodes Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE](#), 16#0030, Seite 348.

4.6.6 ML_SetIpAddress

Kurzbeschreibung Dieser Funktionsbaustein setzt die IP-Adresse des Engineering-Ports der Steuerung. Das Gateway wird durch Schreiben des Parameters C-0-0200 gesetzt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	IP-Settings

Tab. 4-67: Bibliothekszuordnung

Schnittstellenbeschreibung

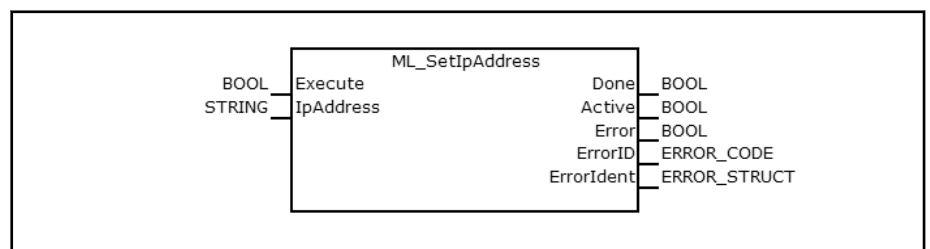


Abb. 4-32: Funktionsbaustein ML_SetIpAddress

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
	IpAddress	STRING(255)	Zu schreibende IP-Adresse
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 4-68: Schnittstelle ML_SetIpAddress

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
IpAddress	STRING			"	steigende Flanke an "Execute"

Tab. 4-69: Eingangsverhalten ML_SetIpAddress

Funktionsbeschreibung Dieser Funktionsbaustein setzt die IP-Adresse des Engineering-Ports der Steuerung. Die IP-Adresse wird durch Schreiben des Parameters C-0-0200 gesetzt.

Der Funktionsbaustein ML_SetIpAddress setzt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Execute" die IP-Adresse der Steuerung. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt.

ML_Base.library

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

IpAddress

Am Eingang "IpAddress" wird die zu setzende IP-Adresse der Steuerung übergeben. Die Angabe erfolgt in der Regel als aus vier durch Punkte getrennte Dezimalzahlen zwischen 0 und 255.

Beispiel:

192.168.1.1

Fehlercodes

Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#).

4.6.7ML_SetSubnetMask

Kurzbeschreibung

Dieser Funktionsbaustein setzt die Subnet-Maske der Steuerung. Die Subnet-Maske wird durch Schreiben des Parameters C-0-0201 gesetzt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	IP-Settings

Tab. 4-70: Bibliothekszuordnung

Schnittstellenbeschreibung

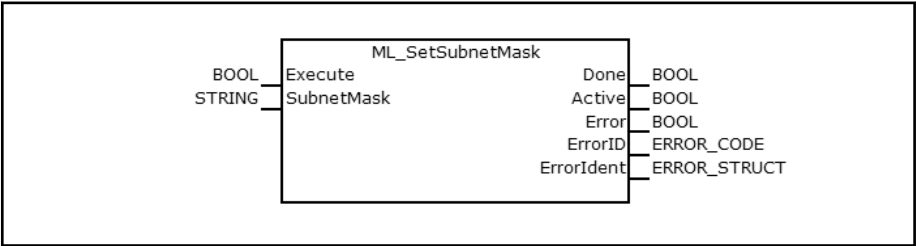


Abb. 4-33: Funktionsbaustein ML_SetSubnetMask

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
	SubnetMask	STRING(255)	zu schreibende Subnet-Maske
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 4-71: Schnittstelle ML_SetSubnetMask

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
SubnetMask	STRING			"	steigende Flanke an "Execute"

Tab. 4-72: Eingangsverhalten ML_SetSubnetMask

Funktionsbeschreibung Dieser Funktionsbaustein setzt die Subnet-Maske der Steuerung. Die Subnet-Maske wird durch Schreiben des Parameters C-0-0201 gesetzt.

Der Funktionsbaustein ML_SetSubnetMask setzt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Execute" die Subnet-Maske der Steuerung. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

SubnetMask

Am Eingang "SubnetMask" wird die zu setzende Subnet-Maske der Steuerung übergeben. Die Angabe erfolgt in der Regel als IP-Adresse. Diese besteht aus vier durch Punkte getrennte Dezimalzahlen zwischen 0 und 255.

Beispiel:

192.168.1.255

Fehlercodes Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#).

4.7 SystemTime - Systemzeit

4.7.1 IL_SetDateTime

Kurzbeschreibung Dieser Funktionsbaustein setzt die Systemzeit der Steuerung und der angeschlossenen Antriebe.

Bibliothek	Bereich
ML_Base_xxx_yyyy	SystemTime

Tab. 4-73: Bibliothekszuordnung

Schnittstellenbeschreibung

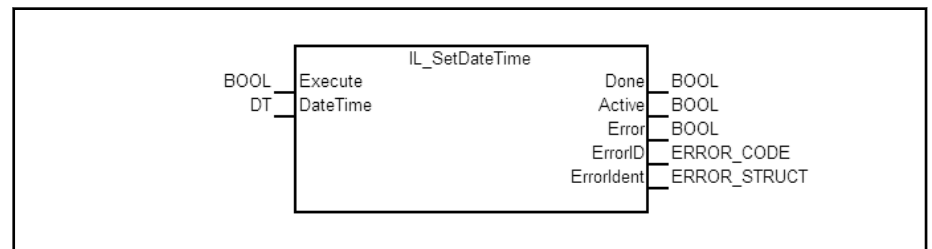


Abb. 4-34: Funktionsbaustein IL_SetDateTime

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
	DateTime	DT	Einzustellende Zeit
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 4-74: Schnittstelle IL_SetDateTime

ML_Base.library

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
DateTime	DT	DT#1970-01-01-00:00:0		DT#1970-01-01-00:00:00	steigende Flanke an "Execute"

Tab. 4-75: Eingangsverhalten IL_SetDateTime

Der Funktionsbaustein IL_SetDateTime setzt die am Eingang "DateTime" angegebene Uhrzeit auf der Steuerung, den angeschlossenen Antrieben sowie der batteriegepufferte Echtzeituhr.

Ist "DateTime" DT#1970-01-01-00:00:00, so wird die aktuelle Steuerungszeit in die Echtzeituhr und die Antriebe übernommen.

Bei steigender Flank an "Execute" wird die Bearbeitung gestartet. Die am Eingang "DateTime" angegebene Uhrzeit wird zuerst in der Steuerung gesetzt und in die Echtzeituhr übernommen. Danach wird die Uhrzeit an alle angeschlossenen Antriebe übertragen. Nicht alle Antriebe unterstützen das Setzen der Uhrzeit. Diese Antriebe werden nicht beschrieben. Die Bearbeitung dauert mehrere SPS-Zyklen. Nach erfolgreicher Abarbeitung wird der Ausgang "Done" gesetzt.

Tritt während der Abarbeitung ein Fehler auf, so wird der Ausgang "Error" gesetzt und der Fehlercode an den Ausgängen "ErrorID" und "ErrorIdent" ausgegeben.

Fehlercodes

Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#) sowie die Fehlertabelle F_RELATED_TABLE, 16#0170.

4.7.2 ML_GetSystemTimeUTC

Kurzbeschreibung

Dieser Funktionsbaustein liest die Systemzeit der Steuerung in UTC. Die Systemzeit wird durch Auslesen des Parameters C-0-0050 ermittelt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	SystemTime

Tab. 4-76: Bibliothekszuordnung

Schnittstellenbeschreibung

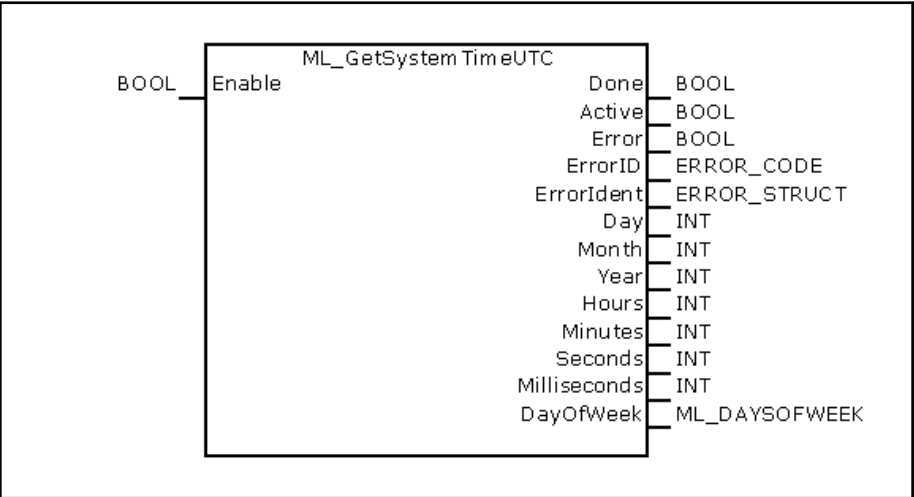


Abb. 4-35: Funktionsbaustein ML_GetSystemTimeUTC

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Day	INT	Tag des Monats
	Month	INT	Monat des Jahres
	Year	INT	Jahr
	Hours	INT	Stunden
	Minutes	INT	Minuten
	Seconds	INT	Sekunden
	Milliseconds	INT	Millisekunden
	DayOfWeek	ML_DAYSOFWEEK	Wochentag

Tab. 4-77: Schnittstelle ML_GetSystemTimeUTC

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Enable	BOOL			FALSE	kontinuierlich

Tab. 4-78: Eingangsverhalten ML_GetSystemTimeUTC

Funktionsbeschreibung

Dieser Funktionsbaustein liest die Systemzeit der Steuerung in UTC. Die Systemzeit wird durch Auslesen des Parameters C-0-0050 ermittelt.

Der Funktionsbaustein ML_GetSystemTimeUTC übermittelt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Enable" die Systemzeit der Steuerung. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt und die Ausgänge "Day", "Month", "Year", "Hours", "Minutes", "Seconds", "Milliseconds" und "DayOfWeek" sind gültig.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

Day, Month, Year

Die Ausgänge "Day", "Month" und "Year" bilden das Datum der Systemzeit ab. Die Angabe erfolgt als Integer.

Beispiel:

Day: 22, Month: 6, Year: 2010

Hours, Minutes, Seconds

Die Ausgänge "Hours", "Minutes", "Seconds" und "Milliseconds" bilden die Zeit der Systemzeit ab. Die Angabe erfolgt als Integer.

Beispiel:

Hours: 13, Minutes: 36, Seconds: 42, Milliseconds: 123

ML_Base.library

DayOfWeek

Der Ausgang "DayOfWeek" gibt den Wochentag der Systemzeit aus. Die Angabe erfolgt als Aufzählungstyp. Der verwendete Aufzählungstyp wird in [ML_DAYSOFWEEK, Seite 155](#), erläutert.

Beispiel:

DayOfWeek: Friday

Fehlercodes Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#).

4.7.3 ML_SetSystemTimeUTC

Kurzbeschreibung Dieser Funktionsbaustein setzt die Systemzeit der Steuerung in UTC. Die Systemzeit wird durch Schreiben des Parameters C-0-0050 gesetzt.

Bibliothek	Bereich
ML_Base_xxx_yyyy	SystemTime

Tab. 4-79: Bibliothekszuordnung

Schnittstellenbeschreibung

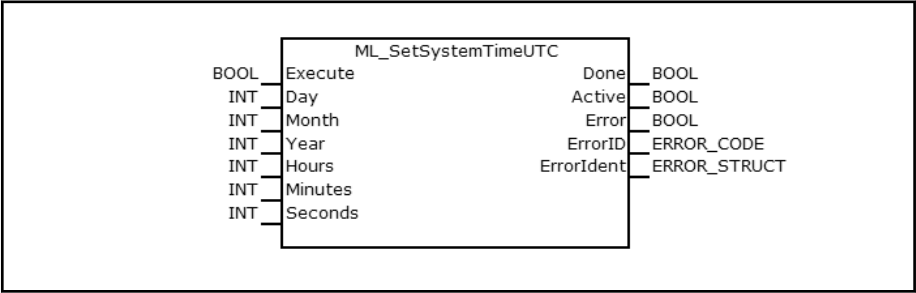


Abb. 4-36: Funktionsbaustein ML_SetSystemTimeUTC

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
	Day	INT	Tag des Monats
	Month	INT	Monat des Jahres
	Year	INT	Jahr
	Hours	INT	Stunde
	Minutes	INT	Minute
	Seconds	INT	Sekunde
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 4-80: Schnittstelle ML_SetSystemTimeUTC

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich
Day	INT	1	31	1	steigende Flanke an "Execute"
Month	INT	1	12	1	steigende Flanke an "Execute"
Year	INT	1	9999	1	steigende Flanke an "Execute"
Hours	INT	0	23	0	steigende Flanke an "Execute"
Minutes	INT	0	59	0	steigende Flanke an "Execute"
Seconds	INT	0	59	0	steigende Flanke an "Execute"

Tab. 4-81: Eingangsverhalten ML_SetSystemTimeUTC

Funktionsbeschreibung

Dieser Funktionsbaustein setzt die Systemzeit der Steuerung in UTC. Die Systemzeit wird durch Schreiben des Parameters C-0-0050 gesetzt.

Der Funktionsbaustein ML_SetSystemTimeUTC setzt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Execute" die Systemzeit der Steuerung. Die Eingänge "Day", "Month", "Year", "Hour", "Minutes" und "Seconds" werden bei steigender Flanke an Execute übernommen. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

Day, Month, Year

Die Ausgänge "Day", "Month" und "Year" bilden das Datum der Systemzeit ab. Die Angabe erfolgt als Integer.

Beispiel:

Day: 22, Month: 6, Year: 2010

Hours, Minutes, Seconds

Die Ausgänge "Hours", "Minutes" und "Seconds" bilden die Zeit der Systemzeit ab. Die Angabe erfolgt als Integer.

Beispiel:

Hours: 13, Minutes: 36, Seconds: 42

Fehlercodes

Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE, 16#0030, Seite 348](#).

4.7.4 ML_DAYSOFWEEK

Bibliothek	Bereich
ML_Base_xxx_yyyy	SystemTime

Tab. 4-82: Bibliothekszuordnung

ML_Base.library

Der Datentyp "ML_DAYSOFWEEK" ist ein Aufzählungstyp und hat die folgenden Einträge:

Element	Wert	Beschreibung
Monday	1	Montag
Tuesday	2	Dienstag
Wednesday	3	Mittwoch
Thursday	4	Donnerstag
Friday	5	Freitag
Saturday	6	Samstag
Sunday	7	Sonntag

Tab. 4-83: Definition des Aufzählungstyps ML_DAYSOFWEEK

4.8 Target - Zielsystem

4.8.1 ML_GetCpuLoad

Kurzbeschreibung Diese Funktion gibt die aktuelle CPU-Last in Prozent zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-84: Bibliothekszuordnung

Schnittstellenbeschreibung

	Name	Typ	Kommentar
RETURN	ML_GetCpuLoad	REAL	CPU-Last in Prozent

Tab. 4-85: Schnittstelle ML_GetCpuLoad

Funktionsbeschreibung Diese Funktion gibt die aktuelle CPU-Last in Prozent zurück.
Die Funktion ML_GetCpuLoad liest den Parameter C-0-0415 und gibt die darin enthaltene CPU-Auslastung in Prozent zurück.

4.8.2 ML_GetDiagnosisLog

Kurzbeschreibung Die Funktion ML_GetDiagnosisLog liest Einträge aus dem Diagnose-Logbuch.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Error-/Diagnostics Memory

Tab. 4-86: Bibliothekszuordnung

Schnittstellenbeschreibung

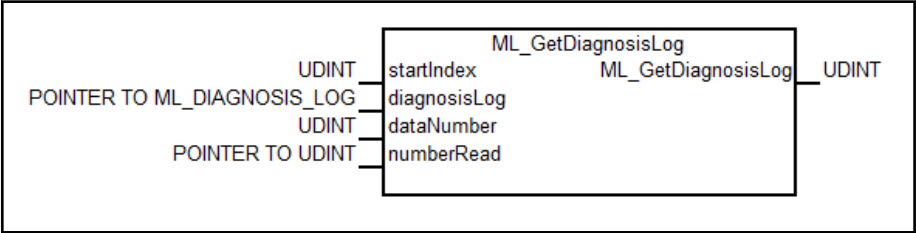


Abb. 4-37: Funktion ML_GetDiagnosisLog

	Name	Typ	Kommentar
VAR_INPUT	startIndex	UDINT	Startindex ab dem das Diagnose-Logbuchs gespeichert werden soll
	diagnosisLog	POINTER TO ML_DIAGNOSIS_LOG	Adresse auf der die Einträge des Diagnose-Logbuchs gespeichert werden sollen siehe Kap. 4.8.17 "ML_DIAGNOSIS_LOG" auf Seite 163
	dataNumber	UDINT	Größe von diagnosisLog (Anzahl Einträge)
	numberRead	POINTER TO UDINT	Adresse der Anzahl der gelesenen Logbuch-Einträge
RETURN		UDINT	Kurzhinweis auf die Fehlerursache

Tab. 4-87: Schnittstelle von FUN ML_GetDiagnosisLog

Funktionsbeschreibung Die Funktion liest eine Anzahl von Diagnose-Logbuch-Einträgen ab dem "startIndex". Die gelesene Anzahl von Einträgen wird in "numberRead" zurückgegeben. Sind weniger als "dataNumber" Einträge verfügbar, so wird nur diese Anzahl von Einträgen gelesen.

4.8.3 ML_GetFirmwareString

Kurzbeschreibung Diese Funktion gibt den Firmwarestring der Steuerung zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-88: Bibliothekszuordnung

Schnittstellenbeschreibung

	Name	Typ	Kommentar
RETURN	ML_GetFirmwareString	STRING	Firmware-Kennung der Steuerung

Tab. 4-89: Schnittstelle ML_GetFirmwareString

Funktionsbeschreibung Diese Funktion gibt den Firmwarestring der Steuerung zurück.

Die Funktion ML_GetFirmwareString liest den Parameter C-0-0012 und gibt den darin enthaltenen Firmwarestring als Zeichenkette zurück.

Beispiel:

CML65s-MLC-12V02.0072

4.8.4 ML_GetHardwareInfoCtrl

Kurzbeschreibung Diese Funktion gibt Informationen über die Hardware der Steuerung zurück.



Die Funktion enthält keinen Eingang, keinen Rückgabewert, aber vier Ausgänge (VAR_OUTPUT).

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-90: Bibliothekszuordnung

ML_Base.library

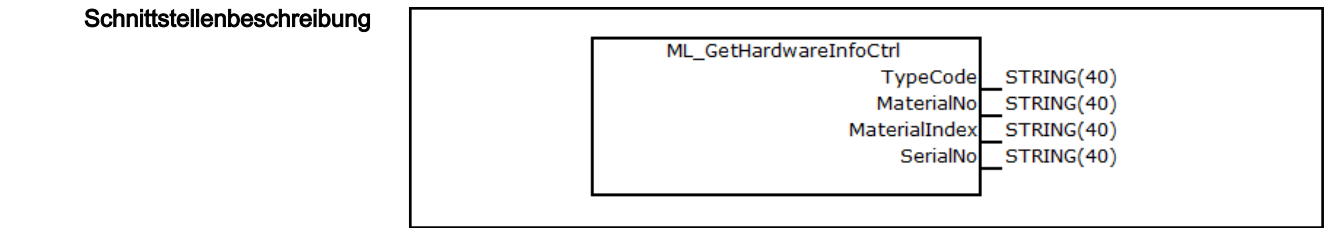


Abb. 4-38: ML_GetHardwareInfoCtrl

	Name	Typ	Kommentar
VAR_OUTPUT	TypeCode	STRING(40)	Typ-Code der Steuerungshardware (C-0-0010)
	MaterialNo	STRING(40)	Materialnummer der Steuerungshardware (C-0-0021)
	MaterialIndex	STRING(40)	Materialindex der Steuerungshardware (C-0-0021)
	SerialNo	STRING(40)	Seriennummer der Steuerung

Tab. 4-91: Schnittstelle ML_GetHardwareInfoCtrl

Funktionsbeschreibung

Diese Funktion gibt Informationen über die Hardware der Steuerung zurück.
Nutzung der Funktion in einem Programm

```
VAR
  strTypeCode: STRING;
  strMaterialNo: STRING;
  strMaterialIndex: STRING;
  strSerialNo: STRING;
END_VAR

ML_GetHardwareInfoCtrl (TypeCode=> strTypeCode, MaterialNo=> strMaterialNo,
MaterialIndex=> strMaterialIndex, SerialNo=> strSerialNo);
```

Abb. 4-39: Antwort der Funktion ML_GetHardwareInfoCtrl

4.8.5 ML_GetInfoApplication

Kurzbeschreibung

Dieser Funktionsbaustein liest die auf der Steuerung vorhandenen SPS-Applikationen aus und schreibt legt Status im Ausgang "Info" ab.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-92: Bibliothekszuordnung

Schnittstellenbeschreibung

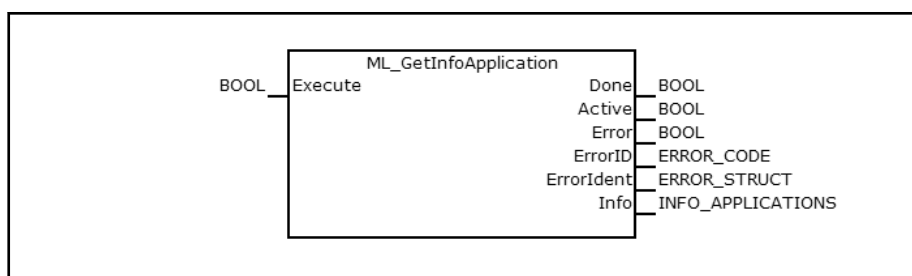


Abb. 4-40: Funktionsbaustein ML_GetInfoApplication

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe
VAR_OUTPUT	Done	BOOL	Operation ausgeführt
	Active	BOOL	Bestätigung der Bearbeitungsfreigabe
	Error	BOOL	Zeigt an, dass ein Fehler innerhalb des Funktionsbausteins aufgetreten ist.
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Info	INFO_APPLICATIONS	Applikationsinformationen

Tab. 4-93: Schnittstelle ML_GetInfoApplication

Name	Typ	Min-Wert	Max-Wert	Default-Wert	Übernahme
Execute	BOOL			FALSE	kontinuierlich

Tab. 4-94: Eingangsverhalten ML_GetInfoApplication

Funktionsbeschreibung

Dieser Funktionsbaustein liest die auf der Steuerung vorhandenen SPS-Applikationen aus und schreibt den Status im Ausgang "Info" ab.

Der Funktionsbaustein ML_GetInfoApplication übermittelt nach der Bearbeitungsfreigabe des Funktionsbausteins mittels "Execute" die auf der Steuerung vorhandenen Applikationen und deren Status. Nach erfolgreicher Abarbeitung des Funktionsbausteins wird der Ausgang "Done" gesetzt und der Ausgang "Info" ist gültig.

Tritt während der Bearbeitung des Funktionsbausteins ein Fehler auf, so wird dieser durch den Ausgang "Error" signalisiert sowie durch die in diesem Fall aktualisierten Ausgänge "ErrorID" und "ErrorIdent" spezifiziert.

IpAddress

Der Ausgang "Info" gibt die Struktur [INFO_APPLICATIONS](#), Seite 163, aus, die die Anzahl der Applikationen, deren Namen und Status enthält.

Fehlercodes

Der Funktionsbaustein nutzt die Fehlertabelle [MLC_TABLE](#), 16#0030, Seite 348.

4.8.6 ML_GetMaxTemperature

Kurzbeschreibung

Diese Funktion gibt die höchste erreichte Temperatur der Steuerung zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-95: Bibliothekszuordnung

Schnittstellenbeschreibung

ML_Base.library

	Name	Typ	Kommentar
RETURN	ML_GetMaxTemperature	REAL	maximale Temperatur der Steuerung

Tab. 4-96: Schnittstelle ML_GetMaxTemperature

Funktionsbeschreibung

Diese Funktion gibt die maximale Temperatur der Steuerung zurück.

Die Funktion ML_GetMaxTemperature liest den Parameter C-0-0071 und gibt die darin enthaltene maximale Temperatur der Steuerung zurück.

Die Einheit der Temperatur kann mit der Funktion [ML_GetScalingType](#), Seite 160, ausgelesen werden.

Die Maximaltemperatur wird beim Einschalten der Steuerung zurückgesetzt.

4.8.7 ML_GetNewestDiagnosisIndex

Die Funktion ML_GetNewestDiagnosisIndex gibt den neuesten Index des Diagnose-Logbuchs zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Error-/Diagnostics Memory

Tab. 4-97: Bibliothekszuordnung

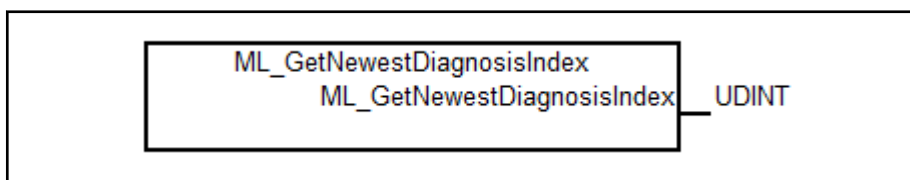


Abb. 4-41: Funktion ML_GetNewestDiagnosisIndex

	Name	Typ	Kommentar
RETURN		UDINT	Neuester Index des Diagnose-Logbuchs

Tab. 4-98: Schnittstelle von FUN ML_GetNewestDiagnosisIndex

4.8.8 ML_GetOhcFan

Kurzbeschreibung

Diese Funktion gibt die Betriebsstunden des Lüfters der Steuerung zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-99: Bibliothekszuordnung

Schnittstellenbeschreibung

	Name	Typ	Kommentar
RETURN	ML_GetOhcFan	REAL	Betriebsstunden des Lüfters der Steuerung

Tab. 4-100: Schnittstelle ML_GetOhcFan

Funktionsbeschreibung

Diese Funktion gibt die Betriebsstunden des Lüfters der Steuerung zurück.

Die Funktion ML_GetOhcFan liest den Parameter C-0-0052 und gibt die darin enthaltenen Betriebsstunden des Steuerungslüfters zurück.

4.8.9 ML_GetScalingType

Kurzbeschreibung

Diese Funktion gibt die Einheit aller Temperaturangaben der Steuerung zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-101: Bibliothekszuordnung

Schnittstellenbeschreibung

	Name	Typ	Kommentar
RETURN	ML_GetScalingType	E_SCALING_TYPE	Einheit aller Temperaturangaben der Steuerung

Tab. 4-102: Schnittstelle ML_GetScalingType

Funktionsbeschreibung

Diese Funktion gibt die Einheit aller Temperaturangaben der Steuerung zurück.

Die Funktion ML_GetScalingType liest den Parameter C-0-0063 und gibt die darin enthaltene Einheit der Temperaturen der Steuerung zurück.

Der Rückgabewert der Funktion ist vom Aufzählungstyp [E_SCALING_TYPE](#), Seite 162.

4.8.10 ML_GetTargetInfo

Kurzbeschreibung Diese Funktion gibt Informationen über das Zielsystem zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-103: Bibliothekszuordnung

Schnittstellenbeschreibung

	Name	Typ	Kommentar
RETURN	ML_GetTargetInfo	TARGET_INFO	Informationen über das Zielsystem

Tab. 4-104: Schnittstelle ML_GetTargetInfo

Funktionsbeschreibung

Diese Funktion gibt Informationen über das Zielsystem zurück.

Der Rückgabewert der Funktion ist vom Aufzählungstyp [TARGET_INFO](#), Seite 163.

4.8.11 ML_GetTemperature

Kurzbeschreibung Diese Funktion gibt die aktuelle Temperatur der Steuerung zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-105: Bibliothekszuordnung

Schnittstellenbeschreibung

	Name	Typ	Kommentar
RETURN	ML_GetTemperature	REAL	aktuelle Temperatur der Steuerung

Tab. 4-106: Schnittstelle ML_GetTemperature

Funktionsbeschreibung

Diese Funktion gibt die aktuelle Temperatur der Steuerung zurück.

Die Funktion ML_GetTemperature liest den Parameter C-0-0070 und gibt die darin enthaltene aktuelle Temperatur der Steuerung zurück.

Die Einheit der Temperatur kann mit der Funktion [ML_GetScalingType](#), Seite 160, ausgelesen werden.

ML_Base.library

4.8.12 ML_GetWarningCpuLoad

Kurzbeschreibung Diese Funktion gibt die eingestellte maximale CPU-Last in Prozent zurück.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-107: Bibliothekszuordnung

Schnittstellenbeschreibung

	Name	Typ	Kommentar
RETURN	ML_GetWarningCpuLoad	REAL	eingestellte maximale CPU-Last in Prozent

Tab. 4-108: Schnittstelle ML_GetWarningCpuLoad

Funktionsbeschreibung Diese Funktion gibt die eingestellte maximale CPU-Last in Prozent zurück.
Die Funktion ML_GetWarningCpuLoad liest den Parameter C-0-0418 und gibt die darin eingestellte maximale CPU-Auslastung in Prozent zurück.

4.8.13 ML_SetWarningCpuLoad

Kurzbeschreibung Diese Funktion stellt die maximal erlaubte CPU-Last in Prozent ein.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-109: Bibliothekszuordnung

Schnittstellenbeschreibung

	Name	Typ	Kommentar
VAR_INPUT	CpuLoad	REAL	maximal erlaubte CPU-Last in Prozent
RETURN	ML_SetWarningCpuLoad	REAL	

Tab. 4-110: Schnittstelle ML_SetWarningCpuLoad

Funktionsbeschreibung Diese Funktion stellt die maximal erlaubte CPU-Last in Prozent ein.
Die Funktion ML_SetWarningCpuLoad schreibt den am Eingang CpuLoad enthaltenen Wert in den Parameter C-0-0418.

4.8.14 APP_INFO

Der Datentyp "APP_INFO" wird in der Struktur [INFO_APPLICATIONS](#), Seite 163, verwendet.

Der Datentyp "APP_INFO" ist eine Struktur und hat die folgenden Einträge:

Element	Typ	Beschreibung
Name	STRING	Name der SPS-Applikation
State	STRING	Zustand der SPS-Applikation

Tab. 4-111: Definition der Struktur APP_INFO

4.8.15 E_SCALING_TYPE

Der Datentyp "E_SCALING_TYPE" wird in der Funktion [ML_GetScalingType](#), Seite 160, verwendet.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Target

Tab. 4-112: Bibliothekszuordnung

Der Datentyp "E_SCALING_TYPE" ist ein Aufzählungstyp und hat die folgenden Einträge:

Element	Wert	Beschreibung
CELSIUS	16#0	Alle Temperaturangaben erfolgen in °C
FARENHEIT	16#1	Alle Temperaturangaben erfolgen in F

Tab. 4-113: Definition des Aufzählungstyps E_SCALING_TYPE

4.8.16 INFO_APPLICATIONS

Der Datentyp "INFO_APPLICATIONS" wird im Funktionsbaustein [ML_GetInfoApplication](#), Seite 158, verwendet.

Der Datentyp "INFO_APPLICATIONS" ist eine Struktur und hat die folgenden Einträge:

Element	Typ	Beschreibung
NoOfApplications	DINT	Anzahl der SPS-Applikationen auf der Steuerung
AppInfo	ARRAY[1..] OF APP_INFO	Informationen zu den einzelnen SPS-Applikationen

Tab. 4-114: Definition der Struktur INFO_APPLICATIONS

4.8.17 ML_DIAGNOSIS_LOG

Kurzbeschreibung Diese Struktur enthält einen Eintrag des Diagnose-Logbuchs.

Bibliothek	Bereich
ML_Base_xxx_yyyy	Data Types/Info

Tab. 4-115: Bibliothekszuordnung

Strukturelement	Typ	Beschreibung
number	UDINT	Fehlernummer
state	UDINT	Status des Diagnoseeintrags
dateTime	SYS_TIME_DATE	Zeitpunkt der Diagnose
logicalAddress	DINT	Logische Adresse des Geräts (z.B. Achse, Kinematik, ...)
physicalAddress	DINT	Physikalische Adresse des Geräts (z.B. Antrieb)
despatcher	UDINT	Quelle der Diagnose
index	UDINT	Index der Diagnose im Diagnose-Logbuch
text	STRING(123)	Diagnosetext

Tab. 4-116: Datenstruktur ML_DIAGNOSIS_LOG

4.8.18 TARGET_INFO

Der Datentyp "TARGET_INFO" wird in der Funktion [ML_GetTargetInfo](#), Seite 161, verwendet.

ML_Base.library

Der Datentyp "TARGET_INFO" ist eine Struktur und hat die folgenden Einträge:

Element	Typ	Beschreibung
TargetType	DWORD	Zielsystemtyp
TargetId	DWORD	Zielsystem ID
TargetVersion	DWORD	Version des Zielsystems
DeviceName	STRING	Name der Steuerung
NodeName	STRING	Name des Knotens

Tab. 4-117: Definition der Struktur TARGET_INFO

5 Funktionsbausteine nach PLCopen-Spezifikation

Die beiden Bibliotheken ML_PLCopen und RMB_PLCopenFieldbus implementieren die PLCopen-Spezifikation für die beiden Feldbusse sercos und Profibus.

Die [ML_PLCopen.library auf Seite 169](#), beinhaltet die Funktionsbausteine für das sercos II und das sercos III Interface.

Es werden neben Rexroth IndraDrives auch alle anderen sercos fähigen Antriebe unterstützt.

Die [RMB_PLCopenFieldbus.library auf Seite 359](#), beinhaltet Funktionsbausteine und Datentypen für die Ansteuerung von Rexroth IndraDrives über den Profibus.

Unterstützte Funktionsbausteine

Die nachfolgende Tabelle stellt die implementierten Funktionsbausteine der beiden Bibliotheken gegenüber.

Funktionsbaustein	ML_PLCopen	RMB_PLCopenFieldbus
MC_Power	X	X
MC_Reset	X	X
MC_Stop	X	X
MB_Stop	-	X
MB_Home	X	X
MC_MoveAbsolute	X	X
MC_MoveAdditive	X	X
MC_MoveRelative	X	X
MC_MoveVelocity	X	X
MB_SetPositionControlMode	X	-
MB_ChangeCamData	X	-
MB_ChangeFlexEventSet	X	-
MB_ChangeFlexProfileSet	X	-
MB_ChangeProfileSet	X	-
MB_ChangeProfileStep	X	-
MB_GearInPos	X	-
MB_MotionProfile	X	-
MB_Phasing	X	-
MB_PhasingSlave	X	-
MC_CamIn	X	-
MC_CamOut	X	-
MC_GearIn	X	-
MC_GearOut	X	-
MC_TorqueControl	X	-

Funktionsbausteine nach PLCopen-Spezifikation

Funktionsbaustein	ML_PLCopen	RMB_PLCopenFieldbus
ML_FlexProfile	X	-
ML_OpenCyclicAnalogChannel	X	-
ML_OpenCyclicPositionChannel	X	-
ML_OpenCyclicTorqueChannel	X	-
ML_OpenCyclicVelocityChannel	X	-
ML_WriteCyclicAnalog	X	-
ML_WriteCyclicPosition	X	-
ML_WriteCyclicTorque	X	-
ML_WriteCyclicVelocity	X	-
ML_ControlOff	X	-
ML_ControlOn	X	-
ML_ControlWriteActualData	X	-
ML_OpenCyclicAnalogChannelCtrl	X	-
ML_OpenCyclicPositionChannelCtrl	X	-
ML_OpenCyclicTorqueChannelCtrl	X	-
ML_OpenCyclicVelocityChannelCtrl	X	-
ML_WriteCyclicAnalogCtrl	X	-
ML_WriteCyclicPositionCtrl	X	-
ML_WriteCyclicTorqueCtrl	X	-
ML_WriteCyclicVelocityCtrl	X	-
MB_Command	X	-
MB_ReadLintParameter	X	-
MB_ReadListParameter	X	X
MB_ReadListSegParameterPtr	-	X
MB_ReadLrealListParameter	X	-
MB_ReadLrealParameter	X	-
MB_ReadMaxRealValue	X	X
MB_ReadMaxValue	X	X
MB_ReadMinRealValue	X	X
MB_ReadMinValue	X	X
MB_ReadName	X	-
MB_ReadParameter	X	X
MB_ReadRealParameter	X	X
MB_ReadSercosAttribute	X	X

Funktionsbausteine nach PLCopen-Spezifikation

Funktionsbaustein	ML_PLCopen	RMB_PLCopenFieldbus
MB_ReadSercosDataStatus	X	-
MB_ReadStringParameter	X	X
MB_ReadUnit	X	-
MB_WriteLintParameter	X	-
MB_WriteListParameter	X	X
MB_WriteListSegParameterPtr	-	X
MB_WriteLrealParameter	X	-
MB_WriteParameter	X	X
MB_WriteRealParameter	X	X
MB_WriteStringParameter	X	X
MC_RealActualPosition	X	X
MC_ReadAxisError	X	X
MC_ReadStatus	X	X
ML_AbortTrigger	X	-
ML_InitTouchProbe	X	-
ML_TouchProbe	X	-
ML_TouchProbeContinuous	X	-
ML_WriteExpectWindow	X	-
MP_Drive	-	X
MP_DriveParameter	-	X
MP_FB_IndraDrv_AcyclicCommunication	-	X

Tab. 5-1: Unterstützte Funktionsbausteine in ML_PLCopen.library / RMB_PLCopenFieldBus.library

6 ML_PLCopen.library

6.1 ML_PLCopen, Allgemeines

Die Bibliothek ML_PLCopen enthält die für die Realisierung der Systeme IndraLogic XLC / IndraMotion MLC wesentlichen Funktionsbausteine für sercos basierte Achsen.

Die von den Bausteinen an ihrer Schnittstelle (VAR_INPUT, VAR_OUTPUT, VAR_IN_OUT, VAR_GLOBAL / EXTERNAL) benötigten Datentypen sind elementar in der IEC-Norm EN 61131-3 enthalten oder durch die Firma definiert in nachstehenden Bibliotheken:

- [RIL_CommonTypes, Seite 35](#), Datentypen zur allgemeinen Nutzung in allen IndraWorks-Systemen,
- [ML_Base, Seite 79](#), Datentypen des Systems IndraLogic XLC und IndraMotion MLC für sercos basierte AAchsen.

Die Bibliothek weist folgende Struktur auf:

Bibliotheks-Info und Zugriffsfunktionen, diese Bestandteile der Bibliothek werden automatisch für die jeweilige Bibliothek erzeugt und enthalten keine spezifischen Anteile.

Globale Variablen

- [ProbeSelect, Seite 172](#), definiert Feld, welches die Daten der Messtaster 1..100 enthält.

Data Types

- *Data Types of TouchProbe*
 - [PROBECONFIG, Seite 172](#), definiert den Datentyp für die Daten der Messtaster 1..100.

POUs

- *Controlled Motion*
 - [ML_ControlOff, Seite 267](#),
 - [ML_ControlOn, Seite 266](#),
 - [ML_ControlWriteActualData, Seite 268](#),
 - [ML_OpenCyclicAnalogChannelCtrl, Seite 269](#),
 - [ML_OpenCyclicPositionChannelCtrl, Seite 271](#),
 - [ML_OpenCyclicTorqueChannelCtrl, Seite 274](#),
 - [ML_OpenCyclicVelocityChannelCtrl, Seite 276](#),
 - [ML_WriteCyclicAnalogCtrl, Seite 270](#),
 - [ML_WriteCyclicPositionCtrl, Seite 272](#),
 - [ML_WriteCyclicTorqueCtrl, Seite 275](#),
 - [ML_WriteCyclicVelocityCtrl, Seite 277](#).
- *Motion*
 - [MB_ChangeCamData, Seite 225](#),
 - [MB_ChangeFlexEventSet, Seite 242](#),
 - [MB_ChangeFlexProfileSet, Seite 238](#),
 - [MB_ChangeProfileSet, Seite 217](#),
 - [MB_ChangeProfileStep, Seite 222](#),
 - [MB_GearInPos, Seite 247](#),

ML_PLCopen.library

- [MB_Home](#), Seite 197,
- [MB_MotionProfile](#), Seite 211,
- [MB_Phasing](#), Seite 202,
- [MB_PhasingSlave](#), Seite 204,
- [MB_SetPositionControlMode](#), Seite 196,
- [MC_AddAxisToGroup](#), Seite 250,



Die Funktionsbausteine "MC_AddAxisToGroup" und "MC_RemoveAxisFromGroup" wurden von ML_Robot nach ML_PLCopen verlegt, um sie auch für IndraLogic XLC zugänglich zu machen.

- [MC_CamIn](#), Seite 206,
- [MC_CamOut](#), Seite 210,
- [MC_GearIn](#), Seite 243,
- [MC_GearOut](#), Seite 246,
- [MC_MoveAbsolute](#), Seite 180,
- [MC_MoveAdditive](#), Seite 183,
- [MC_MoveRelative](#), Seite 186,
- [MC_MoveVelocity](#), Seite 190,
- [MC_Power](#), Seite 176,
- [MC_RemoveAxisFromGroup](#), Seite 251,



Die Funktionsbausteine "MC_AddAxisToGroup" und "MC_RemoveAxisFromGroup" wurden von ML_Robot nach ML_PLCopen verlegt, um sie auch für IndraLogic XLC zugänglich zu machen.

- [MC_Reset](#), Seite 177,
- [MC_Stop](#), Seite 179,
- [MC_TorqueControl](#), Seite 193,
- [ML_FlexProfile](#), Seite 227,
- [ML_OpenCyclicAnalogChannel](#), Seite 254,
- [ML_OpenCyclicPositionChannel](#), Seite 256,
- [ML_OpenCyclicTorqueChannel](#), Seite 261,
- [ML_OpenCyclicVelocityChannel](#), Seite 263,
- [ML_WriteCyclicAnalog](#), Seite 255,
- [ML_WriteCyclicPosition](#), Seite 258,
- [ML_WriteCyclicTorque](#), Seite 262,
- [ML_WriteCyclicVelocity](#), Seite 264,

- *Parameter*

- [MB_Command](#), Seite 281,
- [MB_ReadLintParameter](#), Seite 285,
- [MB_ReadListParameter](#), Seite 287,
- [MB_ReadLrealListParameter](#), Seite 290,
- [MB_ReadLrealParameter](#), Seite 292,
- [MB_ReadMaxRealValue](#), Seite 294,

- [MB_ReadMaxValue, Seite 295,](#)
- [MB_ReadMinRealValue, Seite 297,](#)
- [MB_ReadMinValue, Seite 299,](#)
- [MB_ReadName, Seite 300,](#)
- [MB_ReadParameter, Seite 302,](#)
- [MB_ReadRealParameter, Seite 303,](#)
- [MB_ReadSercosAttribute, Seite 305,](#)
- [MB_ReadSercosDataStatus, Seite 306,](#)
- [MB_ReadStringParameter, Seite 308,](#)
- [MB_ReadUnit, Seite 310,](#)
- [MB_WriteLintParameter, Seite 311,](#)
- [MB_WriteListParameter, Seite 313,](#)
- [MB_WriteLrealParameter, Seite 315,](#)
- [MB_WriteParameter, Seite 317.](#)
- [MB_WriteRealParameter, Seite 318,](#)
- [MB_WriteStringParameter, Seite 320,](#)
- [MC_ReadActualPosition, Seite 321,](#)
- [MC_ReadAxisError, Seite 323,](#)
- [MC_ReadStatus, Seite 324,](#)

Die Bausteine IL_SercosAttribute, IL_SercosElementsToParaNo und IL_SercosStringToParaNo sind ab 12VRS Bestandteil der ML_Base_xx_yy.libraries.



Es können pro Parameterkanal maximal 100 Parameteranfragen gleichzeitig gestellt werden.

Es existieren die folgenden Parameterkanäle:

- A-Parameter
- C-Parameter
- S-/P-Parameter
- K-Parameter
- ...

• *Touch Probe*

- [ML_AbortTrigger, Seite 327](#)
- [ML_InitTouchProbe, Seite 328](#)
- [ML_TouchProbe, Seite 334](#)
- [ML_TouchProbeContinuous, Seite 337](#)
- [ML_WriteExpectWindow, Seite 339](#)

Beginnt der jeweilige Bausteinname mit "MC_" bedeutet dies, dass er mit der Vorlage der PLCopen bestmöglich abgestimmt wurde, und dass somit für unterschiedliche Systeme wie XLC/MLC, MLD, usw. vergleichbare Bausteine existieren (Beispiel: MC_GearIn).

Beginnt der jeweilige Bausteinname mit "MB_" bedeutet dies, dass für unterschiedliche Systeme wie XLC, MLC, MLD, usw. vergleichbare Bausteine existieren (Beispiel: MB_GearInPos).

Beginnt der jeweilige Bausteinname mit "ML_" bedeutet dies, dass er für das System IndraMotion MLC entwickelt wurde, um dieses bestmöglich zu unterstützen (Beispiel: ML_FlexProfile).



Da eine Migration von ML_ nach MB_ / MC_ langfristig im Interesse des Kunden liegt, wird diese angestrebt.



Nicht mehr in dieser Bibliothek enthalten sind die Bausteine und Datentypen der Bewegungssteuerung "Coordinated Motion".
Diese wurden in der ML_Robot.library zusammengefasst.

6.2 Globale_Variablen

6.2.1 ProbeSelect

Kurzbeschreibung

Die Globale Variablenliste **ProbeSelect** enthält den Datenbestand, der den Messtastern 1..100 zu zuordnen ist.
ProbeSelect verwendet den Datentyp [PROBECONFIG](#), [Seite 172](#).

Bibliothek	Bereich
ML_PLCopen	Globale_Variablen

Tab. 6-1: Bibliothekszuordnung
Deklaration der Globalen Variablenliste ProbeSelect

```
VAR_GLOBAL
  ProbeSelect:PROBECONFIG;
END_VAR
```

6.3 Data Types/ Data Types of TouchProbe

6.3.1 PROBECONFIG

Kurzbeschreibung

Der Datentyp definiert ein Feld zur Aufnahme von 100 Messtaster-Datensätzen.
Die Datensätze MB_PROBE_CONFIG sind in der Bibliothek RIL_Common-Types vereinbart.

Bibliothek	Bereich
ML_PLCopen	Data types/ Data types of TouchProbe

Tab. 6-2: Bibliothekszuordnung
Typdeklaration PROBECONFIG

```
TYPE PROBECONFIG :
STRUCT
  arProbeConfig: ARRAY [1..100] OF MB_PROBE_CONFIG;
END_STRUCT
END_TYPE
```

6.4 Zustandsdiagramm einer realen Achse

Das nachstehende Zustandsdiagramm definiert das Verhalten von IndraMotion-Achsen aus der Sicht des Anwenders, wenn mehrere IndraMotion-Funktionsbausteine gleichzeitig aktiviert sind. Diese Kombination von Bewegungsprofilen ist bei der Erzeugung komplizierterer Profile und bei der Reaktion auf Ausnahmesituationen in Programmen nützlich.

Grundsätzlich werden Motion-Kommandos sequentiell gegeben und abgearbeitet, selbst wenn die SPS in der Lage ist, eine parallele Verarbeitung vorzunehmen. Die Motion-Kommandos folgen dem Zustandsdiagramm der IndraMotion-Achsen.

Jede Achse ist in genau einem der definierten Zustände (siehe nachstehendes Zustandsdiagramm). Jedes Bewegungskommando stellt einen Zustandsübergang für die konkrete Achse dar. Die Folge dieser Übergänge beschreibt das gesamte Verhalten der Achse.

Die möglichen Status werden im Parameter A-0-0022, Erweiterter Achsstatus angezeigt.

Das Diagramm beschreibt das Verhalten einer Einzelachse.

Multiachs-Funktionsbausteine wie MB_MotionProfile, MB_FlexProfile, MC_CamIn, MC_GearIn und MB_GearInPos können so gesehen werden, dass sich jede Achse aus der Sicht des Betrachters in ihrem speziellen Zustand im Zustandsdiagramm befindet, z.B. kann sich die CAM-Leitachse im Zustand 'Continuous Motion' befinden, während sich die korrespondierende Folgeachse im Zustand 'Synchronized Motion' befindet.

Das Anhängen einer Folgeachse an eine Leitachse hat keinen Einfluss auf das Verhalten der Leitachse.

Funktionsbausteine mit "Verwaltungscharakter", wie z.B. Funktionsbausteine zum Schreiben und Lesen von Parametern haben keinen Einfluss auf das Zustandsdiagramm.

ML_PLCopen.library

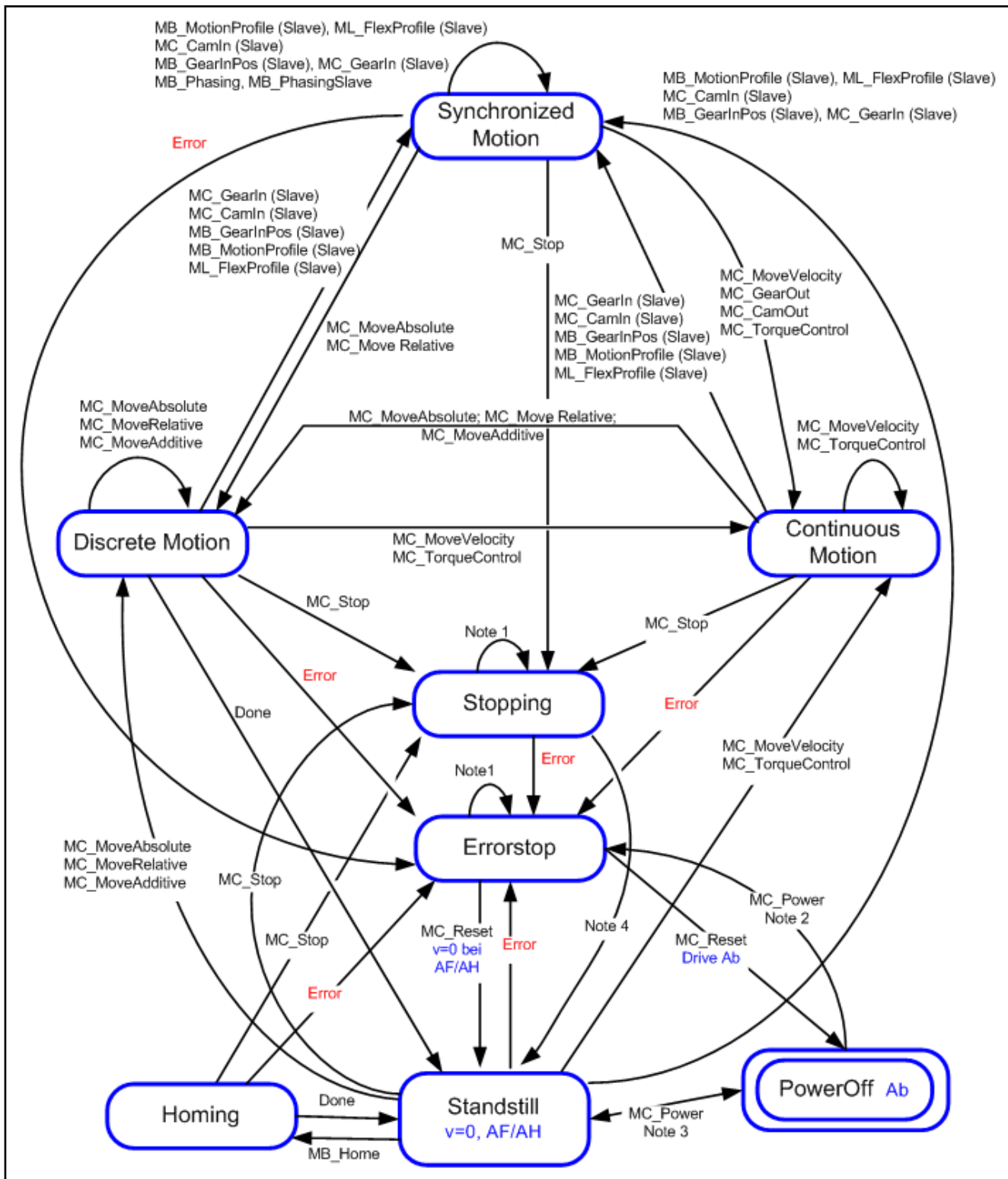


Abb. 6-1: IndraMotion-Zustandsdiagramm einer realen Achse

Legende

(Note 1) - Alle Funktionsbausteine können gerufen werden, ohne dass sie bearbeitet werden, mit Ausnahme von MC_Reset und Error - diese erzeugen jeweils den Übergang zu StandStill oder ErrorStop.

(Note 2) - Power.Enable= TRUE und Fehler in der Achse.

(Note 3) - Power.Enable= TRUE und fehlerfreie Achse.

(Note 4) - MC_Stop.Done AND NOT MC_Stop.Execute

Ein Fehler im Antrieb führt abhängig von der programmierten Fehlerreaktion irgendwann zur Leistungsabschaltung. Die Zustandsmaschine bleibt aber im ErrorStop, MC_Reset führt bei abgeschalteter Leistung in den Power-Off-Zustand.

Das Zustandsdiagramm wird ergänzt durch

1. den Zustand **Controlled Motion**, Seite 265, für geregelte Bewegungen.

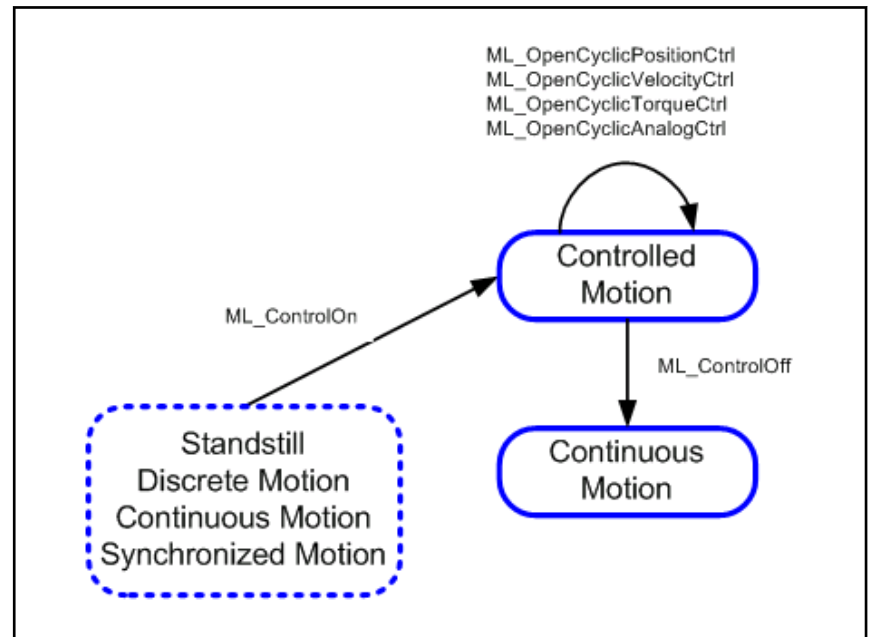


Abb. 6-2: Übergang zum Zustand "Controlled Motion"

Der Zustand 'Controlled Motion' dient dazu, eine Achse einem Regler in der SPS fest zuzuordnen und die Kommandierung über andere Motion Kommandos während dieser Zeit zu verhindern. Der Zustand wird mit dem Funktionsbaustein ML_ControlOn betreten und mit ML_ControlOff verlassen. In diesem Zustand können nur zyklische Kanäle über die Funktionsbausteine ML_OpenCyclic*Ctrl geöffnet werden.

2. die Zustandsdiagramme 'Coordinated Motion', für Achsgruppen (K-0-0022, Erweiterter Kinematik-Status).

ML_PLCopen.library

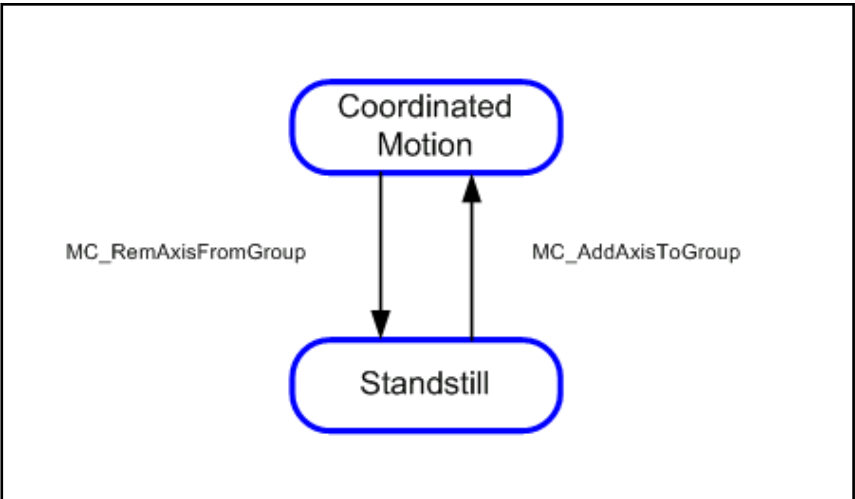


Abb. 6-3: *Übergang zum Zustand "Coordinated Motion"*

Der Zustand 'Coordinated Motion' kann mit Hilfe des Funktionsbausteins MC_AddAxisToGroup erreicht werden. Die Achse ist dann einer Kinematik zugeordnet. Die Kommandierung der Achse durch andere Motion Funktionsbausteine ist untersagt. Mit dem Funktionsbaustein MC_RemoveAxisFromGroup kann die Zuordnung der Achse zu einer Kinematik wieder aufgehoben werden. Die Kommandierung durch andere Motion-Funktionsbausteine ist wieder erlaubt.

6.5 Funktionsbausteine zur Einzelachssteuerung

6.5.1 MC_Power

Kurzbeschreibung Der Funktionsbaustein steuert die Leistungszuschaltung (ein oder aus).
Sie benötigen **für jede reale Achse eine separate Instanz** dieses Bausteines.
Das Aktivierens des Bausteins ist Voraussetzung für jede Bewegung, für Beschleunigen oder Bremsen.
Eine Sondersituation ergibt sich beim Baustein [MC_Stop, Seite 179](#). Hier wird die Position durch den Antrieb aktiv gehalten. Damit kann bei aktivem MC_Stop der Baustein MC_Power nicht abgeschaltet werden.

 Der Baustein führt in Verbindung mit einer virtuellen Achse zu einer Fehlermeldung.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-3: *Bibliothekszuordnung*

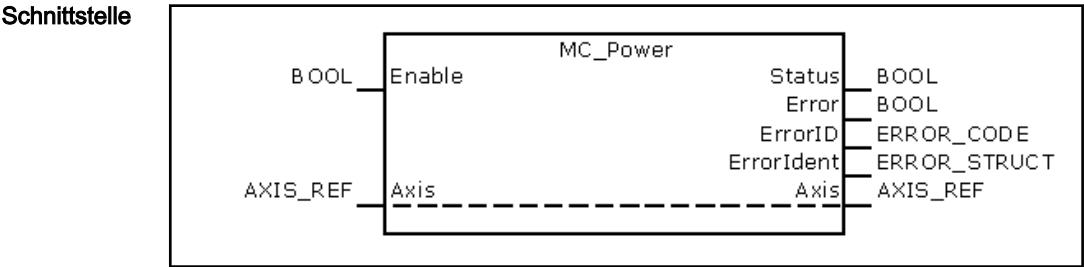


Abb. 6-4: *FB MC_Power*

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Solange "Enable" TRUE ist, ist die Leistung zugeschaltet
VAR_OUTPUT	Status	BOOL	Zustand der Leistungszuschaltung Siehe "Besonderheiten des MC_Power" auf Seite 177
	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

Tab. 6-4: Schnittstelle des FB MC_Power

Besonderheiten des MC_Power

Anwendungsfall "Leistungsabschaltung über MC_Power" durch FALSE am Enable-Eingang.

Die Auswertung des Antriebsstatus sollte nicht über den Statusausgang des Bausteins erfolgen, sondern über einen SPS-Zugriff auf Parameterwerte über AxisData[] - Achsdaten im Shared Memory.

- `AxisData[Drive1.AxisNo].Axis_Power;`

AH_AF" - Antrieb Halt, Antrieb Freigabe

Der Antrieb gibt Leistung ab, d.h. er ist momentan behaftet. Die Antrieb-Halt-Funktion dient dem Stillsetzen einer Achse mit definierter Beschleunigung und definiertem Ruck. "Antrieb Freigabe" bedeutet, dass der Antrieb in der gewünschten Betriebsart freigegeben ist; A-0-0021.15 -RD-

- `AxisData[Drive1.AxisNo].PowerOn;`

Leistungszuschaltung erfolgt; A-0-0022.03 -RD-

6.5.2 MC_Power, Besonderheiten bei Sercos basierten Achsen

- Fehlerbehandlung:**
- Eine Fehlermeldung wird abgesetzt, wenn bei gesetztem "Enable", nach Ablauf einer Wartezeit, der "Status" immer noch FALSE ist. Dies zeigt an, dass ein Hardwareproblem bei der Leistungszuschaltung aufgetreten ist.
 - Eine Fehlermeldung wird ferner dann abgesetzt, wenn versucht wird, die Leistung in einem Achszustand abzuschalten, in dem dies nicht erlaubt ist.
 - Falls die Leistungszuschaltung fehlschlägt oder während einer Bewegung ausfällt, wird für die Achse "ErrorStop" generiert.

Der Funktionsbaustein MC_Power erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.5.3 MC_Reset

Kurzbeschreibung

Der Funktionsbaustein realisiert den Übergang vom Zustand 'ErrorStop' in den Zustand 'StandStill' durch Rücksetzen aller achs- und steuerungsinternen Fehler. Die achsinternen Fehler werden über die entsprechende Antriebsnummer im AXIS_REF gelöscht. Mit dem ENUM NO_OBJECT oder der Achsreferenz NoAxis im AXIS_REF lassen sich die Steuerungsfehler zurücksetzen. Der Funktionsbaustein MC_Reset benutzt im Parametriemodus (Phase 2) den Parameter C-0-1040 oder A-0-1040. Im Betriebsmodus (Phase 4) werden MLC interne Funktionen ausgeführt

ML_PLCopen.library



Die Ausgänge der Funktionsbausteininstanzen, die den/die Fehler angezeigt haben, werden nicht beeinflusst.

Steht kein Fehler an, bleibt das Auslösen des MC_Resets ohne Wirkung.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-5: Bibliothekszuordnung

Schnittstellenbeschreibung

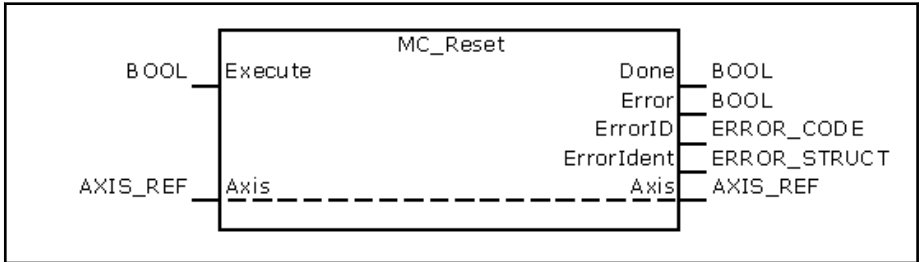


Abb. 6-5: FB MC_Reset

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigenden Flanke
VAR_OUTPUT	Done	BOOL	Fehler sind zurückgesetzt
	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

Tab. 6-6: Schnittstelle des FB MC_Reset

Signal-Zeit-Diagramm

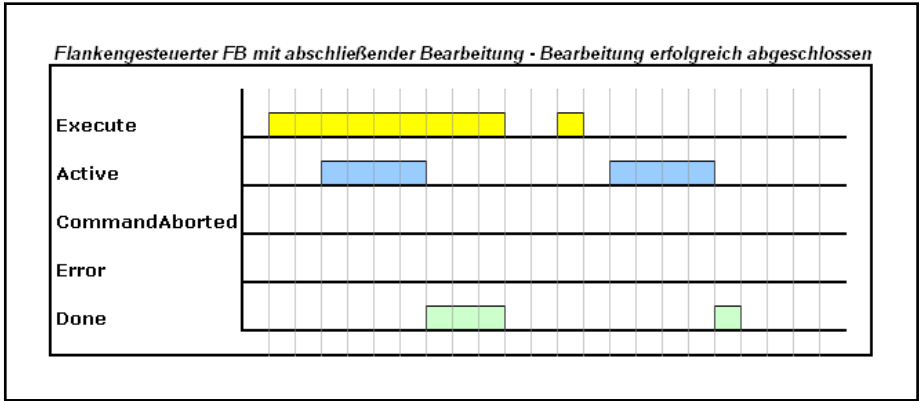


Abb. 6-6: Zeitdiagramm des MC_Reset

6.5.4 MC_Reset, Besonderheiten bei SERCOS-basierten Achsen

Der Baustein benutzt den Parameter C-0-1030, Kommando: Alle Steuerungsfehler löschen.

Fehlerbehandlung

Der Funktionsbaustein MC_Reset erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.5.5 MC_Stop

Kurzbeschreibung

Der Funktionsbaustein realisiert einen gesteuerten Bewegungsstop und überführt die Achse in den Zustand 'Stopping'. Dies stoppt jede beliebige Funktionsbausteinabarbeitung. Mit Erreichen des Stillstandes wird der "Done"-Ausgang gesetzt.

Die Position wird aktiv gehalten.



Solange "Execute" TRUE ist, verbleibt die Achse im Zustand 'Stopping' und kann durch kein anderes Kommando bewegt werden.

Im Zustand 'Stopping' ist es nicht möglich, die Leistung ([MC_Power, Seite 176,](#)) der Achse abzuschalten. Wird das "Enable" der MC_Power-Instanz deaktiviert, bleibt dessen "Status" trotzdem, bis das "Execute" der MC_Stop-Instanz gleichfalls deaktiviert wird, eingeschaltet.



Der MC_Stop wird nicht für einen IndraDrive mit aktivierten Betriebshalt (P-0-0558 Bit 0 = 1) unterstützt.

Der Funktionsbaustein setzt den Done-Ausgang nie auf TRUE.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-7: Bibliothekszuordnung

Schnittstellenbeschreibung

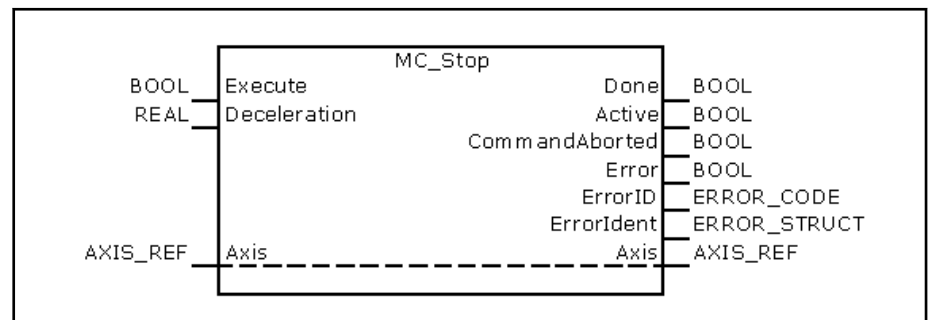


Abb. 6-7: FB MC_Stop

	Name	Typ	Kommentar ¹⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Solange "Execute" TRUE ist, wird die Achse gestoppt bzw. im Zustand "Stopping" gehalten
	Deceleration	REAL	Verzögerung (stets größer 0, sonst kein "Stoppen").
VAR_OUTPUT	Done	BOOL	Stillstand erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando (einer anderen MC_Stop-Instanz)
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist

¹⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

ML_PLCopen.library

	Name	Typ	Kommentar ¹⁾
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 6-8: Schnittstelle des FB MC_Stop

Signal-Zeit-Diagramm

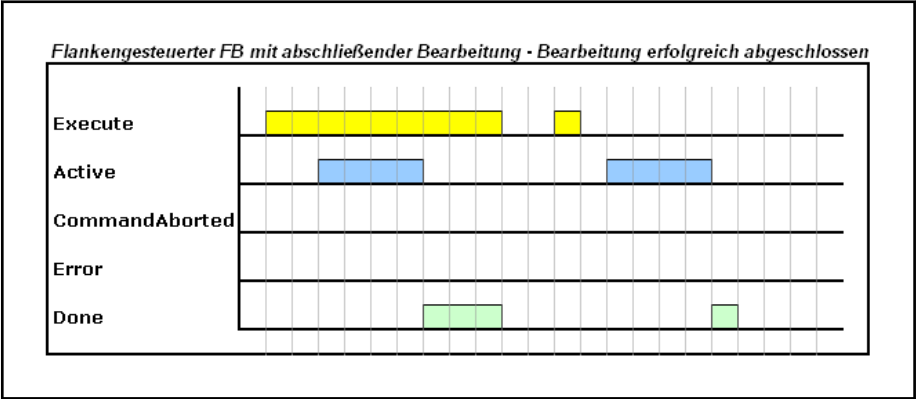


Abb. 6-8: Zeitdiagramm von MC_Stop

6.5.6 MC_Stop, Besonderheiten bei SERCOS-basierten Achsen

Fehlerbehandlung Der Funktionsbaustein MC_Stop erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.5.7 MC_MoveAbsolute

Kurzbeschreibung Der Funktionsbaustein kopiert die absolut vorgegebene "Position" in die "Zielposition" und fährt diese an.



Die Anwendung des Bausteins setzt die vorherige Leistungszuschaltung über eine Instanz des [MC_Power, Seite 176](#), (bei realen Achsen) voraus.

1) Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.



1. Zusätzlich zu den Bausteineingängen wirkt bei dem Funktionsbaustein der Ruckgrenzwert, der über Parameter A-0-0216 eingestellt wird.

Bei eingeschalteter Ruckbegrenzung ($A-0-0216 > 0$) tritt das nachfolgend beschriebene Verhalten auf

2. In der Beschleunigungsphase wirkt immer der Acceleration-Eingang des Bausteins.

Muss in der Beschleunigungsphase die Geschwindigkeit der Achse reduziert werden (z. B. Überschwingen der Ist-Geschwindigkeit) gilt auch hier der Acceleration-Eingang als Grenzwert für die notwendige Verzögerung.

In folgendem Fall kann es damit vorkommen, dass die Achse mit einer größeren Verzögerung abgebremst wird als dies am Deceleration-Eingang festgelegt wurde:

- Ruckbegrenzung ist aktiv ($A-0-0216 > 0$) UND
- Acceleration-Eingang $>$ Deceleration-Eingang UND
- Achse befindet sich in der Beschleunigungsphase UND
- Erneuter Start der Bewegung mit reduzierter Positioniergeschwindigkeit

3. Ohne Ruckbegrenzung ($A-0-0216 = 0$) tritt das Verhalten nicht auf

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-9: Bibliothekszuordnung

Schnittstellenbeschreibung

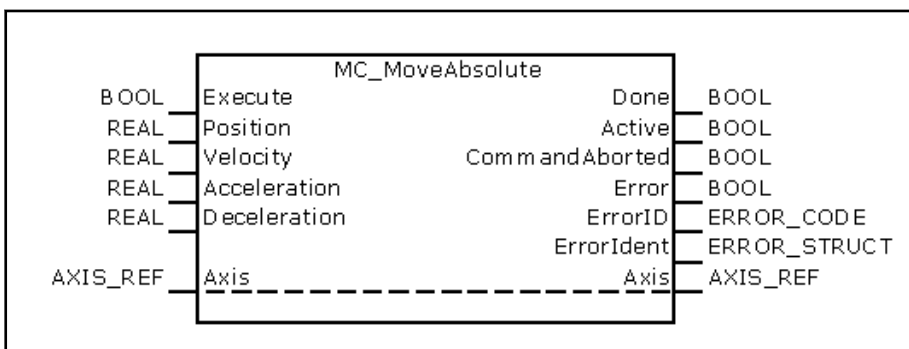


Abb. 6-9: FB MC_MoveAbsolute

	Name	Typ	Kommentar ²⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Position	REAL	Zielposition für die Bewegung (+/-)
	Velocity	REAL	Maximalwert der Geschwindigkeit (stets +) (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).

²⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

ML_PLCopen.library

	Name	Typ	Kommentar ²⁾
	Deceleration	REAL	Verzögerung (stets +).
VAR_OUTPUT	Done	BOOL	gewünschte Position erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-10: Schnittstelle von FB MC_MoveAbsolute

Implementationsbeispiel: In folgenden Bild sind zwei Beispiele für die Zusammenarbeit von zwei FB-Instanzen des "MC_MoveAbsolute" dargestellt:

1. Der linke Teil des Zeitdiagramms zeigt den Fall, dass Instance2 nach Instance1 gerufen wird. Nachdem Instance1 die gewünschte Position 6000 (Geschwindigkeit 0) realisiert hat, aktiviert sein Ausgang "Done" Instance2 mit der Zielposition 10000.
2. Die rechte Seite des Zeitdiagramms zeigt wie die Instance2 bereits aktiviert wird, während die Instance1 noch arbeitet. In diesem Fall wird die Bewegung von Instance1 durch das "Test"-Signal gestoppt und abgebrochen. Instance2 fährt seine Zielposition 10000 direkt an, obwohl die Position 6000 noch nicht erreicht wurde.

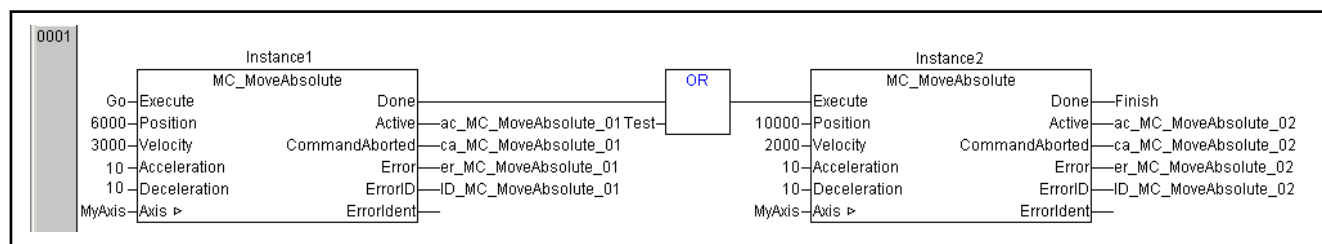


Abb. 6-10: Zwei Instanzen von MC_MoveAbsolute

²⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

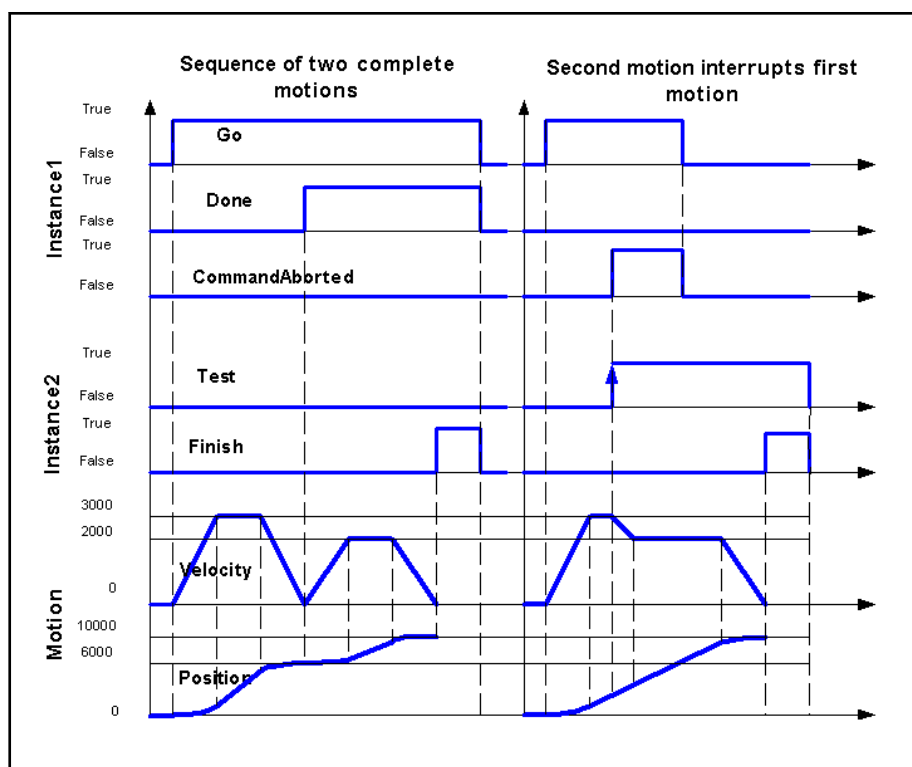


Abb. 6-11: Zeitdiagramm: zwei Instanzen von MC_MoveAbsolute

6.5.8 MC_MoveAbsolute, Besonderheiten bei SERCOS-basierten Achsen

Verwendete Parameter

Der Funktionsbaustein ermittelt die neue Zielposition (A-0-2201, Zielposition) durch Übernahme der Sollposition (A-0-2200, Sollposition) in den Parameter A-0-2201 und fährt diese an.



Das Erreichen der Zielposition wird im Modulo-Betrieb durch den Parameter A-0-0203, Bewegungsrichtung, beeinflusst.

Sonderfunktion:

Die FB's MC_MoveAbsolute, MC_MoveAdditive und MC_MoveRelative akzeptieren eine Geschwindigkeitsvorgabe von Vsoll = 0.

Mit Aufruf dieser FB's (Vsoll = 0) wird die Achse angehalten. Die Positionierung wird lediglich unterbrochen. Die Zielposition in der Achse bleibt erhalten.

Folgt anschließend der Aufruf eines MC_MoveAdditive (Vsoll ≠ 0) wird der Restweg verfahren. Erst mit Aufruf eines anderen Motion-FB's wird die Zielposition überschrieben.

Fehlerbehandlung:

Der Funktionsbaustein MC_MoveAbsolute erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.5.9 MC_MoveAdditive

Kurzbeschreibung

Der Funktionsbaustein addiert die "Distanz" zur "Zielposition" und fährt diese an.



Die Anwendung des Bausteins setzt die vorherige Leistungszuschaltung über eine Instanz des [MC_Power, Seite 176](#), (bei realen Achsen) voraus.

ML_PLCopen.library



1. Zusätzlich zu den Bausteineingängen wirkt bei dem Funktionsbaustein der Ruckgrenzwert, der über Parameter A-0-0216 eingestellt wird.

Bei eingeschalteter Ruckbegrenzung ($A-0-0216 > 0$) tritt das nachfolgend beschriebene Verhalten auf

2. In der Beschleunigungsphase wirkt immer der Acceleration-Eingang des Bausteins.

Muss in der Beschleunigungsphase die Geschwindigkeit der Achse reduziert werden (z. B. Überschwingen der Ist-Geschwindigkeit) gilt auch hier der Acceleration-Eingang als Grenzwert für die notwendige Verzögerung.

In folgendem Fall kann es damit vorkommen, dass die Achse mit einer größeren Verzögerung abgebremst wird als dies am Deceleration-Eingang festgelegt wurde:

- Ruckbegrenzung ist aktiv ($A-0-0216 > 0$) UND
- Acceleration-Eingang > Deceleration-Eingang UND
- Achse befindet sich in der Beschleunigungsphase UND
- Erneuter Start der Bewegung mit reduzierter Positioniergeschwindigkeit

3. Ohne Ruckbegrenzung ($A-0-0216 = 0$) tritt das Verhalten nicht auf

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-11: Bibliothekszuordnung

Schnittstellenbeschreibung

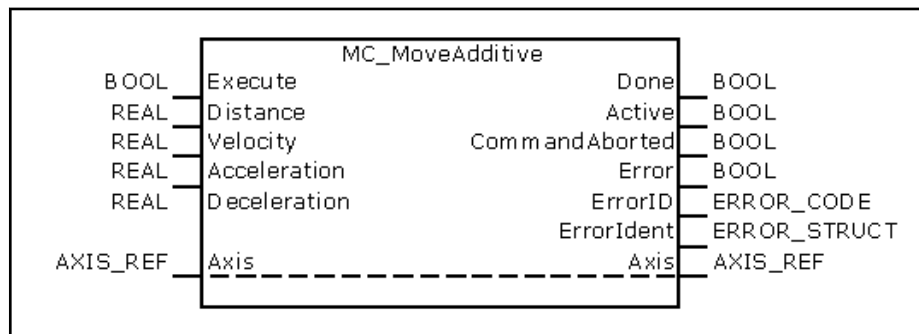


Abb. 6-12: FB MC_MoveAdditive

	Name	Typ	Kommentar ³⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Distance	REAL	Relativentfernung für die Bewegung (+/-).
	Velocity	REAL	Maximalwert der Geschwindigkeit (stets +) (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).
	Deceleration	REAL	Verzögerung (stets +).

³⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

	Name	Typ	Kommentar ³⁾
VAR_OUTPUT	Done	BOOL	gewünschte Position erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-12: Schnittstelle des FBs MC_MoveAdditive

Implementationsbeispiel:

Das nachfolgende Bild zeigt zwei Beispiele der Kombination von zwei FB-Instanzen während die Achse im "Discrete Motion State" ist:



Hier sind zwei unterschiedliche FB-Typen in Benutzung:

- Instance1: MC_MoveAbsolute
- Instance2: MC_MoveAdditive.

1. Der linke Teil des Zeitdiagramms zeigt den Fall, dass der zweite FB nach dem ersten gestartet wird. Der erste erreicht die gewünschte Entfernung 6000 (bei Geschwindigkeit 0). Danach bewirkt der Ausgang "Done", dass der zweite FB die Distanz bis 10000 überwindet.
2. Der rechte Teil des Zeitdiagramms zeigt den Fall, bei dem der zweite FB bereits gestartet wird, bevor der erste seine Arbeit abschließen konnte. In diesem Fall wird die erste Bewegung bei konstanter Geschwindigkeit durch das Signal "Test" gestoppt und abgebrochen. Der zweite FB addiert seine Distanz von 4000 zur ursprünglich befohlenen Position 6000 und bewegt die Achse auf die resultierende Position 10000 mit der am zweiten FB vorgegebenen Geschwindigkeit.

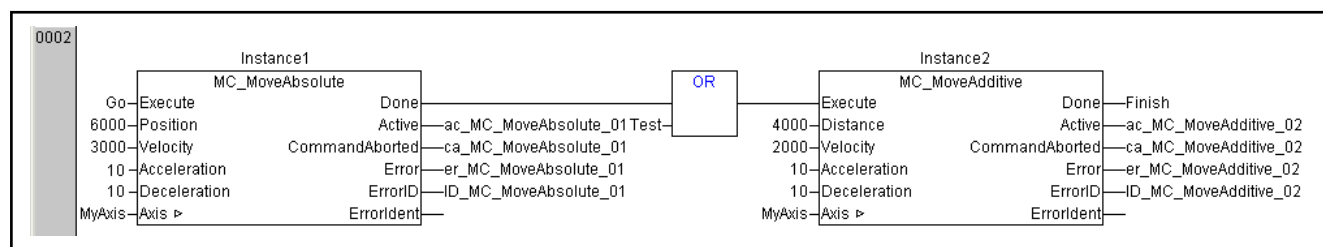


Abb. 6-13: Instanzen von MC_MoveAbsolute und MC_MoveAdditive

³⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

ML_PLCopen.library

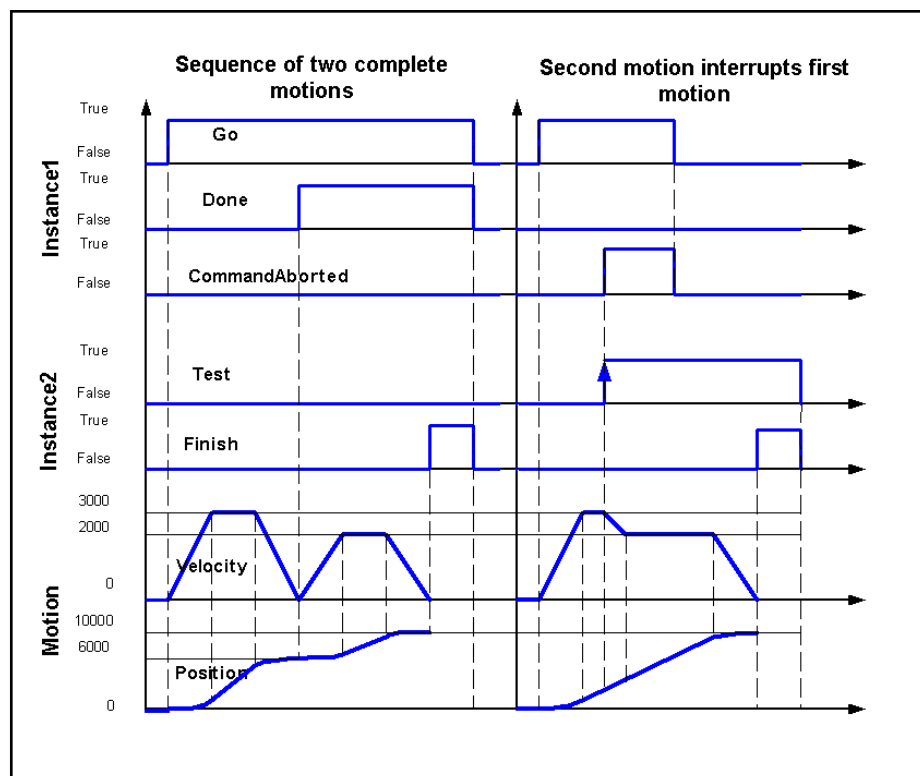


Abb. 6-14: Zeitdiagramm: Instanzen von MC_MoveAbsolute und MC_MoveAdditive

6.5.10 MC_MoveAdditive, Besonderheiten bei SERCOS-basierten Achsen

Verwendete Parameter

Der Funktionsbaustein ermittelt die neue Zielposition (A-0-2201, Zielposition) durch Addition der Distanz (A-0-2200, Sollposition) zur vorherigen Zielposition und fährt diese an.



Die Distanz wird relativ zur Istposition angegeben. Der Parameter A-0-0203, Bewegungsrichtung, ist unwirksam und überschreibt das Vorzeichen der Distanz nicht!

Sonderfunktion: Die FB's MC_MoveAbsolute, MC_MoveAdditive und MC_MoveRelative akzeptieren eine Geschwindigkeitsvorgabe von $V_{soll} = 0$.

Mit Aufruf dieser FB's ($V_{soll} = 0$) wird die Achse angehalten. Die Positionierung wird lediglich unterbrochen. Die Zielposition in der Achse bleibt erhalten.

Folgt anschließend der Aufruf eines MC_MoveAdditive ($V_{soll} \neq 0$) wird der Restweg verfahren. Erst mit Aufruf eines anderen Motion-FB's wird die Zielposition überschrieben.

Fehlerbehandlung: Der Funktionsbaustein MC_MoveAdditive erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.5.11 MC_MoveRelative

Kurzbeschreibung

Der Funktionsbaustein ermittelt die neue "Zielposition" durch Addition der "Distanz" und der "Istposition" und fährt diese an.



Die Anwendung des Bausteins setzt die vorherige Leistungszuschaltung über eine Instanz des **MC_Power**, Seite 176, (bei realen Achsen) voraus.



1. Zusätzlich zu den Bausteineingängen wirkt bei dem Funktionsbaustein der Ruckgrenzwert, der über Parameter A-0-0216 eingestellt wird.

Bei eingeschalteter Ruckbegrenzung ($A-0-0216 > 0$) tritt das nachfolgend beschriebene Verhalten auf

2. In der Beschleunigungsphase wirkt immer der Acceleration-Eingang des Bausteins.

Muss in der Beschleunigungsphase die Geschwindigkeit der Achse reduziert werden (z. B. Überschwingen der Ist-Geschwindigkeit) gilt auch hier der Acceleration-Eingang als Grenzwert für die notwendige Verzögerung.

In folgendem Fall kann es damit vorkommen, dass die Achse mit einer größeren Verzögerung abgebremst wird als dies am Deceleration-Eingang festgelegt wurde:

- Ruckbegrenzung ist aktiv ($A-0-0216 > 0$) UND
- Acceleration-Eingang $>$ Deceleration-Eingang UND
- Achse befindet sich in der Beschleunigungsphase UND
- Erneuter Start der Bewegung mit reduzierter Positioniergeschwindigkeit

3. Ohne Ruckbegrenzung ($A-0-0216 = 0$) tritt das Verhalten nicht auf

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-13: Bibliothekszuordnung

Schnittstellenbeschreibung

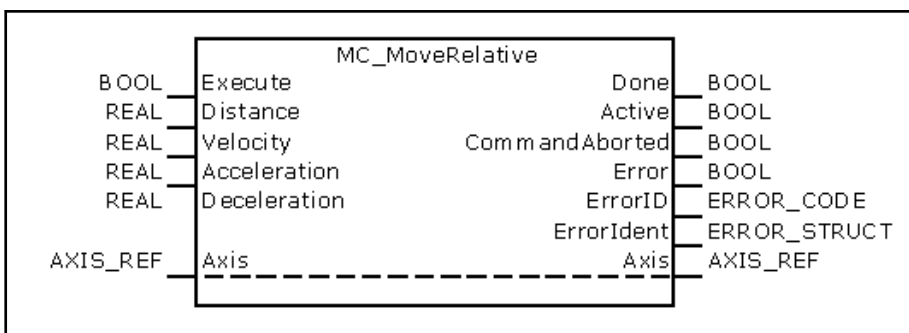


Abb. 6-15: FB MC_MoveRelative

	Name	Typ	Kommentar ⁴⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Distance	REAL	Relativentfernung für die Bewegung (+/-).

⁴⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

ML_PLCOpen.library

	Name	Typ	Kommentar ⁴⁾
	Velocity	REAL	Maximalwert der Geschwindigkeit (stets +) (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).
	Deceleration	REAL	Verzögerung (stets +).
VAR_OUTPUT	Done	BOOL	Gewünschte Position erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-14: Schnittstelle von FB MC_MoveRelative

Signal-Zeit-Diagramm

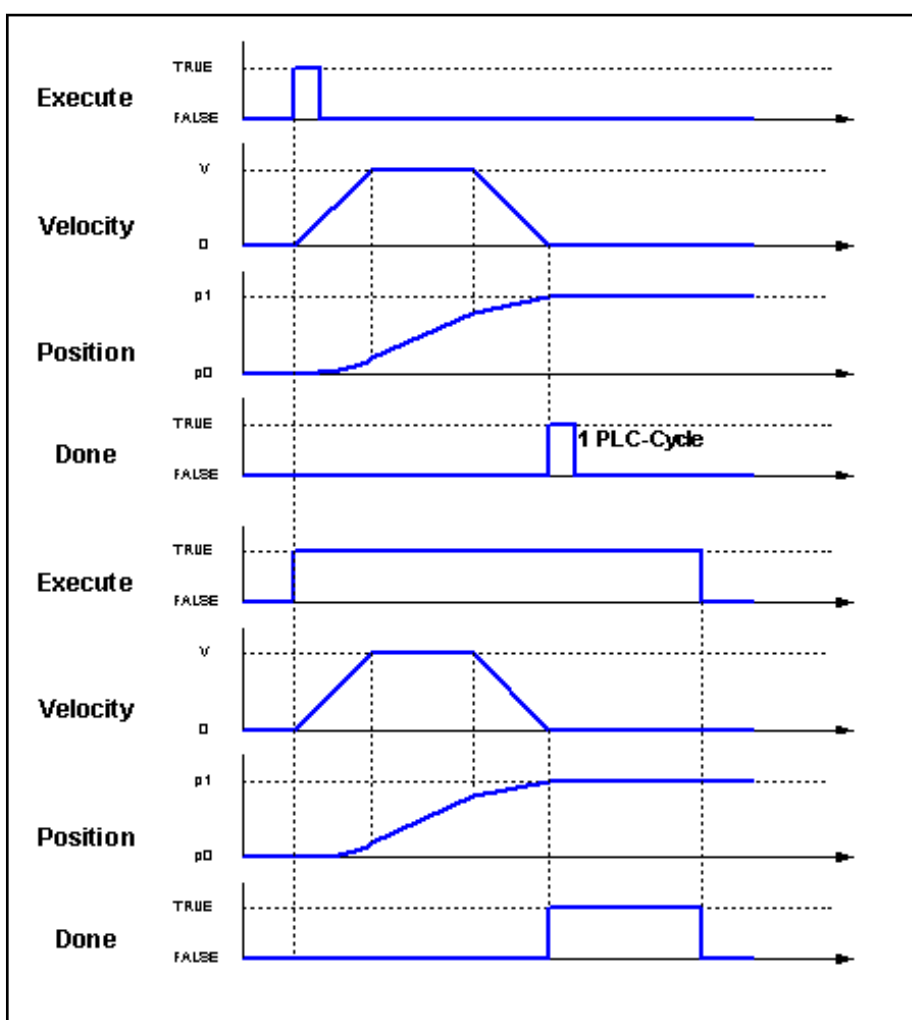


Abb. 6-16: Zeitdiagramme zu MC_MoveRelative

⁴⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

Implementationbeispiel

Das folgende Beispiel zeigt die Kombination von zwei FB-Instanzen MC_MoveRelative.

1. Der linke Teil des Zeitdiagramms zeigt den Fall, in dem Instance2 nach Abarbeitung von Instance1 gerufen wird. Nachdem Instance1 die gewünschte Distanz 6000 (Geschwindigkeit 0) erreicht hat, verursacht sein Ausgang "Done" über Instance2 eine weitere Bewegung um 4000 auf die Gesamtentfernung 10000.
2. Der rechte Teil zeigt, wie Instance2 bereits während der Arbeit von Instance1 durch das Signal "Test" aktiviert wird. In diesem Fall wird die erste Bewegung während der konstanten Geschwindigkeit von Instance1 abgebrochen. Instance1 addiert zur aktuellen Position (3250) noch seine Entfernung von 4000. Die Achse bewegt sich zur resultierenden Position von 7250.

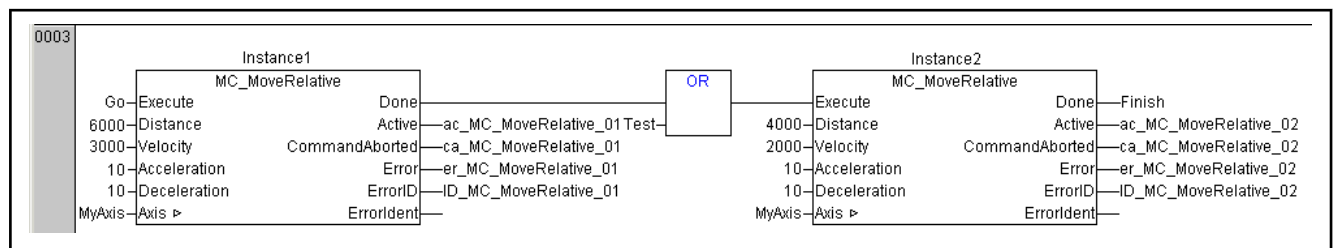


Abb. 6-17: Zwei Instanzen von MC_MoveRelative

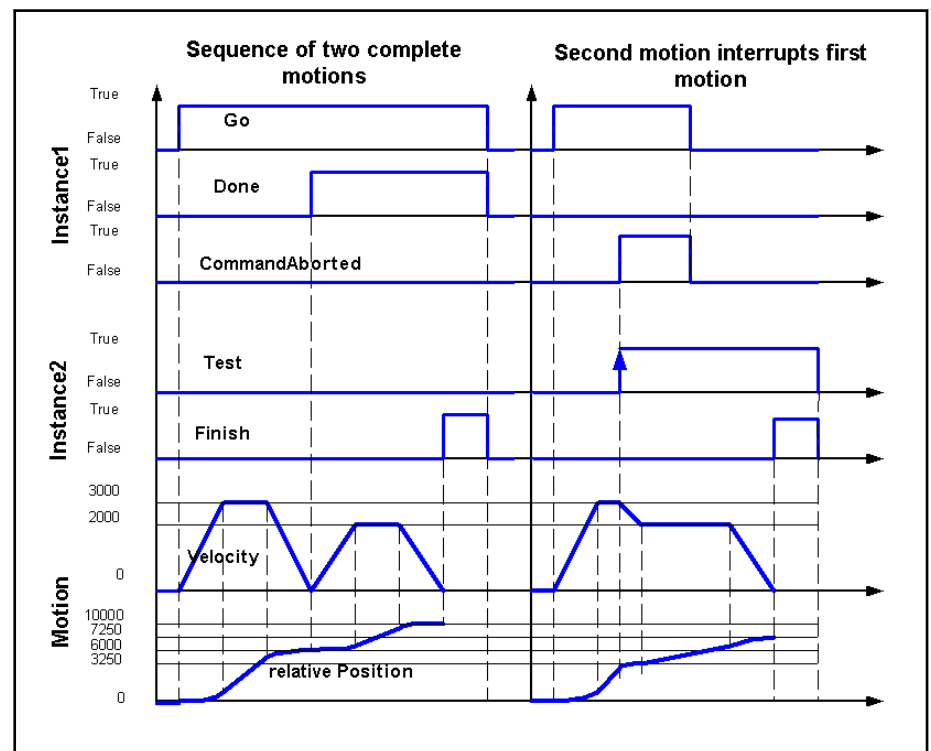


Abb. 6-18: Zeitdiagramm: zwei Instanzen von MC_MoveRelative

6.5.12 MC_MoveRelative, Besonderheiten bei SERCOS-basierten Achsen

Verwendete Parameter

Der Funktionsbaustein ermittelt die neue Zielposition (A-0-2201, Zielposition) durch Addition der Distanz (A-0-2200, Sollposition) und der Istposition (A-0-0100, Positionswert) und fährt diese an.

ML_PLCopen.library



Die Distanz wird relativ zur Istposition angegeben. Der Parameter A-0-0203, Bewegungsrichtung, ist unwirksam und überschreibt das Vorzeichen der Distanz nicht!

Sonderfunktion: Die FB's MC_MoveAbsolute, MC_MoveAdditive und MC_MoveRelative akzeptieren eine Geschwindigkeitsvorgabe von Vsoll = 0.

Mit Aufruf dieser FB's (Vsoll = 0) wird die Achse angehalten. Die Positionierung wird lediglich unterbrochen. Die Zielposition in der Achse bleibt erhalten.

Folgt anschließend der Aufruf eines MC_MoveAdditive (Vsoll <> 0) wird der Restweg verfahren. Erst mit Aufruf eines anderen Motion - FB's wird die Zielposition überschrieben.

Fehlerbehandlung: Der Funktionsbaustein MC_MoveRelative erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.5.13 MC_MoveVelocity

Kurzbeschreibung Der Funktionsbaustein steuert eine "endlose" Bewegung mit der jeweils festgelegten Geschwindigkeit. Um die Bewegung zu beenden, muss die FB-Instanz durch eine andere FB-Instanz, die ein neues Kommando ausgibt, unterbrochen werden.



Die Anwendung des Bausteins setzt die vorherige Leistungsschaltung über eine Instanz des [MC_Power, Seite 176](#), (bei realen Achsen) voraus.



1. Zusätzlich zu den Bausteineingängen wirkt bei dem Funktionsbaustein der Ruckgrenzwert, der über Parameter A-0-0216 eingestellt wird.

Bei eingeschalteter Ruckbegrenzung (A-0-0216 > 0) tritt das nachfolgend beschriebene Verhalten auf

2. In der Beschleunigungsphase wirkt immer der Acceleration-Eingang des Bausteins.

Muss in der Beschleunigungsphase die Geschwindigkeit der Achse reduziert werden (z. B. Überschwingen der Ist-Geschwindigkeit) gilt auch hier der Acceleration-Eingang als Grenzwert für die notwendige Verzögerung.

In folgendem Fall kann es damit vorkommen, dass die Achse mit einer größeren Verzögerung abgebremst wird als dies am Deceleration-Eingang festgelegt wurde:

- Ruckbegrenzung ist aktiv (A-0-0216 > 0) UND
- Acceleration-Eingang > Deceleration-Eingang UND
- Achse befindet sich in der Beschleunigungsphase UND
- Erneuter Start der Bewegung mit reduzierter Positioniergeschwindigkeit

3. Ohne Ruckbegrenzung (A-0-0216 = 0) tritt das Verhalten nicht auf

Das Signal "InVelocity" wird zurückgesetzt, wenn die FB-Instanz durch einen andere FB-Instanz unterbrochen wird (CommandAborted) oder bei fallender Flanke seines "Execute"-Eingangs.



Bei Abschalten des "Execute" läuft der Antrieb mit der gleichen Geschwindigkeit weiter.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-15: Bibliothekszuordnung

Schnittstellenbeschreibung

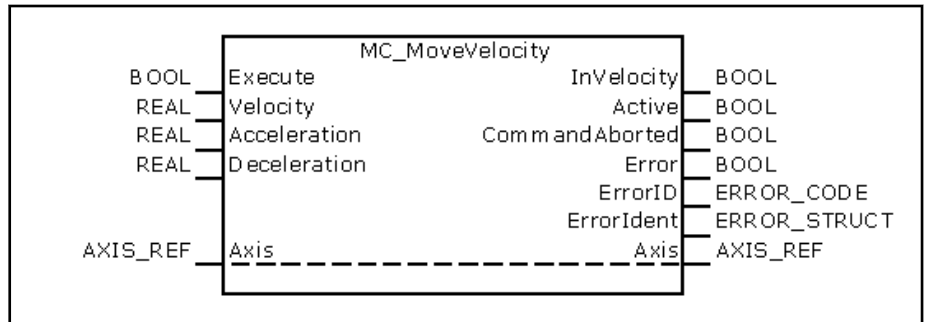


Abb. 6-19: FB MC_MoveVelocity

	Name	Typ	Kommentar ⁵⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Velocity	REAL	Maximalwert der Geschwindigkeit (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).
	Deceleration	REAL	Abbremsen (stets +).
VAR_OUTPUT	InVelocity	BOOL	Geschwindigkeit erreicht (erstmalig erreicht)
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-16: Schnittstelle von FB MC_MoveVelocity

⁵⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

ML_PLCopen.library

Signal-Zeit-Diagramm

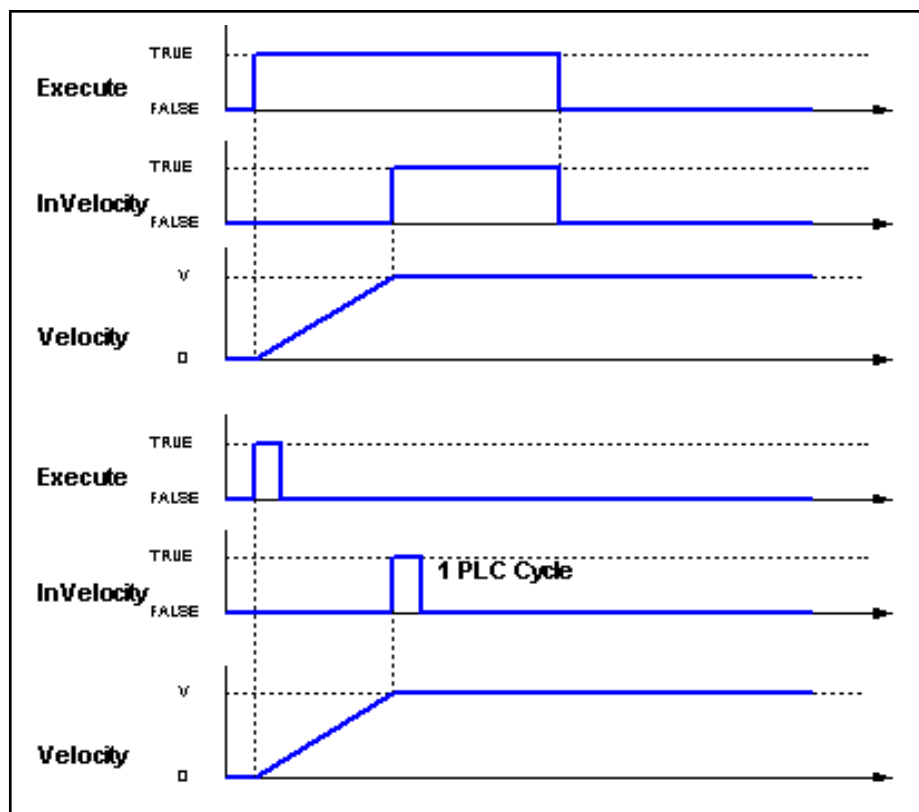


Abb. 6-20: Zeitdiagramm zu MC_MoveVelocity (ohne Kommandoabbruch)

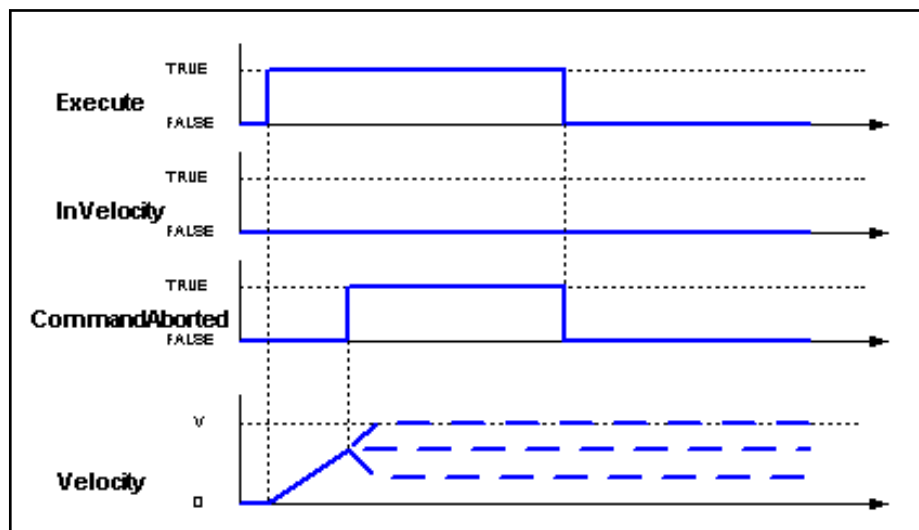


Abb. 6-21: Zeitdiagramm zu MC_MoveVelocity (mit Kommandoabbruch)

Implementationsbeispiel:

Das nachstehende Bild zeigt zwei Beispiele der Kombination von zwei Instanzen der FBs MC_MoveVelocity:

1. Der linke Teil des Zeitdiagramms zeigt den Fall, bei dem die zweite Instanz gerufen wird, nachdem die Arbeit der ersten abgeschlossen ist. Nachdem der erste die geforderte Geschwindigkeit 3000 erreicht hat, bewirken der Ausgang "Instance1.InVelocity" UND das Signal "Next", dass der zweite FB die Bewegung bei Geschwindigkeit 2000 fortsetzt.
2. Der rechte Teil des Zeitdiagramms zeigt den Fall, bei dem der erste FB noch nicht die gewünschte Geschwindigkeit erreicht hat, wenn der zweite gestartet wird. Nachfolgend ist gezeigt: Die erste Bewegung wird er-

neut gestartet durch "Go" am Eingang "Instance1.Execute". Während der erste FB noch beschleunigt, um die Geschwindigkeit 3000 zu erreichen, wird er durch das Signal "Test" unterbrochen, dass den zweiten FB startet. Jetzt läuft der zweite FB und vermindert die Geschwindigkeit auf 2000.

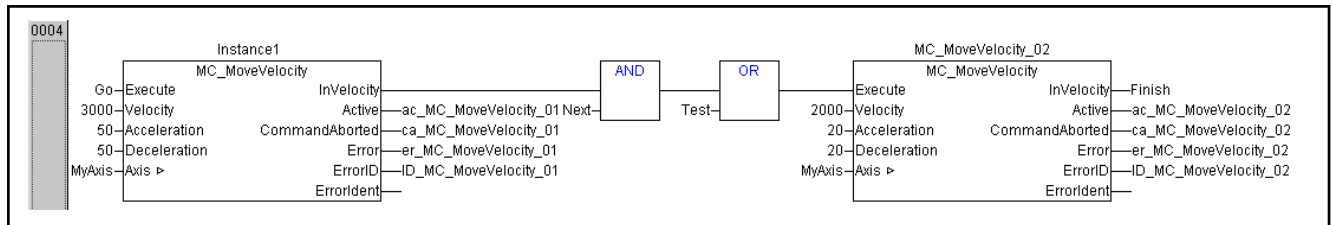


Abb. 6-22: Zwei Instanzen von MC_MoveVelocity

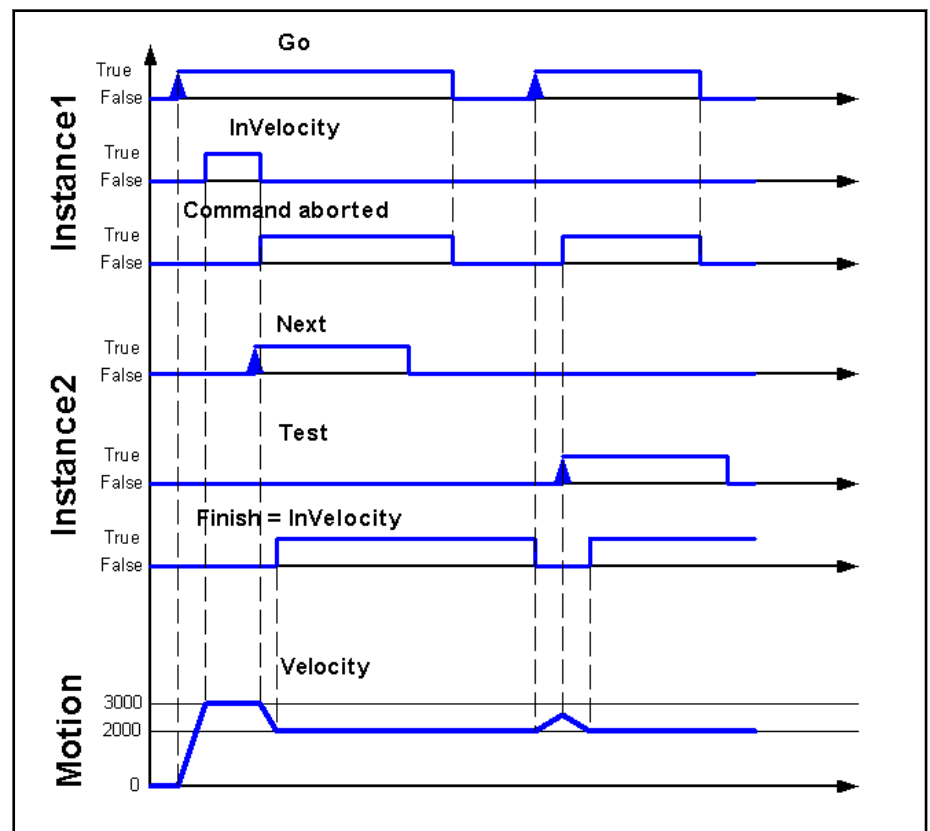


Abb. 6-23: Zeitdiagramm: zwei Instanzen von MC_MoveVelocity

6.5.14 MC_MoveVelocity, Besonderheiten bei SERCOS-basierten Achsen

Fehlerbehandlung: Der Funktionsbaustein MC_MoveVelocity erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.5.15 MC_TorqueControl

Kurzbeschreibung Der Funktionsbaustein steuert eine "kontinuierlich vorgebbare Kraft mit einer definierten Krafterampe in einer niemals endenden Bewegung". Der Funktionsbaustein ist je nach Wichtung für Kraft- und Drehmomentenvorgabe geeignet (positive Kraft-/ Drehmomentenvorgabe im Sinne positiver Geschwindigkeitsvorgabe). Um die Bewegung zu beenden, muss die FB-Instanz durch

ML_PLCopen.library

eine andere FB-Instanz, die ein neues Kommando ausgibt, unterbrochen werden.

 Die Anwendung des Bausteins setzt die vorherige Leistungsschaltung über eine Instanz des [MC_Power, Seite 176](#), (bei realen Achsen) voraus.

 In Abhängigkeit der Achskonfiguration Interpolation im Antrieb / Steuerung wirkt der MC_TorqueControl wie folgt:

Bei **Interpolation im Antrieb** wird die Drehmomentenrampe im Antrieb berechnet. Der MC_TorqueControl schreibt auf die Parameter:

- Torque-Eingang wirkt auf S-0-0080, Drehmoment-/Kraft-Sollwert
- TorqueRamp-Eingang wirkt auf S-0-0822, Drehmoment-/Kraft-Rampe
- S-0-0823, Drehmoment-/Kraft-Rampenzeit. Die Steuerungsfirmware konfiguriert den Parameter fest mit 1000ms

Bei **Interpolation in der Steuerung** wird die Drehmomentenrampe in der Steuerung berechnet.

- Die Steuerung schreibt zyklisch auf „S-0-0080, Drehmoment-/Kraft-Sollwert“.
- Die Bildung der Drehmomentenrampe im Antrieb ist deaktiviert (S-0-0822 = 0, S-0-0823 = 0).
- Der TorqueRamp-Eingang entspricht einer Drehmomentenveränderung innerhalb 1000ms.

 *Die Funktionalität wird von folgenden Achstypen unterstützt:*

- IndraDrive mit Interpolation im Antrieb
- IndraDrive mit Interpolation in der Steuerung
- SercosDrive
- Reglerachse

Das Signal "InTorque" wird zurückgesetzt, wenn die FB-Instanz durch eine andere FB-Instanz unterbrochen wird (CommandAborted) oder bei fallender Flanke ihres "Execute"-Eingangs.

 Bei Abschalten des "Execute" läuft der Antrieb mit der gleichen Geschwindigkeit weiter.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-17: Bibliothekszuordnung

Schnittstellenbeschreibung

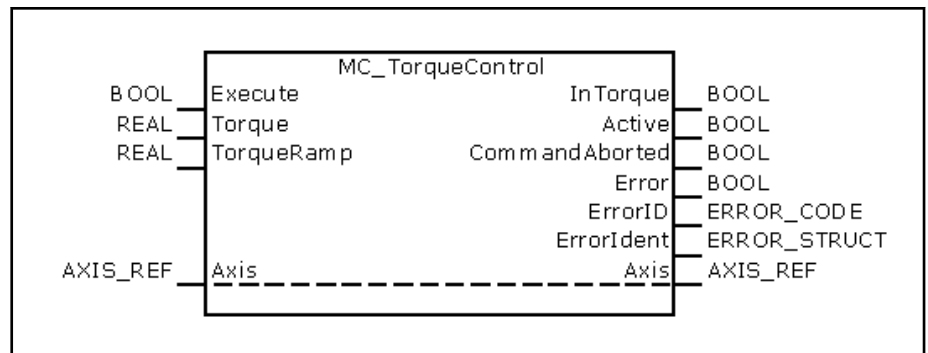
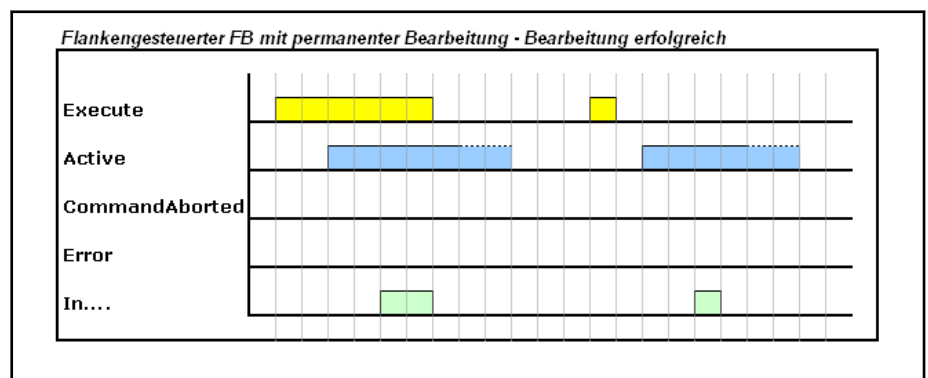


Abb. 6-24: FB MC_TorqueControl

	Name	Typ	Kommentar ⁶⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Torque	REAL	Maximalwert der Kraft/ des Drehmomentes (stets positiv, Wert muss nicht unbedingt erreicht werden).
	TorqueRamp	REAL	Kraft-/ Drehmoment-Differenz, die innerhalb der Rampenzeit erreicht werden kann
VAR_OUTPUT	InTorque	BOOL	Kraft/ Drehmoment erreicht (erstmalig erreicht)
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-18: Schnittstelle von FB MC_TorqueControl

Signal-Zeit-Diagramm



In.... InTorque

Abb. 6-25: Signal-Zeitverlauf des MC_TorqueControl

6.5.16 MC_TorqueControl, Besonderheiten bei Sercos basierten Achsen

Verwendete Parameter

Der Baustein arbeitet mit folgenden Sercos Parametern zusammen:

- S-0-0091, Geschwindigkeitsgrenzwert bipolar (vor FB-Aufruf setzen!)

⁶⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung, der Verzögerung und der Kraft / des Drehmomentes sind von der wirksamen Wichtung abhängig.

- S-0-0823, Rampenzeit (vor FB-Aufruf setzen!)



Rampenzeit auf 1000ms einstellen, zu programmierende Drehmoment-/ Kraftdifferenz bezogen auf eine Sekunde am Baustein-eingang vorgeben.

- S-0-0822, Drehmoment-/ Kraftrampe, wird vom Baustein übernommen.



Die Parameter S-0-0822 und S-0-0823 werden von den Antriebsvarianten ab HNC3x-Sercos und MPH04 unterstützt.



Die Funktionalität wird von folgenden Achstypen unterstützt:

- IndraDrive mit Interpolation im Antrieb
- IndraDrive mit Interpolation in der Steuerung
- SercosDrive
- Reglerachse

Fehlerbehandlung: Der Funktionsbaustein MC_TorqueControl erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.5.17 MB_SetPositionControlMode

Kurzbeschreibung Der Funktionsbaustein "MB_SetPositionControlMode" dient zur Einstellung des Lagereglers.

Mit dem Funktionsbaustein können die Art der Lageregelung (schleppfehlerbehaftet bzw. schleppfehlerfrei) und die Auswahl des Positionsiswertes (Motorgeber bzw. externer Geber) eingestellt werden.

Die Bezugswerte für das relative Positionieren können eingestellt werden.

Bibliothek	Bereich
ML_PLCopen.lib	Motion

Tab. 6-19: Bibliothekszuordnung

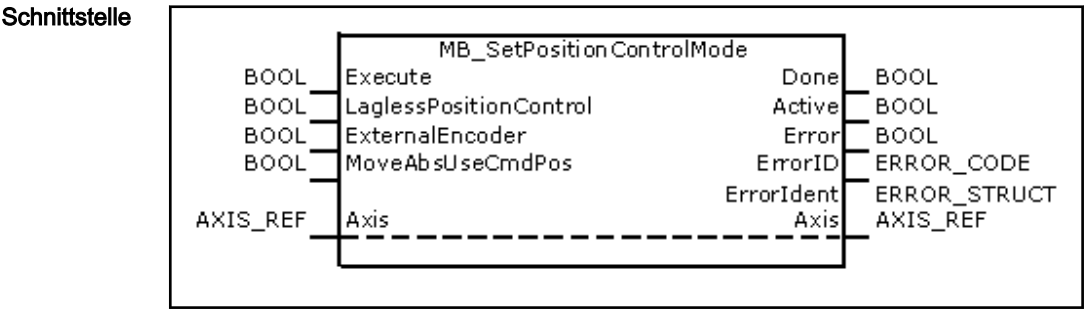


Abb. 6-26: MB_SetPositionControlMode

	Name	Datentyp	Beschreibung
VAR_IN_OUT	Axis	AXIS_REF	Enthält Information zur konkreten Achse
VAR_INPUT	Execute	BOOL	Bearbeitungsfreigabe des Funktionsbausteins (einmalig, flankengesteuert)

	Name	Datentyp	Beschreibung
	LaglessPosition-Control	BOOL	Beeinflusst Bit 0 von "A-0-0007, Achskonfiguration": FALSE: Lageregelung mit Schleppfehler TRUE: schleppfehlerfreie Lageregelung
	ExternalEncoder	BOOL	Beeinflusst Bit 1 des "A-0-0007, Achskonfiguration": FALSE: Regelung mit Motorgeber (Lageistwert-1) TRUE: Regelung mit externem Geber (Lageistwert-2)
	MoveAbsUseCmd-Pos	BOOL	Beeinflusst Bit 2 von "A-0-0201, Interpolatorkonfiguration": FALSE: aktueller Positionswert als Bezugspunkt für relative Positionswerte TRUE: letzte wirksame Zielposition als Bezugspunkt für relative Positionswerte
VAR_OUTPUT	Done	BOOL	"Done"-TRUE zeigt an, dass die Bearbeitung des Auftrages erfolgreich abgeschlossen ist
	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

Tab. 6-20: Schnittstelle des FBs MB_SetPositionControlMode

Funktionsbeschreibung

Der Eingang "LaglessPositionControl" legt fest, ob durch die Geschwindigkeitsvorsteuerung schleppfehlerfrei geregelt werden soll.

Mit dem Eingang "ExternalEncoder" wird der Lageistwert ausgewählt, mit dem der Lageregelkreis geschlossen wird. Unterstützt werden die Achstypen "IndraDrive" und "HydraulicDrive".

Bei Konfiguration eines externen Gebers als Messradgeber wird der Messradbetrieb ein- und ausgeschaltet.



Die Achse am Eingang "Axis", darf nicht als Leitachse für eine reale Achse verwendet sein, bzw. darf auch keine Leitachsen in einem MLC-Querverbund produzieren.

Fehlerbehandlung

Der Funktionsbaustein MB_SetPositionControlMode erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, ERROR_TABLE 16#0030](#) , Seite 348 - Fehler beim Zugriff auf die Achse / die Steuerung.

6.5.18 MB_Home

Kurzbeschreibung

Der Funktionsbaustein steuert **Achsen mit Inkrementalgeber** bei der Ausführung der "search home"-Sequenz. Die Details der Sequenz sind über Achsparameter voreinzustellen. Das "Homing" endet im Stillstand.



Der Funktionsbaustein MB_Home darf nicht in Verbindung mit virtuellen Achsen verwendet werden.

ML_PLCopen.library

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-21: Bibliothekszuordnung

Schnittstellenbeschreibung

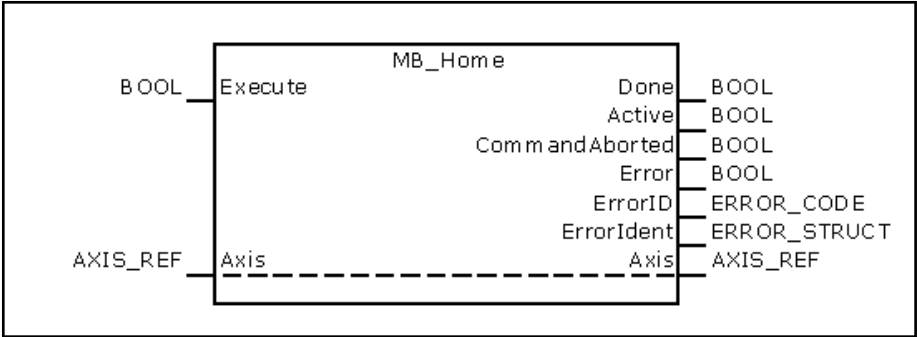


Abb. 6-27: FB MB_Home

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
VAR_OUTPUT	Done	BOOL	Homing abgeschlossen, Geschwindigkeit 0
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-22: Schnittstelle von FB MB_Home

Signal-Zeit-Diagramm

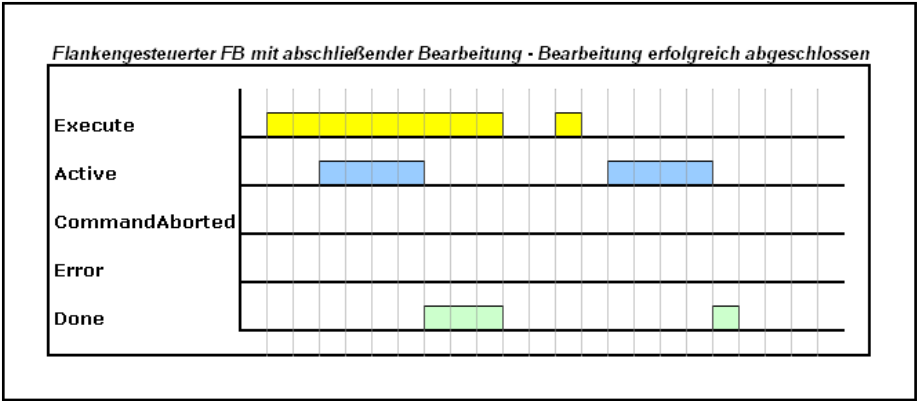


Abb. 6-28: Zeitdiagramm von MB_Home

6.5.19 MB_Home, Besonderheiten bei Sercos basierten Achsen

Der Funktionsbaustein MB_Home darf nicht in Verbindung mit virtuellen Achsen verwendet werden.

Fehlerbehandlung: Der Funktionsbaustein MB_Home erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.6 Funktionsbausteine zur Multiachssteuerung

6.6.1 Die MotionControl-Wirkungskette

Die Funktionbausteine MC_CamIn, MB_GearInPos, MC_GearIn, MB_MotionProfile und ML_FlexProfile wurden jeweils zu MotionControl-Wirkungsketten erweitert.

Zusätzlich zu den bisher möglichen sercos Zykluszeiten von 2, 4, 8, 16ms ist eine **sercos Zykluszeit von 1ms** einstellbar.

Nachstehende Antriebe sind verwendbar:

Unterstützung IndraDrive

- MPx03,
- MPx04 (BA03),
- MPD,
- MPx05 (BA03),
- MPx06,
- MPx07,
- MPx08,
- MPx16,
- MPx17,
- MI.

Unterstützung SercosDrive (Pack Profile)

- Antriebe (DIAX02, 03, 04),

Unterstützung IndraWorks

- IndraDrive (Pack Profile),
- EcoCS (Pack Profile).

ML_PLCopen.library

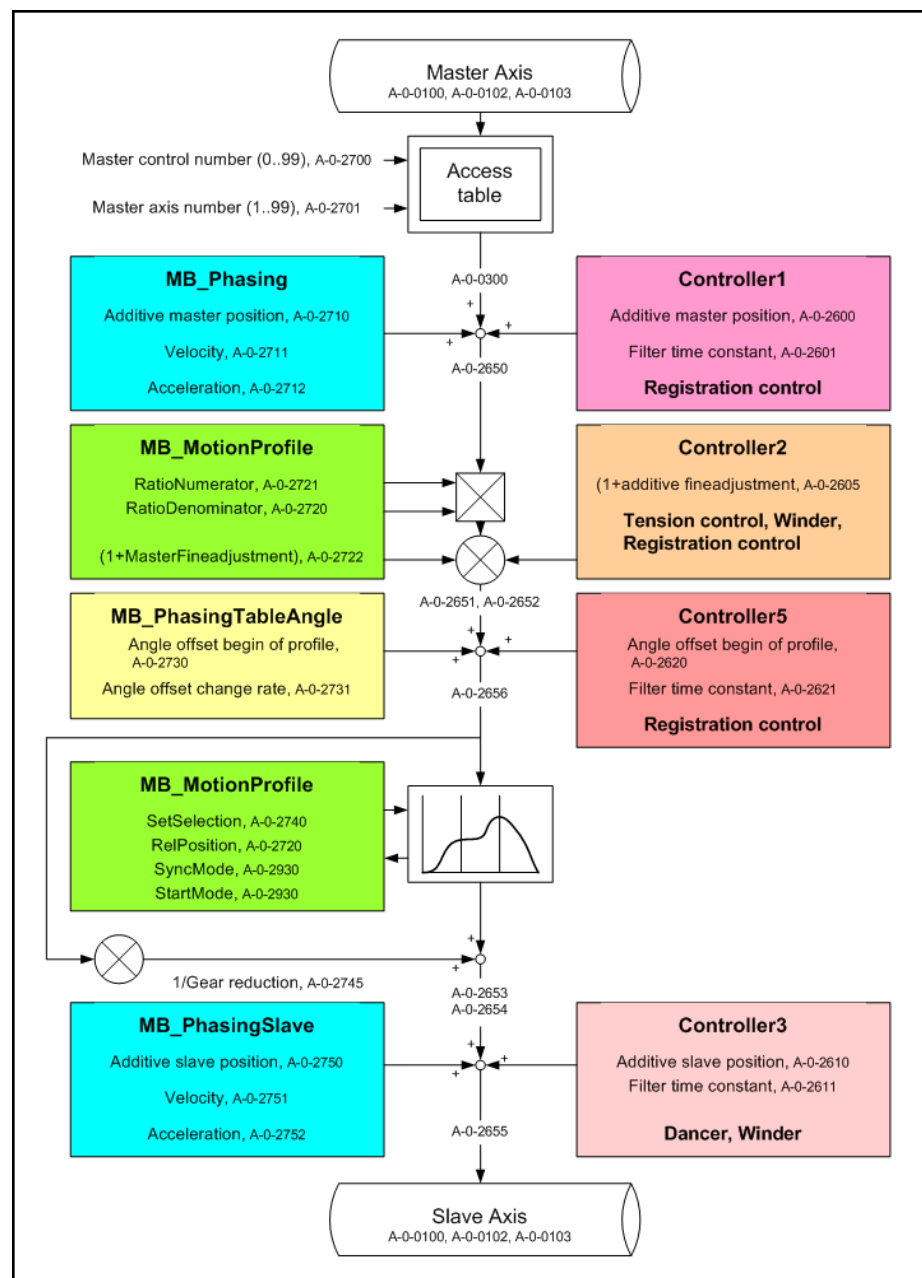


Abb. 6-29: Wirkungskette MotionProfile als Beispiel

Bei aktivem MB_MotionProfile ist es möglich die FBs MB_Phasing, MB_PhasingSlave und später auch "MB_PhasingTableAngle" (nicht in der aktuellen Version) parallel auszuführen.

Aus "MB_Phasing" heraus programmierte Bewegungen überlagern sich mit der des Haupt-Funktionsbausteines "MB_MotionProfile". Das gleiche gilt für "MB_PhasingSlave" und später auch für "MB_PhasingTableAngle".

Alle genannten FBs sind mit eigenen Parametern und eigenem Interpolator ausgestattet.

Zu jedem parallelen FB existiert zusätzlich ein Controller. Die additiven Positionen aus den Controllern wirken an der gleichen Summationsstelle wie der zugehörige parallele FB. Die über die Controller summierten Positionen werden nicht interpoliert, sondern wirken über ein Filter 1.Ordnung mit programmierbarer Zeitkonstante.

Am Ende der Wirkungskette entsteht A-0-2655, ELS, Positionssollwert. Die Folgeachse kann eine virtuelle Achse (kaskadierbar) oder eine reale generische Achse sein. Der Lagesollwert der Folgeachse wird über sercos Lagesollwertschnittstelle dem Antrieb direkt vorgegeben.

Für die meisten Funktionsbausteine ist es (ab Version MLC 03VRS) möglich entweder im Antrieb oder auf der Steuerung gerechnet zu werden. Diese Entscheidung wird beim Hochlauf der Steuerung pro Achse, für alle Bausteine dieser Achse gemeinsam, in Abhängigkeit des Antriebstyps (virtuelle Achse, generische Achse, IndraDrive, EcoDrive CS) entschieden.



Ein neu einzuführender Parameter kann diese Entscheidung noch einmal drehen, sie gilt dann jedoch, wie bisher, für alle Funktionsbausteine einer Achse gleichermaßen.

Die Funktionsbausteine "MB_MotionProfile" und "ML_FlexProfile" werden nur auf der Steuerung gerechnet.

Sie können nur dann programmiert werden, wenn alle Funktionsbausteine dieser Achse auf der Steuerung gerechnet werden.

Bei Nichtbeachtung wird eine Fehlermeldung erzeugt.



Der Funktionsbaustein "MC_CamIn" wird nur im intelligenten Antrieb (z.B. IndraDrive) unterstützt, er kann nur dann programmiert werden, wenn alle Funktionsbausteine dieser Achse auf dem intelligenten Antrieb (z.B. IndraDrive) gerechnet werden.

Der Baustein "MC_CamIn" wird nicht auf der Steuerung gerechnet.

sercos III, C2C-Verbund

Im Unterschied zu sercos 2, MLC-Verbund, bei dem in der lokalen Steuerung eine Verbundachse zu definieren war, die stellvertretend als Leitachse (Master) dienen konnte, ist im sercos III, C2C-Verbund, ein direkter Leitachs-Zugriff möglich.

Dazu sind zunächst in einer Globalen Variablenliste die für den Verbund relevanten Achsen zu vereinbaren:

Adressvereinbarung C2C-Verbund

```
VAR GLOBAL CONSTANT
M_RA11: AXIS_REF := (CntrlNo:=CNTRL_1, AxisNo:=AXIS_1);
M_RA13: AXIS_REF := (CntrlNo:=CNTRL_1, AxisNo:=AXIS_3);
S_RA21: AXIS_REF := (CntrlNo:=CNTRL_2, AxisNo:=AXIS_1);
S_RA22: AXIS_REF := (CntrlNo:=CNTRL_2, AxisNo:=AXIS_2);
END_VAR
```

Wird nun in Steuerung CNTRL_2 für Achse RA21 eine Leitachse RA11 von Steuerung CNTRL_1 benötigt, so ergibt sich unter Nutzung der automatisch erzeugten Globalen Variablenliste von Steuerung CNTRL_2 die nachfolgende Funktionsbausteinbeschaltung.

Auszug aus MlcVarGlobal von Steuerung CNTRL_2

```
VAR_GLOBAL CONSTANT
(*real axis "RA22" with axis number 2*)
RA22: AXIS_REF := (CntrlNo:=LOCAL_CNTRL, AxisNo:=AXIS_2);
(*real axis "RA21" with axis number 1*)
RA21: AXIS_REF := (CntrlNo:=LOCAL_CNTRL, AxisNo:=AXIS_1);
(* .... *)
END_VAR
```


ML_PLCopen.library

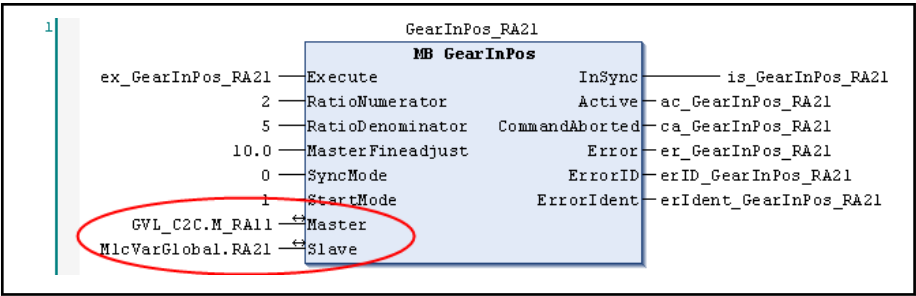


Abb. 6-30: Baustein mit Leitachse aus entfernter Steuerung

6.6.2 MB_Phasing

Kurzbeschreibung Mit dem Funktionsbaustein MB_Phasing wird ein Offset zur Leitachse, die als Eingang an einem Synchronisations-Baustein (MC_CamIn, MB_GearInPos, MC_GearIn, MB_MotionProfile oder ML_FlexProfile) anliegt, verändert.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-23: Bibliothekszuordnung

Schnittstellenbeschreibung

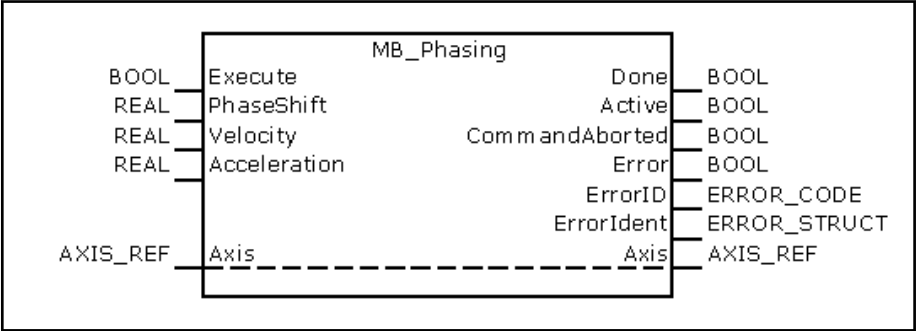


Abb. 6-31: FB MB_Phasing

	Name	Typ	Kommentar ⁷⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur Folgeachse
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
	PhaseShift	REAL	Positionsdiffrenz für die Bewegung (+/-)
	Velocity	REAL	Maximalwert der Geschwindigkeit in Einheiten der Leitachse (stets +) (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung in Einheiten der Leitachse (stets +).
VAR_OUTPUT	Done	BOOL	Der Positionsversatz wurde erfolgreich verfahren
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-24: Schnittstelle des FB MB_Phasing

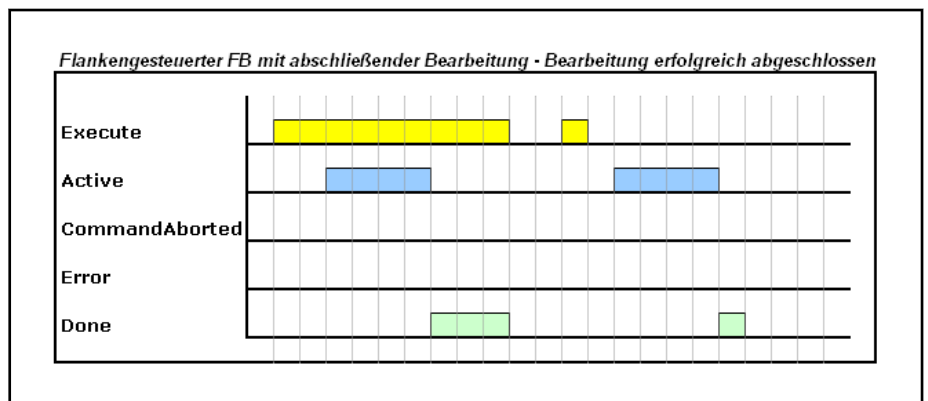
Signal-Zeit-Diagramm

Abb. 6-32: Signal-Zeit-Diagramm

Funktionsbeschreibung

Die Eingänge PhaseShift, Velocity und Acceleration bestimmen das Bewegungsprofil, mit dem der Weg verfahren wird.

Die Verfahrbewegung findet nur statt, wenn eine Synchronbetriebsart aktiv ist und ist dann bezogen auf die gültige Leitachse.

Ist keine Synchronbetriebsart aktiv, so wird der Wert "PhaseShift" sofort übernommen und beim Übergang in eine Synchronbetriebsart berücksichtigt. Da dann keine Leitachse gültig ist, wird auf die im A-0-0200, Default-Leitachse, angegebene Leitachse Bezug genommen. Ist auch dort keine gültige Leitachse eingetragen, so wird ein Fehler generiert.

Die Einheiten der Eingänge, welche die Achsbewegung definieren, sind den Wichtungparametern der Leitachse zu entnehmen.

Der Funktionsbaustein arbeitet parallel zum jeweiligen Haupt-Funktionsbaustein (MC_GearIn, MB_GearInPos, MC_CamIn, MB_MotionProfile, ML_FlexProfile).

Es werden dazu die nachstehenden Steuerungsparameter verwendet:

⁷⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung der Leitachse abhängig.

ML_PLCopen.library

- A-0-2710, ELS, Phasing, Positionssollwert additiv,
- A-0-2711, ELS, Phasing, Positionssollwert additiv, Geschwindigkeit, und
- A-0-2712, ELS, Phasing, Positionssollwert additiv, Beschleunigung.

Die Bearbeitung erfolgt

- für virtuelle und generische Achsen und wenn für IndraDrive gewünscht, als Funktion in der Steuerung.
- für IndraDrive im Antrieb nach Übergabe der A-Parameter an:
 - P-0-0054, Leitachspannung additiv,
 - P-0-0688, Leitachspannung additiv Positioniergeschwindigkeit, und
 - P-0-0689, Leitachspannung additiv Positionierbeschleunigung.

Bei generischen / virtuellen Achsen wird der Phasenoffset sofort aufgeschaltet, sofern keine synchrone Betriebsart aktiv ist.

Den Zustand des Done-Ausgangs wird auch auf das Echtzeitbit "Axis_InMasterPhaseOffset" der Achsdatenstruktur abgebildet.

Der Baustein verfährt immer den kürzesten Weg.

Der Wert "PhaseShift" wird nur absolut verfahren, d.h. startet man den Baustein zunächst mit dem Wert 50 und dannach erneut mit dem Wert 70, so beträgt der Versatz nach dem zweiten Starten 70 und nicht $50 + 70 = 120$.

Es wird nur ein Versatz verfahren, der kleiner als der Modulowert der jeweiligen Folgeachse ist. Ist der Wert dennoch größer, so wird zunächst eine Modulooperation durchgeführt und anschließend die Bewegung gestartet.



Für Master-Achsen, welche über den sercos III-Steuerungsverbund ausgetauscht werden, wird immer rotatorische Wichtung mit 360° Modulowert zur Berechnung verwendet (die Wichtungsinformationen werden nicht über den Steuerungsverbund übertragen).

Fehlerbehandlung Der Funktionsbaustein MB_Phasing erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.3 MB_PhasingSlave

Kurzbeschreibung Der Funktionsbaustein MB_PhasingSlave realisiert eine Verstellung der Folgeachse mit vorgegebener Geschwindigkeit und Beschleunigung.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-25: Bibliothekszuordnung

Schnittstelle

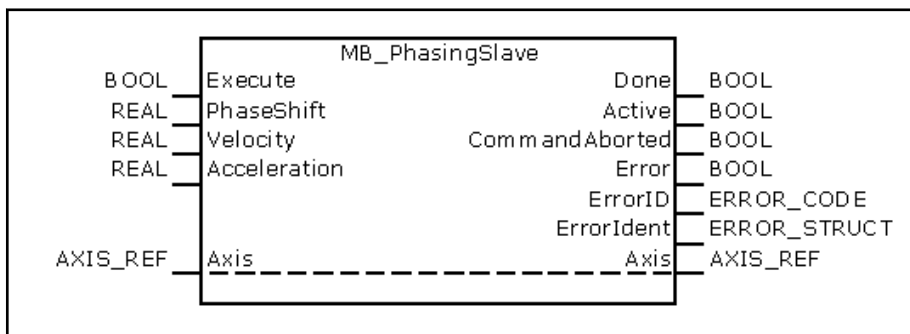


Abb. 6-33: FB MB_PhasingSlave

	Name	Typ	Kommentar ⁸⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	PhaseShift	REAL	Positionsdivergenz für die Bewegung (+/-).
	Velocity	REAL	Maximalwert der Geschwindigkeit (stets +) (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).
VAR_OUTPUT	Done	BOOL	gewünschte Differenz erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-26: Schnittstelle des FB MB_PhasingSlave

Signal-Zeit-Diagramm

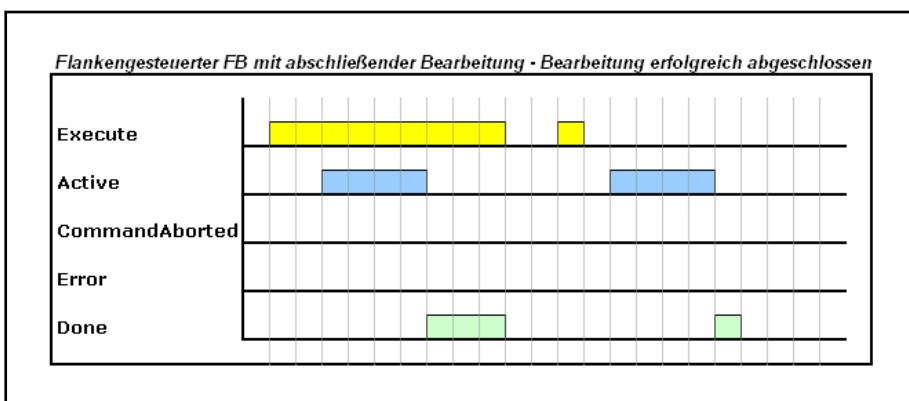


Abb. 6-34: Signal-Zeit-Diagramm

Funktionsbeschreibung

Die Eingänge PhaseShift, Velocity und Acceleration bestimmen das Bewegungsprofil, mit dem der Weg verfahren wird.

Die Verfahrbewegung findet nur statt, wenn eine Synchronbetriebsart aktiv ist.

Ist keine Synchronbetriebsart aktiv, so wird der Wert "PhaseShift" sofort übernommen und beim Übergang in eine Synchronbetriebsart berücksichtigt.

Die Einheiten der Eingänge, welche die Antriebsbewegung definieren, sind den Wichtungsparemtern der Folgeachse zu entnehmen.

Der Funktionsbaustein arbeitet parallel zum jeweiligen Haupt-Funktionsbaustein (MC_CamIn, MB_GearInPos, MC_GearIn, MB_MotionProfile oder ML_FlexProfile).



Der Baustein ist nicht für MPx3 / MC_CamIn geeignet, da diese die benötigten P-0-0686, P-0-0687 nicht besitzen.

Es werden dazu die nachstehenden Steuerungsparameter verwendet:

⁸⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung der Folgeachse abhängig.

ML_PLCopen.library

- A-0-2750, ELS, PhasingSlave, Positionssollwert additiv,
- A-0-2751, ELS, PhasingSlave, Positionssollwert additiv, Geschwindigkeit, und
- A-0-2752, ELS, PhasingSlave, Positionssollwert additiv, Beschleunigung.

Die Bearbeitung erfolgt

- für virtuelle Achsen und SercosDrives als Funktion in der Steuerung.
- für IndraDrive im Antrieb nach Übergabe der A-Parameter an:
 - S-0-0048, Lagesollwert additiv,
 - P-0-0686, Lagesollwert additiv - Positioniergeschwindigkeit, und
 - P-0-0687, Lagesollwert additiv - Positionierbeschleunigung.

Befindet sich die Folgeachse in einer Synchronbetriebsart und dieser Baustein wird gestartet, so nimmt das "InSync"-Bit der synchronen Bausteine (MB_GearInPos, MC_CamIn, MB_MotionProfile) solange den Wert "FALSE" an, bis der Baustein MB_PhasingSlave den Done-Ausgang setzt.

Bei Interpolation im Antrieb wird das "InSync"-Bit immer für einen Takt auf FALSE gesetzt, auch wenn keine Positionsdifferenz verfahren werden soll.

Der Baustein verfährt immer den kürzesten Weg.

Der Wert "PhaseShift" wird nur absolut verfahren, d.h. startet man den Baustein zunächst mit dem Wert 50 und dannach erneut mit dem Wert 70, so beträgt der Versatz nach dem zweiten Starten 70 und nicht $50 + 70 = 120$.

Es wird nur ein Versatz verfahren, der kleiner als der Modulowert der jeweiligen Folgeachse ist. Ist der Wert dennoch größer, so wird zunächst eine Modulooperation durchgeführt und anschließend die Bewegung gestartet.

Fehlerbehandlung

Der Funktionsbaustein MB_PhasingSlave erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, ERROR_TABLE 16#0030](#) , [Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.4 MC_CamIn**Kurzbeschreibung**

Der Funktionsbaustein MC_CamIn realisiert eine elektronische Kurvenscheibenfunktion.



Kurventabellen-Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation im Antrieb

Der Baustein "MC_CamIn" kann nicht auf der Steuerung gerechnet werden.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-27: Bibliothekszuordnung

Schnittstellenbeschreibung

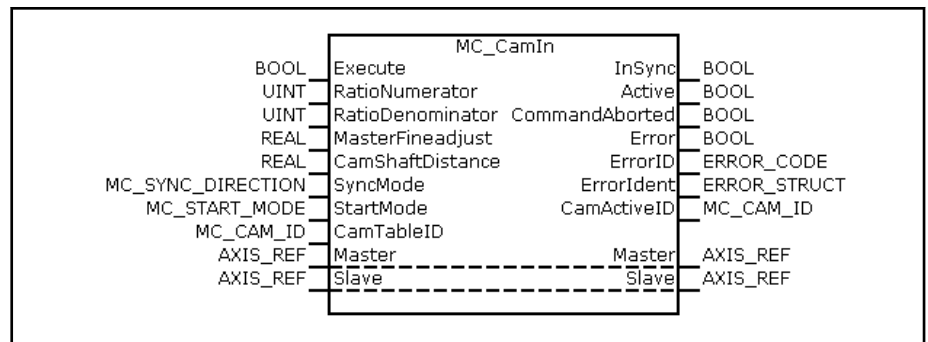


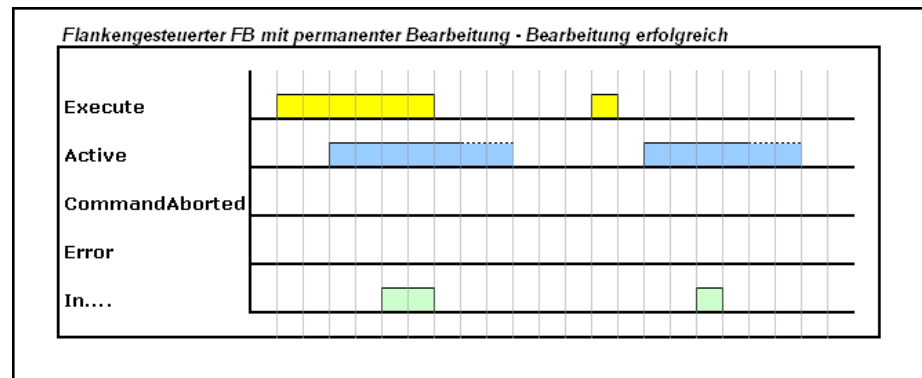
Abb. 6-35: FB MC_CamIn

	Name	Typ	Kommentar
VAR_IN_OUT	Master	AXIS_REF	enthält Informationen zur Leitachse
	Slave	AXIS_REF	enthält Informationen zur Folgeachse, die der Leitachse folgt
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
	RatioNumerator	UINT	Getriebeübersetzungsverhältnis - Zähler "Leitachsgetriebe Ausgangsumdrehungen" min: 1; max: 65535; default: 1
	RatioDenominator	UINT	Getriebeübersetzungsverhältnis - Nenner "Leitachsgetriebe Eingangsumdrehungen" min: 1; max: 65535; default: 1
	MasterFineadjust	REAL	"Feinabgleich Getriebeübersetzung" in % ; default: 0.0, min.: -799,9999%, max.: 799,9999
	CamShaftDistance	REAL	"Kurvenscheibenhub"
	SyncMode	MC_SYNC_DIRECTION	Aufsynchronisier-Richtung: 0:= kürzeste Entfernung, 1:= positive Richtung, 2:= negative Richtung
	StartMode	MC_START_MODE	Art des Aufsynchronisierens: 0:= absolut, 1:= relative
	CamTableID	MC_CAM_ID	Nummer der zu aktivierenden Kurvenscheibe 1..4(...8)
VAR_OUTPUT	InSync	BOOL	Folgeachse folgt Leitachse; Kopplung erfolgreich durchgeführt
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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
	CamActiveID	MC_CAM_ID	Nummer der im Baustein aktiven Kurvenscheibe

Tab. 6-28: Schnittstelle des FB MC_CamIn

ML_PLCopen.library

Signal-Zeit-Diagramm



in... InSync

Abb. 6-36: Signal-Zeit-Diagramm

6.6.5 MC_CamIn, Besonderheiten bei Sercos basierten Achsen

Funktionsbeschreibung Der Funktionsbaustein MC_CamIn realisiert zwei Aufgaben:

- Festlegen, wie die Achse die auf ihr gespeicherten Kurvenscheibendaten zu interpretieren bzw. abzuarbeiten hat,
- Erteilen des Auftrages, die Achse in den CAM-Bearbeitungsmodus zu schalten und einen Bewegungszyklus (definiert durch den Sercos-Zyklus) zu bearbeiten. Nach erfolgreicher Abarbeitung des Bausteines verbleibt die Achse im CAM-Bearbeitungsmodus bis sie ein anderes Kommando erhält oder ein Fehler eintritt.

Ziel ist es, eine in der Kurvenscheibentabelle niedergelegte Abhängigkeit zwischen der Leitachse (A-0-2700, ELS, Leitachse, Steuerungsnummer, A-0-2701, ELS, Leitachse, Achsnummer) und der ausgewählten Folgeachse zu realisieren.



Kurventabellen-Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation im Antrieb

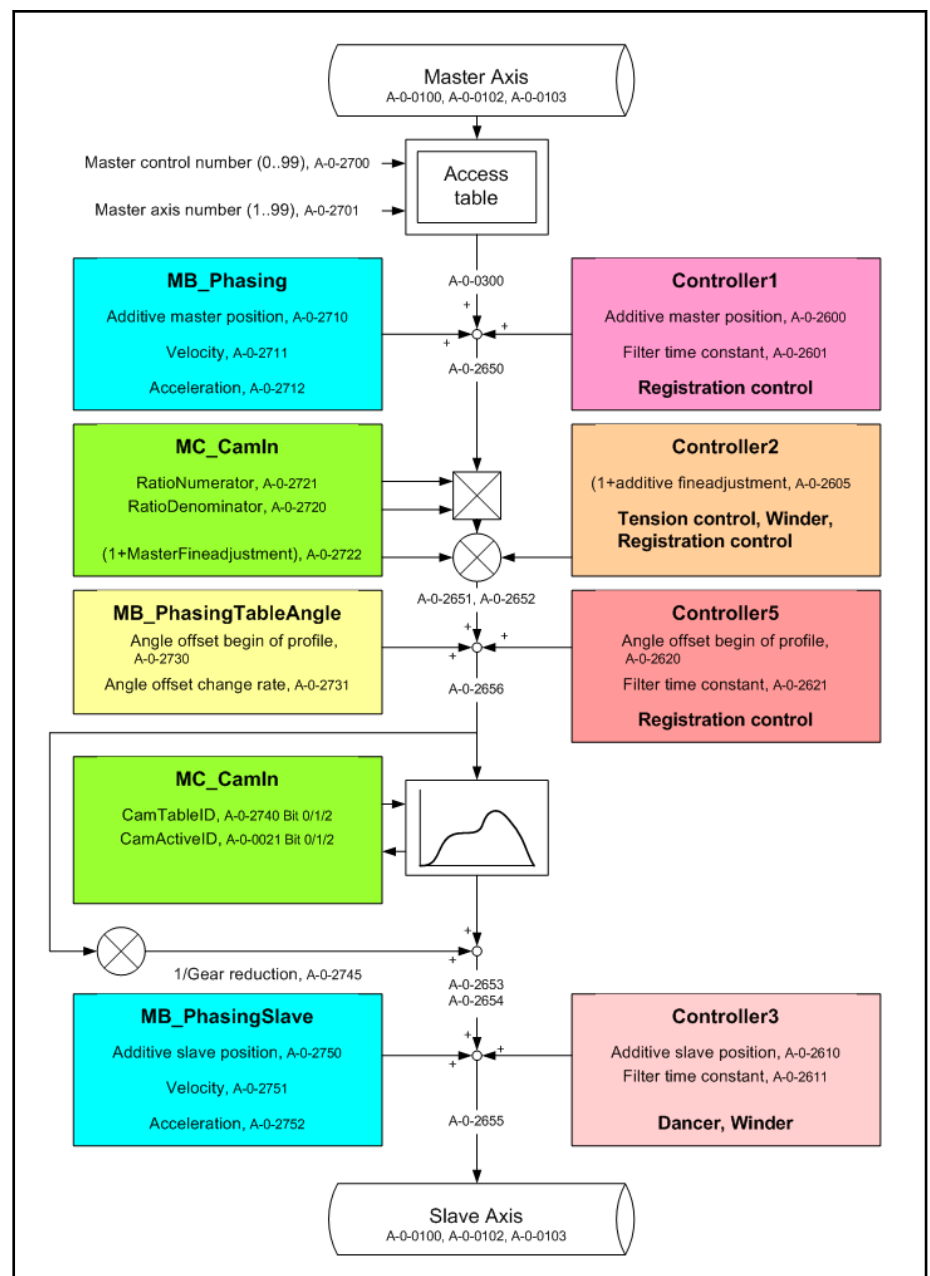


Abb. 6-37: MotionControl-Wirkungskette MC_CamIn

Der Baustein enthält folgende funktionelle Gruppen:

- das elektronische Getriebe, welches definiert ist durch das Verhältnis von A-0-2721, ELS, Leitachsgetriebe, Ausgangsumdrehungen, und A-0-2720, ELS, Leitachsgetriebe, Eingangsumdrehungen,
- den A-0-2722, ELS, Feinabgleich der Getriebeübersetzung. Der Wert des Feinabgleichs wird als %-Zahl vorgeben und als $(1 + \text{Feinabgleich})$ gerechnet,
- die Auswahl einer der möglichen Kurvenscheiben-Tabellen, die vorher im Antrieb abzulegen sind: A-0-2740, ELS, CamIn, Kurventabellenvorwahl
- die Wichtung folgeachsseitig: A-0-2741, ELS, CamIn, Kurvenscheibenhub

ML_PLCopen.library

Der Baustein ist in der Lage, die Synchronisation, ausgehend vom ankommenden Lage- und Geschwindigkeitssignal, vor dem Einrasten von Leit- und Folgeachse, herbeizuführen.

- Fehlerbehandlung
- Der Funktionsbaustein MC_CamIn erzeugt folgende Fehlermeldungen:
- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.6 MC_CamOut

- Kurzbeschreibung
- Der Funktionsbaustein MC_CamOut beendet die vom Funktionsbaustein MC_CamIn aktivierte elektronische Kurvenscheibenfunktion.
- Der Baustein realisiert zwei Aufgaben:
- Beenden des Kurvenscheibenbetriebs der Folgeachse,
 - Überführen der Folgeachse in die Betriebsart "Continuous Motion" unter Beibehaltung des letzten Geschwindigkeitswertes, für den Fall, dass kein anderes Kommando ansteht.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-29: Bibliothekszuordnung

Schnittstellenbeschreibung

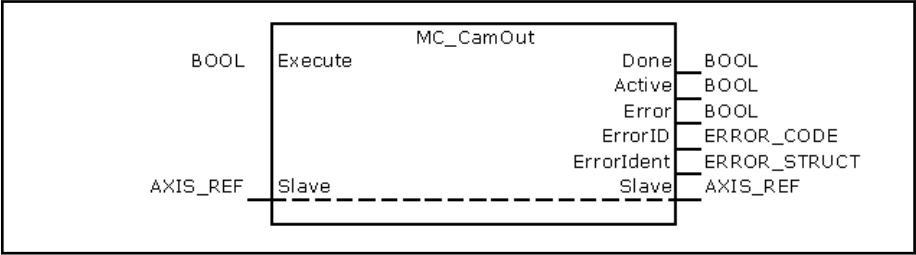


Abb. 6-38: FB MC_CamOut

	Name	Typ	Kommentar
VAR_IN_OUT	Slave	AXIS_REF	enthält Informationen zur Folgeachse, die der Leitachse folgt
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigenden Flanke
VAR_OUTPUT	Done	BOOL	Entkopplung erfolgreich durchgeführt
	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

Tab. 6-30: Schnittstelle des FB MC_CamOut

Signal-Zeit-Diagramm

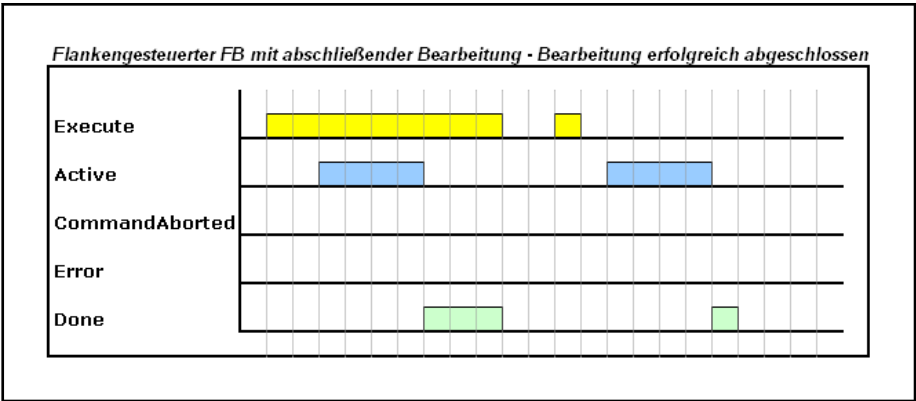


Abb. 6-39: Signal-Zeitdiagramm

6.6.7 MC_CamOut, Besonderheiten bei SERCOS-basierten Achsen

Fehlerbehandlung Der Funktionsbaustein MC_CamOut erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.8 MB_MotionProfile

Kurzbeschreibung Der Funktionsbaustein MB_MotionProfile realisiert eine elektronische Kurvenfunktion. Die Kurvenfunktionen sind dabei analytisch definiert durch die Parameter, die zu der Betriebsart "MotionProfile" gehören.

-  Die Anwendung des Bausteins setzt die vorherige Leistungszuschaltung über eine Instanz des [MC_Power, Seite 176](#), (bei realen Achsen) voraus.
-  Der Baustein wird durch den Baustein [MC_GearOut, Seite 246](#), deaktiviert.
-  *MotionProfile-Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:*
 - IndraDrive mit Interpolation in der Steuerung
 - IndraDrive mit Interpolation in der Steuerung
 - SercosDrive
 - Virtuelle Achse
 - Reglerachse

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-31: Bibliothekszuordnung

ML_PLCOpen.library

Schnittstellenbeschreibung

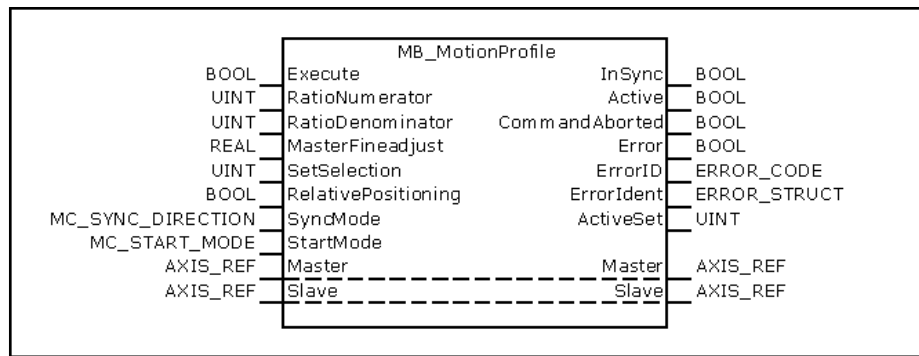


Abb. 6-40: FB MB_MotionProfile

	Name	Typ	Kommentar
VAR_IN_OUT	Master	AXIS_REF	Auswahl der Leitachse
	Slave	AXIS_REF	Auswahl der Folgeachse, die der Leitachse folgt
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
	RatioNumerator	UINT	Getriebeübersetzungsverhältnis - Zähler "Leitachsgetriebe Ausgangsumdrehungen" min: 1; max: 65535; default: 1
	RatioDenominator	UINT	Getriebeübersetzungsverhältnis - Nenner "Leitachsgetriebe Eingangsumdrehungen" min: 1; max: 65535; default: 1
	MasterFineadjust	REAL	"Feinabgleich Getriebeübersetzung" in % ; default: 0.0, min.: -799,9999%, max.: 799,9999
	SetSelection	UINT	Auswahl des MotionProfiles (0 oder 1)
	RelativePositioning	BOOL	RelativePositioning legt fest, ob der Offset-Winkel aus dem Parameter A-0-2901 aktiv sein sollte. 0: Offset-Winkel nicht aktiv (Default) 1: Offset-Winkel aktiv
	SyncMode	MC_SYNC_DIRECTION	Aufsynchronisier-Richtung: 0:= kürzeste Entfernung, 1:= positive Richtung, 2:= negative Richtung
	StartMode	MC_START_MODE	Art des Aufsynchronisierens: 0:= absolut Synchronisation, ohne Aufsynchronisieren bei erneutem Bausteinanruf 1:= relative Synchronisation, ohne Aufsynchronisieren bei erneutem Bausteinanruf 2:= absolut Synchronisation, mit Aufsynchronisieren bei erneutem Bausteinanruf 3:= relative Synchronisation, mit Aufsynchronisieren bei erneutem Bausteinanruf
VAR_OUTPUT	InSync	BOOL	Folgeachse ist synchron
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando

	Name	Typ	Kommentar
	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
	ActiveSet	UINT	Nummer des aktiven MotionProfile

Tab. 6-32: Schnittstelle des FB MB_MotionProfile

Funktionsbeschreibung

Beim MotionProfile mit realer/virtueller Leitachse besteht eine feste Beziehung zwischen Leitachsposition (master position) und Folgeachse (slave position). Die Leitachsposition wird von einer realen oder virtuellen Leitachse vorgegeben.

- 2 Bewegungsabläufe pro Achse mit bis zu 16 Bewegungsschritten sind parametrierbar.
- Ein Bewegungsschritt wird durch eine Kurventabelle oder durch ein normiertes Profil (z.B. Polynom 5. Ordnung) festgelegt.
- Norm-Profile Rast in Rast, Rast in Geschwindigkeit, Geschwindigkeit in Rast, Geschwindigkeit in Geschwindigkeit und konstante Geschwindigkeit sind wählbar.
- Die Realisierung der Norm-Profile erfolgt durch ein Polynom bis 5. Ordnung oder bei Rast in Rast alternativ durch eine geneigte Sinuslinie.
- Werden Kurventabellen benötigt, sind Kurventabellen mit 3 bis maximal 1024 Stützpunkten wählbar.

Die Kurventabellen sind auf der Steuerung unter den C-Parametern C-0-2001..2099 bevorratet. Der Baustein [MB_ChangeCamData](#), [Seite 225](#), erlaubt es, diese Parameter mit Daten aus der SPS zu füllen.

- Zwischen den Kurventabellen-Stützpunkten kann kubische Spline-Interpolation oder Linear-Interpolation gewählt werden.
- Jeder Bewegungsschritt hat einen eigenen Hub.
- Absolute oder relative Verarbeitung der Bewegungsschritte sind wählbar.
- Ein elektronisches Getriebe mit Feinabgleich-Getriebeübersetzung steht zur Verfügung.
- Zusätzliche Prozessreglerparameter Lagesollwert additiv, Leitachsposition additiv (mit Filter 1. Ordnung), Winkelverschiebung Tabellenanfang (mit Filter 1. Ordnung) und Feinabgleich-Getriebeübersetzung können innerhalb der MotionControl-Wirkungskette gewählt werden..
- Der Aufsynchronisier-Modus ist wählbar.
- Die Leitachspolarität ist einstellbar.

Das folgende Bild zeigt einen Bewegungsablauf, der aus 5 Schritten besteht.

ML_PLCopen.library

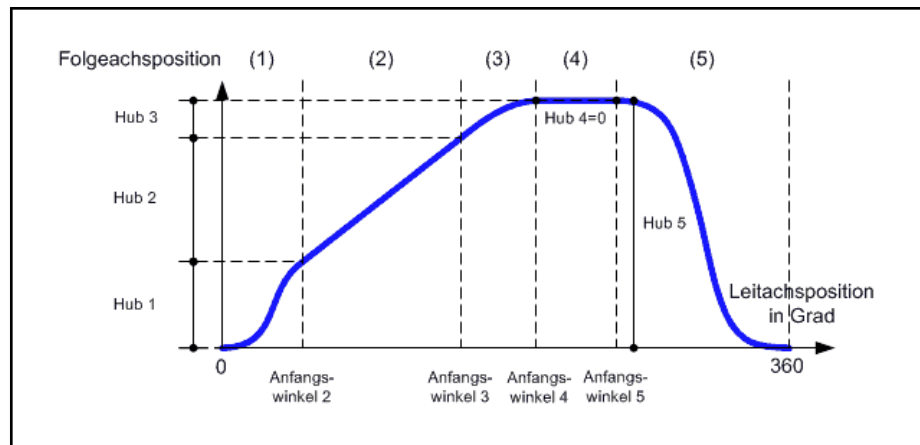


Abb. 6-41: Beispiel eines Bewegungsprofils

Benutzte Bewegungsgesetze

- Rast in Geschwindigkeit: Bewegungsschritt ①,
- Geschwindigkeit in Geschwindigkeit: Bewegungsschritt ②,
- Geschwindigkeit in Rast: Bewegungsschritt ③,
- Geschwindigkeit in Geschwindigkeit: Bewegungsschritt ④,
- Rast in Rast: Bewegungsschritt ⑤.

6.6.9 MB_MotionProfile, Besonderheiten bei Sercos basierten Achsen

Funktionsbeschreibung



MotionProfile-Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation in der Steuerung
- IndraDrive mit Interpolation in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

Der Funktionsbaustein MB_MotionProfile realisiert zwei Aufgaben:

- Festlegen, wie das auf der Steuerung gespeicherte MotionProfile zu interpretieren bzw. abzuarbeiten ist,
- Erteilen des Auftrages, das MotionProfile in der Bearbeitung zu starten und zu bearbeiten. Nach erfolgreicher Abarbeitung des Bausteines verbleibt die Achse im MotionProfile-Bearbeitungsmodus bis sie ein anderes Kommando erhält oder ein Fehler eintritt.

Ziel ist es, eine im MotionProfile niedergelegte Abhängigkeit zwischen der Leitachse ("A-0-2700, ELS, Leitachse, Steuerungsnummer", "A-0-2701, ELS, Leitachse, Achsnummer") und der ausgewählten Folgeachse zu realisieren.

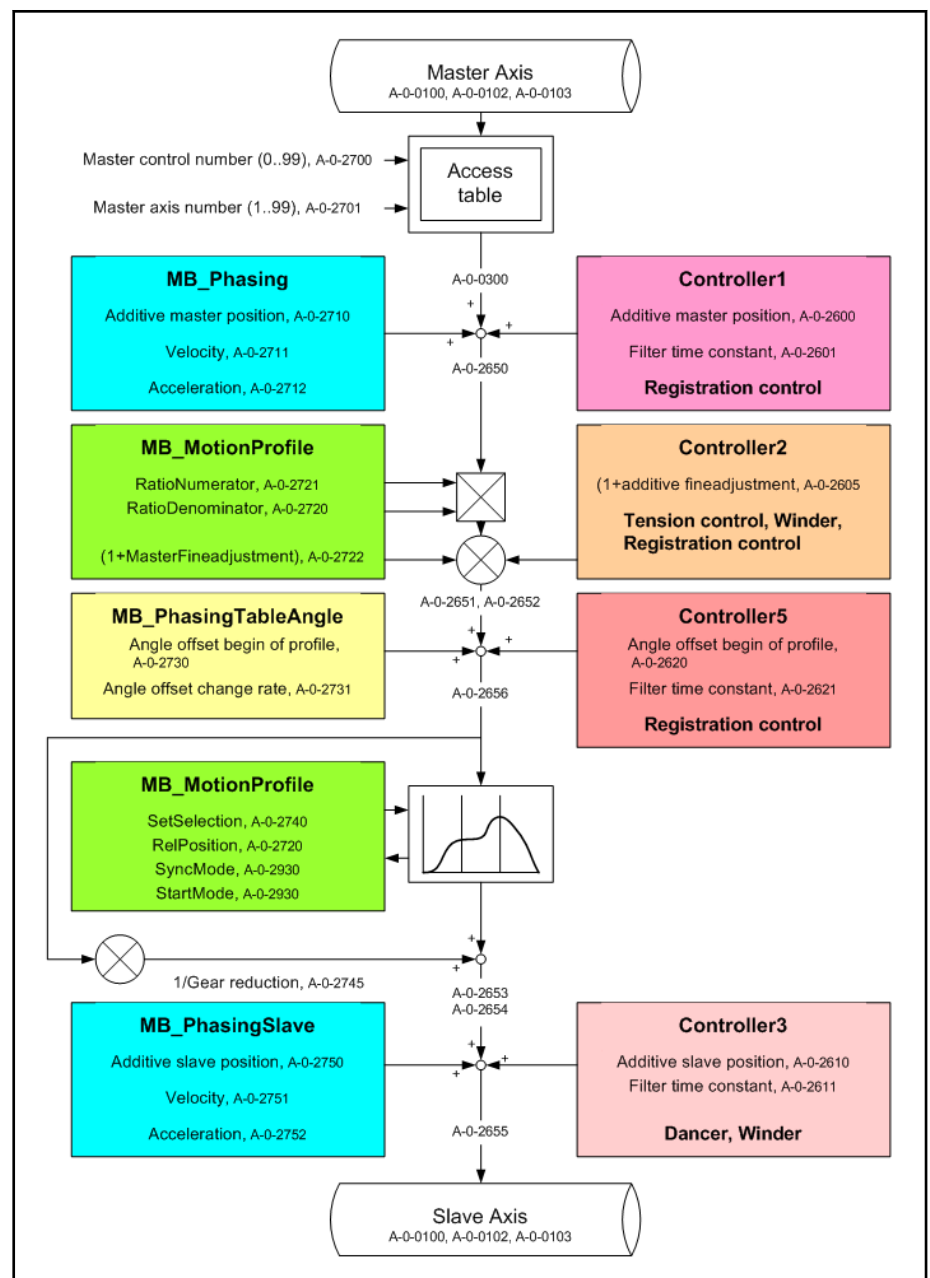


Abb. 6-42: MotionControl-Wirkungskette MB_MotionProfile

Der Baustein enthält folgende funktionelle Gruppen:

- das elektronische Getriebe, welches definiert ist durch das Verhältnis von "A-0-2721, ELS, Leitachsgetriebe, Ausgangsumdrehungen" und "A-0-2720, ELS, Leitachsgetriebe, Eingangsumdrehungen",
- den "A-0-2722, ELS, Feinabgleich der Getriebeübersetzung". Der Wert des Feinabgleichs wird als %-Zahl vorgeben und als $(1 + \text{Feinabgleich})$ gerechnet.
- die Auswahl eines der beiden möglichen MotionProfiles, die vorher in der MLC abzulegen sind:

Gemeinsam für beide Bewegungsprofile (0 und 1) genutzte Parameter zur Konfiguration:

- A-0-2900, MotionProfile, Leitachs-Umschaltposition

ML_PLCopen.library

- A-0-2901, MotionProfile, Bewegungsschritt 1, Folgeachsanfangsposition
- A-0-2902, MotionProfile 0, Diagnose
- A-0-2903, MotionProfile 1, Diagnose
- A-0-2904, MotionProfile, Basis-Konfiguration

Achs-Parameter Bewegungsprofil 0:

- A-0-2910, MotionProfile 0, Anzahl der Bewegungsschritte
- A-0-2911, MotionProfile 0, Leitachsgeschwindigkeit
- A-0-2912, MotionProfile 0, Liste der Leitachsanfangspositionen
- A-0-2913, MotionProfile 0, Liste der Bewegungsgesetze
- A-0-2914, MotionProfile 0, Liste der Hübe
- A-0-2915, MotionProfile 0, Liste der Folgeachsgeschwindigkeiten

Achs-Parameter Bewegungsprofil 1:

- A-0-2920, MotionProfile 1, Anzahl der Bewegungsschritte
- A-0-2921, MotionProfile 1, Leitachsgeschwindigkeit
- A-0-2922, MotionProfile 1, Liste der Leitachsanfangspositionen
- A-0-2923, MotionProfile 1, Liste der Bewegungsgesetze
- A-0-2924, MotionProfile 1, Liste der Hübe
- A-0-2925, MotionProfile 1, Liste der Folgeachsgeschwindigkeiten

Gemeinsam für beide Bewegungsprofile (0 und 1) genutzte Parameter im laufenden Betrieb:

- A-0-2930, MotionProfile, Steuerwort
- A-0-2931, MotionProfile, Statuswort

Der Baustein ist in der Lage, die Synchronisation, ausgehend vom ankommenden Lage- und Geschwindigkeitssignal, vor dem Einrasten von Leit- und Folgeachse, herbeizuführen.

- Fehlerbehandlung** Der Funktionsbaustein MB_MotionProfile erzeugt folgende Fehlermeldungen:
- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.10 PLCopen-Funktionsbausteine zur Unterstützung von MotionProfiles

Allgemeines

Eine MLC kann bis maximal 64 Achsen⁹⁾ verwalten.

Einer Achse können dabei je zwei MotionProfile zugeordnet werden, von denen eines aktiviert werden kann. Bei Bedarf kann auf das andere MotionProfile umgeschaltet werden.

Jedes MotionProfile kann maximal 16 Schritte aufnehmen.

Die einzelnen Bewegungsschritte könne entweder über bestimmte Bewegungsgesetze (VDI 2143, und spezielle Bewegungsgesetze) oder über die darauf referenzierte Kurventabellen definiert werden.

⁹⁾ Steuerungstypabhängig

Zur Aufnahme der Kurventabellen stellt die Steuerung in ihrer CompactFlash (CF) den Speicherplatz für maximal 99 Kurventabellen mit je 3 bis maximal 1024 Punkten zur Verfügung. Die Kurventabellen sind adressierbar über die Parameter C-0-2001 ... 2099.

Die Kurventabellen können entweder über den SPS-Funktionsbaustein [MB_ChangeCamData](#), [Seite 225](#), oder über die IndraWorks-Oberfläche konfiguriert werden.



MotionProfile-Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation in der Steuerung
- IndraDrive mit Interpolation in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

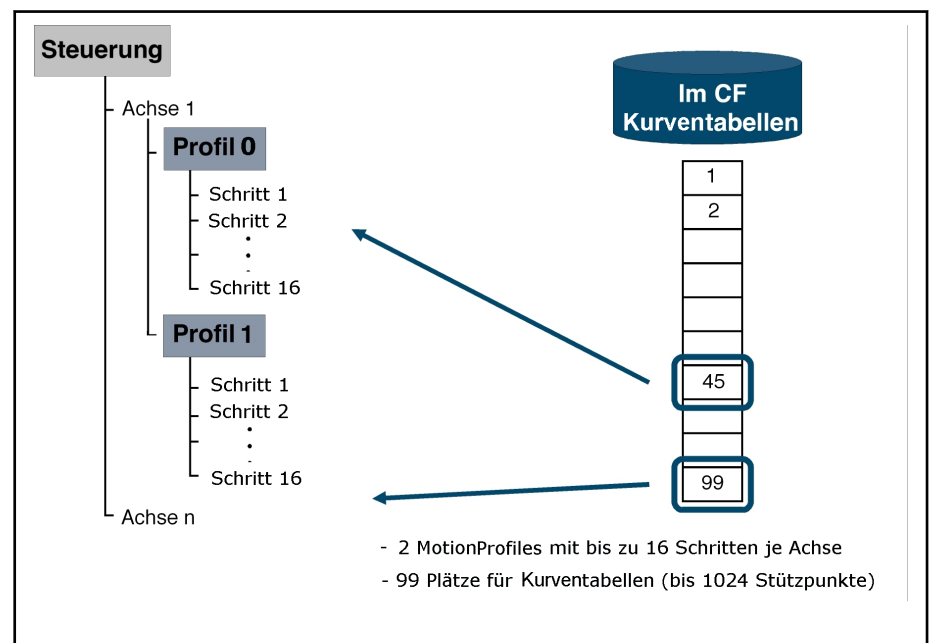


Abb. 6-43: Zuordnung Kurventabellen / MotionProfile

- MC_STEP_MODE, RIL_CommonTypes.library: Aufzählungsdatentyp, der für einen Schritt des Bewegungs- oder des Flexprofils den funktionellen Zusammenhang zwischen der Position der Folgeachse und der Leitachse beschreibt.
- [MB_ChangeProfileSet](#), [Seite 217](#): Schreiben eines kompletten MotionProfiles, von der SPS in die Motion
- [MB_ChangeProfileStep](#), [Seite 222](#): Ersetzen eines Schrittes des MotionProfiles,
- [MB_ChangeCamData](#), [Seite 225](#): Übertragen einer Kurventabelle aus dem SPS-Speicher in die Motion, C-0-2001...2099.

MB_ChangeProfileSet

Kurzbeschreibung

Der Funktionsbaustein MC_ChangeProfileSet gestattet es, ein komplettes MotionProfile mit maximal 16 Schritten (Steps) zu beschreiben bzw. ein bereits vorhandenes auszutauschen.

ML_PLCopen.library

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-33: Bibliothekszuordnung

Schnittstellenbeschreibung

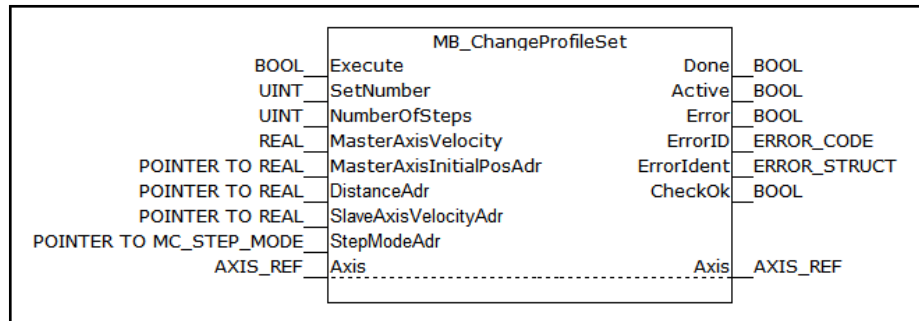


Abb. 6-44: FB MB_ChangeProfileSet

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur Achse, deren Bewegungsprofilsatz ausgetauscht werden soll.
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
	SetNumber	UINT	Auswahl des MotionProfiles (0 oder 1)
	Number of Steps	UINT	Anzahl der Schritte des MotionProfiles (1...16)
	MasterAxis-Velocity	REAL	Leitachsgeschwindigkeit für das Profil
	MasterAxis-InitialPosAdr	POINTER TO REAL	Zeiger auf eine Liste der Leitachspositionen am Anfang des jeweiligen Schrittes
	DistanceAdr	POINTER TO REAL	Zeiger auf eine Liste der Hübe am Ende des jeweiligen Schrittes
	SlaveAxis-VelocityAdr	POINTER TO REAL	Zeiger auf eine Liste der Folgeachsgeschwindigkeiten am Ende des jeweiligen Schrittes
	StepModeAdr	POINTER TO MC_STEP_MODE	Zeiger auf eine Liste der Bewegungsgesetze der Schritte
VAR_OUTPUT	Done	BOOL	Daten wurden übernommen.
	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
	CheckOk	BOOL	reserviert

Tab. 6-34: Schnittstelle des FB MB_ChangeProfileSet

Signal-Zeit-Diagramm

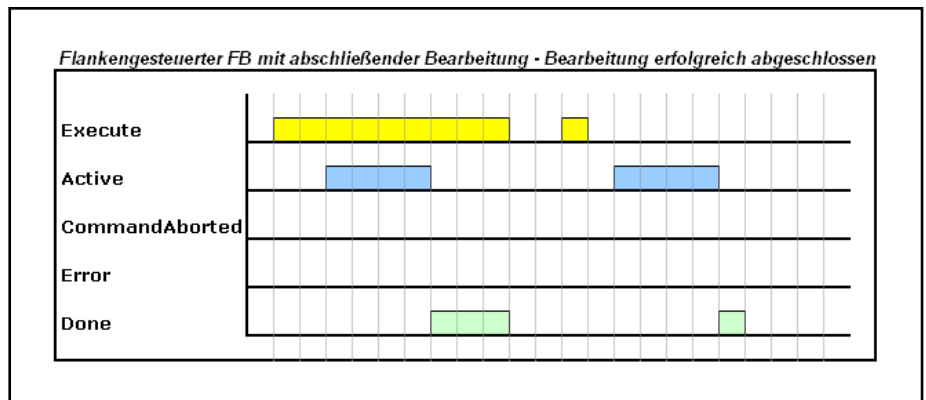


Abb. 6-45: Signal-Zeit-Diagramm: MB_ChangeProfileSet

Funktionsbeschreibung

Der Funktionsbaustein erlaubt es, ein komplettes MotionProfile mit 1...16 Schritten von der SPS an die Motionkomponente der Steuerung zu übergeben.

Die Daten, die die jeweiligen Schritte charakterisieren sind in Liste abgelegt, deren Anfang im SPS-Speicherraum jeweils durch einen Pointer (of REAL) zu beschreiben ist.

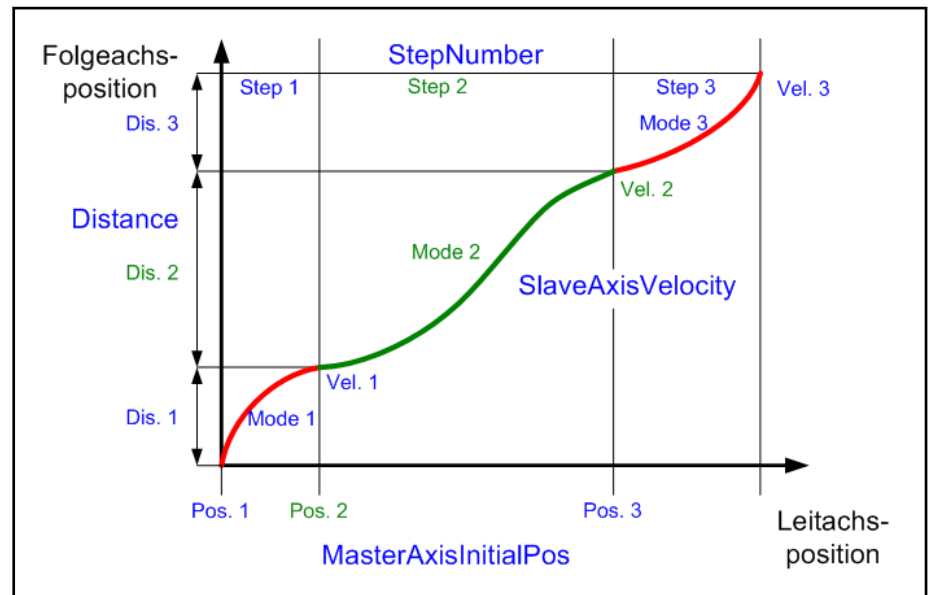


Abb. 6-46: Datensätze der einzelnen Schritte eines Bewegungsprofils

Die Modi der jeweiligen Schritte sind dem Aufzählungstyp "MC_STEP_MODE" aus der RIL_CommonTypes.library zu entnehmen:



Der Bewegungsmodus (siehe MC_STEP_MODE) ist ein Aufzählungstyp, der neben Funktionen wie

- REST_IN_REST_POLY5
- REST_IN_REST_INCLINEDSINE
- REST_IN_VELOCITY_POLY5
- VELOCITY_IN_VELOCITY_POLY5 und
- CONSTANT_VELOCITY

auch die Auswahl der CAM_TABLE_1 ... CAM_TABLE_99 zulässt.

ML_PLCopen.library

MB_ChangeProfileSet, Besonderheiten bei SERCOS-basierten Achsen

Für den jeweils geänderten Datensatz ist es möglich im Parameter "A-0-2930, MotionProfile, Steuerwort" (WORD) Voreinstellungen zu treffen.

Bit 11:

Dieses Bit legt fest, ob bei einer Kurventabellenabarbeitung zwischen den Stützpunkten eine Linear- oder Spline-Interpolation gerechnet wird. Während einer Bearbeitung sollte der Zustand nicht mehr geändert werden.

Bit 12:

Dieses Bit legt den Verarbeitungsmodus bei einer Parameteränderung fest.

- Bit 12 = 0: Parametersammeln ist nicht aktiv, jede Parameteränderung wird sofort getestet und, wenn das Profil fehlerfrei ist, übernommen.
- Bit 12 = 1: Parametersammeln ist aktiv, jede Parameteränderung wird gespeichert. Eine Übernahme der Änderung in die internen Arbeitsdaten, sowie die Testung der geänderten Parameter, erfolgt erst nach dem Rücksetzen von Bit 12.

Bit 13:

Dieses Bit legt fest, ob eine Kurvenscheibe während der Bearbeitung schreibgeschützt ist. Der Schreibschutz gilt jeweils im aktiven Schritt. Kurvenscheiben in nicht aktiven Schritten können geschrieben werden. Während einer Bearbeitung sollte der Zustand nicht mehr geändert werden.



Der Test läuft quasi automatisch, wenn Bit 12 = 0 voreingestellt ist.

Der Test läuft er nach dem Aktivwerden des Done, d.h. das Testergebnis muss dem Parameter "A-0-2931, MotionProfile, Statuswort" entnommen werden!

**Einstellung Betriebsart "Testen",
A-0-2930, Bit 12 = 0**

- Lesen des Parameters "A-0-2930, MotionProfile, Steuerwort" mittels MB_ReadParameter (Wert als DINT),
- DINT_TO_WORD,
- AND 16#EFFF, Rücksetzen Bit 12, Auslösen des Testvorganges, WORD_TO_DINT,
- Schreiben des Parameters "A-0-2930, MotionProfile, Steuerwort" mittels MB_WriteParameter (Wert als DINT).

**Lesen des Status zur Kontrolle,
dass das gesamte Profil getestet
wurde.**

- Lesen des Parameters "A-0-2931, MotionProfile, Statuswort" mittels MB_ReadParameter (Wert als DINT),
- DINT_TO_DWORD,
- **Bit 12/13:**

Bit 12 gibt an, ob das MotionProfile in Satz 0 überprüft wurde.

Dann ist aus Bit 13 das Ergebnis der Prüfung zu entnehmen. Nähere Informationen über ein negatives Prüfergebnis "F2229235, Fehler im MotionProfile" befinden sich in "A-0-2902, MotionProfile 0, Diagnose".

Bit 14/15:

Bit 14 gibt an, ob das MotionProfile in Satz 1 überprüft wurde.

Dann ist aus Bit 15 das Ergebnis der Prüfung zu entnehmen. Nähere Informationen über ein negatives Prüfergebnis "F2229235, Fehler im MotionProfile" befinden sich in "A-0-2903, MotionProfile 1, Diagnose".

Parameterzuordnung

Gemeinsam für beide MotionProfiles (0 und 1) genutzte Parameter zur Konfiguration:

- A-0-2900, MotionProfile, Leitachs-Umschaltposition
- A-0-2901, MotionProfile, Bewegungsschritt 1, Folgeachsanfangsposition
- A-0-2902, MotionProfile 0, Diagnose
- A-0-2903, MotionProfile 1, Diagnose
- A-0-2904, MotionProfile, Basis-Konfiguration

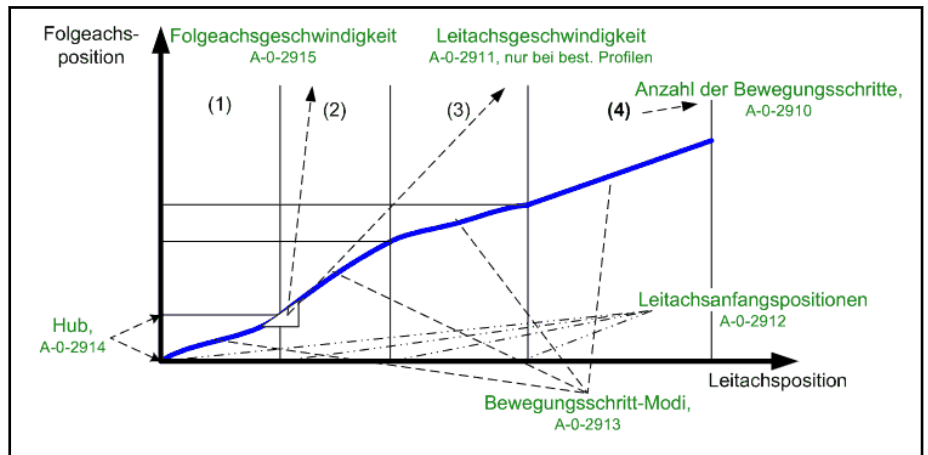


Abb. 6-47: Überblick über die Parameter des MotionProfiles 0
Achsen-Parameter, MotionProfile 0

- A-0-2910, MotionProfile 0, Anzahl der Bewegungsschritte
- A-0-2911, MotionProfile 0, Leitachsgeschwindigkeit
- A-0-2912, MotionProfile 0, Liste der Leitachsanfangspositionen
- A-0-2913, MotionProfile 0, Liste der Bewegungsgesetze
- A-0-2914, MotionProfile 0, Liste der Hübe
- A-0-2915, MotionProfile 0, Liste der Folgeachsgeschwindigkeiten

Achsen-Parameter, MotionProfile 1

- A-0-2920, MotionProfile 1, Anzahl der Bewegungsschritte
- A-0-2921, MotionProfile 1, Leitachsgeschwindigkeit
- A-0-2922, MotionProfile 1, Liste der Leitachsanfangspositionen
- A-0-2923, MotionProfile 1, Liste der Bewegungsgesetze
- A-0-2924, MotionProfile 1, Liste der Hübe
- A-0-2925, MotionProfile 1, Liste der Folgeachsgeschwindigkeiten

Gemeinsam für beide MotionProfiles (0 und 1) genutzte Steuer- und Statusworte:

- A-0-2930, MotionProfile, Steuerwort
- A-0-2931, MotionProfile, Statuswort

Fehlerbehandlung

Der Funktionsbaustein MB_ChangeProfileSet erzeugt folgende Arten von Fehlermeldung:

ML_PLCopen.library

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

MB_ChangeProfileStep

Kurzbeschreibung

Der Funktionsbaustein MB_ChangeProfileStep ändert die vier Parameter, die einen Schritt (Step) eines MotionProfile-Satzes (Set) definieren.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-35: Bibliothekszuordnung

Schnittstellenbeschreibung

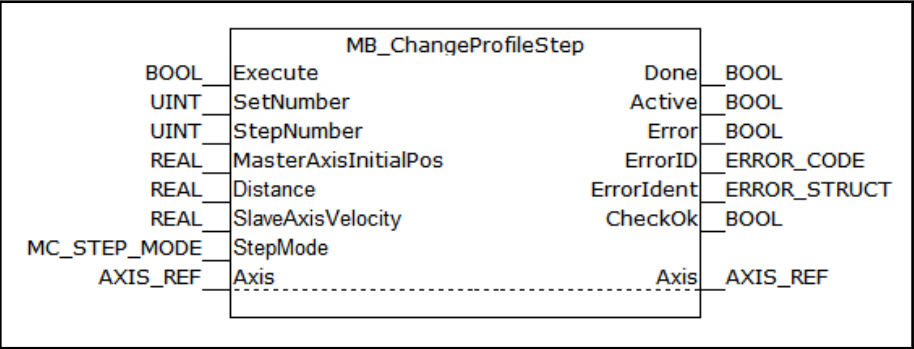


Abb. 6-48: FB MB_ChangeProfileStep

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur Achse, deren Bewegungsprofilsatz geändert werden soll.
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
	SetNumber	UINT	Auswahl des MotionProfiles (0 oder 1)
	StepNumber	UINT	Auswahl des zu ändernden Bewegungsschrittes (1...16)
	MasterAxis-InitialPos	REAL	Leitachsanfangsposition des Schrittes in Grad
	Distance	REAL	Folgeachsweg des Schrittes
	SlaveAxis-Velocity	REAL	Folgeachsgeschwindigkeit innerhalb des Schrittes
	StepMode	MC_STEP_MODE	Art des Schrittes
VAR_OUTPUT	Done	BOOL	Daten wurden übernommen.
	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
	CheckOk	BOOL	reserviert

Tab. 6-36: Schnittstelle des FB MB_ChangeProfileStep

Signal-Zeit-Diagramm

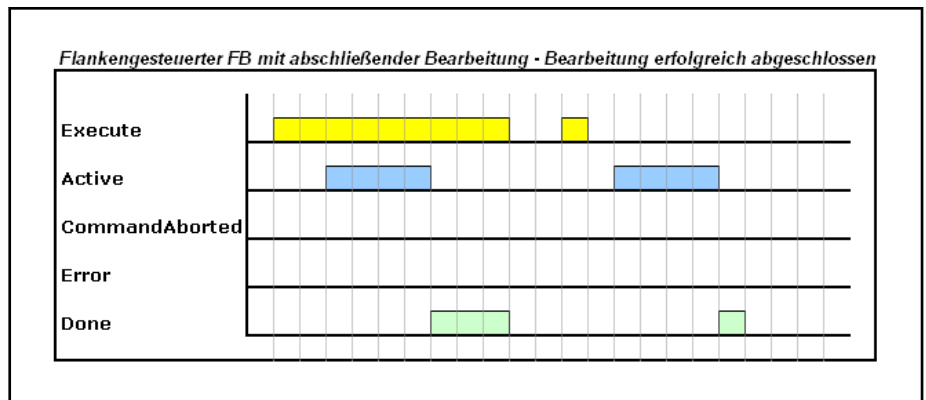


Abb. 6-49: Signal-Zeit-Diagramm: MB_ChangeProfileStep

Funktionsbeschreibung

Der Baustein dient zum Ersatz eines Schrittes durch einen anderen.

Die Eingangsgrößen des Bausteins sind damit die Nummer des MotionProfile (SetNumber), 0 oder 1, die Nummer des Schrittes (StepNumber), 1...16, also der Ort des Schrittes, der ersetzt werden soll, und die Daten des neuen Schrittes: Leitachs-Anfangsposition (MasterAxisInitialPosition), Hub (Distance) und Folgeachsgeschwindigkeit (SlaveAxisVelocity) am Ende des Schrittes, sowie der Bewegungsschritt-Modus (Funktion oder Kurventabelle) zur Auswahl.



Der Bewegungsmodus (siehe MC_STEP_MODE) ist ein Aufzählungstyp, der neben Funktionen wie

- REST_IN_REST_INCLINEDSINE
- REST_IN_REST_POLY5
- REST_IN_VELOCITY_POLY5
- VELOCITY_IN_VELOCITY_POLY5 und
- CONSTANT_VELOCITY

auch die Auswahl der CAM_TABLE_1 ... CAM_TABLE_99 zulässt.

Die Einheiten für die Bausteineingänge Distance und SlaveAxisVelocity sind durch die Wichtungparameter der Achse festgelegt.

ML_PLCopen.library

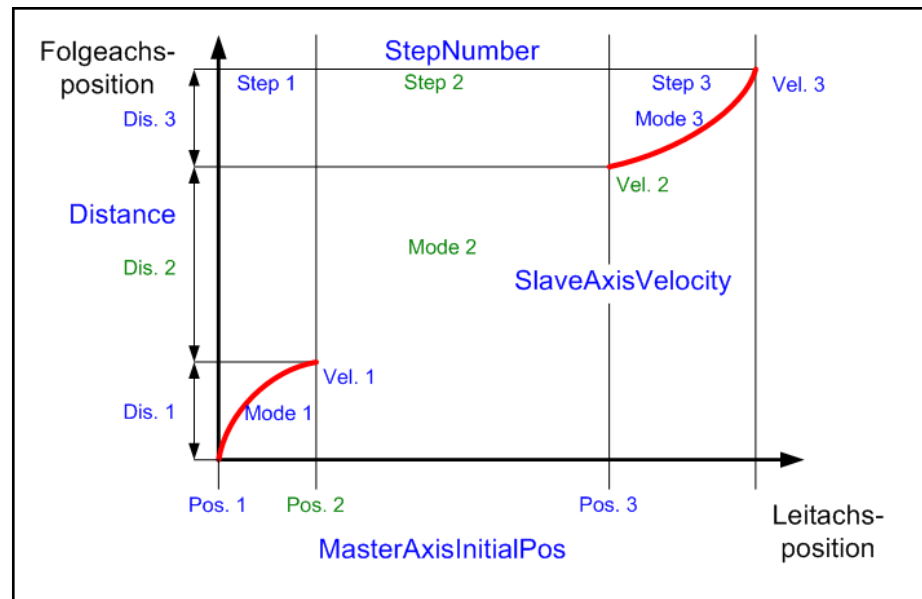


Abb. 6-50: MotionProfile mit 3 Schritten, Schritt 2 wird verändert

MB_ChangeProfileStep, Besonderheiten bei SERCOS-basierten Achsen

Plausibilitätsprüfung und Sammeln

Jeder Schreibvorgang auf ein Bewegungsprofil löst einen Plausibilitätstest bezüglich der Passfähigkeit des neuen Schrittes aus.

Theoretisch ist es möglich, einen die Übergangsbedingungen erfüllenden Schritt in das aktive Bewegungsprofil zu schreiben, jedoch nur wenn mit diesem Satz kein Bewegungsbefehl ausgeführt wird.

Werden mehrere Schritte im Bewegungsprofil getauscht, ist es möglich den Plausibilitätstest zur Vermeidung von Fehlermeldungen zunächst zu blockieren und erst dann auszulösen, wenn alle gewünschten Schritte getauscht sind (Sammeln).

Die Entscheidung sammeln oder sofort testen kann für jede Achse im Parameter "A-0-2930, MotionProfile, Steuerwort, Bit 12", getroffen werden.

Beispiel zur Nutzung des MB_ChangeProfileStep für zwei (oder mehrere) Schritte im Profil

Nachstehend wird der Arbeitsablauf zur Nutzung des "MB_ChangeProfileStep" am Beispiel zweier auszutauschender Schritte beschrieben:

1. Einstellung Betriebsart "Sammeln"
 - Lesen des Parameters "A-0-2930, MotionProfile, Steuerwort" mittels MB_ReadParameter (Wert als DINT),
 - DINT_TO_WORD,
 - OR 16#1000, Setzen Bit 12, sammeln, WORD_TO_DINT,
 - Schreiben des Parameters "A-0-2930, MotionProfile, Steuerwort" mittels MB_WriteParameter (Wert als DINT).
2. Tausch des ersten Schrittes
 - MB_ChangeProfileStep für den ersten Datensatz (wird ohne Test durchgeführt).

Die Statusänderung wird im Parameter "A-0-2931, MotionProfile, Statuswort" wirksam.
3. Tausch des zweiten Schrittes
 - MB_ChangeProfileStep für den zweiten Datensatz (wird ohne Test durchgeführt).

4. Einstellung Betriebsart "Testen"

- Lesen des Parameters "A-0-2930, MotionProfile, Steuerwort" mittels MB_ReadParameter (Wert als DINT),
- DINT_TO_WORD,
- AND 16#EFFF, Rücksetzen Bit 12, Auslösen des Testvorganges, WORD_TO_DINT,
- Schreiben des Parameters "A-0-2930, MotionProfile, Steuerwort" mittels MB_WriteParameter (Wert als DINT).

5. Lesen des Status zur Kontrolle, dass das gesamte Profil getestet wurde.

- Lesen des Parameters "A-0-2931, MotionProfile, Statuswort" mittels MB_ReadParameter (Wert als DINT),
- DINT_TO_DWORD,
- **Bit 12/13:**

Bit 12 gibt an, ob das Bewegungsprofil in Satz 0 überprüft wurde.

Dann ist aus Bit 13 das Ergebnis der Prüfung zu entnehmen. Nähere Informationen über ein negatives Prüfergebnis "F2229235, Fehler im MotionProfile" befinden sich in "A-0-2902, MotionProfile 0, Diagnose".

Bit 14/15:

Bit 14 gibt an, ob das Bewegungsprofil in Satz 1 überprüft wurde.

Dann ist aus Bit 15 das Ergebnis der Prüfung zu entnehmen. Nähere Informationen über ein negatives Prüfergebnis "F2229235, Fehler im MotionProfile" befinden sich in "A-0-2903, MotionProfile 1, Diagnose".

Fehlerbehandlung

Der Funktionsbaustein MB_ChangeProfileStep erzeugt folgende Arten von Fehlermeldung:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

MB_ChangeCamData**Kurzbeschreibung**

Der Funktionsbaustein MB_ChangeCamData unterstützt das Laden von Kurventabellen-Daten unter Echtzeitbedingungen, die auf der SPS-Seite gespeichert oder / und bearbeitet wurden, auf die Kurventabellenparameter, C-0-2001 ... 2099 (Motion-Seite).

Der Funktionsbaustein transportiert "NumberOfElements * 4 Byte" Daten, beginnend ab dem Ort im SPS-Speicher auf den der "CamTableAdr"-Pointer zeigt, auf die mit "CamTableID" ausgewählte Kurventabelle (RAM, C-Parameter).

Done zeigt den erfolgreich abgeschlossenen Kopiervorgang an.



Die zu übertragenden Einzeldaten dürfen im Bereich -799.999.999 bis +799.999.999 liegen.



Der Funktionsbaustein führt keine inhaltliche Prüfung der zu übertragenden Daten durch, so dass der Anwender für die korrekte Datenmenge und den richtigen Datentyp (DINT) der zu übertragenden Information zu sorgen hat.

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.11 ML_FlexProfile

Kurzbeschreibung Der Funktionsbaustein "ML_FlexProfile" realisiert eine elektronische Kurventabelle (FlexProfile).

FlexProfile sind dabei durch die Parameter analytisch definiert, die zu der Betriebsart "Elektronisches Bewegungsprofil" gehören.



Die Anwendung des Bausteins setzt die vorherige Leistungszuschaltung über eine Instanz des [MC_Power, Seite 176](#), (bei realen Achsen) voraus.



Der Baustein wird durch den Baustein [MC_GearOut, Seite 246](#), deaktiviert.



FlexProfile-Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation in der Steuerung
- HydraulicDrive mit Interpolation in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-39: Bibliothekszuordnung

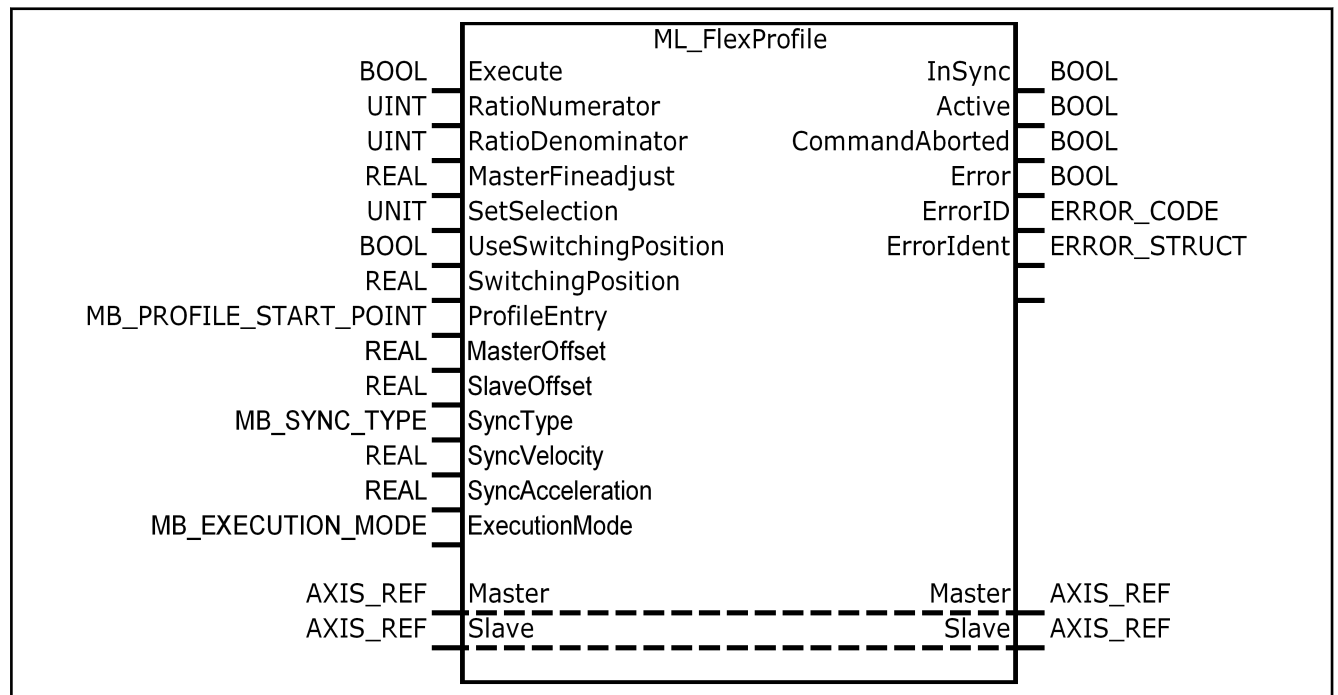


Abb. 6-53: Funktionsbaustein, ML_FlexProfile

Schnittstellenbeschreibung

ML_PLCopen.library

	Name	Typ	Kommentar
VAR_IN_OUT	Master	AXIS_REF	Auswahl der Leitachse
	Slave	AXIS_REF	Auswahl der Folgeachse, die der Leitachse folgt
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
	RatioNumerator	UINT	Getriebeübersetzungsverhältnis - Zähler "Leitachsgetriebe Ausgangsumdrehungen" min: 1; max: 65535; default: 1
	RatioDenominator	UINT	Getriebeübersetzungsverhältnis - Nenner "Leitachsgetriebe Eingangsumdrehungen" min: 1; max: 65535; default: 1
	MasterFineadjust	REAL	"Feinabgleich Getriebeübersetzung" in % ; default: 0.0, min.: -799,9999%, max.: 799,9999
	SetSelection	UINT	Auswahl des FlexProfile (0 ...3)
	UseSwitchingPosition	BOOL	UseSwitchingPosition legt fest, ob die Umschaltposition bzw. Anfangsposition aktiviert (berücksichtigt) werden soll. Falls TRUE, dann erfolgt der Bewegungsstart bzw. das Umschalten auf ein anderes Profil erst, wenn die Leitachse die am Eingang "SwitchingPosition" angelegte Position erreicht.
	SwitchingPosition	REAL	Umschaltposition bzw. Anfangsposition, an der die Umschaltung auf ein anderes Profil bzw. der Profilstart erfolgen soll.
	ProfileEntry	MB_PROFI- LE_START_POINT	Beschreibt wie bei einem Profilstart oder Profilwechsel der Ursprung des Profils interpretiert werden soll. 0 - Absoluter Einstieg in das Profil 1 - Profil startet an der aktuellen Leit- und Folgeachseposition 2 - Profil startet an der aktuellen Leitachseposition und an der absoluten Folgeachseposition 3 - Profil startet an der aktuellen Folgeachseposition und der absoluten Leitachseposition 4 - Profil startet an der absoluten Leitachseposition und der automatisch ermittelten Folgeachseposition
	SlaveOffset	REAL	Offset der Folgeachse
	MasterOffset	REAL	Offset der Leitachse
	SyncType	MB_SYNC_TYPE	Aufsynchronisierart 0 - weiches Umschalten kürzester Weg 1 - weiches Umschalten Uhrzeigersinn 2 - weiches Umschalten gegen Uhrzeigersinn 3 - hartes Umschalten
	SyncVelocity	REAL	Aufsynchronisierungsgeschwindigkeit
	SyncAcceleration	REAL	Aufsynchronisierungsbeschleunigung

	Name	Typ	Kommentar
	ExecutionMode	MB_EXECUTION_MODE	Abarbeitungstyp 0 - Zyklische FlexProfile-Abarbeitung 1 - Einmalige FlexProfile-Abarbeitung
VAR_OUTPUT	InSync	BOOL	Folgeachse ist synchron
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	CommandAborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando.
	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

Tab. 6-40: Schnittstelle des FB ML_FlexProfile

Funktionsbeschreibung

Bei FlexProfilen mit realer/virtueller Leitachse besteht eine feste Beziehung zwischen Leitachsisposition (master position) und Folgeachse (slave position). Die Leitachsisposition wird von einer realen oder virtuellen Leitachsquelle vorgegeben.

- 4 FlexProfile pro Achse mit bis zu 64 Bewegungsschritten sind parametrierbar.
- Ein Bewegungsschritt wird durch eine Kurvenscheibentabelle oder durch ein genormtes Profil (z.B. Polynom 5. Ordnung, Polynom 7. Ordnung, Sinoide, ...) festgelegt.
- In einem FlexProfile können bis zu 16 Bedingungen (Reaktion auf die Position, Zeit, Steuerbit...) definiert werden.
- Profilstartpunkt kann beim Profilstart oder Profilwechsel definiert werden.
- Die Leitachse kann entweder als Zeitachse oder Positionsachse konfiguriert werden.
- Norm-Profile Rast in Rast, Rast in Geschwindigkeit, Geschwindigkeit in Rast, Geschwindigkeit in Geschwindigkeit und konstante Geschwindigkeit sind wählbar.
- Werden Kurvenscheiben benötigt, sind Kurvenscheibentabellen mit 3 bis maximal 1024 Stützpunkten wählbar.

Die Kurventabellen sind auf der Steuerung unter den C-Parametern C-0-2001...2099 bevorratet. Der Baustein [MB_ChangeCamData](#) erlaubt es, diese Parameter mit Daten aus der SPS zu füllen.

- Kurventabellen bis 100 Punkte werden mit kubischen Splines, Kurventabellen mit mehr als 100 Punkten linear interpoliert.
- Jeder Bewegungsschritt hat einen eigenen Hub.
- Der Umschalttyp (SyncType) ist wählbar. D.h., die Profile können hart, oder mit einem automatischen Aufsynchronisiervorgang gewechselt werden.
- Aufsynchronisierungsgeschwindigkeit und Aufsynchronisierungsbeschleunigung stehen zur Verfügung.
- Ein elektronisches Getriebe mit Feinabgleich-Getriebeübersetzung steht zur Verfügung.
- Die Umschaltposition bzw. der FlexProfile-Anfang kann konfiguriert und aktiviert werden.

ML_PLCopen.library

- Zusätzliche Prozessreglerparameter Lagesollwert additiv, Leitachsposition additiv (mit Filter 1. Ordnung), Winkelverschiebung Tabellenanfang (mit Filter 1. Ordnung) und Feinabgleich-Getriebeübersetzung können innerhalb der MotionControl-Wirkungskette gewählt werden.

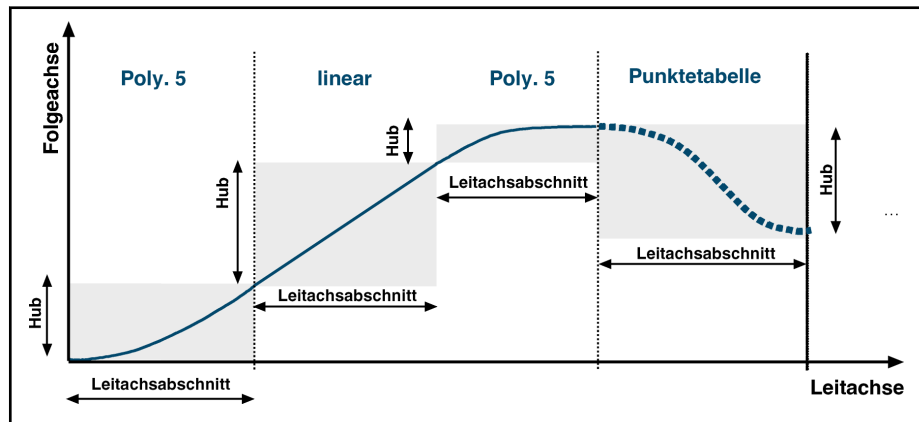


Abb. 6-54: Beispiel eines FlexProfile

6.6.12 ML_FlexProfile, Besonderheiten bei sercos basierten Achsen

Funktionsbeschreibung



FlexProfile-Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation in der Steuerung
- HydraulicDrive mit Interpolation in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

Der Funktionsbaustein ML_FlexProfile realisiert zwei Aufgaben:

- Festlegen, wie das auf der Steuerung gespeicherte FlexProfile zu interpretieren bzw. abzuarbeiten ist,
- Erteilen des Auftrages, das FlexProfile in der Bearbeitung zu starten und zu bearbeiten. Nach erfolgreicher Abarbeitung des Bausteines verbleibt die Achse im FlexProfile-Bearbeitungsmodus bis sie ein anderes Kommando erhält oder ein Fehler eintritt.

Ziel ist es, eine im FlexProfile niedergelegte Abhängigkeit zwischen der Leitachse ("A-0-2700, ELS, Leitachse, Steuerungsnummer" und "A-0-2701, ELS, Leitachse, Achsnummer") und der ausgewählten Folgeachse zu realisieren.

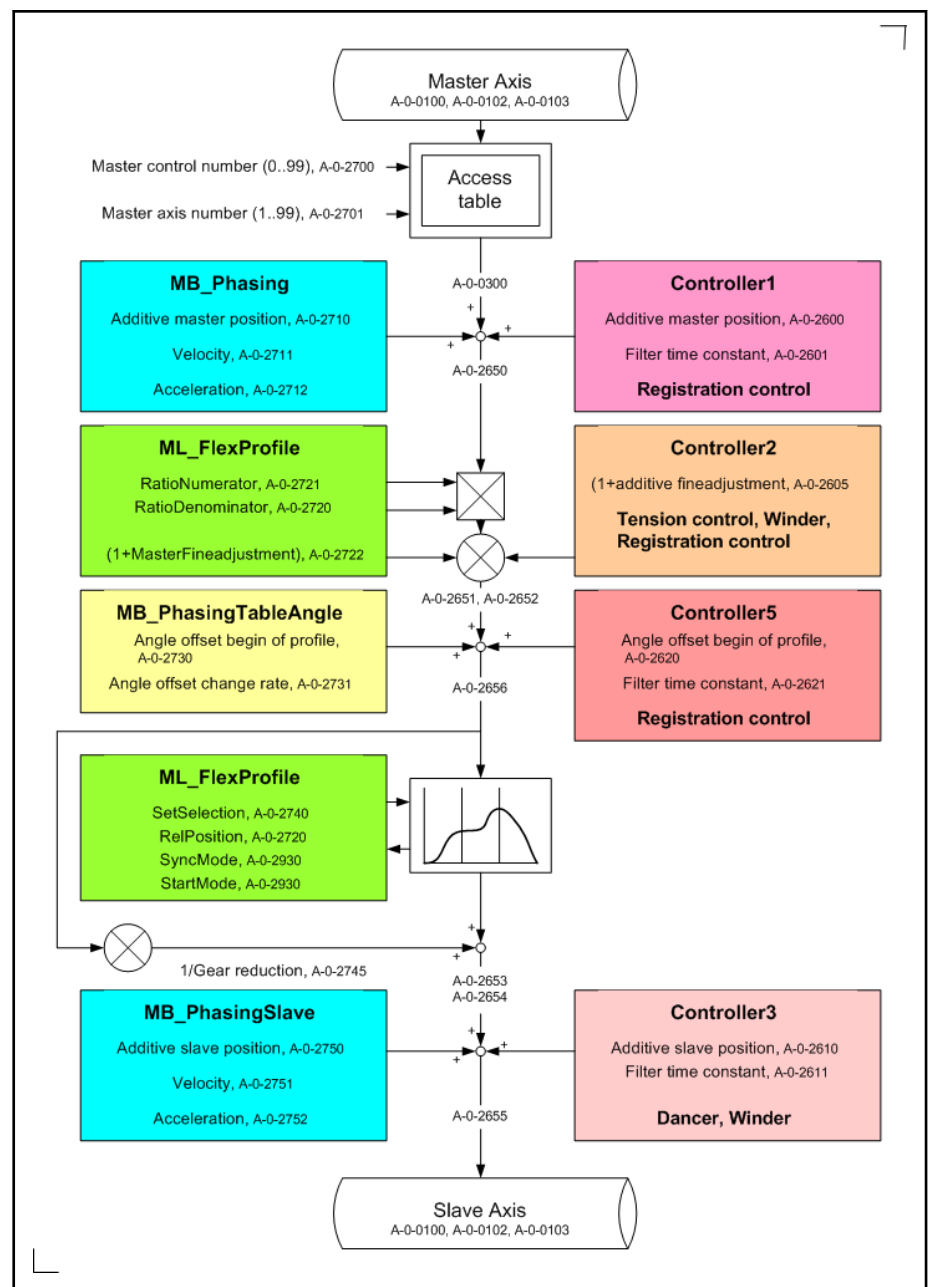


Abb. 6-55: MotionControl-Wirkungskette ML_FlexProfile

Der Baustein enthält folgende funktionelle Gruppen:

- das elektronische Getriebe, das durch das Verhältnis von "A-0-2721, ELS, Leitachsgetriebe, Ausgangsumdrehungen" und "A-0-2720, ELS, Leitachsgetriebe, Eingangsumdrehungen" definiert ist,
- den "A-0-2722, ELS, Feinabgleich der Getriebeübersetzung".
Der Wert des Feinabgleichs wird als %-Zahl vorgeben und als (1 + Feinabgleich) gerechnet.
- die Auswahl eines der vier möglichen FlexProfiles, die vorher in der SPS abzulegen sind:

Gemeinsame Parameter für alle vier Sätze

- A-0-3000, FlexProfile, Aktueller Satz
- A-0-3001, FlexProfile, Aktueller Schritt

ML_PLCopen.library

- A-0-3002, FlexProfile, Leitachs-Umschaltposition
- A-0-3003, FlexProfile, Folgeachse, Positionssollwert additiv
- A-0-3004, FlexProfile, Masterachse, Positionssollwert additiv
- A-0-3009, FlexProfile, Steuerwort
- A-0-3010, FlexProfile, Statuswort
- A-0-3011, FlexProfile, Bedingungen Steuervariablen
- A-0-3012, FlexProfile, Bedingungen Statusvariablen
- A-0-3013, FlexProfile, Wert, Aktuelle Zeit-Master.

Parametername	FlexProfile 0	FlexProfile 1	FlexProfile 2	FlexProfile 3
Leitachsgeschwindigkeit	A-0-3020	A-0-3050	A-0-3080	A-0-3110
Liste der Leitachsquellen	A-0-3021	A-0-3051	A-0-3081	A-0-3111
Liste der Folgeachsbezüge (Hub)	A-0-3022	A-0-3052	A-0-3082	A-0-3112
Liste der Leitachsabschnitte (Range)	A-0-3023	A-0-3053	A-0-3083	A-0-3113
Liste der Leitachsabschnitte Einheiten	A-0-3024	A-0-3054	A-0-3084	A-0-3114
Liste der Bewegungsgesetze	A-0-3025	A-0-3055	A-0-3085	A-0-3115
Liste der Bewegungsschritt-Typen	A-0-3026	A-0-3056	A-0-3086	A-0-3116
Liste der Folgeachsanzugsgeschwindigkeiten	A-0-3027	A-0-3057	A-0-3087	A-0-3117
Liste der Folgeachsanzugsbeschleunigungen	A-0-3028	A-0-3058	A-0-3088	A-0-3118
Liste der Folgeachsanzugsrucke	A-0-3029	A-0-3059	A-0-3089	A-0-3119
Liste der Folgeachsabzugsgeschwindigkeiten	A-0-3030	A-0-3060	A-0-3090	A-0-3120
Liste der Folgeachsabzugsbeschleunigungen	A-0-3031	A-0-3061	A-0-3091	A-0-3121
Liste der Folgeachsabzugsrucke	A-0-3032	A-0-3062	A-0-3092	A-0-3122
Liste der Folgeachsverfahrgeschwindigkeiten	A-0-3033	A-0-3063	A-0-3093	A-0-3123
Liste der Folgeachsverfahrbeschleunigungen	A-0-3034	A-0-3064	A-0-3094	A-0-3124
Liste der Folgeachsverfahrrucke	A-0-3035	A-0-3065	A-0-3095	A-0-3125
Liste der Wendepunkteverschiebungen (WPV)	A-0-3036	A-0-3066	A-0-3096	A-0-3126
Events, Liste Bewegungsschrittzunordnungen	A-0-3040	A-0-3070	A-0-3100	A-0-3130
Events, Liste Auslöser-Modi	A-0-3041	A-0-3071	A-0-3101	A-0-3131
Events, Liste Aktions-Modi	A-0-3042	A-0-3072	A-0-3102	A-0-3132
Events, Liste Auslöser-Werte	A-0-3043	A-0-3073	A-0-3103	A-0-3133
Events, Liste Auslöser-Werte Einheiten	A-0-3044	A-0-3074	A-0-3104	A-0-3134
Kommando: Event prüfen und übernehmen	A-0-3046	A-0-3076	A-0-3106	A-0-3136
Diagnose	A-0-3047	A-0-3077	A-0-3107	A-0-3137
Kommando: Profil prüfen und übernehmen	A-0-3048	A-0-3078	A-0-3108	A-0-3138
Prüfsumme	A-0-3049	A-0-3079	A-0-3109	A-0-3139

Der Baustein ist in der Lage, die Synchronisation, ausgehend vom ankommenden Lage- und Geschwindigkeitssignal, vor dem Einrasten von Leit- und Folgeachse, herbeizuführen.

Fehlerbehandlung Der Funktionsbaustein ML_FlexProfile erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.13 FlexProfile Umschalten und Aufsynchronisieren

Auswahl des Koordinatensystems

Die Einordnung der verschiedenen Profile im Koordinaten-System kann im Bezug auf die jeweilige Leit- und Folgeachse entweder relativ oder absolut sein. Daraus ergibt sich die Möglichkeiten die zugelassenen Einstellungsmöglichkeiten beliebig zu kombinieren (siehe auch MB_PROFILE_START_POINT, ML_Base.library).

Die rote Linie soll das aktuelle Profil symbolisieren und die blaue das Profil auf das umgeschaltet werden soll.

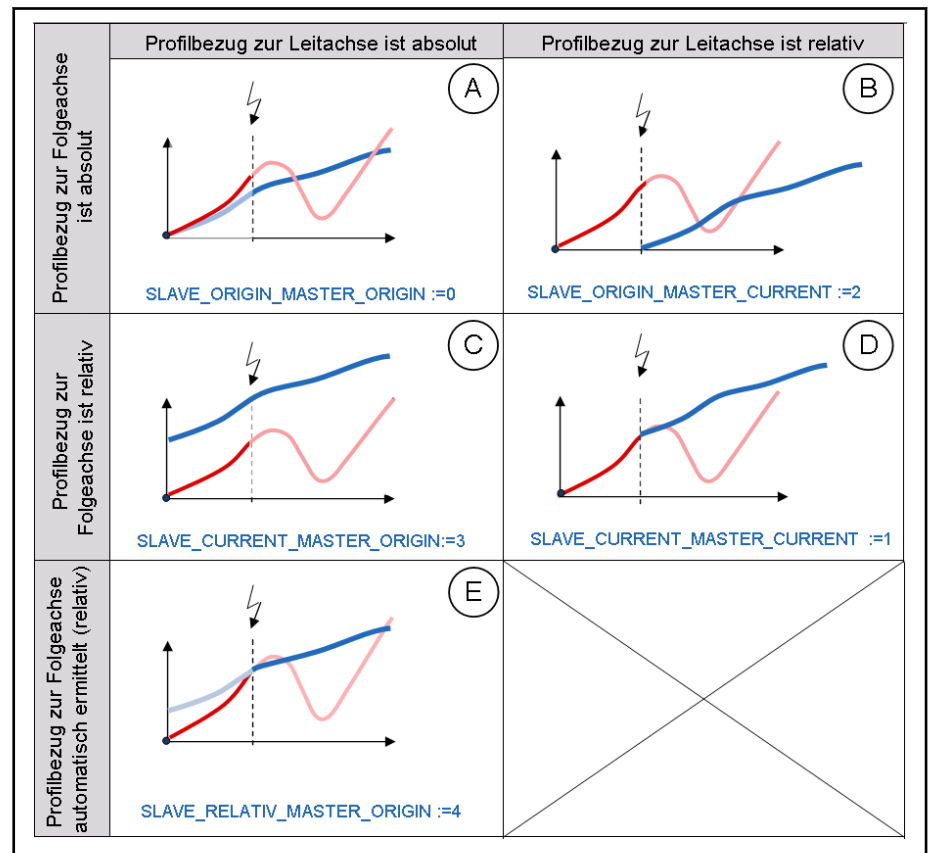


Abb. 6-56: Auswahl des Koordinatensystems

Umschalten und Aufsynchronisieren

Nachdem die Zuordnung von Profilen definiert ist, kann sowohl der Umschaltcharakter (Umschalten auf Winkel oder Sofort) als auch die Aufsynchronisierungsart (mit oder ohne Aufsynchronisieren) ausgewählt werden (siehe auch MB_EXECUTION_MODE, ML_Base.library).

Bei dem Umschalten auf Winkel wird ab dem Freigabesignal solange gewartet, bis der eingegebene Winkel erreicht ist und genau in dem Punkt findet das Umschalten auf das andere Profil bzw. im ausgekuppelten Zustand der Start des neuen Profils statt. Das sofortige Umschalten wird gleichzeitig mit der Eingabe einer positiven Flanke ausgeführt. Bei der Verwendung dieser Funktionalität sollte ein besonderes Augenmerk auf die Deckungsgleichheit der FlexProfile gelegt werden. Wird diese Regel ignoriert, kann es bei manchen Konfigurationen zu Sollwertsprüngen kommen.

In den nachstehenden Abbildungen werden mögliche Kombinationen für das Umschalten und Aufsynchronisieren in Verbindung mit beliebigen Koordinatensystemen vorgestellt.

Im unteren Bild wird auf ein anderes Bewegungsprofil umgeschaltet. Der Ursprung des zweiten Bewegungsprofils ist deckungsgleich mit dem des ersten Bewegungsprofils und liegt im Nullpunkt des Koordinatensystems (absolute Leitachseposition und absolute Folgeachseposition - Fall A). Die Bewegungsprofile werden nicht verschoben.

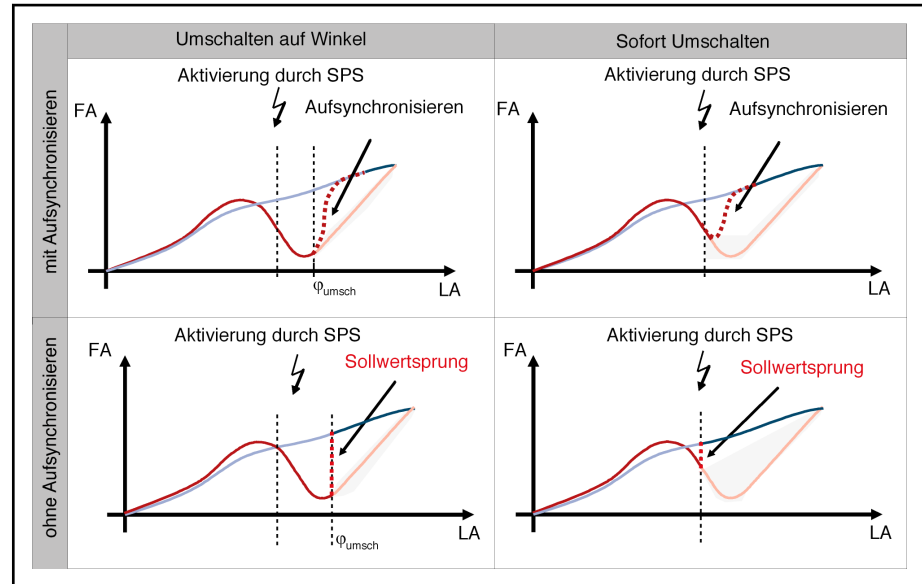


Abb. 6-57: Fall A - Umschalten und Aufsynchronisieren mit absoluter Leitachseposition und absoluter Folgeachseposition

Im unteren Bild wird auf ein anderes Bewegungsprofil umgeschaltet. Der Ursprung des zweiten Bewegungsprofils wird an die aktuelle Leitachse- und Folgeachseposition des ersten Bewegungsprofils verschoben (relative Leitachseposition und relative Folgeachseposition - Fall D).

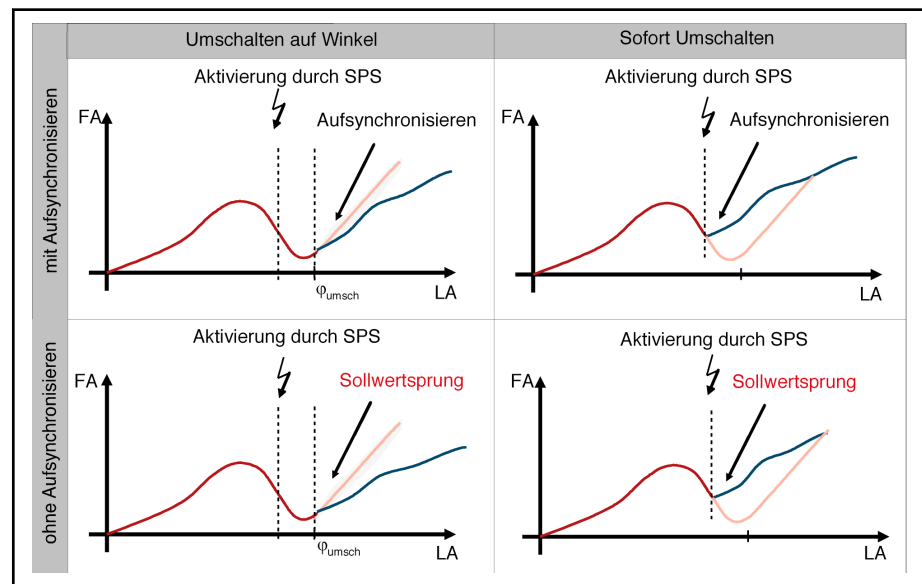


Abb. 6-58: Fall D - Umschalten und Aufsynchronisieren mit relativer Leitachseposition und relativer Folgeachseposition –

Im unteren Bild wird auf ein anderes Bewegungsprofil umgeschaltet. Der Ursprung des zweiten Bewegungsprofils wird an die aktuelle Leitachseposition

des ersten Bewegungsprofils verschoben (relative Leitachseposition und absolute Folgeachseposition - Fall B). Es erfolgt keine Verschiebung des Bewegungsprofils in Richtung der Folgeachse.

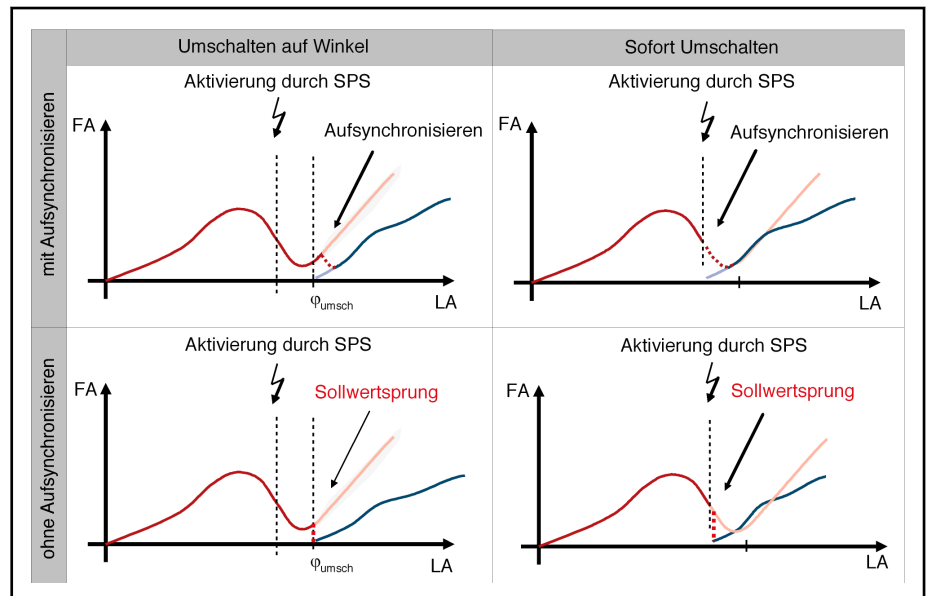


Abb. 6-59: Fall B - Umschalten und Aufsynchronisieren mit relativer Leitachseposition und absoluter Folgeachseposition

Im unteren Bild wird auf ein anderes Bewegungsprofil umgeschaltet. Der Ursprung des zweiten Bewegungsprofils wird an die aktuelle Folgeachseposition des ersten Bewegungsprofils verschoben (Absolute Leitachseposition und relative Folgeachseposition - Fall C). Es erfolgt keine Verschiebung des Bewegungsprofils in Richtung der Leitachse.

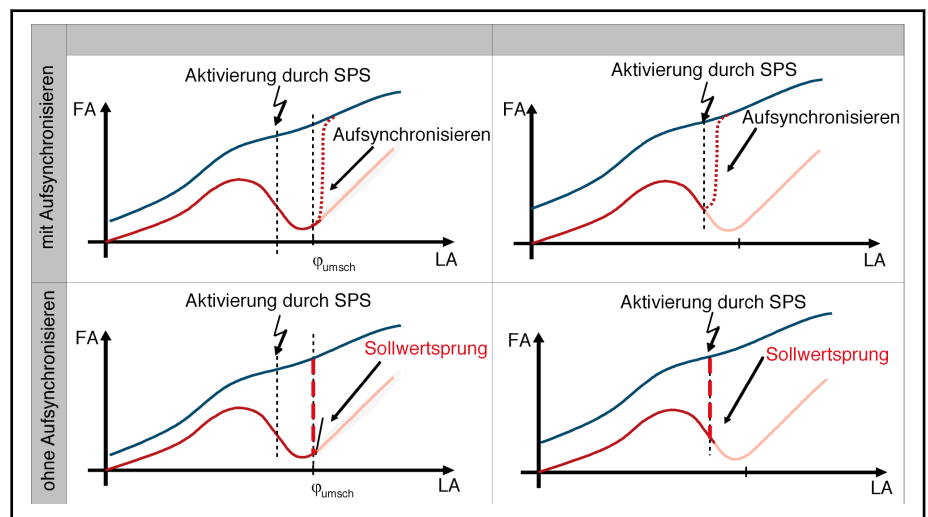


Abb. 6-60: Fall C - Umschalten und Aufsynchronisieren mit Absoluter Leitachseposition und relativer Folgeachseposition

Im unteren Bild wird auf ein anderes Bewegungsprofil umgeschaltet. Spielt ein Phasenversatz der Folgeachse keine Rolle und soll die Folgeachse beim Aufsynchronisieren mit stehender Leitachse keine Bewegung ausführen, dann eignet sich hierfür Modus E. Der Profileinstieg erfolgt dann relativ. Bei einer stehenden Leitachse würde die Folgeachse so keine Aufsynchronisierungsbewegung durchführen. Mit Start der Leitachse folgt die Folgeachse der Leitachse wie durch das Bewegungsprofil gegeben.

ML_PLCopen.library

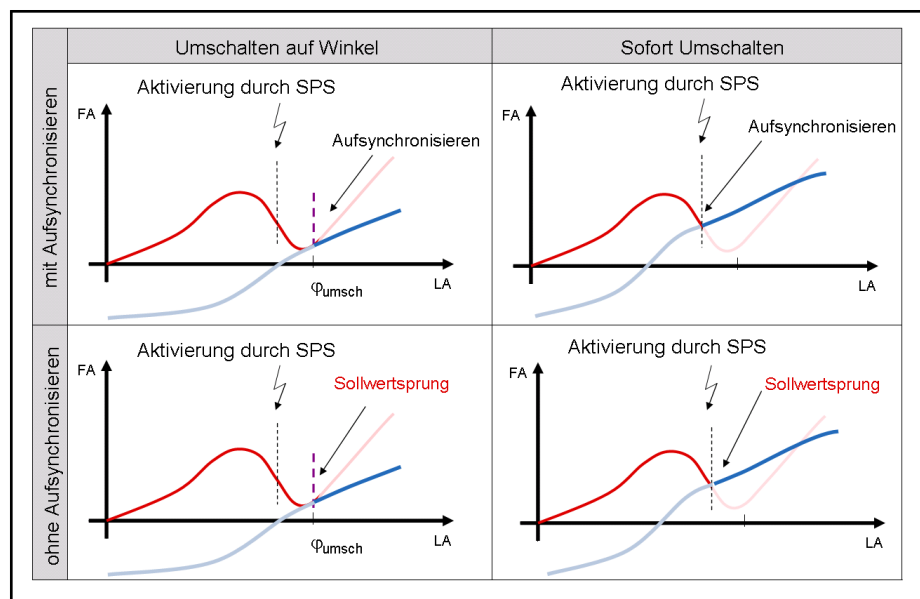


Abb. 6-61: Fall E - Profilbezug zur Folgeachse automatisch ermittelt (relativ)

Art der Abarbeitung

Es kann zwischen zwei Abarbeitungsarten unterschieden werden (siehe auch MB_EXECUTION_MODE, ML_Base.library):

- Kontinuierliche Abarbeitung
- Einmalige Abarbeitung

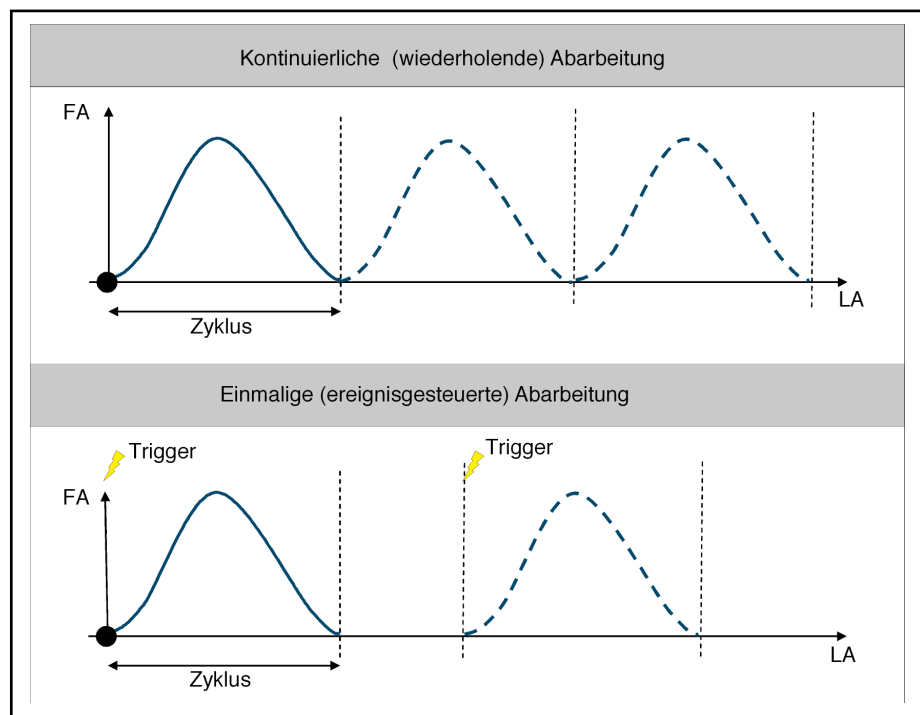


Abb. 6-62: Kontinuierliche Abarbeitung und Einmalige Abarbeitung

Bei der kontinuierlichen Abarbeitung wird das aktive FlexProfil zyklisch durchgeführt. Es ist zu beachten, dass die Profilendgeschwindigkeit des letzten Bewegungsschritts mit der Anfangsgeschwindigkeit des gesamten Profils übereinstimmen muss. Gegebenenfalls wird eine Fehlermeldung ausgegeben. Bei der einmaligen Abarbeitung wird das ausgewählte Profil nur einmal

lig ausgeführt. Der mehrmalige Start des gleichen Profils wird durch den erneuten Aufruf des Bausteins "MC_FlexProfile" erreicht.

6.6.14 PLCopen-Funktionsbausteine zur Unterstützung von Flexprofilen

Allgemeines

Eine XLC/MLC kann bis maximal 64 Achsen¹⁰⁾ verwalten.

Einer Achse können dabei je vier Profile zugeordnet werden, von denen eines aktiviert werden kann. Bei Bedarf kann auf die anderen Profile umgeschaltet werden.

Jedes Profil kann maximal 64 Schritte aufnehmen.

Die einzelnen Bewegungsschritte können entweder über bestimmte Bewegungsgesetze (VDI 2143, und spezielle Bewegungsgesetze) oder über die darauf referenzierte Kurventabellen definiert werden.

Zur Aufnahme der Kurventabellen stellt die Steuerung in ihrer CompactFlash (CF) den Speicherplatz für maximal 99 Punktetabellen mit je 3 bis maximal 1024 Punkten zur Verfügung. Die Punktetabellen sind adressierbar über die Parameter C-0-2001 ... 2099.

Die Kurventabellen können entweder über den SPS-Funktionsbaustein [MB_ChangeCamData, Seite 225](#), oder über die IndraWorks-Oberfläche konfiguriert werden.



FlexProfile-Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation in der Steuerung
- HydraulicDrive mit Interpolation in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

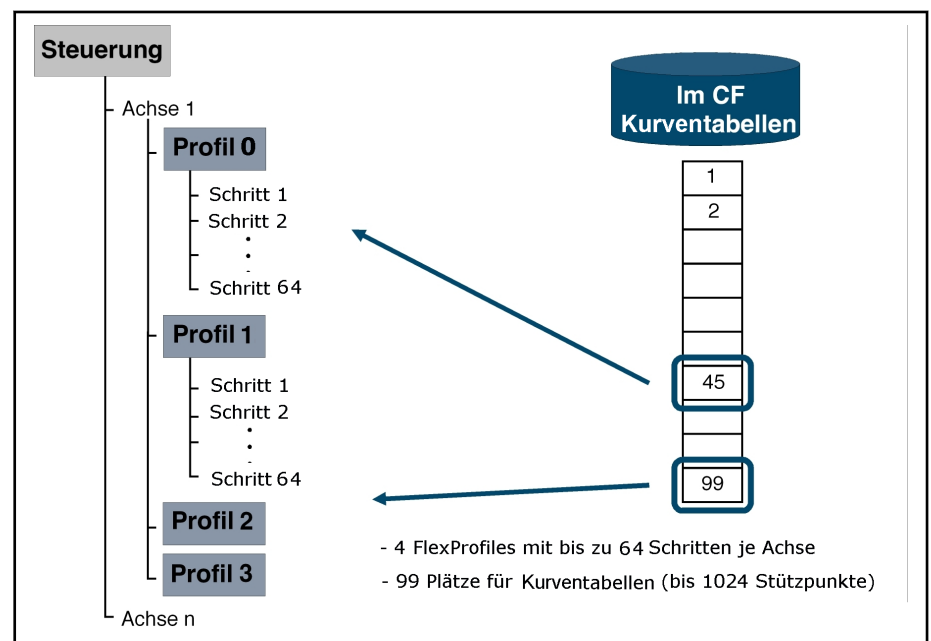


Abb. 6-63: Zuordnung Kurventabellen / FlexProfile

¹⁰⁾ Steuerungstypabhängig

ML_PLCopen.library

Zur Bearbeitung von Flexprofilen und Kurventabellen auf der MLC stehen folgende Bausteine zur Verfügung:

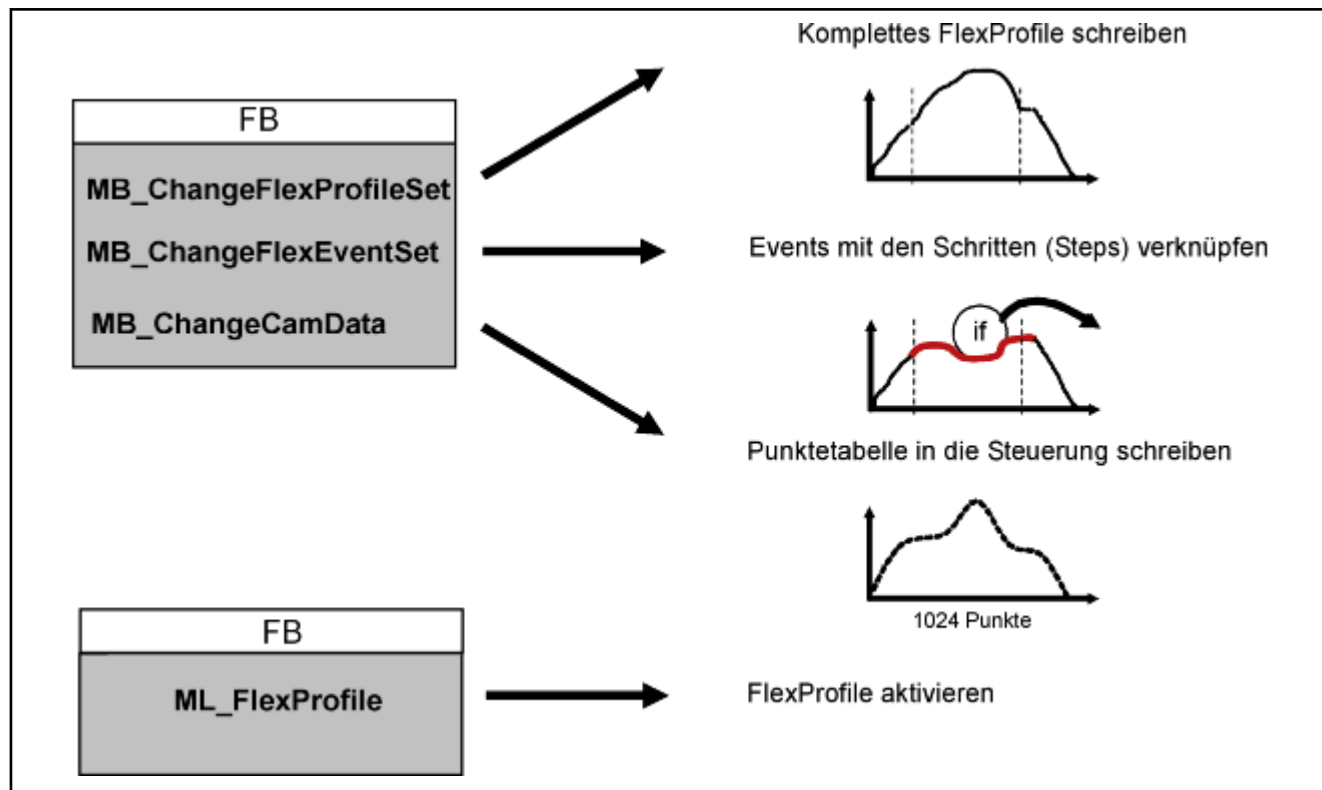


Abb. 6-64: Übersicht über die SPS-Bausteine

- [MB_ChangeFlexProfileSet](#), Seite 238: Schreiben eines kompletten Flexprofiles, von der SPS in die Motion
- [MB_ChangeFlexEventSet](#), Seite 242: Konfiguration von Events.
- [MB_ChangeCamData](#), Seite 225.: Übertragen einer Kurventabelle aus dem SPS-Speicher in die Motion, C-0-2001...2099 .

MB_ChangeFlexProfileSet

Kurzbeschreibung

Der Funktionsbaustein "MB_ChangeFlexProfileSet" gestattet es, ein komplettes Bewegungsprofil (FlexProfile) mit maximal 64 Schritten (Steps) zu beschreiben bzw. ein bereits vorhandenes auszutauschen.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-41: Bibliothekszuordnung

Schnittstellenbeschreibung

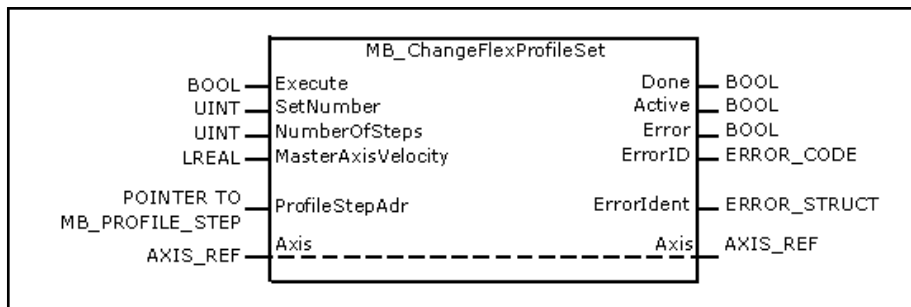


Abb. 6-65: Funktionsbaustein, MB_ChangeFlexProfileSet

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur Achse, deren FlexProfile-Satz ausgetauscht werden soll.
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke.
	SetNumber	UINT	Auswahl des FlexProfiles (0 ... 3)
	NumberOfSteps	UINT	Anzahl der Schritte des FlexProfiles (1...64)
	MasterAxisVelocity	LREAL	Normierte Leitachsgeschwindigkeit, auf die sich die angegebenen Geschwindigkeiten beziehen
	ProfileStepAdr	POINTER TO MB_PROFILE_STEP	Zeiger auf Struktur-Datenfeld, welches die Definition des FlexProfiles enthält.
VAR_OUTPUT	Done	BOOL	Daten wurden übernommen.
	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.

Tab. 6-42: Schnittstelle des FB MB_ChangeFlexProfileSet

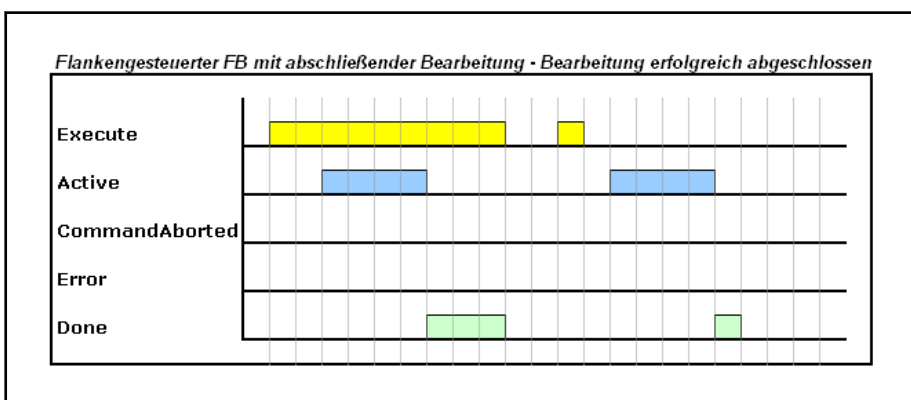
Signal-Zeit-Diagramm

Abb. 6-66: Signal-Zeit-Diagramm: MB_ChangeFlexProfileSet

Funktionsbeschreibung

Der Funktionsbaustein erlaubt es, ein komplettes FlexProfile mit 1...64 Schritten von der SPS an die Motionkomponente der Steuerung zu übergeben.

Die Daten, die die jeweiligen Schritte charakterisieren, sind in der Liste abgelegt, deren Anfang im SPS-Speicherraum jeweils durch einen Pointer (of LREAL) zu beschreiben ist.

Fehlerbehandlung

Der Funktionsbaustein MB_ChangeProfileSet erzeugt folgende Arten von Fehlermeldung:

- [MLC_TABLE](#), [ERROR_TABLE 16#0030](#) , [Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

Eigenschaften der Ereignissteuerung

In fast allen Verarbeitungsmaschinen sind mehrere Achsen am Prozess beteiligt. Die Bewegungsaufgaben, die die einzelnen Achsen dabei zu erledigen haben, sind in den meisten Fällen voneinander abhängig und ergeben sich aus der Anwendungsaufgabe.

So muss die Bewegung einer Achse oft genau an der Stelle beginnen, an der eine andere Achse eine ganz spezielle Position erreicht hat oder sich für eine

gewisse Zeit in einem Bewegungsschritt befunden hat. Im folgenden Bild werden diese Zusammenhänge exemplarisch veranschaulicht.

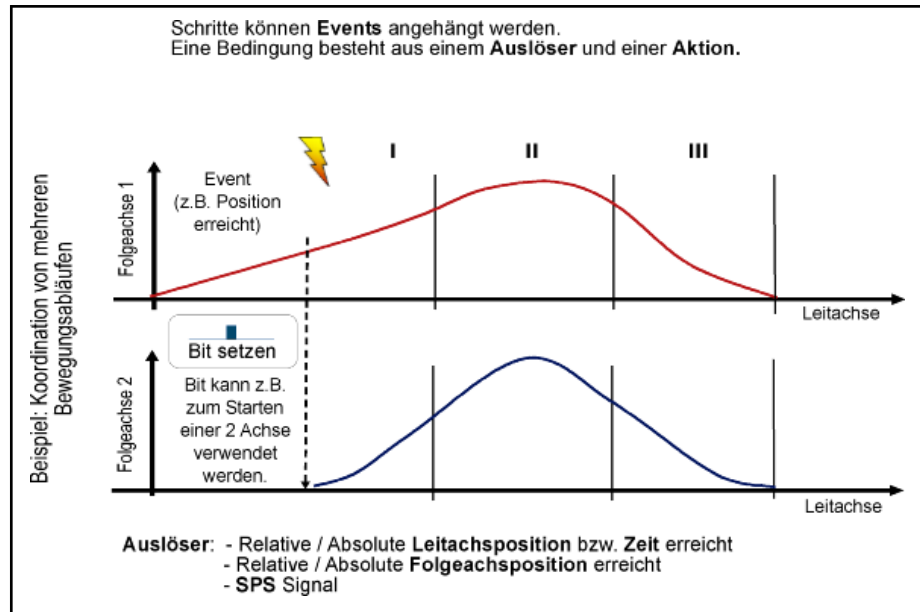


Abb. 6-67: Beispiel einer Ereignissteuerung

Events eignen sich hier sehr gut, um diese Punkte innerhalb des Bewegungsprofils zu markieren. Auf diese kann dann sehr bequem innerhalb einer SPS über die Axisdata-Struktur reagiert werden. Es können Anschlussbewegungen anderer Achsen gestartet oder über den weiteren Verlauf der Abarbeitung entschieden werden. Es ergibt sich eine dynamische "Bewegungskette", die in ihrer Realisierung innerhalb einer SPS, die Abhängigkeiten der Maschine widerspiegelt.

An die Bewegungsschritte können Events angehängt werden. Eine Bedingung besteht aus einem Auslöser und einer Aktion.

Der Auslöser dient zur Erkennung von Zuständen der Steuervariablen, diese sind:

- **Relative/Absolute Leitachseposition erreicht**

Beschreibt das Erreichen einer relativen/absoluten Position der Leitachse. Ähnlich einem Nockenschaltwerk kann die aktuelle Position einer Leitquelle überwacht werden. Überfährt diese im aktuellen Bewegungsschritt eine vorgegebene Position, dann tritt das Ereignis ein.

- **Relative/Absolute Folgeachseposition erreicht**

Das Erreichen einer relativen/absoluten Position der Folgeachse. Gleiches Konzept lässt sich auch auf die Position der Folgeachse anwenden.

- **SPS-Signal**

Aktionen können durch eine SPS ausgelöst werden. Events werden an die SPS gemeldet oder starten direkt eine Aktion.

- **Zeitgesteuertes Abbrechen**

Der Schritt selbst ist nicht zeitgesteuert, besitzt also keinen Zeit-Master, sondern eine Leitachse zu der synchron gefahren werden soll. Dies lässt sich erreichen, durch Verknüpfung eines Ereignisses, welches die Verfahrzeit seit Bewegungsschrittbeginn überwacht, und einer Aktion, die einen Abbruch des aktuellen Bewegungsschrittes bewirkt. Das resul-

tierende Bewegungsprofil lässt sich damit generisch für verschiedene Leitachsgeschwindigkeiten verwenden.

- **Überwachung für Zeit-Segmente**

In Bezug auf die Zeit-Master-Funktionalität lassen sich die Ereignisse auch zur Überwachung der verschiedenen Leitquellen nutzen. Je nach Bewegungsaufgabe kann zum Beispiel ein Event so gesetzt werden, dass während eines Bewegungsschrittes der an einen Zeit-Master gekoppelt ist, der Fahrweg der Leitachse überwacht wird. Auf diese Weise könnte eine Fehlerreaktion durchgeführt werden, falls die Leitachse sich in dieser Zeit zu weit bewegt hat. Verschiedene Prozesszustände können vom Anwender mit Hilfe der Events auf diese Weise sehr einfach und komfortabel abgefangen werden. In einer SPS lässt sich dann darauf reagieren.

Die Aktion ist ein aktives Element, die für die Aktivierung von bestimmten Variablen verantwortlich ist.

Mögliche Aktionen sind:

- Bit sofort setzen

Es wird eines der 16 zur Verfügung stehenden Bits im AxisData sofort gesetzt.

- Bit am Ende des Bewegungsschritts setzen

Es wird eines der 16 zur Verfügung stehenden Bits im AxisData am Ende des Bewegungsschritts gesetzt.

- Bit am Ende des Profils setzen

Es wird eines der 16 zur Verfügung stehenden Bits im AxisData am Ende des Profils gesetzt.

- Bit sofort setzen und mit dem nächsten Bewegungsschritt fortfahren

Bewirkt das sofortige Abbrechen des aktuellen Bewegungsschrittes. Dieser wird also nicht zu Ende abgearbeitet, sondern es wird aus der laufenden Bewegung direkt mit dem nächsten Bewegungsschritt begonnen. Handelt es sich dabei um einen Flex-Bewegungsschritt, kann dieser Wechsel stoß- und ruckfrei erfolgen.

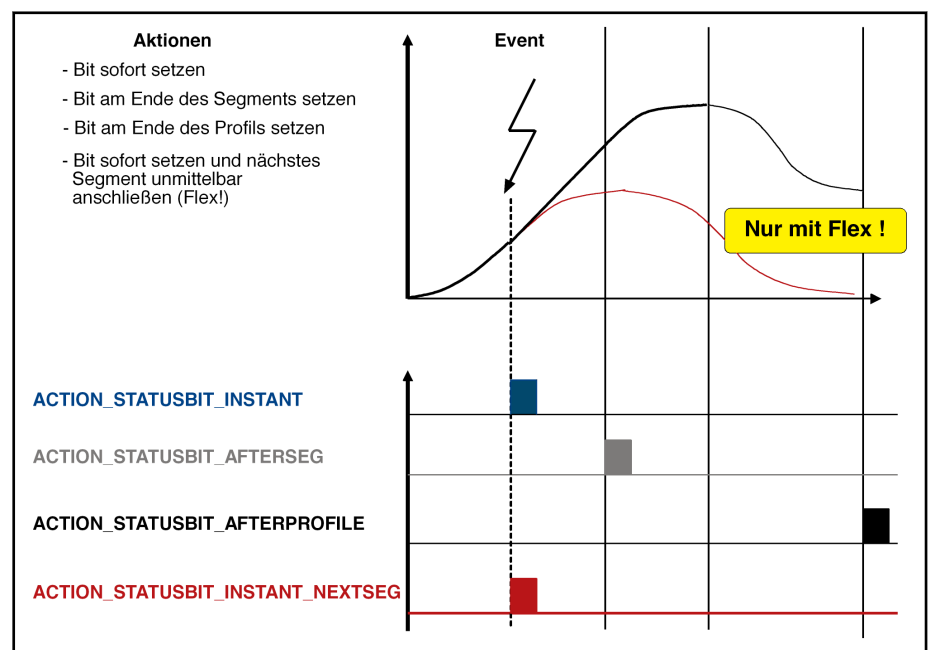


Abb. 6-68: Vorstellung der möglichen Aktionen

ML_PLCopen.library

Die Ereignisse können entweder über den SPS-Funktionsbaustein "MB_ChangeFlexEventSet" oder über die IndraWorks-Oberfläche konfiguriert werden.

MB_ChangeFlexEventSet

Kurzbeschreibung Der Funktionsbaustein "MB_ChangeFlexEventSet" wird zur Konfiguration von Events eingesetzt.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-43: Bibliothekszuordnung

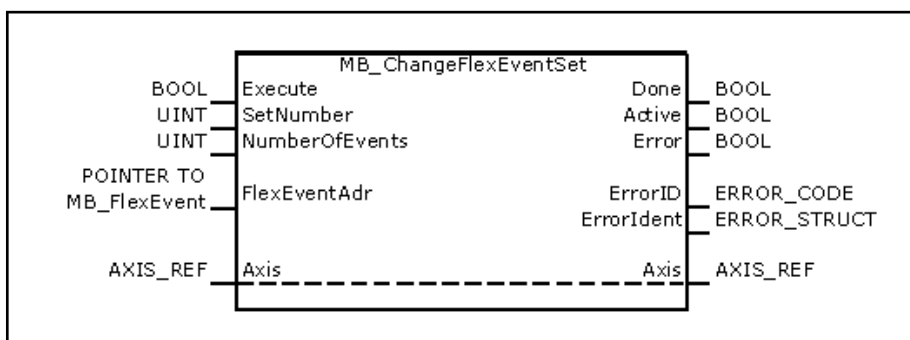


Abb. 6-69: Funktionsbaustein, MB_ChangeFlexEventSet

Schnittstellenbeschreibung

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur Achse, deren Bewegungsprofilsatz ausgetauscht werden soll.
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke.
	SetNumber	UINT	Auswahl des FlexProfiles (0..3)
	NumberOf-Events	UINT	Anzahl der Events die definiert werden (1..16)
	FlexEvent-Adr	POINTER TO MB_FLEX_EVENT	Definitionsliste der Events(1..16) Max. Anzahl der Events pro FlexProfile-Satz beträgt 16. Die Events können beliebig über den FlexProfile-Satz verteilt werden
VAR_OUTPUT	Done	BOOL	Daten wurden übernommen
	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

Tab. 6-44: Schnittstelle des FB MB_ChangeFlexEventSet

- Es können maximal 16 Events pro FlexProfile definiert werden.
- Alle definierten Events können beliebig über den gesamten FlexProfile-Satz verteilt werden. D.h., alle 16 Events können auch mit einen gemeinsamen Bewegungsschritt verknüpft werden.
- Ein Event kann nur einmal referenziert werden.

- Fehlerbehandlung**
- Falls gleiche Events auf mehreren Bewegungsschritten referenziert werden sollen, müssen diese dupliziert werden.
- Der Funktionsbaustein MB_ChangeFlexEventSet erzeugt folgende Fehlermeldungen:
- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.15 MC_GearIn

Kurzbeschreibung Der Funktionsbaustein MC_GearIn realisiert eine elektronische Getriebefunktion mit Feinabgleich.



Die Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation im Antrieb und in der Steuerung
- HydraulicDrive mit Interpolation im Antrieb und in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-45: Bibliothekszuordnung

Schnittstellenbeschreibung

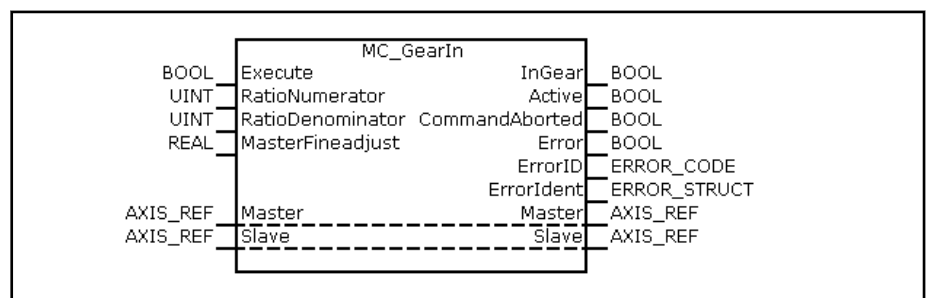


Abb. 6-70: FB MC_GearIn

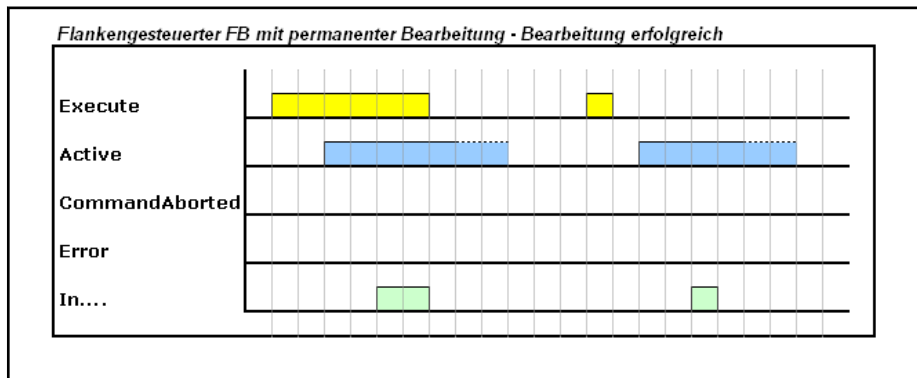
	Name	Typ	Kommentar
VAR_IN_OUT	Master	AXIS_REF	enthält Informationen zur Leitachse
	Slave	AXIS_REF	enthält Informationen zur Folgeachse, die der Leitachse folgt
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigenden Flanke
	RatioNumerator	UINT	Getriebeübersetzungsverhältnis - Zähler "Leitachsgetriebe Ausgangsumdrehungen" min: 1; max: 65535; default: 1
	RatioDenominator	UINT	Getriebeübersetzungsverhältnis - Nenner "Leitachsgetriebe Eingangsumdrehungen" min: 1; max: 65535; default: 1
	MasterFineadjust	REAL	"Feinabgleich Getriebeübersetzung" in %; default: 0.0, min.: -799,9999%, max.: 799,9999
VAR_OUTPUT	InGear	BOOL	Folgeachse folgt Leitachse; Kopplung erfolgreich durchgeführt

ML_PLCOpen.library

	Name	Typ	Kommentar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-46: Schnittstelle des FB MC_GearIn

Signal-Zeit-Diagramm



In... InGear

Abb. 6-71: Signal-Zeit-Diagramm

6.6.16 MC_GearIn, Besonderheiten bei Sercos basierten Achsen

Funktionsbeschreibung

Die Getriebefunktion wird mittels antriebsinterner Getriebefunktionalität realisiert oder auf der Steuerung gerechnet. Ziel ist ein geschwindigkeitssynchroner (driftfreier) Lauf zwischen der Leitachse (A-0-2700, ELS, Leitachse, Steuerungsnummer, A-0-2701, ELS, Leitachse, Achsnummer) und der ausgewählten Folgeachse.



Die Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation im Antrieb und in der Steuerung
- HydraulicDrive mit Interpolation im Antrieb und in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

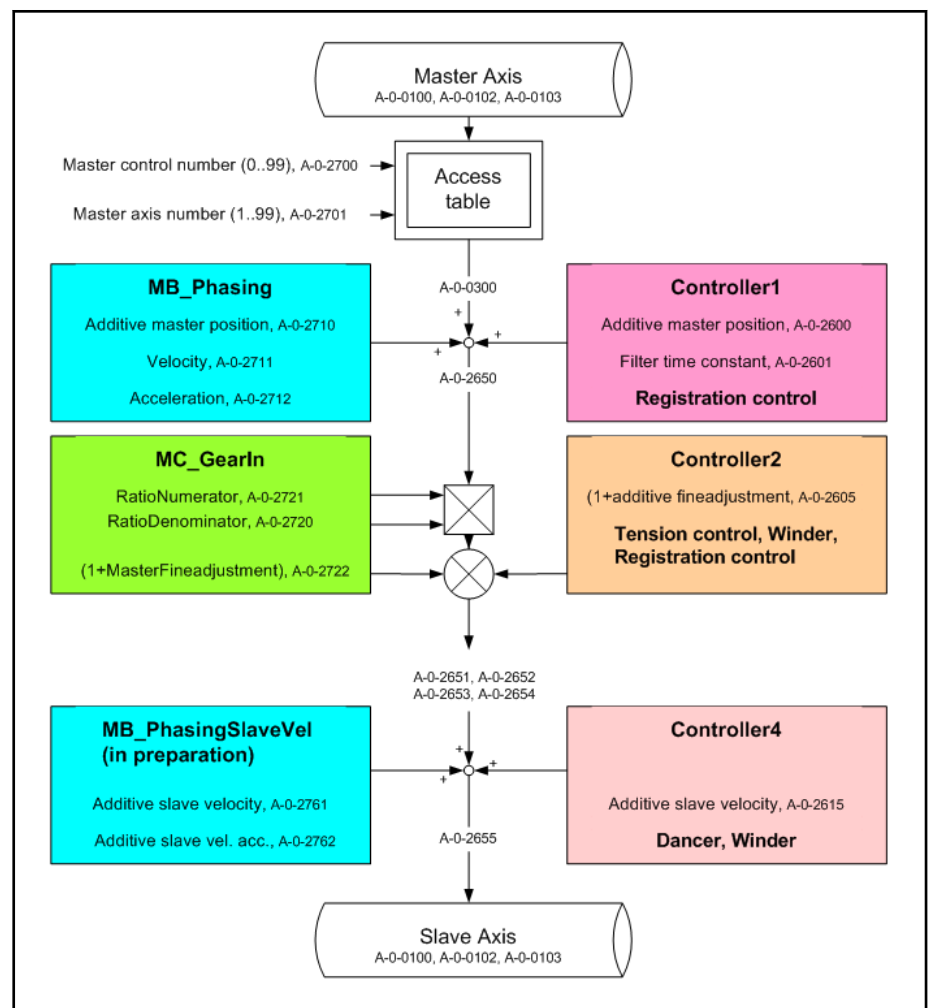


Abb. 6-72: MotionControl-Wirkungskette MC_GearIn

Der Baustein enthält zwei funktionelle Gruppen:

- das elektronische Getriebe, welches definiert ist durch das Verhältnis von A-0-2721, ELS, Leitachsgetriebe, Ausgangsumdrehungen, und A-0-2720, ELS, Leitachsgetriebe, Eingangsumdrehungen, und
- den A-0-2722, ELS, Feinabgleich der Getriebeübersetzung. Der Wert des Feinabgleichs wird als %-Zahl vorgeben und als (1 + Feinabgleich) verrechnet.

Wird der Baustein im Antrieb (IndraDrive) gerechnet, werden die A-Parameter nachstehenden S- und P-Parametern übergeben:

- A-0-2721 nach P-0-0157,
- A-0-2720 nach P-0-0156,
- A-0-2722 nach P-0-0083.

Der Baustein ist in der Lage, die Synchronisation, ausgehend vom ankommenden Geschwindigkeitssignal, vor dem Einrasten von Leit- und Folgeachse, herbeizuführen.

Fehlerbehandlung

Der Funktionsbaustein MC_GearIn erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

ML_PLCopen.library

6.6.17 MC_GearOut

Kurzbeschreibung Der Funktionsbaustein MC_GearOut beendet die von einem der Funktionsbausteine MC_GearIn oder MB_GearInPos aktivierte elektronische Getriebe-funktion mit Feinabgleich bzw. die von einem der Funktionsbausteine MB_MotionProfile oder ML_FlexProfile aktivierte Profile.

Der Baustein realisiert zwei Aufgaben:

- Beenden des geschwindigkeitssynchronen Laufs der Folgeachse
- Überführen der Folgeachse in die Betriebsart "Continuous Motion" unter Beibehaltung des letzten Geschwindigkeitswertes, für den Fall, dass kein anderes Kommando ansteht.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-47: Bibliothekszuordnung

Schnittstellenbeschreibung

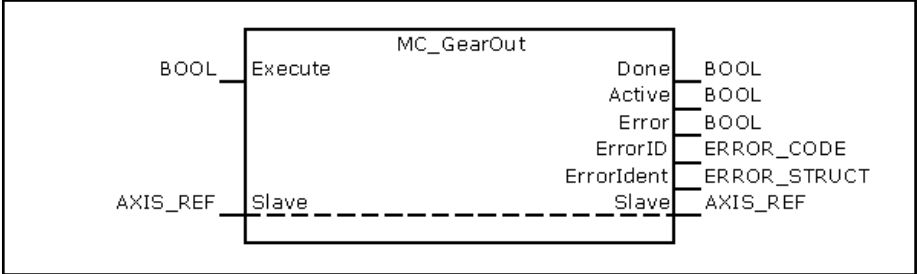


Abb. 6-73: FB MC_GearOut

	Name	Typ	Kommentar
VAR_IN_OUT	Slave	AXIS_REF	enthält Informationen zur Folgeachse, die der Leitachse folgt
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigenden Flanke
VAR_OUTPUT	Done	BOOL	Entkopplung erfolgreich durchgeführt
	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

Tab. 6-48: Schnittstelle des FB MC_GearOut

Signal-Zeit-Diagramm

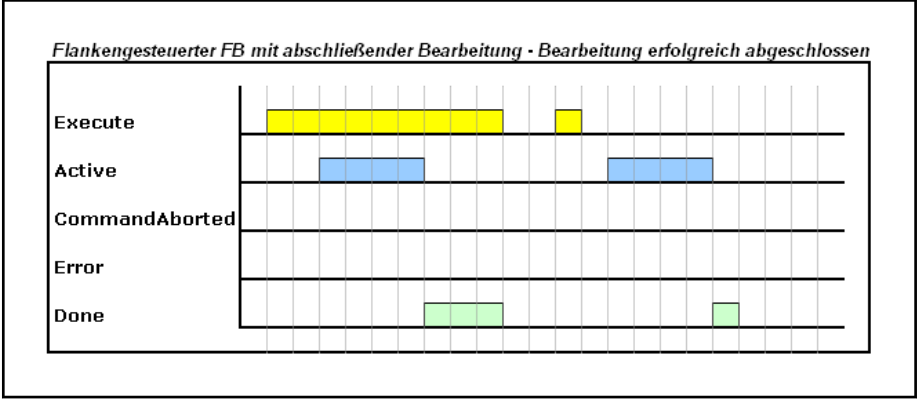


Abb. 6-74: Signal-Zeit-Diagramm

6.6.18 MC_GearOut, Besonderheiten bei SERCOS-basierten Achsen

Fehlerbehandlung Der Funktionsbaustein MC_GearOut erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

6.6.19 MB_GearInPos

Kurzbeschreibung Der Funktionsbaustein MB_GearInPos realisiert eine elektronische Getriebe-funktion mit Feinabgleich.

Ziel ist ein phasen- und geschwindigkeitssynchroner (driftfreier) Lauf zwischen der Leitachse und der ausgewählten Folgeachse.



Die Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation im Antrieb und in der Steuerung
- HydraulicDrive mit Interpolation im Antrieb und in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-49: Bibliothekszuordnung

Schnittstellenbeschreibung

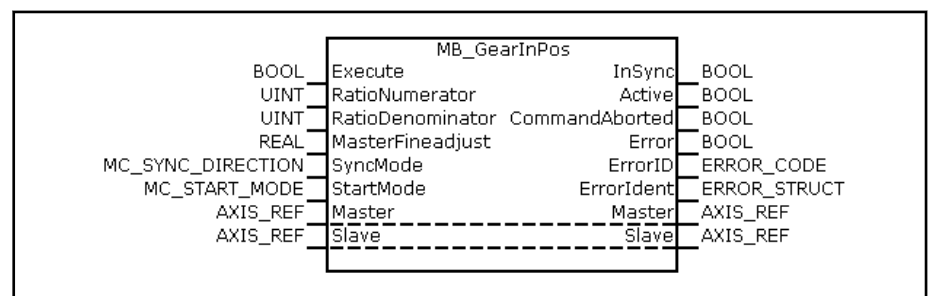


Abb. 6-75: FB MB_GearInPos

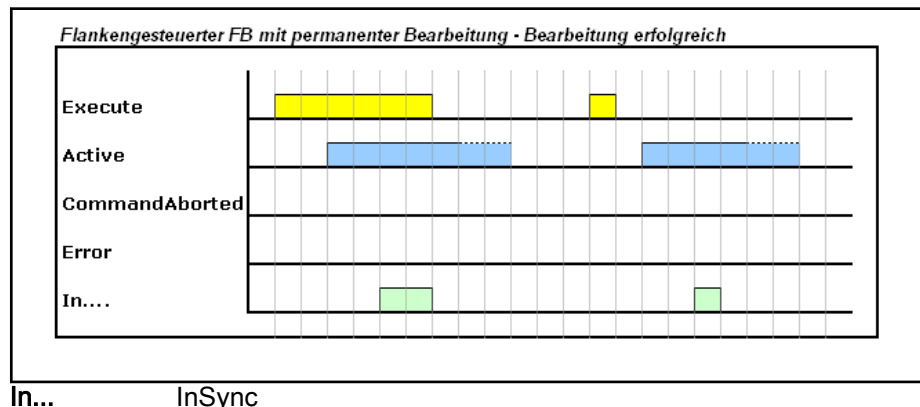
	Name	Typ	Kommentar
VAR_IN_OUT	Master	AXIS_REF	enthält Informationen zur Leitachse
	Slave	AXIS_REF	enthält Informationen zur Folgeachse, die der Leitachse folgt
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigenden Flanke
	RatioNumerator	UINT	Getriebeübersetzungsverhältnis - Zähler "Leitachsgetriebe Ausgangsumdrehungen" min: 1; max: 65535; default: 1
	RatioDenominator	UINT	Getriebeübersetzungsverhältnis - Nenner "Leitachsgetriebe Eingangsumdrehungen" min: 1; max: 65535; default: 1
	MasterFineadjust	REAL	"Feinabgleich Getriebeübersetzung" in %; default: 0.0, min.: -799,9999%, max.: 799,9999

ML_PLCopen.library

	Name	Typ	Kommentar
	SyncMode	MC_SYNC_DIRECTION	Aufsynchronisier-Richtung: 0:= kürzeste Entfernung, 1:= positive Richtung, 2:= negative Richtung
	StartMode	MC_START_MODE	Art des Aufsynchronisierens: 0:= absolut, 1:= relative
VAR_OUTPUT	InSync	BOOL	Synchroner Lauf erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-50: Schnittstelle des FB MC_GearInPos

Signal-Zeit-Diagramm



In... InSync

Abb. 6-76: Signal-Zeit-Diagramm

6.6.20 MB_GearInPos, Besonderheiten bei Sercos basierten Achsen

Funktionsbeschreibung

Die Getriebefunktion wird mittels antriebsinterner Getriebefunktionalität realisiert oder auf der Steuerung gerechnet. Ziel ist ein phasen- und geschwindigkeitssynchroner (driftfreier) Lauf zwischen der Leitachse ("A-0-2700, ELS, Leitachse, Steuerungsnummer", "A-0-2701, ELS, Leitachse, Achsnummer") und der ausgewählten Folgeachse.



Die Funktionalität wird von folgenden Achstypen (Folgeachse) unterstützt:

- IndraDrive mit Interpolation im Antrieb und in der Steuerung
- HydraulicDrive mit Interpolation im Antrieb und in der Steuerung
- SercosDrive
- Virtuelle Achse
- Reglerachse

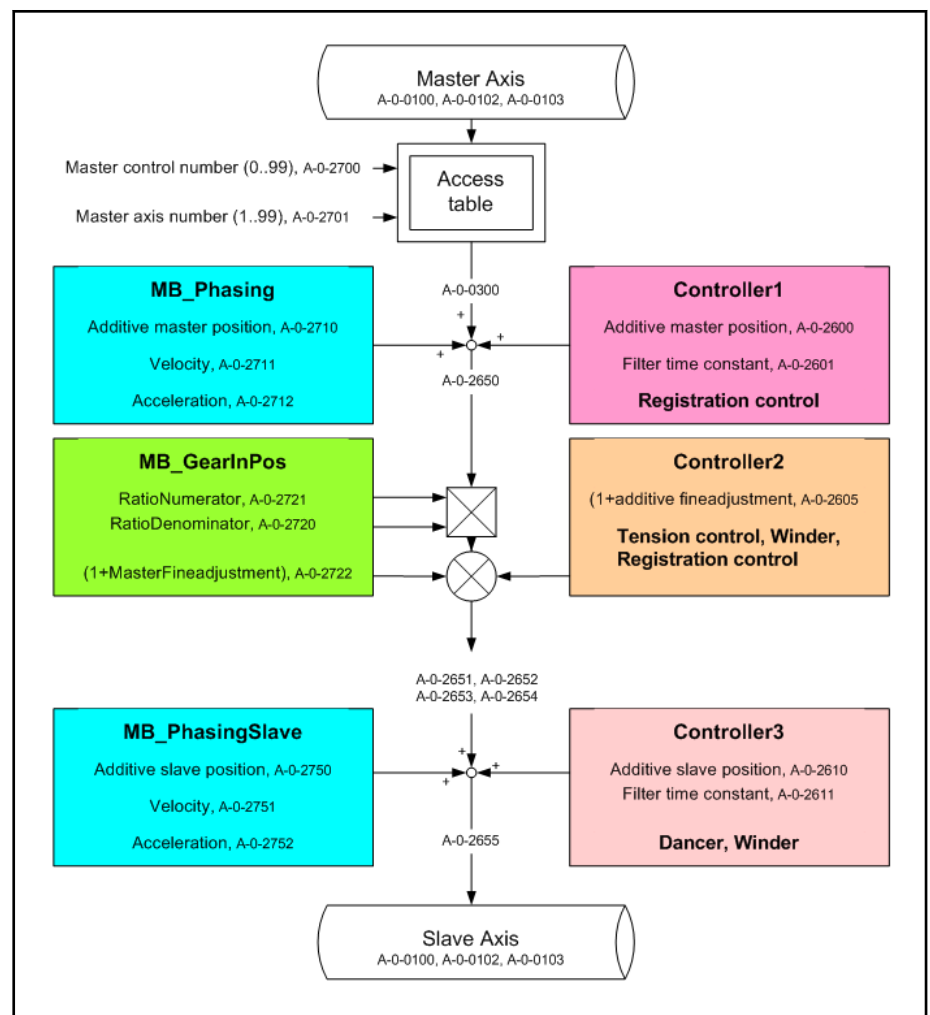


Abb. 6-77: MotionControl-Wirkungskette MB_GearInPos

Der Baustein enthält zwei funktionelle Gruppen:

- das elektronische Getriebe, welches definiert ist durch das Verhältnis der "A-0-2721, ELS, Leitachsgetriebe, Ausgangsumdrehungen" und der "A-0-2720, ELS, Leitachsgetriebe, Eingangsumdrehungen" und
- den "A-0-2722, ELS, Feinabgleich der Getriebeübersetzung". Der Wert des Feinabgleichs wird als %-Zahl vorgeben und als $(1 + \text{Feinabgleich})$ verrechnet.

Wird der Baustein im Antrieb (IndraDrive) gerechnet, werden die A-Parameter nachstehenden S- und P-Parametern übergeben:

- A-0-2721 nach P-0-0157,
- A-0-2720 nach P-0-0156,
- A-0-2722 nach P-0-0083.

Der Baustein ist in der Lage, die Synchronisation, ausgehend von den ankommenden Lage- und Geschwindigkeitssignalen, vor dem Einrasten von Leit- und Folgeachse, herbeizuführen.

Fehlerbehandlung

Der Funktionsbaustein MB_GearInPos erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse / die Steuerung.

ML_PLCopen.library

6.7 Funktionsbausteine zur Achsgruppensteuerung

6.7.1 MC_AddAxisToGroup

Kurzbeschreibung Der Funktionsbaustein MC_AddAxisToGroup fügt eine Achse zur Achsgruppe hinzu. Dieser Funktionsbaustein löst keine Bewegung aus. Die "Group State Machine" wechselt in den Zustand "GroupStandStill, no IRDProgram-Selected", wenn alle für die Gruppe notwendigen Achsen hinzugeführt werden.



Die Funktionsbausteine "MC_AddAxisToGroup" und "MC_RemoveAxisFromGroup" wurden von ML_Robot nach ML_PLCopen verlegt, um sie auch für IndraLogic XLC zugänglich zu machen.



Die erforderlichen Achsen der Gruppe sind im IndraWorks zu konfigurieren. Die Ergebnisse der Konfiguration werden im Parameter "K-0-0008, Achskonfigurationsliste" abgelegt.

Die Identifikation einer Achse innerhalb der Gruppe ist ebenfalls mit dem Parameter K-0-0008 gegeben.

Bibliothek	Bereich
ML_PLCopen	Motion

Tab. 6-51: Bibliothekszuordnung

Schnittstellenbeschreibung

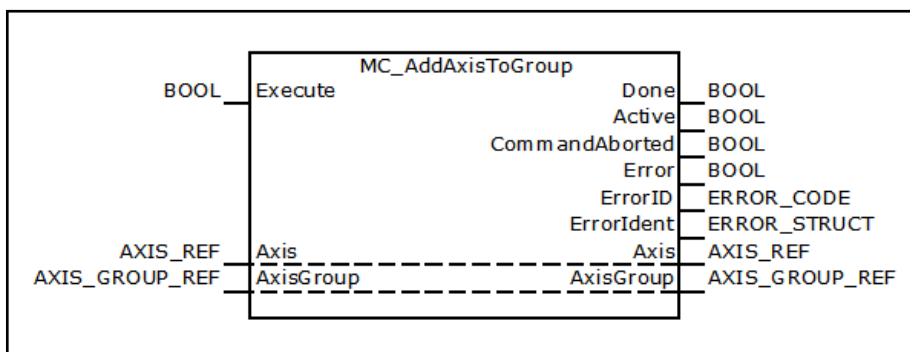


Abb. 6-78: MC_AddAxisToGroup

	Name	Typ	Kommentar
VAR_IN_OUT	AxisGroup	AXIS_GROUP_REF	Steuerungsnummer und Objektnummer der Achsgruppe
	Axis	AXIS_REF	Steuerungsnummer und Objektnummer der Achse
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
VAR_OUTPUT	Done	BOOL	Die Achse ist erfolgreich in die Achsgruppe aufgenommen worden
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch ein nachfolgendes Achskommando
	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

Tab. 6-52: Schnittstelle des FB MC_AddAxisToGroup

Signal-Zeit-Diagramm

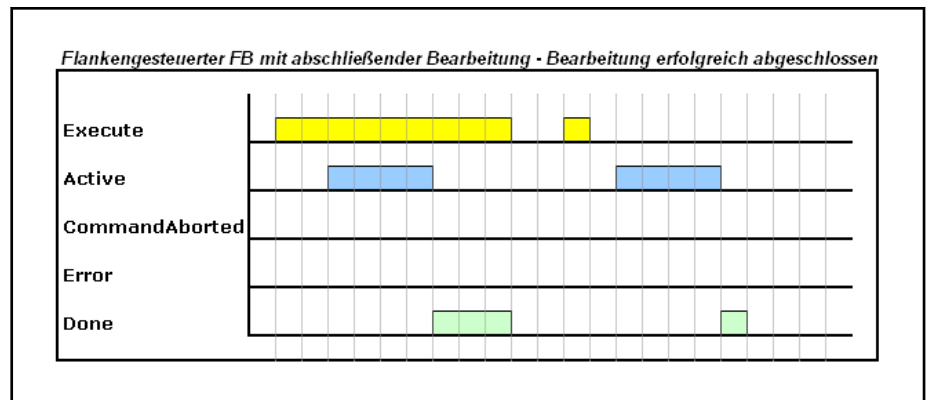


Abb. 6-79: Signal-Zeit-Diagramm: MC_AddAxisToGroup

6.7.2 MC_RemoveAxisFromGroup

Kurzbeschreibung

Der Funktionsbaustein MC_RemoveAxisFromGroup entfernt eine Achse aus der Achsgruppe. Dieser Funktionsbaustein löst keine Bewegung aus. Die "Group State Machine" wechselt in den Zustand "GroupUndefined".



Die Funktionsbausteine "MC_AddAxisToGroup" und "MC_RemoveAxisFromGroup" wurden von ML_Robot nach ML_PLCopen verlegt, um sie auch für IndraLogic XLC zugänglich zu machen.



Die erforderlichen Achsen der Gruppe sind über den Parameter "K-0-0008, Achskonfigurationsliste" abgelegt. Die Identifikation einer Achse innerhalb der Gruppe ist ebenfalls mit dem Parameter K-0-0008 gegeben.

Bibliothek	Bereich
ML_PLCopen	Motion

Tab. 6-53: Bibliothekszuordnung

Schnittstellenbeschreibung

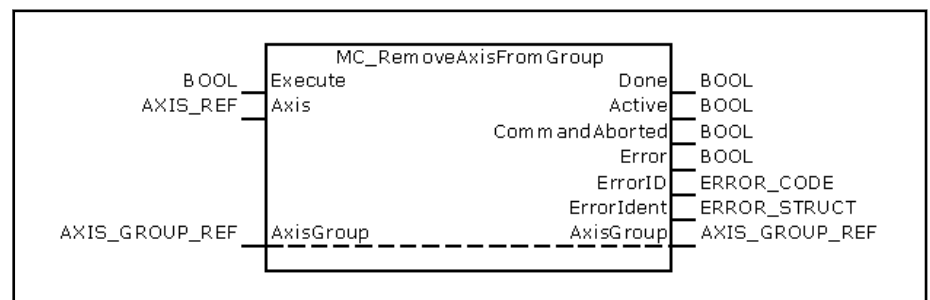


Abb. 6-80: MC_RemoveAxisFromGroup

	Name	Typ	Kommentar
VAR_IN_OUT	AxisGroup	AXIS_GROUP_REF	Steuerungsnummer und Objektzahl der Achsgruppe
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigender Flanke
	Axis	AXIS_REF	Steuerungsnummer und Objektzahl der Achse
VAR_OUTPUT	Done	BOOL	Die Achse ist erfolgreich aus der Achsgruppe entfernt worden
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft

ML_PLCopen.library

	Name	Typ	Kommentar
	Command-Aborted	BOOL	Kommandoabbruch durch ein nachfolgendes Achs-Kommando
	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

Tab. 6-54: Schnittstelle des FB MC_RemoveAxisFromGroup

Signal-Zeit-Diagramm

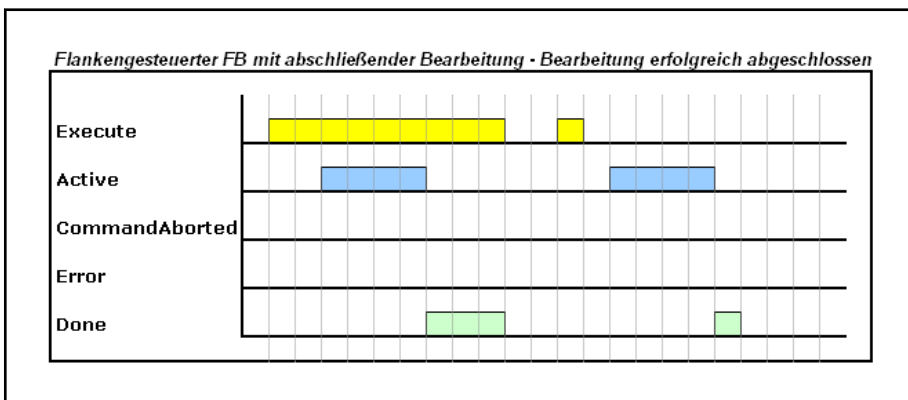


Abb. 6-81: Signal-Zeit-Diagramm: MC_RemoveAxisFromGroup

6.8 Funktionsbausteine zur Nutzung des zyklischen Datenkanals

6.8.1 Allgemeines zur Nutzung des zyklischen Datenkanals

Der sercos Bus bietet die Möglichkeit der Nutzung eines zyklischen Datenkanals zur Übertragung von Informationen von der Steuerung zu der jeweils adressierten Achse.

Dazu wird zunächst mit dem Baustein "ML_OpenCyclic***Channel" die adressierte Achse in die Betriebsart "Cyclic *** Command" geschaltet und der betreffende zyklische Sollwertkanal geöffnet.

Bei geöffnetem Sollwertkanal ist es dann möglich, mit dem Bausteinen "ML_WriteCyclic***" jeden Takt einen Sollwert für die Achse zu schreiben, welcher bereits im nächsten Zyklus auf der Achse wirksam ist.

Folgende Bausteine stehen zur Verfügung:

Analog-Sollwertkanal, Betriebsart "Cyclic Analog Command"

- [ML_OpenCyclicAnalogChannel](#), Seite 254, mit
- [ML_WriteCyclicAnalog](#), Seite 255,

Lage-Sollwertkanal, Betriebsart "Cyclic Position Command"

- [ML_OpenCyclicPositionChannel](#), Seite 256, mit
- [ML_WriteCyclicPosition](#), Seite 258,

Kraft-Sollwertkanal, Betriebsart "Cyclic Torque Command"

- [ML_OpenCyclicTorqueChannel](#), Seite 261, mit
- [ML_WriteCyclicTorque](#), Seite 262,

Geschwindigkeit-Sollwertkanal, Betriebsart "Cyclic Velocity Command"

- [ML_OpenCyclicVelocityChannel](#), Seite 256, mit
- [ML_WriteCyclicVelocity](#), Seite 264,

Jeweils zwei der Bausteine müssen zusammen verwendet werden. Folgendes Beispiel beschreibt, wie die Funktionsbausteine ML_OpenCyclicPositionChannel und ML_WriteCyclicPosition zusammen verwendet werden sollten.

Ziel ist es zyklisch aus der SPS Lage-Sollwerte auf eine Achse zu kommandieren. Zu Beginn ist es nicht bekannt, ob der zyklische Lagesollwertkanal bereits geöffnet ist, oder nicht. Die Funktionsbausteine ML_OpenCyclicPositionChannel und ML_WriteCyclicPosition werden in einer SPS-Task benutzt die synchron zur Motionberechnung läuft. Die Bausteine sollten wie folgt verschaltet werden:

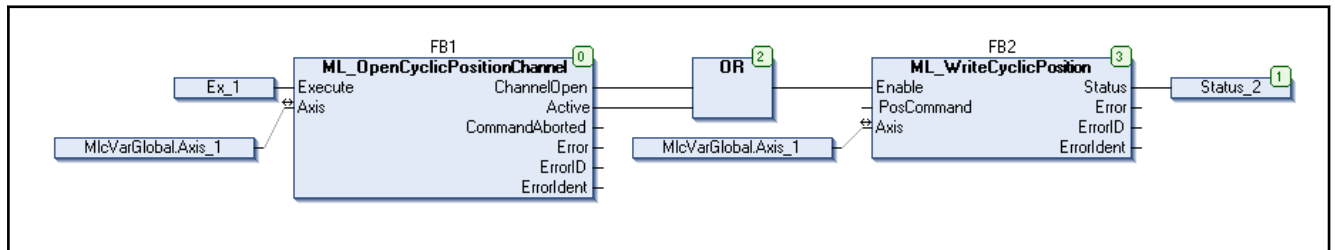


Abb. 6-82: Verschaltung der Funktionsbausteine ML_OpenCyclicPositionChannel und ML_WriteCyclicPosition

Erster Zyklus: Soll die eigene Sollwertvorgabe gestartet werden, muss der Execute-Eingang des Bausteins OpenCyclicTorqueChannel auf TRUE gesetzt werden. Wenn danach der Error-Ausgang auf TRUE ist, war der zyklische Positions-Sollwertkanal bereits von jemand anders geöffnet. In diesem Fall darf kein Positionssollwert über den Funktionsbaustein ML_WriteCyclicPosition geschrieben werden. Es würde der Sollwert des anderen Benutzers des zyklischen Kanals überschrieben werden. Wenn Active auf TRUE ist, muss der erste Sollwert mit dem Baustein ML_WriteCyclicPosition geschrieben werden. Die Ausgänge CommandAborted und ChannelOpen können im ersten Takt nicht TRUE werden.

Aktivierungszyklen: Während der zyklische Sollwertkanal geöffnet wird, ist der Ausgang ChannelOpen auf FALSE. Wird während dessen versucht den zyklischen Sollwertkanal von jemand anders zu öffnen, wird der Ausgang Error TRUE. Wird der Ausgang CommandAborted TRUE, wurde das Öffnen des zyklischen Kanals durch ein anderes Motion Kommando abgebrochen. Während dieser Zeit muss bereits der Sollwert geschrieben werden, der aktiv werden soll, wenn der Kanal geöffnet wurde.

Arbeitszyklen: Während der zyklische Sollwertkanal geöffnet ist, ist der Ausgang ChannelOpen auf TRUE. Es muss der Achse ein zyklischer Lagesollwert über den Funktionsbaustein ML_WriteCyclicPosition geschrieben werden.

Schließen des Sollwertkanals: Ein zyklischer Kanal kann nicht explizit geschlossen werden. Er kann nur durch andere Motion-Funktionsbaustein abgebrochen, und somit geschlossen werden (z.B. MC_Stop). Sobald der Ausgang CommandAborted am Funktionsbaustein ML_OpenCyclicPositionChannel TRUE ist, darf kein zyklischer Lagesollwert mehr über den Funktionsbaustein ML_WriteCyclicPosition geschrieben werden. Der Execute-Eingang am Funktionsbaustein ML_OpenCyclicPositionChannel kann auf FALSE gesetzt werden.

Zusammenfassung:

Ausgang	Beschreibung/Aktion
Active	Der zyklische Sollwertkanal wird geöffnet. Der zyklische Lagesollwert muss über den Funktionsbaustein ML_WriteCyclicPosition geschrieben werden.
ChannelOpen	Der zyklische Sollwertkanal ist geöffnet. Der zyklische Lagesollwert muss über den Funktionsbaustein ML_WriteCyclicPosition geschrieben werden.
Error	Der zyklische Sollwertkanal wird von jemand anderem benutzt. Es darf kein Sollwert über den Funktionsbaustein ML_WriteCyclicPosition geschrieben werden.
CommandAborted	Der zyklische Sollwertkanal wurde geschlossen. Es darf kein Sollwert über den Funktionsbaustein ML_WriteCyclicPosition geschrieben werden.

Tab. 6-55: Zusammenfassung Ausgänge ML_OpenCyclicPositionChannel
Die Funktionsbausteine für den zyklischen Geschwindigkeits-/Kraft- und Analog-Kanal können analog verwendet werden.

6.8.2 ML_OpenCyclicAnalogChannel

Kurzbeschreibung Der Funktionsbaustein schaltet die Achse in den Modus 'Cyclic Analog Command' und öffnet den zyklischen Analog-Sollwertkanal.



Wenn ML_OpenCyclicAnalogChannel aktiviert wird und der zyklische Analog-Sollwertkanal ist bereits geöffnet, so wird ein Fehler generiert.

Der Funktionsbaustein muss im MotionTask verwendet werden (synchron zum sercos Takt), ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich

Wenn der "Execute"-Eingang deaktiviert wird, werden alle Ausgänge auf FALSE gesetzt. Der zyklische Analog-Sollwertkanal bleibt geöffnet, wenn er zuvor geöffnet war.

Wenn der Funktionsbaustein durch einen anderen Motion-Funktionsbaustein unterbrochen wurde, wird der Funktionsbaustein abgebrochen und der zyklische Analog-Sollwertkanal geschlossen.

Der Funktionsbaustein steht nur bei der realen Achse vom Typ HNC100-3X zur Verfügung

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-56: Bibliothekszuordnung

Schnittstellenbeschreibung

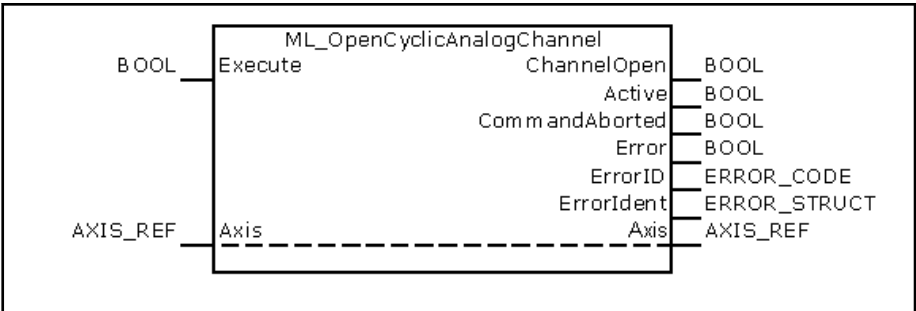


Abb. 6-83: FB ML_OpenCyclicAnalogChannel

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
VAR_OUTPUT	ChannelOpen	BOOL	Indiziert, dass der zyklische Analog-Sollwertkanal geöffnet ist, und die Achse dem Analogwert des letzten Zyklus folgt.
	Active	BOOL	Indiziert, dass der FB die Achse in den Modus 'Cyclic Analog Command' bringt.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	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

Tab. 6-57: Schnittstelle von FB ML_OpenCyclicAnalogChannel

Fehlerbehandlung Der Funktionsbaustein ML_OpenCyclicAnalogChannel erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.8.3 ML_WriteCyclicAnalog

Kurzbeschreibung Der Funktionsblock ML_WriteCyclicAnalog schreibt ein Analogwert für eine Achse. Nur wenn der zyklische Analog-Sollwertkanal geöffnet ist, wird der Analogwert im nächsten Zyklus auf der Achse wirksam.



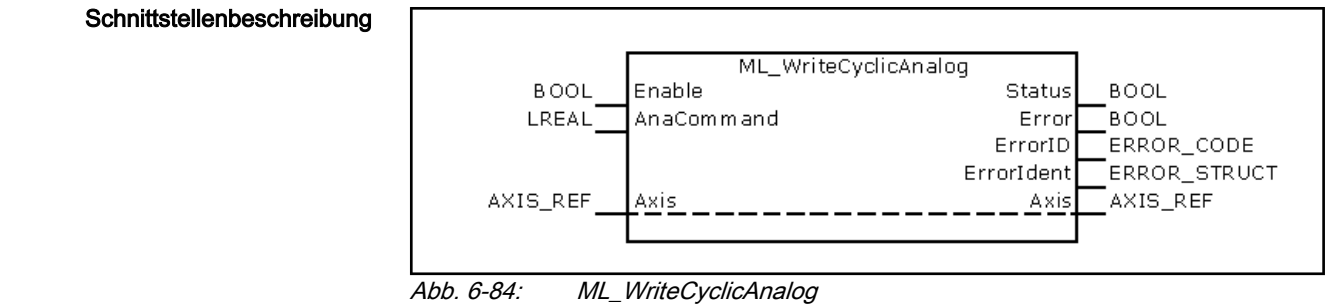
Um den zyklischen Analog-Sollwertkanal zu öffnen, muss der Funktionsbaustein [ML_OpenCyclicAnalogChannel, Seite 256](#), benutzt werden.

Der Funktionsbaustein muss in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-58: Bibliothekszuordnung

ML_PLCopen.library



	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, wird "AnaCommand" auf die Achse geschrieben. Wirksam wird der Analogwert erst, wenn der zyklische Analog-Sollwertkanal geöffnet ist.
	AnaCommand	LREAL	gibt den Analogwert an, die die Achse im nächsten Zyklus anfahren soll. Wirksam wird der Analogwert erst, wenn der zyklische Analog-Sollwertkanal geöffnet ist.
VAR_OUTPUT	Status	BOOL	zeigt an, dass der Analogwert auf die Achse geschrieben wurde. Wirksam wird der Analogwert erst, wenn der zyklische Analog-Sollwertkanal geöffnet ist.
	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

Tab. 6-59: Schnittstelle des FB ML_WriteCyclicAnalog

Besonderheiten des AnaCommand-Eingangs	1. Die Werte des "AnaCommand"-Eingangs werden im nächsten Zyklus wirksam, wenn der zyklische Analog-Sollwertkanal geöffnet ist. 2. Der Wert des "AnaCommand"-Eingangs hat die Skalierung von der zugeordneten Achse.
Fehlerbehandlung	Der Funktionsbaustein ML_WriteCyclicAnalog erzeugt folgende Fehlermeldungen in: MLC_TABLE, Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.8.4 ML_OpenCyclicPositionChannel

Kurzbeschreibung	Der Funktionsbaustein schaltet die Achse in den Modus 'Cyclic Position Command' und öffnet den zyklischen Lagesollwertkanal.



Wenn ML_OpenCyclicPositionChannel aktiviert wird und der zyklische Lagesollwertkanal ist bereits geöffnet, so wird ein Fehler generiert.

Der Funktionsbaustein muss im MotionTask verwendet werden (synchron zum sercos Takt), ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich

Wenn der "Execute"-Eingang deaktiviert wird, werden alle Ausgänge auf FALSE gesetzt. Der zyklische Lagesollwertkanal bleibt geöffnet, wenn er zuvor geöffnet war.

Wenn der Funktionsbaustein durch einen anderen Motion-Funktionsbaustein unterbrochen wurde, wird der Funktionsbaustein abgebrochen und der zyklische Lagesollwertkanal geschlossen (siehe auch Signal-Zeitdiagramm).

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-60: Bibliothekszuordnung

Schnittstellenbeschreibung

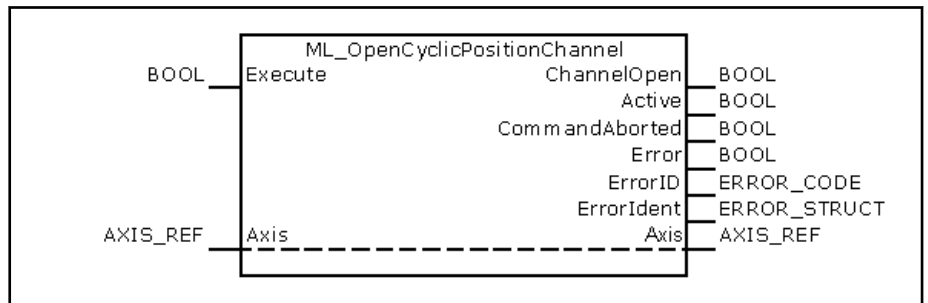


Abb. 6-85: FB ML_OpenCyclicPositionChannel

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
VAR_OUTPUT	ChannelOpen	BOOL	Indiziert, dass der zyklische Lagesollwertkanal geöffnet ist, und die Achse dem Positionswert des letzten Zyklus folgt.
	Active	BOOL	Indiziert, dass der FB die Achse in den Modus 'Cyclic Position Command' bringt.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	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

Tab. 6-61: Schnittstelle von FB ML_OpenCyclicPositionChannel

Signal-Zeit-Diagramm

Die Arbeitsweise des Bausteines wird deutlich, wenn man nachstehendes Impulsdiagramm betrachtet. Der Baustein arbeitet hier in Konkurrenz zu einer Instanz eines [MC_MoveAbsolute](#), Seite 180.

ML_PLCopen.library

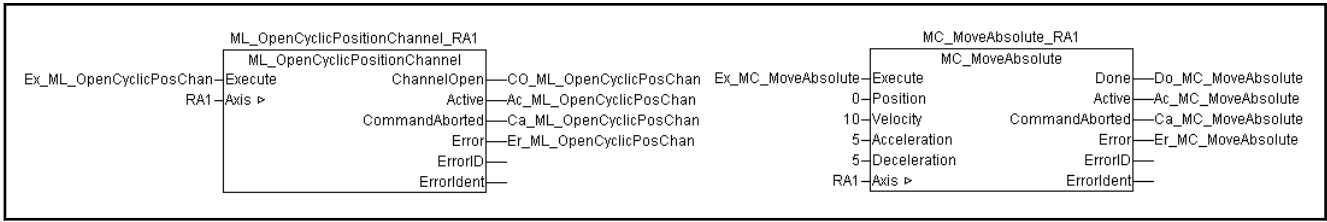


Abb. 6-86: Zeitverhalten ML_OpenCyclicPositionChannel

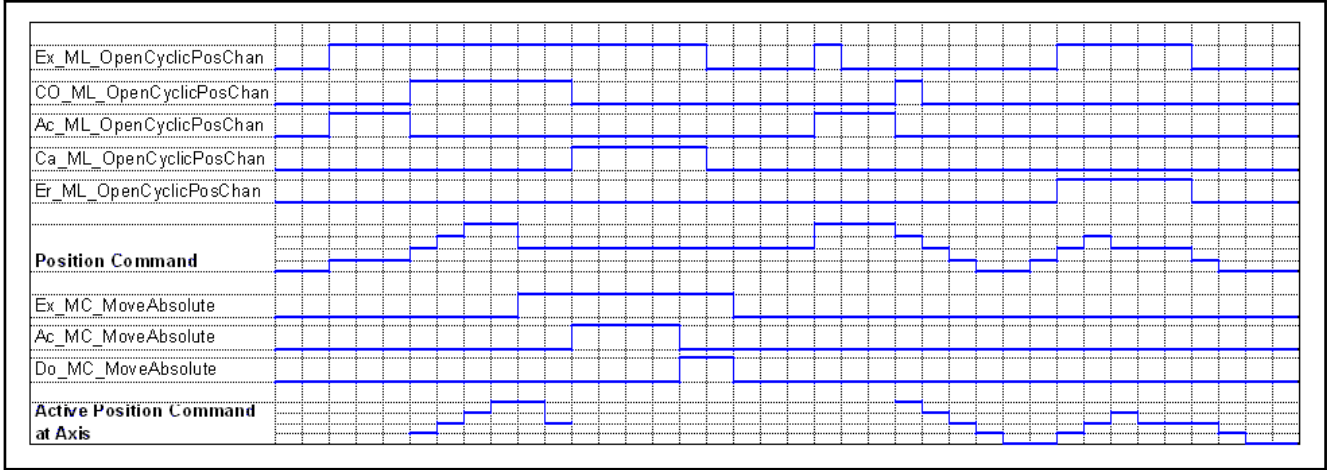


Abb. 6-87: Zeitverhalten ML_OpenCyclicPositionChannel

- Fehlerbehandlung** Der Funktionsbaustein ML_OpenCyclicPositionChannel erzeugt folgende Fehlermeldungen:
- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.8.5 ML_WriteCyclicPosition

Kurzbeschreibung Der Funktionsblock ML_WriteCyclicPosition schreibt ein Positionskommando für eine Achse. Nur wenn der zyklische Lagesollwertkanal geöffnet ist, wird das Positionskommando im nächsten Zyklus auf der Achse wirksam.

 Um den zyklischen Lagesollwertkanal zu öffnen, muss der Funktionsbaustein [ML_OpenCyclicPositionChannel, Seite 256](#), benutzt werden.

Der Funktionsbaustein muss in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-62: Bibliothekszuordnung

Schnittstellenbeschreibung

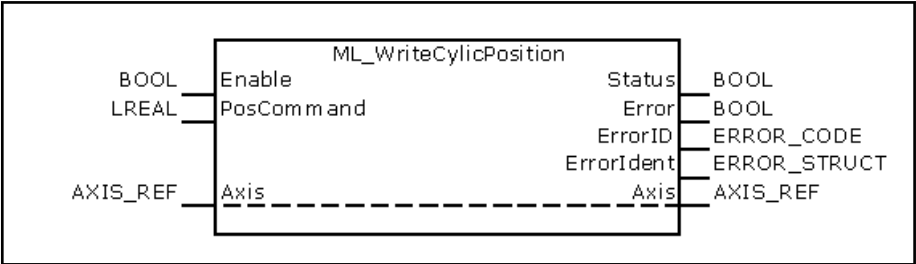


Abb. 6-88: ML_WriteCyclicPosition

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, wird "PosCommand" auf die Achse geschrieben. Wirksam wird das Positionskommando erst, wenn der zyklische Lagesollwertkanal geöffnet ist.
	PosCommand	LREAL	gibt die Position an, die die Achse im nächsten Zyklus anfahren soll. Wirksam wird das Positionskommando erst, wenn der zyklische Lagesollwertkanal geöffnet ist.
VAR_OUTPUT	Status	BOOL	zeigt an, dass das Positionskommando auf die Achse geschrieben wurde. Wirksam wird das Positionskommando erst, wenn der zyklische Lagesollwertkanal geöffnet ist.
	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

Tab. 6-63: Schnittstelle des FB ML_WriteCyclicPosition

Besonderheiten des PosCommand-Eingangs

1. Die Werte des "PosCommand"-Eingangs werden im nächsten Zyklus wirksam, wenn der zyklische Lagesollwertkanal geöffnet ist.
2. Die Werte des "PosCommand"-Eingangs sollten sich nahe an der nächsten Position des Bewegungsvektors befinden.
3. Der Wert des "PosCommand"-Eingangs hat die Skalierung von der zugeordneten Achse (linear, rotatorisch, modulo).

Fehlerbehandlung

Der Funktionsbaustein ML_WriteCyclicPosition erzeugt folgende Fehlermeldungen in:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

Implementationsbeispiel:**Gemeinsame Benutzung von ML_OpenCyclicPositionChannel und ML_WriteCyclicPosition**

Der "ChannelOpen"-Ausgang ODER-verknüpft mit dem "Active"-Ausgang vom FB1 kann als "Enable"-Eingang vom FB2 benutzt werden. Der "PosCommand"-Eingang vom FB2 muss mit gültigen Werten aus der Positionstabelle belegt werden: Der erste Wert aus der Positionstabelle muss auf den "PosCommand"-Eingang geschrieben werden, solange der "Active"-Ausgang vom FB1 gesetzt ist. Nachfolgende Einträge aus der Positionstabelle müssen auf den "PosCommand"-Eingang geschrieben werden, solange der "ChannelOpen"-Ausgang vom FB1 gesetzt ist.

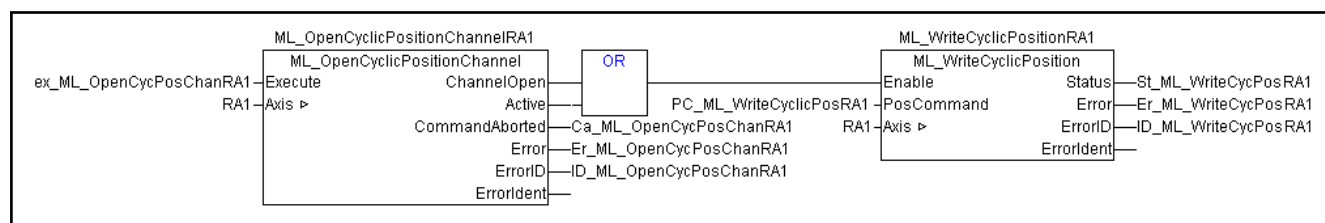


Abb. 6-89: Bsp. ML_OpenCyclicPositionChannel und ML_WriteCyclicPosition

Erster Zyklus:

- Der Execute-Eingang wird am FB ML_OpenCyclicPositionChannel gesetzt.

ML_PLCopen.library

- Wenn danach der "Error"-Ausgang gesetzt ist, war der zyklische Lagesollwertkanal bereits geöffnet und wird von einer anderen Bausteininstanz genutzt.
- Wenn danach der "Active"-Ausgang gesetzt ist, muss am FB ML_WriteCyclicPosition der "Enable"-Eingang gesetzt werden und der "PosCommand"-Eingang auf den ersten Eintrag der Positionstabelle gesetzt werden.

Der "CommandAborted"- und "ChannelOpen"-Ausgang können im ersten Zyklus nicht gesetzt sein.

Aktivierungszyklen ("ChannelOpen" ist nicht gesetzt)

- Wenn der "Error"-Ausgang gesetzt ist, wurde der zyklische Lagesollwertkanal von jemand anderem während der Initialisierungsphase geöffnet. Der "Enable"-Eingang am FB ML_WriteCyclicPosition kann deaktiviert werden.
- Wenn der "CommandAborted"-Ausgang gesetzt ist, wurde ML_OpenCyclicPositionCommand durch einen anderen Bewegungs-Funktionsblock unterbrochen. Der "Enable"-Eingang am FB ML_OpenCyclicPositionCommand kann deaktiviert werden.
- Wenn der "Active"-Ausgang gesetzt ist, müssen am FB ML_WriteCyclicPosition der "Enable"-Eingang gesetzt und der "PosCommand"-Eingang auf den ersten Eintrag der Positionstabelle gesetzt werden.

Arbeitszyklen

- "ChannelOpen"-Ausgang ist gesetzt. Der Wert des "PosCommand"-Eingangs am FB ML_OpenCyclicPositionCommand vom letzten Zyklus ist an der Achse wirksam. Am FB ML_WriteCyclicPosition müssen der "Enable"-Eingang gesetzt und der PosCommand-Eingang auf den nachfolgenden Eintrag der Positionstabelle gesetzt werden.

Abbruch

- Der zyklische Lagesollwertkanal wird geschlossen, wenn ein anderer Bewegungsfunktionsblock (z.B. MC_Stop) auf der Achse aktiv wird.
- Wenn der "CommandAborted"-Ausgang aktiv ist, wurde der zyklische Lagesollwertkanal geschlossen. In diesem Zyklus wird kein Positionskommando am Antrieb wirksam. Der "Enable"-Eingang am FB ML_WriteCyclicPosition kann deaktiviert werden. Der "Execute"-Eingang am FB ML_OpenCyclicPositionCommand kann deaktiviert werden.

Zusammenfassung

Ausgang	Beschreibung/Aktion
Active	Der zyklische Lagesollwertkanal wird initialisiert. Am FB ML_WriteCyclicPosition müssen der "Enable"-Eingang aktiviert und der "PosCommand"-Eingang auf den ersten Eintrag der Positionstabelle gesetzt werden.
ChannelOpen	Der zyklische Lagesollwertkanal ist geöffnet. Am FB ML_WriteCyclicPosition müssen der "Enable"-Eingang aktiviert und der "Pos-Command"-Eingang auf den nachfolgenden Eintrag der Positionstabelle gesetzt werden.
Error	Der zyklische Lagesollwertkanal war bereits geöffnet und wird von jemand anderem benutzt. Der "Enable"-Eingang am FB ML_WriteCyclicPosition muss deaktiviert werden.
CommandAborted	Der zyklische Lagesollwertkanal wurde geschlossen. Der "Enable"-Eingang am FB ML_WriteCyclicPosition kann deaktiviert werden.

Tab. 6-64: Zusammenfassung

6.8.6 ML_OpenCyclicTorqueChannel

Kurzbeschreibung Der Funktionsbaustein schaltet die Achse in den Modus 'Cyclic Torque Command' und öffnet den zyklischen Kraft-Sollwertkanal.



Wenn ML_OpenCyclicTorqueChannel aktiviert wird und der zyklische Kraft-Sollwertkanal ist bereits geöffnet, so wird ein Fehler generiert.

Der Funktionsbaustein muss im MotionTask verwendet werden (synchron zum sercos Takt), ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich

Wenn der "Execute"-Eingang deaktiviert wird, werden alle Ausgänge auf FALSE gesetzt. Der zyklische Kraft-Sollwertkanal bleibt geöffnet, wenn er zuvor geöffnet war.

Wenn der Funktionsbaustein durch einen anderen Motion-Funktionsbaustein unterbrochen wurde, wird der Funktionsbaustein abgebrochen und der zyklische Kraft-Sollwertkanal geschlossen.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-65: Bibliothekszuordnung

Schnittstellenbeschreibung

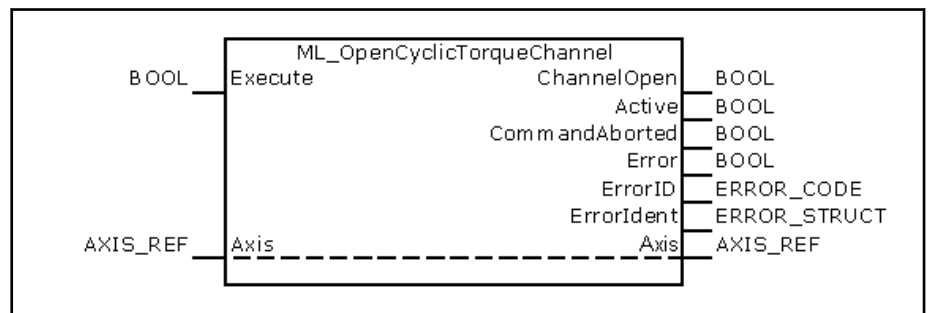


Abb. 6-90: FB ML_OpenCyclicTorqueChannel

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
VAR_OUTPUT	ChannelOpen	BOOL	Indiziert, dass der zyklische Kraft-Sollwertkanal geöffnet ist, und die Achse dem Kraftwert des letzten Zyklus folgt.
	Active	BOOL	Indiziert, dass der FB die Achse in den Modus 'Cyclic Torque Command' bringt.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	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

Tab. 6-66: Schnittstelle von FB ML_OpenCyclicKraftChannel

Fehlerbehandlung Der Funktionsbaustein ML_OpenCyclicTorqueChannel erzeugt folgende Fehlermeldungen:

ML_PLCopen.library

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.8.7 **ML_WriteCyclicTorque**

Kurzbeschreibung Der Funktionsblock ML_WriteCyclicTorque schreibt ein Kraftwert für eine Achse. Nur wenn der zyklische Kraft-Sollwertkanal geöffnet ist, wird der Kraftwert im nächsten Zyklus auf der Achse wirksam.



Um den zyklischen Kraft-Sollwertkanal zu öffnen, muss der Funktionsbaustein [ML_OpenCyclicTorqueChannel, Seite 256](#), benutzt werden.

Der Funktionsbaustein muss in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-67: Bibliothekszuordnung

Schnittstellenbeschreibung

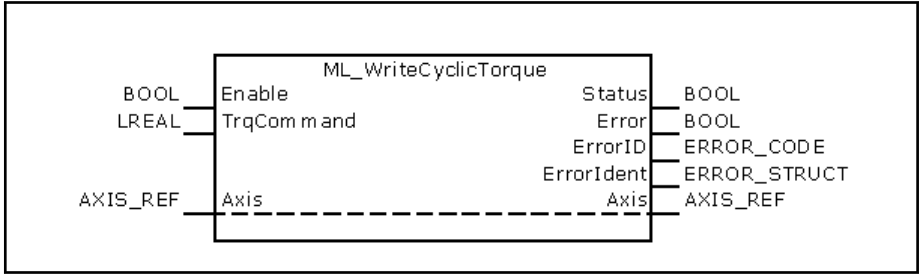


Abb. 6-91: ML_WriteCyclicTorque

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, wird "TrqCommand" auf die Achse geschrieben. Wirksam wird der Kraftwert erst wenn der zyklische Kraft-Sollwertkanal geöffnet ist.
	TrqCommand	LREAL	gibt den Kraftwert an, die die Achse im nächsten Zyklus anfahren soll. Wirksam wird der Kraftwert erst wenn der zyklische Kraft-Sollwertkanal geöffnet ist.
VAR_OUTPUT	Status	BOOL	zeigt an, dass der Kraftwert auf die Achse geschrieben wurde. Wirksam wird der Kraftwert erst wenn der zyklische Kraft-Sollwertkanal geöffnet ist.
	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

Tab. 6-68: Schnittstelle des FB ML_WriteCyclicTorque

Besonderheiten des TrqCommand-Eingangs

1. Die Werte des "TrqCommand"-Eingangs werden im nächsten Zyklus wirksam, wenn der zyklische Kraft-Sollwertkanal geöffnet ist.
2. Der Wert des "TrqCommand"-Eingangs hat die Skalierung von der zugeordneten Achse.

- Fehlerbehandlung** Der Funktionsbaustein ML_WriteCyclicTorque erzeugt folgende Fehlermeldungen in:
- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.8.8 ML_OpenCyclicVelocityChannel

Kurzbeschreibung Der Funktionsbaustein schaltet die Achse in den Modus 'Cyclic Velocity Command' und öffnet den zyklischen Geschwindigkeits-Sollwertkanal.



Wenn ML_OpenCyclicVelocityChannel aktiviert wird und der zyklische Geschwindigkeits-Sollwertkanal ist bereits geöffnet, so wird ein Fehler generiert.

Der Funktionsbaustein muss im MotionTask verwendet werden (synchron zum sercos Takt), ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich

Wenn der "Execute"-Eingang deaktiviert wird, werden alle Ausgänge auf FALSE gesetzt. Der zyklische Geschwindigkeits-Sollwertkanal bleibt geöffnet, wenn er zuvor geöffnet war.

Wenn der Funktionsbaustein durch einen anderen Motion-Funktionsbaustein unterbrochen wurde, wird der Funktionsbaustein abgebrochen und der zyklische Geschwindigkeits-Sollwertkanal geschlossen.

Bibliothek	Bereich
ML_PLCopen.library	Motion

Tab. 6-69: Bibliothekszuordnung

Schnittstellenbeschreibung

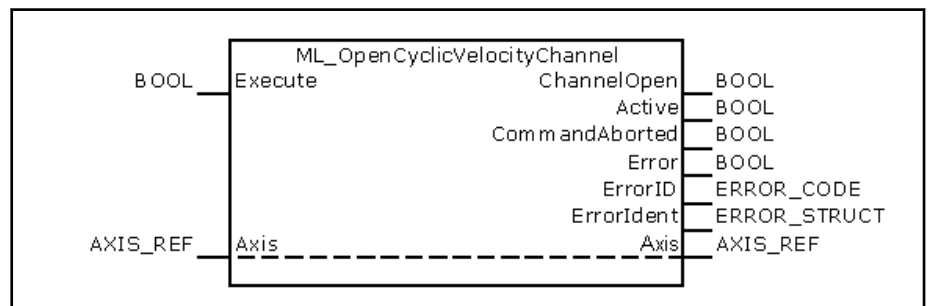


Abb. 6-92: FB ML_OpenCyclicVelocityChannel

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
VAR_OUTPUT	ChannelOpen	BOOL	Indiziert, dass der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist, und die Achse dem Geschwindigkeitswert des letzten Zyklus folgt.
	Active	BOOL	Indiziert, dass der FB die Achse in den Modus 'Cyclic Velocity Command' bringt.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist

ML_PLCOpen.library

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 6-70: Schnittstelle von FB ML_OpenCyclicVelocityChannel

Fehlerbehandlung Der Funktionsbaustein ML_OpenCyclicVelocityChannel erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.8.9 ML_WriteCyclicVelocity

Kurzbeschreibung Der Funktionsblock ML_WriteCyclicVelocity schreibt ein Geschwindigkeitswert für eine Achse. Nur wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist, wird der Geschwindigkeitswert im nächsten Zyklus auf der Achse wirksam.



Um den zyklischen Geschwindigkeits-Sollwertkanal zu öffnen, muss der Funktionsbaustein [ML_OpenCyclicVelocityChannel, Seite 256](#), benutzt werden.

Der Funktionsbaustein muss in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Bibliothek	Bereich
ML_PLCOpen.library	Motion

Tab. 6-71: Bibliothekszuordnung

Schnittstellenbeschreibung

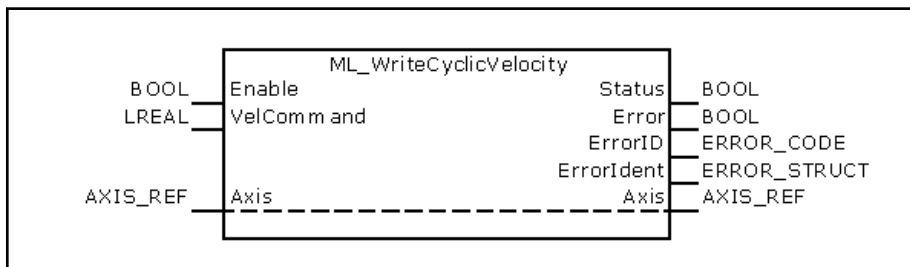


Abb. 6-93: ML_WriteCyclicVelocity

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, wird "VelCommand" auf die Achse geschrieben. Wirksam wird der Geschwindigkeitswert erst wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist.
	VelCommand	LREAL	gibt den Geschwindigkeitswert an, die die Achse im nächsten Zyklus anfahren soll. Wirksam wird der Geschwindigkeitswert erst wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist.
VAR_OUTPUT	Status	BOOL	zeigt an, dass der Geschwindigkeitswert auf die Achse geschrieben wurde. Wirksam wird der Geschwindigkeitswert erst wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist.
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 6-72: Schnittstelle des FB ML_WriteCyclicVelocity

Besonderheiten des VelCommand-Eingangs

1. Die Werte des "VelCommand"-Eingangs werden im nächsten Zyklus wirksam, wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist.
2. Der Wert des "VelCommand"-Eingangs hat die Skalierung von der zugeordneten Achse.

Fehlerbehandlung

Der Funktionsbaustein ML_WriteCyclicVelocity erzeugt folgende Fehlermeldungen in:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9 Funktionsbausteine zur geregelten Bewegung

6.9.1 Allgemeines zur geregelten Bewegung

Dieses Kapitel beschreibt die Möglichkeit, die zyklische Sollwertvorgabe einer Achse unter vollständige Kontrolle der SPS zu stellen. Dies wird oft benötigt, um achsübergreifende Regelalgorithmen, wie z.B. Gantry- oder Gleichlauf-Verbünde, zu implementieren. Während dieser Sollwertvorgabe ist es nicht erlaubt, die Achse über andere Funktionsbausteine, wie z.B. MC_MoveAbsolute, zu kommandieren.

Die geregelte Bewegung wird mit dem Funktionsbaustein ML_ControlOn gestartet. Dieser Funktionsbaustein darf aus den Zuständen 'Standstill', 'Continuous Motion', 'Discrete Motion' und 'Synchronized Motion' aufgerufen werden. Bei Aufruf des Funktionsbausteins wechselt die Achse in den Zustand 'Controlled Motion'. Mit dem Funktionsbaustein ML_ControlOff wird der Zustand verlassen. Es findet ein Wechsel in den Zustand 'Continuous Motion' statt.

ML_PLCopen.library

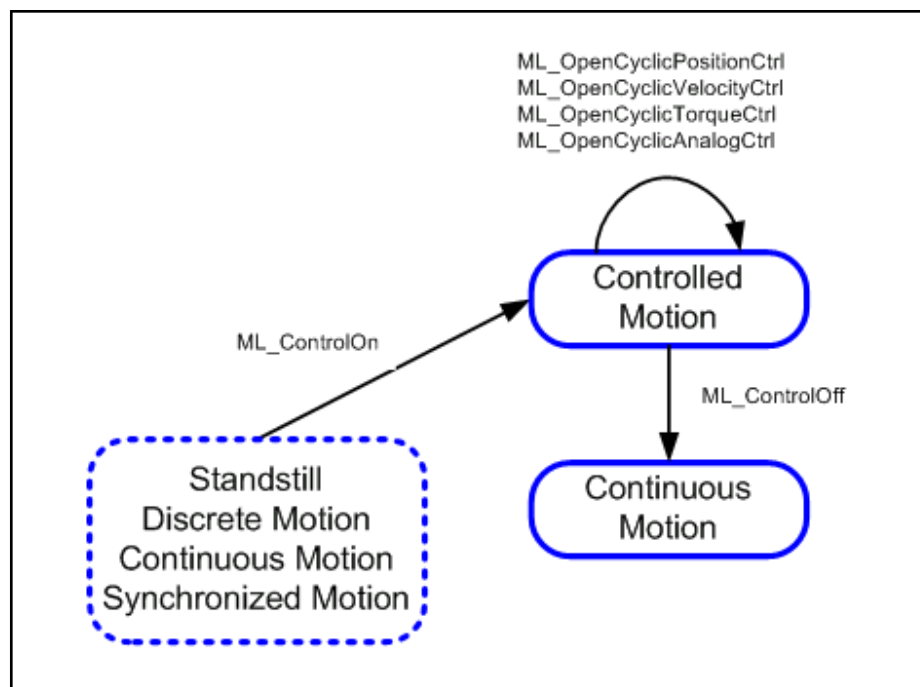


Abb. 6-94: Einbindung des Zustands 'Controlled Motion' in die PLCopen Zustandsmaschine

Während eine Achse sich in dem Zustand 'Controlled Motion' befindet, sind nur Motion Funktionsbausteine vom Typ `ML_OpenCyclic*Ctrl` erlaubt.

Folgende Bausteine stehen zur Verfügung:

Funktionsbausteine zum Betreten und Verlassen der geregelten Bewegung

- [ML_ControlOn](#), Seite 266
- [ML_ControlOff](#), Seite 267

Funktionsbaustein zum Schreiben der aktuellen Werte eines Reglers der SPS in die Reglerachse der MLC

- [ML_ControlWriteActualData](#), Seite 268

Analog-Sollwertkanal, Betriebsart "Cyclic Analog Command"

- [ML_OpenCyclicAnalogChannelCtrl](#), Seite 269
- [ML_WriteCyclicAnalogCtrl](#), Seite 270

Lage-Sollwertkanal, Betriebsart "Cyclic Position Command"

- [ML_OpenCyclicPositionChannelCtrl](#), Seite 271
- [ML_WriteCyclicPositionCtrl](#), Seite 272

Kraft-Sollwertkanal, Betriebsart "Cyclic Torque Command"

- [ML_OpenCyclicTorqueChannelCtrl](#), Seite 274
- [ML_WriteCyclicTorqueCtrl](#), Seite 275

Geschwindigkeit-Sollwertkanal, Betriebsart "Cyclic Velocity Command"

- [ML_OpenCyclicVelocityChannelCtrl](#), Seite 276
- [ML_WriteCyclicVelocityCtrl](#), Seite 277

Je nach Achstyp können die unterstützten Funktionsbausteine variieren.

6.9.2 ML_ControlOn

Kurzbeschreibung

Der Funktionsbaustein `ML_ControlOn` wird dazu verwendet, die zyklische Sollwertvorgabe einer Achse unter vollständige Kontrolle der SPS zu stellen.

Bei Aktivieren des Funktionsbausteins findet ein Zustandswechsel der Achse in den Zustand 'Controlled Motion' statt. Der Zustand 'Controlled Motion' kann mittels des Funktionsbausteins ML_ControlOff verlassen werden.



Wenn ML_ControlOn aktiviert wird bleibt das letzte Motion Kommando noch so lange wirksam bis ein Funktionsbaustein des Typs ML_OpenCyclic*Ctrl ausgeführt wurde.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-73: Bibliothekszuordnung

Schnittstellenbeschreibung

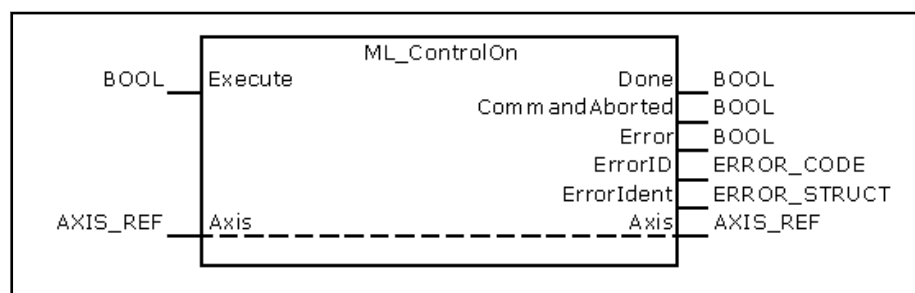


Abb. 6-95: FB ML_ControlOn

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
VAR_OUTPUT	Done	BOOL	Kommandoausführung erfolgreich. Die Achse ist in den Zustand 'Controlled Motion' gewechselt.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	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

Tab. 6-74: Schnittstelle von FB ML_ControlOn

Fehlerbehandlung

Der Funktionsbaustein ML_ControlOn erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.3 ML_ControlOff

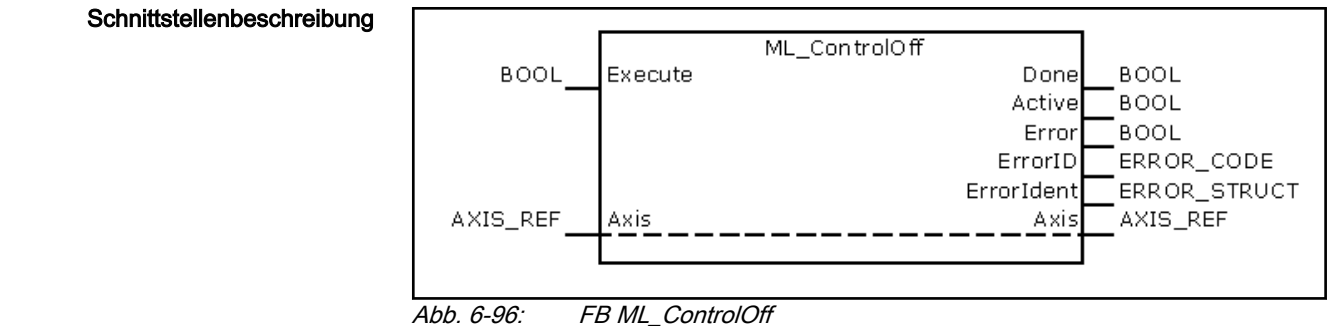
Kurzbeschreibung

Der Funktionsbaustein ML_ControlOff wird dazu verwendet, die zyklische Kontrolle einer Achse aus der SPS zu beenden. Bei Ausführung des Kommandos verlässt die Achse den Zustand 'Controlled Motion' und wird in den Zustand 'Continuous Motion' überführt. Der letzte aktive Sollwert wird beibehalten bis ein neues Motion-Kommando abgesetzt wird.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-75: Bibliothekszuordnung

ML_PLCopen.library



	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
VAR_OUTPUT	Done	BOOL	Kommandoausführung erfolgreich. Die Achse hat den Zustand 'Controlled Motion' verlassen.
	Active	BOOL	Zeigt an, dass der Abschaltvorgang 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

Tab. 6-76: Schnittstelle von FB ML_ControlOff

Fehlerbehandlung

Der Funktionsbaustein ML_ControlOff erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.4 ML_ControlWriteActualData

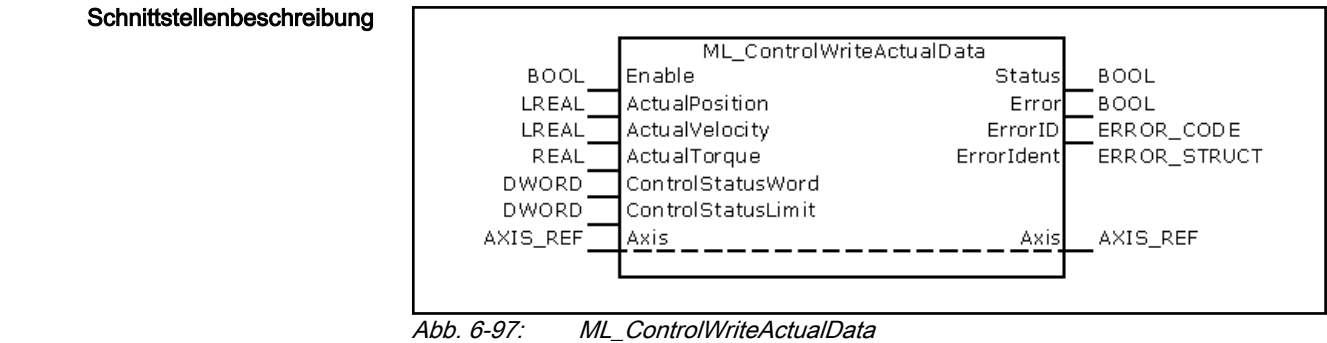
Kurzbeschreibung

Der Funktionsbaustein ML_ControlWriteActualData schreibt die aktuellen Werte eines Reglers der SPS in die Reglerachse der MLC.

Dieser Baustein verhält sich bezüglich des Error-Ausgangs nicht normgerecht. Das bedeutet, dass im Falle eines Fehlers der Error-Ausgang auf TRUE gesetzt wird. Wird die Fehlerursache behoben, wechselt der Error-Ausgang automatisch von TRUE auf FALSE.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-77: Bibliothekszuordnung



	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, werden
	ActualPosition	LREAL	Istposition des Reglers
	ActualVelocity	LREAL	Istgeschwindigkeit des Reglers
	ActualTorque	REAL	Istwert Kraft/Drehmoment des Reglers
	ControlStatusWord	DWORD	Statuswort des Reglers (Siehe A-0-0019)
	ControlStatusLimit	DWORD	Dieser Wert wird aktuell nicht verwendet
VAR_OUTPUT	Status	BOOL	Zeigt an, dass die Istwerte der Achse übermittelt wurden.
	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

Tab. 6-78: Schnittstelle des FB ML_ControlWriteActualData

Fehlerbehandlung

Der Funktionsbaustein ML_ControlWriteActualData erzeugt folgende Fehlermeldungen in:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.5 ML_OpenCyclicAnalogChannelCtrl

Kurzbeschreibung

Der Funktionsbaustein öffnet den zyklischen Analog-Sollwertkanal für die angegebene Achse. Der Baustein ist nur im Zustand 'Controlled Motion' erlaubt. Optional kann eine Leitachse angegeben werden, falls der zyklische Analogwert für diese Achse aus den Werten einer anderen Achse berechnet wurde. Mit dieser Angabe wird verhindert, dass Leitachsrekursionen programmiert werden, die zu Rückkopplungseffekten führen können.



Der Funktionsbaustein sollte in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Wenn der "Execute"-Eingang deaktiviert wird, werden alle Ausgänge auf FALSE gesetzt. Der zyklische Analog-Sollwertkanal bleibt geöffnet, wenn er zuvor geöffnet war.

Wenn der Funktionsbaustein durch einen anderen Motion-Funktionsbaustein unterbrochen wurde, wird der Funktionsbaustein abgebrochen und der zyklische Analog-Sollwertkanal geschlossen.

Der Baustein ist nur bei der HNC100-3x verfügbar.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-79: Bibliothekszuordnung

ML_PLCopen.library

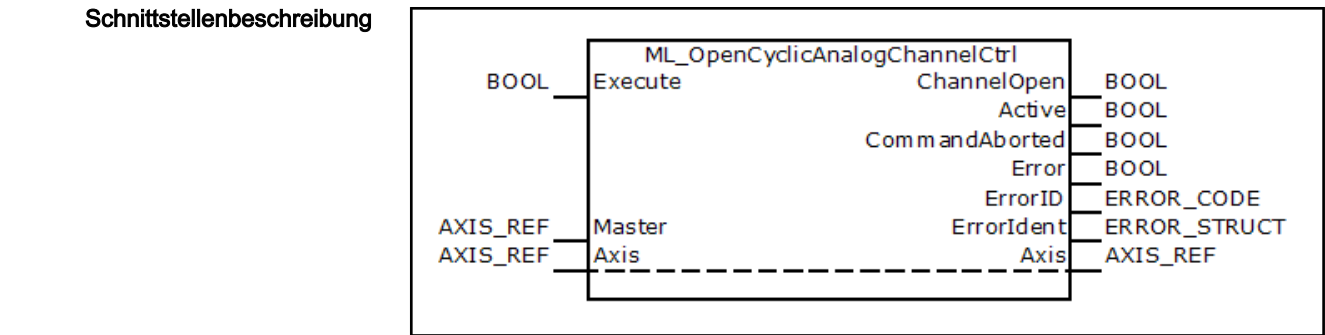


Abb. 6-98: FB ML_OpenCyclicAnalogChannelCtrl

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
	Master	AXIS_REF	Enthält Informationen zur Leitachse.
VAR_OUTPUT	Channel-Open	BOOL	Indiziert, dass der zyklische Analog-Sollwertkanal geöffnet ist, und die Achse dem Analogwert des letzten Zyklus folgt.
	Active	BOOL	Indiziert, dass der zyklische Analog-Sollwertkanal geöffnet wird.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	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

Tab. 6-80: Schnittstelle von FB ML_OpenCyclicAnalogChannelCtrl

- Fehlerbehandlung
- Der Funktionsbaustein ML_OpenCyclicAnalogChannelCtrl erzeugt folgende Fehlermeldungen:
- MLC_TABLE, Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.6 ML_WriteCyclicAnalogCtrl

Kurzbeschreibung

Der Funktionsbaustein ML_WriteCyclicAnalogCtrl schreibt einen zyklischen Analogwert für eine Achse, wenn sich die Achse im Zustand 'Controlled Motion' befindet. Nur wenn der zyklische Analog-Sollwertkanal geöffnet ist, wird der Analogwert im nächsten Zyklus an der Achse wirksam.

Um den zyklischen Analog-Sollwertkanal zu öffnen, muss der Funktionsbaustein [ML_OpenCyclicAnalogChannelCtrl, Seite 269](#), benutzt werden.

Der Funktionsbaustein sollte in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-81: Bibliothekszuordnung

Schnittstellenbeschreibung

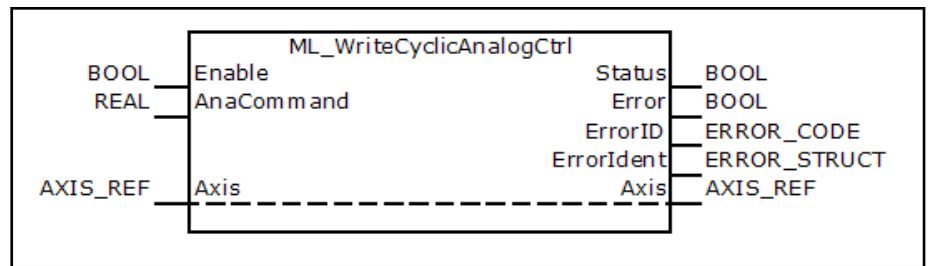


Abb. 6-99: ML_WriteCyclicAnalogCtrl

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, wird der Analogwert "AnaCommand" auf die Achse geschrieben. Wirksam wird der Analogwert erst, wenn der zyklische Analog-Sollwertkanal geöffnet ist.
	AnaCommand	REAL	Gibt den Analogwert an, der an die Achse übertragen werden soll. Wirksam wird der Analogwert erst, wenn der zyklische Analog-Sollwertkanal geöffnet ist.
VAR_OUTPUT	Status	BOOL	Zeigt an, dass der Analogwert an die Achse übermittelt wurde. Wirksam wird der Analogwert erst dann, wenn der zyklische Analog-Sollwertkanal geöffnet ist.
	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

Tab. 6-82: Schnittstelle des FB ML_WriteCyclicAnalogCtrl

Besonderheiten des AnaCommand-Eingangs

1. Der Wert des "AnaCommand"-Eingangs wird im nächsten Zyklus wirksam, wenn der zyklische Analog-Sollwertkanal geöffnet ist.
2. Der Wert des "AnaCommand"-Eingangs hat die Skalierung der zugeordneten Achse.

Fehlerbehandlung

Der Funktionsbaustein ML_WriteCyclicAnalogCtrl erzeugt folgende Fehlermeldungen in:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.7 ML_OpenCyclicPositionChannelCtrl

Kurzbeschreibung

Der Funktionsbaustein öffnet den zyklischen Positions-Sollwertkanal für die angegebene Achse. Der Baustein ist nur im Zustand 'Controlled Motion' erlaubt. Optional kann eine Leitachse angegeben werden, falls der zyklische Positions-Sollwert für diese Achse aus den Werten einer anderen Achse berechnet wurde. Mit dieser Angabe wird verhindert, dass Leitachsrekursionen programmiert werden, die zu Rückkopplungseffekten führen können.



Der Funktionsbaustein sollte in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Wenn der "Execute"-Eingang deaktiviert wird, werden alle Ausgänge auf FALSE gesetzt. Der zyklische Positions-Sollwertkanal bleibt geöffnet, wenn er zuvor geöffnet war.

Wenn der Funktionsbaustein durch einen anderen Motion-Funktionsbaustein unterbrochen wurde, wird der Funktionsbaustein abgebrochen und der zyklische Positions-Sollwertkanal geschlossen.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-83: Bibliothekszuordnung

Schnittstellenbeschreibung

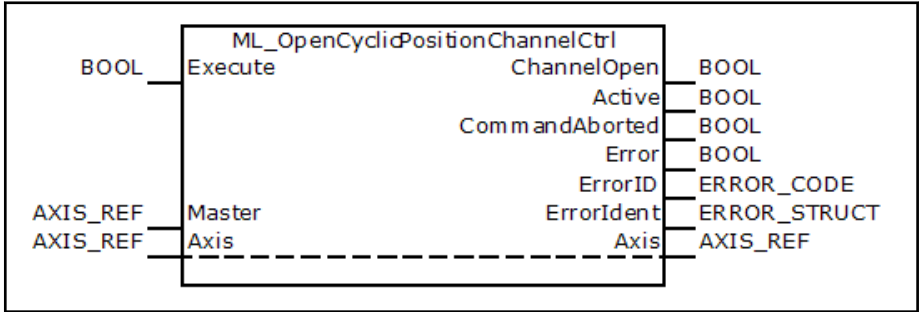


Abb. 6-100: FB ML_OpenCyclicPositionChannelCtrl

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
	Master	AXIS_REF	Enthält Informationen zur Leitachse.
VAR_OUTPUT	Channel-Open	BOOL	Indiziert, dass der zyklische Positions-Sollwertkanal geöffnet ist, und die Achse dem Positionswert des letzten Zyklus folgt.
	Active	BOOL	Indiziert, dass der zyklische Positions-Sollwertkanal geöffnet wird.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	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

Tab. 6-84: Schnittstelle von FB ML_OpenCyclicPositionChannelCtrl

Fehlerbehandlung Der Funktionsbaustein ML_OpenCyclicPositionChannelCtrl erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.8 ML_WriteCyclicPositionCtrl

Kurzbeschreibung Der Funktionsbaustein ML_WriteCyclicPositionCtrl schreibt ein zyklischen Positionswert für eine Achse, wenn sich die Achse im Zustand 'Controlled

Motion' befindet. Nur wenn der zyklische Positions-Sollwertkanal geöffnet ist, wird der Positionswert im nächsten Zyklus an der Achse wirksam.



Um den zyklischen Positions-Sollwertkanal zu öffnen, muss der Funktionsbaustein [ML_OpenCyclicPositionChannelCtrl](#), [Seite 271](#), benutzt werden.

Der Funktionsbaustein muss in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-85: Bibliothekszuordnung

Schnittstellenbeschreibung

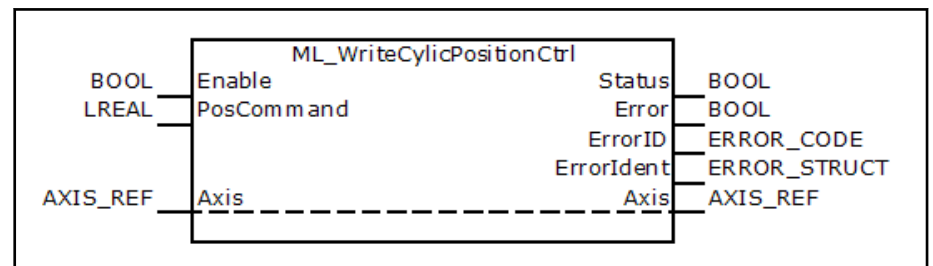


Abb. 6-101: ML_WriteCyclicPositionCtrl

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, wird "PosCommand" auf die Achse geschrieben. Wirksam wird der Positionswert erst, wenn der zyklische Positions-Sollwertkanal geöffnet ist.
	PosCommand	LREAL	Gibt den Positionswert an, der an die Achse übertragen werden soll. Wirksam wird der Positionswert erst, wenn der zyklische Positions-Sollwertkanal geöffnet ist.
VAR_OUTPUT	Status	BOOL	Zeigt an, dass der Positionswert an die Achse übermittelt wurde. Wirksam wird der Positionswert erst, wenn der zyklische Positions-Sollwertkanal geöffnet ist.
	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

Tab. 6-86: Schnittstelle des FB ML_WriteCyclicPositionCtrl

Besonderheiten des PosCommand-Eingangs

1. Der Wert des "PosCommand"-Eingangs wird erst im nächsten Zyklus wirksam, wenn der zyklische Positions-Sollwertkanal geöffnet ist.
2. Der Wert des "PosCommand"-Eingangs hat die Skalierung der zugeordneten Achse.

Fehlerbehandlung

Der Funktionsbaustein ML_WriteCyclicPositionCtrl erzeugt folgende Fehlermeldungen in:

- [MLC_TABLE](#), [Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

ML_PLCopen.library

6.9.9ML_OpenCyclicTorqueChannelCtrl

Kurzbeschreibung Der Funktionsbaustein öffnet den zyklischen Kraft-/Drehmoment-Sollwertkanal für die angegebene Achse. Der Baustein ist nur im Zustand 'Controlled Motion' erlaubt. Optional kann eine Leitachse angegeben werden, falls der zyklische Kraft-/Drehmoment-Wert für diese Achse aus den Werten einer anderen Achse berechnet wurde. Mit dieser Angabe wird verhindert, dass Leitachsrekursionen programmiert werden, die zu Rückkopplungseffekten führen können.



Der Funktionsbaustein sollte in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Wenn der "Execute"-Eingang deaktiviert wird, werden alle Ausgänge auf FALSE gesetzt. Der zyklische Kraft-/Drehmoment-Sollwertkanal bleibt geöffnet, wenn er zuvor geöffnet war.

Wenn der Funktionsbaustein durch einen anderen Motion-Funktionsbaustein unterbrochen wurde, wird der Funktionsbaustein abgebrochen und der zyklische Kraft-/Drehmoment-Sollwertkanal geschlossen.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-87: Bibliothekszuordnung

Schnittstellenbeschreibung

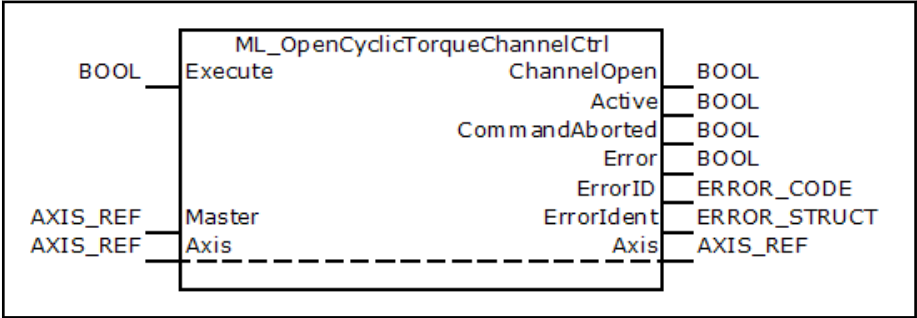


Abb. 6-102: FB ML_OpenCyclicTorqueChannelCtrl

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
	Master	AXIS_REF	Enthält Informationen zur Leitachse.
VAR_OUTPUT	Channel-Open	BOOL	Kommandoausführung erfolgreich. Der zyklische Kraft-/Drehmoment-Sollwertkanal wurde geöffnet. Die Achse folgt dem Kraft-/Drehmoment-Wert.
	Active	BOOL	Indiziert, dass der zyklische Kraft-/Drehmoment-Sollwertkanal geöffnet wird.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 6-88: Schnittstelle von FB ML_OpenCyclicTorqueChannelCtrl

Fehlerbehandlung Der Funktionsbaustein ML_OpenCyclicTorqueChannelCtrl erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.10 ML_WriteCyclicTorqueCtrl

Kurzbeschreibung Der Funktionsbaustein ML_WriteCyclicTorqueCtrl schreibt ein zyklischen Kraft-/Drehmoment-Wert für eine Achse, wenn sich die Achse im Zustand 'Controlled Motion' befindet. Nur wenn der zyklische Kraft-/Drehmoment-Sollwertkanal geöffnet ist, wird der Kraft-/Drehmoment-Wert im nächsten Zyklus an der Achse wirksam.



Um den zyklischen Kraft-/Drehmoment-Sollwertkanal zu öffnen, muss der Funktionsbaustein [ML_OpenCyclicTorqueChannelCtrl, Seite 274](#), benutzt werden.

Der Funktionsbaustein muss in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-89: Bibliothekszuordnung

Schnittstellenbeschreibung

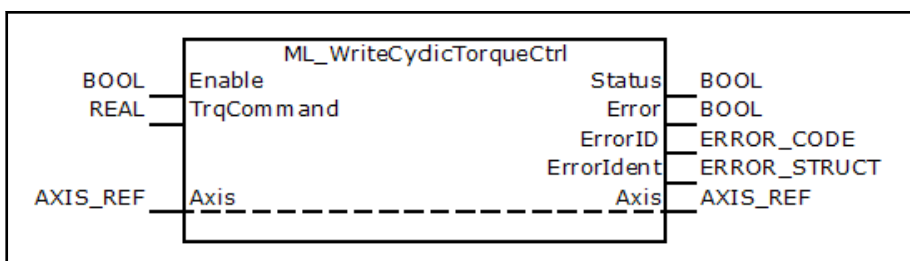


Abb. 6-103: ML_WriteCyclicTorqueCtrl

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, wird "TrqCommand" auf die Achse geschrieben. Wirksam wird der Kraft-/Drehmoment-Wert erst, wenn der zyklische Kraft-/Drehmoment-Sollwertkanal geöffnet ist.
	TrqCommand	REAL	Gibt den Kraft-/Drehmoment-Wert an, die an die Achse übertragen werden soll. Wirksam wird der Kraft-/Drehmoment-Wert erst, wenn der zyklische Kraft-/Drehmoment-Sollwertkanal geöffnet ist.
VAR_OUTPUT	Status	BOOL	Zeigt an, dass der Kraft-/Drehmoment-Wert an die Achse übermittelt wurde. Wirksam wird der Kraft-/Drehmoment-Wert erst, wenn der zyklische Kraft-/Drehmoment-Sollwertkanal geöffnet ist.
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist

ML_PLCopen.library

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 6-90: Schnittstelle des FB ML_WriteCyclicTorqueCtrl

Besonderheiten des TrqCommand-Eingangs

1. Die Wert des "TrqCommand"-Eingangs wird im nächsten Zyklus wirksam, wenn der zyklische Kraft-/Drehmoment-Sollwertkanal geöffnet ist.
2. Der Wert des "TrqCommand"-Eingangs hat die Skalierung von der zugeordneten Achse.

Fehlerbehandlung

Der Funktionsbaustein ML_WriteCyclicTorqueCtrl erzeugt folgende Fehlermeldungen in:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.11 ML_OpenCyclicVelocityChannelCtrl

Kurzbeschreibung

Der Funktionsbaustein öffnet den zyklischen Geschwindigkeits-Sollwertkanal für die angegebene Achse. Der Baustein ist nur im Zustand 'Controlled Motion' erlaubt. Optional kann eine Leitachse angegeben werden, falls der zyklische Geschwindigkeitswert für diese Achse aus den Werten einer anderen Achse berechnet wurde. Mit dieser Angabe wird verhindert, dass Leitachsrekursionen programmiert werden, die zu Rückkopplungseffekten führen können.



Der Funktionsbaustein sollte in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Wenn der "Execute"-Eingang deaktiviert wird, werden alle Ausgänge auf FALSE gesetzt. Der zyklische Geschwindigkeits-Sollwertkanal bleibt geöffnet, wenn er zuvor geöffnet war.

Wenn der Funktionsbaustein durch einen anderen Motion-Funktionsbaustein unterbrochen wurde, wird der Funktionsbaustein abgebrochen und der zyklische Geschwindigkeits-Sollwertkanal geschlossen.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-91: Bibliothekszuordnung

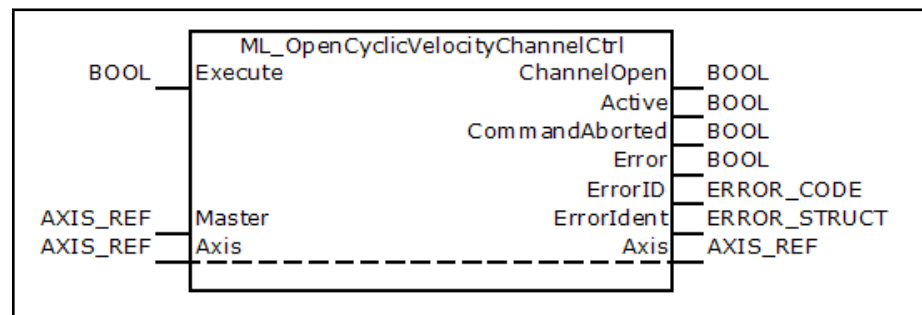
Schnittstellenbeschreibung

Abb. 6-104: FB ML_OpenCyclicVelocityChannelCtrl

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Execute	BOOL	Startet den Funktionsbaustein bei einer steigenden Flanke.
	Master	AXIS_REF	Enthält Informationen zur Leitachse.
VAR_OUTPUT	Channel-Open	BOOL	Indiziert, dass der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist, und die Achse dem Geschwindigkeitswert des letzten Zyklus folgt.
	Active	BOOL	Indiziert, dass der zyklische Geschwindigkeits-Sollwertkanal geöffnet wird.
	Command-Aborted	BOOL	Kommando wurde durch ein anderes Kommando abgebrochen.
	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

Tab. 6-92: Schnittstelle von FB ML_OpenCyclicVelocityChannelCtrl

Fehlerbehandlung

Der Funktionsbaustein ML_OpenCyclicVelocityChannelCtrl erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.9.12 ML_WriteCyclicVelocityCtrl

Kurzbeschreibung

Der Funktionsbaustein ML_WriteCyclicVelocityCtrl schreibt ein zyklischen Geschwindigkeitswert für eine Achse, wenn sich die Achse im Zustand 'Controlled Motion' befindet. Nur wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist, wird der Geschwindigkeitswert im nächsten Zyklus an der Achse wirksam.



Um den zyklischen Geschwindigkeits-Sollwertkanal zu öffnen, muss der Funktionsbaustein [ML_OpenCyclicVelocityChannelCtrl, Seite 276](#), benutzt werden.

Der Funktionsbaustein muss in einer PLC-Task verwendet werden, die synchron zum sercos Takt ist, ansonsten sind ruckhafte Bewegungen der Achse sehr wahrscheinlich.

Bibliothek	Bereich
ML_PLCopen.library	Controlled Motion

Tab. 6-93: Bibliotheksuzuordnung

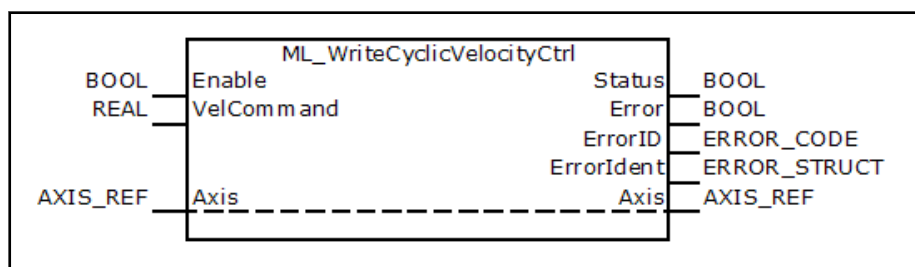
Schnittstellenbeschreibung

Abb. 6-105: ML_WriteCyclicVelocityCtrl

ML_PLCopen.library

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Enthält Informationen zur angesprochenen Achse.
VAR_INPUT	Enable	BOOL	Solange der "Enable"-Ausgang TRUE ist, wird "VelCommand" auf die Achse geschrieben. Wirksam wird der Geschwindigkeitswert erst, wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist.
	VelCom- mand	REAL	Gibt den Geschwindigkeitswert an, der an die Achse übertragen werden soll. Wirksam wird der Geschwindigkeitswert erst, wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist.
VAR_OUTPUT	Status	BOOL	Zeigt an, dass der Geschwindigkeitswert an die Achse übermittelt wurde. Wirksam wird der Geschwindigkeitswert erst, wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist.
	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

Tab. 6-94: Schnittstelle des FB ML_WriteCyclicVelocityCtrl

Besonderheiten des VelCommand-Eingangs

1. Die Wert des "VelCommand"-Eingangs wird im nächsten Zyklus wirksam, wenn der zyklische Geschwindigkeits-Sollwertkanal geöffnet ist.
2. Der Wert des "VelCommand"-Eingangs hat die Skalierung von der zugeordneten Achse.

Fehlerbehandlung

Der Funktionsbaustein ML_WriteCyclicVelocityCtrl erzeugt folgende Fehlermeldungen in:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10 Parameterzugriff

6.10.1 Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes

Aufbau und Zugriff auf Parameter aus dem SPS-Programm

Die Daten einer Achse, einer Steuerung oder eines Nockenschaltwerkes, sind über Parameterzugriff lesend und teilweise auch schreibend erreichbar. Folgende Parameterarten werden bei der IndraMotion MLC unterschieden:

- C-Parameter (control parameters), die sich auf eine konkrete Steuerung, die die Achsen beeinflusst, beziehen.
- A-Parameter (axis parameters), die sich auf jeweils eine Achse beziehen.
- K-Parameter (kinematics parameters), die sich auf eine Kinematik (Achsgruppe) beziehen
(siehe ML_Robot, Zugriff auf Daten einer Kinematik (Achsgruppe)).
- M-Parameter (touch probe parameters), die sich auf jeweils einen Mess-taster beziehen. Sie sind nur über PLCopen-Funktionsbausteine zu erreichen.
- N-Parameter (pls parameters), die sich auf jeweils ein Nockenschaltwerk beziehen.
- O-Parameter (oscilloscope parameters), die sich auf jeweils eine Oszilloskop-Funktion beziehen.

Zusätzlich zu diesen Parametern existieren weitere Gruppen, die auf Geräte an einem bestimmten Bus (S-Parameter, sercos Interface) oder auf ein bestimmtes Produkt (P-Parameter) beziehen.

Alle IndraMotion-Parameter werden nach dem gleichen Bildungsgesetz aufgebaut:

Bsp.: A-0-0001 , mit "A" 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.

Dem Anwender stehen dazu in der ML_Base.library für A-, ..., O-, P- und S-Parameter Vereinbarungsdateien für Globale Konstanten zur Verfügung:

Vereinbarungsdateien für Globale Konstanten

- [GVL_A_Param_ID, Seite 134,](#)
- [GVL_C_Param_ID, Seite 134,](#)
- [GVL_M_Param_ID, Seite 135,](#)
- [GVL_N_Param_ID, Seite 135,](#)
- [GVL_O_Param_ID, Seite 136,](#)
- [GVL_P_Param_ID, Seite 136,](#)
- [GVL_S_Param_ID, Seite 137.](#)

Bei Kenntnis der Nummer des Parameters:

A-Parameter-Adresskonstanten

```
VAR GLOBAL CONSTANT
  FP_A_0_0000: MB_IDN:= 4294967296 ; (* Dummy A parameter *)
  FP_A_0_0001: MB_IDN:= 4294967297 ; (* Logical axis number *)
  FP_A_0_0002: MB_IDN:= 4294967298 ; (* Axis name *)
  (* ..... *)
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
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. 6-95: *Parameter-Offsets*

Mit der Steuerung können mehrere Achsen angesteuert werden.

Aus diesem Grund besitzen die Bausteine einen Adresseingang (VAR_IN_OUT) vom Typ AXIS_REF, RIL_CommonTypes.library, der wie folgt zu belegen ist:

ML_PLCopen.library

Zugriff auf A-Parameter:

- Der Adressbezeichner wird durch IndraWorks MLC automatisch generiert (Achsname/Achsnummer), Typ AXIS_REF.

Zugriff auf N-Parameter:

- Der Adressbezeichner wird durch IndraWorks MLC automatisch generiert oder kann ersatzweise durch den Anwender definiert werden (Objektname/ Objektnummer), Typ AXIS_REF.

Zugriff auf O-Parameter:

- Der Adressbezeichner wird durch IndraWorks MLC automatisch generiert oder kann ersatzweise durch den Anwender definiert werden (Objektname/ Objektnummer), Typ AXIS_REF.

Zugriff auf C-Parameter:

- Verwendung des Adressbezeichners einer beliebigen Achse dieser Steuerung oder Deklaration mit den Werten.
 - ContrlNo:= LOCAL_CNTRL bzw. CNTRL_1...
 - AxisNo:= NO_OBJECT.

Referenzen am Axis-Eingang, Bsp.: MB_ReadParameter

```

VA1:AXIS_REF:=(ContrlNo:=LOCAL_CNTRL,AxisNo:=AXIS_1)
               (*A parameters, axis VA1, axis number 1*);

MLC_A: AXIS_REF:=(ContrlNo:=LOCAL_CNTRL,AxisNo:=NO_OBJECT)
               (* C parameters, control MLC_A *);

PLS_A: AXIS_REF:=(ContrlNo:=LOCAL_CNTRL,AxisNo:=PLS_1)
               (*N parameters, programmable limit switch PLS_A, obj.no.1*);

OSZ1: AXIS_REF:=(ContrlNo:=LOCAL_CNTRL,AxisNo:=OSCI_CHANNEL_1)
               (*O parameters, oscilloscope OSZ1, obj.no.1*);

```

6.10.2 Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen

Die MB_Read***Parameter- und die MB_Write***Parameter-Funktionsbausteine, sowie die MB_GroupRead***Parameter- und die MB_GroupWrite***Parameter-Funktionsbausteine organisieren die Datenübertragung zwischen der SPS und der Motion-Komponente der Steuerung bzw. den Antrieben.

Der jeweilige Funktionsbaustein greift dabei auf den genannten Parameter zu. Die nachstehenden Zeiten sind durchschnittliche Zugriffszeiten und gelten für eine IndraMotion MLC L65.1 (PlcTask: 1ms (zyklisch)) .

Parameter	Anzahl Byte	Funktionsbaustein	durchschnittl. Zugriffszeit	min	max
S-0-0256	UDINT, 4	MB_ReadParameter	9,8 ms	7ms	23ms
P-0-0072	Liste, 1024	MB_WriteListParameter	232ms	219ms	245ms
P-0-0072	Liste, 1024	MB_ReadListParameter	139ms	135ms	157ms
A-0-0224	REAL, 4	MB_ReadRealParameter	2ms	2ms	2ms

Parameter	Anzahl Byte	Funktionsbaustein	durchschnittl. Zugriffszeit	min	max
A-0-0101	LREAL, 8	MB_ReadLrealParameter	2ms	2ms	2ms
A-0-0023	UDINT, 4	MB_ReadParameter	2ms	2ms	2ms
C-0-0852	LREAL, 8	MB_WriteLrealParameter	2 ms	2ms	2ms
A-0-3020	LREAL, 8	MB_WriteLrealParameter	32ms	12ms	79ms
A-0-0030	REAL, 4	MB_WriteRealParameter	2ms	2ms	2ms
A-0-0002	STRING	MB_ReadStringParameter	2ms	2ms	2ms

6.10.3 MB_Command

Kurzbeschreibung Der Funktionsbaustein MB_Command erlaubt die Ausführung eines Kommandos.

Möglich sind dabei das Kommando zu starten, zu steuern und zu beobachten oder abubrechen.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-96: Bibliothekszuordnung

Schnittstellenbeschreibung

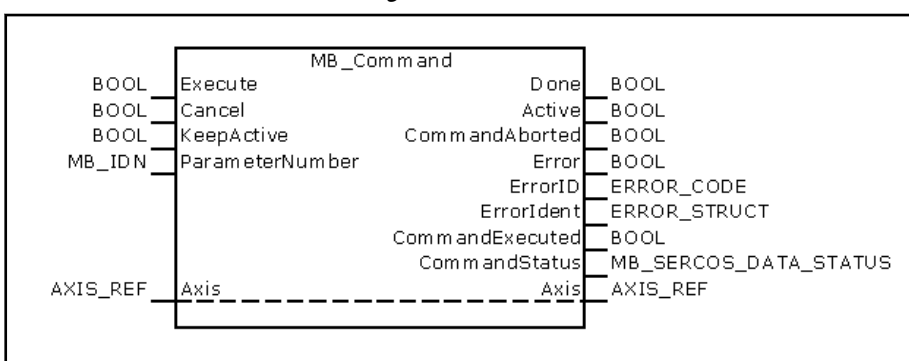


Abb. 6-106: FB MB_Command

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Cancel	BOOL	bricht das laufende Kommando bei steigender Flanke ab
	KeepActive	BOOL	hält das Kommando nach Abarbeitung des Zustandsgraphen aktive
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
VAR_OUTPUT	Done	BOOL	gültiger Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command- Aborted	BOOL	Kommando erfolgreich abgebrochen und zurückgesetzt
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache

ML_PLCOpen.library

	Name	Typ	Kommentar
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Command-Executed	BOOL	Kommando erfolgreich abgeschlossen
	Command-Status	MB_SERCOS_DATA_STATUS	Status des gewählten Parameters

Tab. 6-97: Schnittstelle von FB MB_Command

Beschreibung

Der Funktionsbaustein "MB_Command" startet nach der Bearbeitungsfreigabe mittels "Execute" die Kommandoausführung des über den Eingang "ParameterNumber" festgelegten Kommandos. Das Kommando wird im allgemeinen nach Abschluss der Bearbeitung automatisch zurückgesetzt.

Die Kommandoausführung kann, abhängig vom Kommando selbst, jederzeit über den Eingang "Cancel" abgebrochen werden.

Der Eingang "KeepActive" ermöglicht es, das Kommando nach Ausführung weiter aktiv zu halten. Wird im Folgenden der Eingang "KeepActive" deaktiviert, wird das Kommando ebenfalls automatisch zurückgesetzt. .

Der erfolgreiche Abschluss bzw. erfolgreiche Abbruch eines Kommandos wird über die Ausgänge "Done" bzw. "CommandAborted" für die Zeitdauer von mindestens einem Aufruf signalisiert.

Die aktive Bearbeitung des Kommandos wird über den Ausgang "Active" und das entsprechenden Datum am Ausgang "CommandStatus" dargestellt.

Der aktuelle Status der Kommandoausführung wird permanent am Ausgang "CommandStatus" signalisiert, der Abschluss der Kommandoausführung wird durch den Ausgang "CommandExecuted" quittiert.

Tritt während der Bearbeitung des Kommandos ein Fehler auf, wird dies über den Ausgang "Error", sowie über "ErrorID", "ErrorIdent" für die Zeitdauer von mindestens einem Aufruf signalisiert.

CommandStatus:

- 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

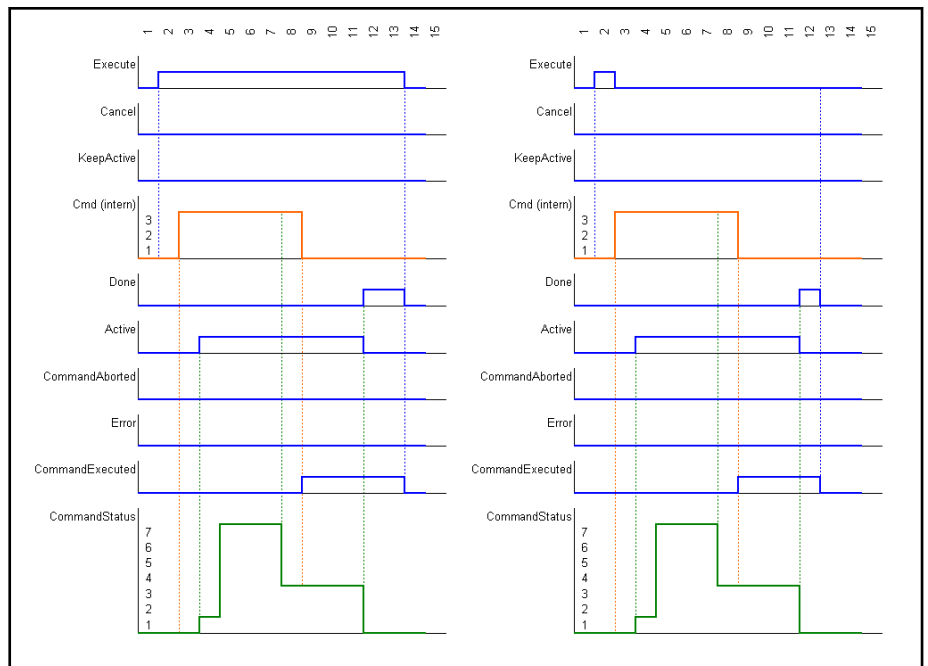


Abb. 6-107: Signal-Zeit-Diagramm für Signalverlauf "Execute-Done"

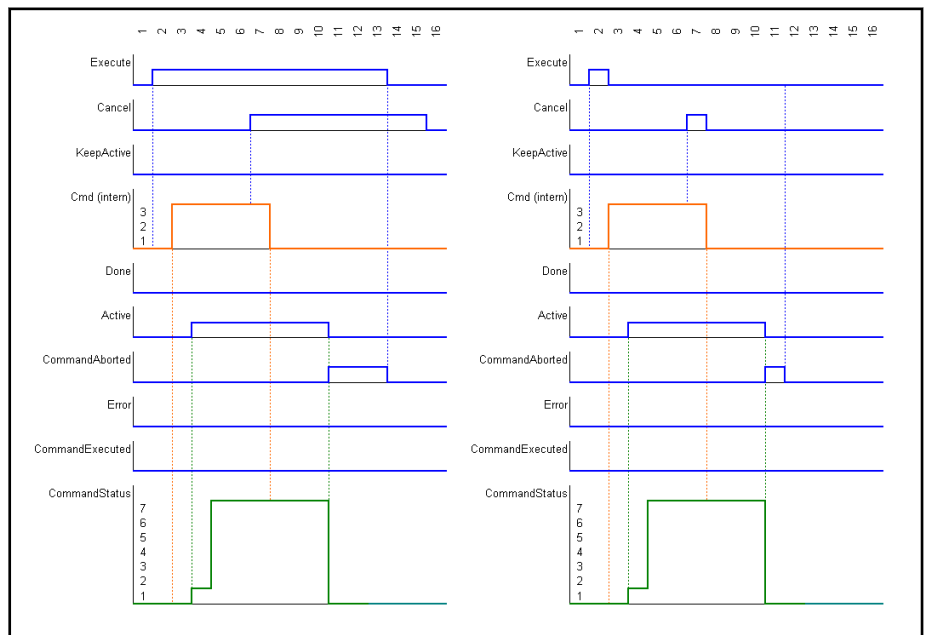


Abb. 6-108: Signal-Zeit-Diagramm für Signalverlauf "Execute-Cancel-Command-Aborted"

ML_PLCopen.library

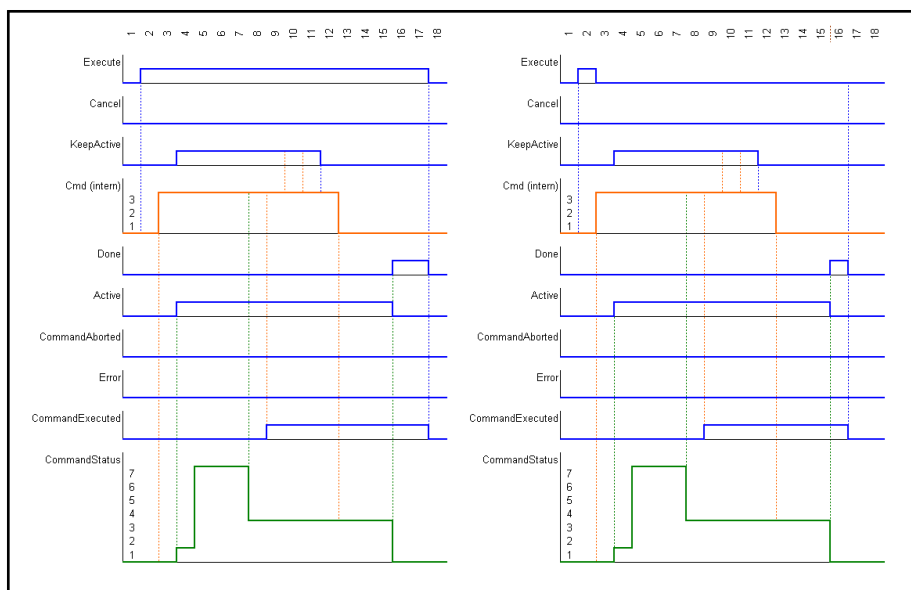


Abb. 6-109: Signal-Zeit-Diagramm für Signalverlauf "Execute-KeepActive-Done"

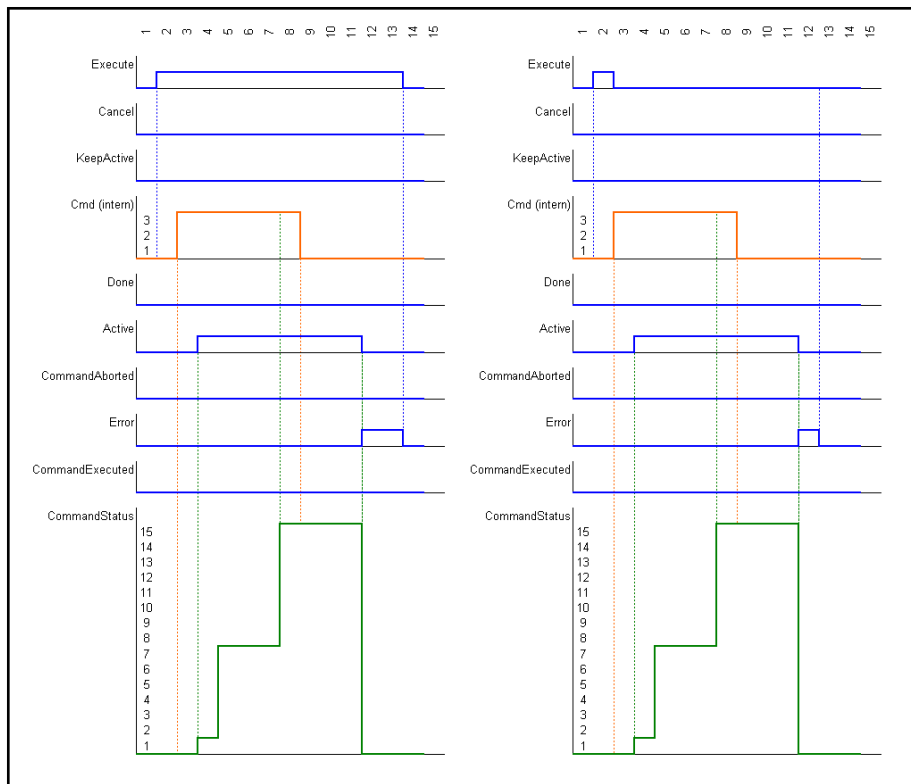


Abb. 6-110: Signal-Zeit-Diagramm für Signalverlauf "Execute-Error"

6.10.4 MB_Command, Besonderheiten bei SERCOS-basierten Achsen



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Ein parametergestütztes Kommando kann auf drei unterschiedliche Arten ausgelöst werden

(Beispiel A-0-1030 Kommando: Alle Achsfehler löschen):

- **Von IndraWorks aus:** Button "Fehler löschen" im Popup-Menü der Achse ⇒ Diagnose ⇒ "Fehler löschen".
- **Durch den Parametereditor:** Eingabe des Binärwertes "0000000000000011" (=0x0003) in den Parameter. Dies löst den Fehlerlöschvorgang aus.

Nach Abschluss des Fehlerlöschvorganges setzt die SPS den Wert des Parameters automatisch zurück auf "0000000000000000".

- **Von der SPS aus:** über den Baustein "MB_Command".

Fehlerbehandlung Der Funktionsbaustein MB_Command erzeugt folgende Arten von Fehlermeldungen:

ERROR_TABLEs

- MLC_TABLE, Tabellennummer 16#0030,
- INDRV_TABLE, Tabellennummer 16#0070,
- F_RELATED_TABLE, Tabellennummer 16#0170.

ErrorID	Tabelle	Additional1	Additional2	Beschreibung
INPUT_RANGE_ERROR	F_RELATED_TABLE	16#0002	16#0001	Retriggerung nicht unterstützt
INPUT_RANGE_ERROR	F_RELATED_TABLE	16#0002	16#0002	Cancel/ Keep Active sind bei ctrl commands nicht erlaubt
INPUT_RANGE_ERROR	F_RELATED_TABLE	16#0002	16#0003	Parameter ist kein Kommando
STATE_MACHINE_ERROR	F_RELATED_TABLE	16x0006	Zustand	Ungültiger Zustand des FBs
DEVICE_ERROR	F_RELATED_TABLE	16x1701	16#0000	Fehler während der Kommandoabarbeitung (Steuerung)
DEVICE_ERROR	INDRV_TABLE	Wert	16#0000	Fehler während der Kommandoabarbeitung (Antrieb) Wert in S-0-0390 enthalten.

Tab. 6-98: Fehlernummern, verursacht durch MB_Command

Zusätzlich können Fehlercodes der MLC_TABLE auftreten, wie bei

- MB_WriteParameter,
- MB_ReadSercosDataStatus,
- MB_ReadParameter,
- MB_ReadSercosAttribute.

6.10.5 MB_ReadLintParameter

Kurzbeschreibung Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als LINT-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

ML_PLCopen.library

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-99: Bibliothekszuordnung

Schnittstellenbeschreibung

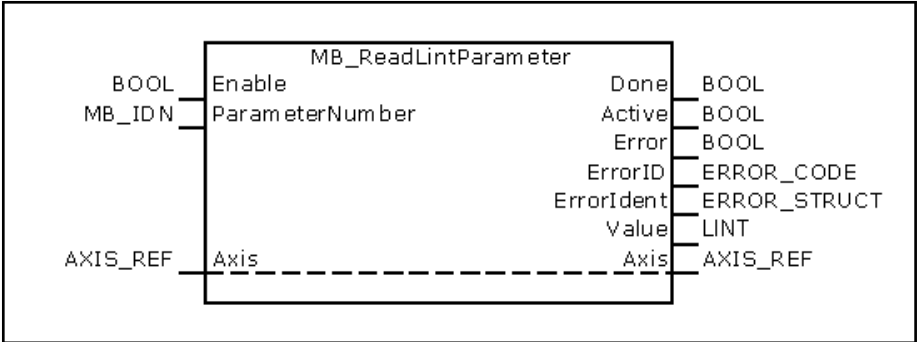


Abb. 6-111: FB MB_ReadLintParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse, bzw. zum adressierten Objekt.
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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	LINT	8-Byte-Wert, als LINT dargestellt

Tab. 6-100: Schnittstelle von FB MB_ReadLintParameter

Signal-Zeit-Diagramm

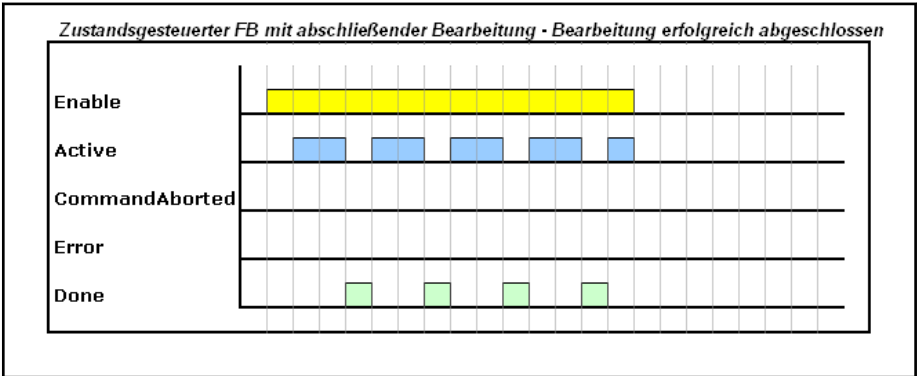


Abb. 6-112: Standard-Signal-Zeit-Diagramm

6.10.6 MB_ReadLintParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadLintParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.7 MB_ReadListParameter

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines anwenderspezifischen 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.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-101: Bibliothekszuordnung

Schnittstellenbeschreibung

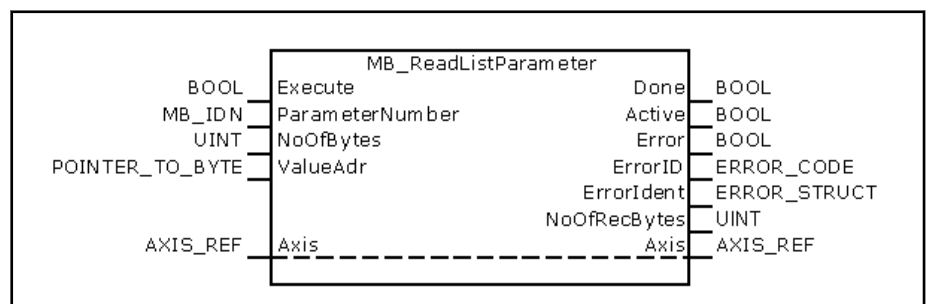


Abb. 6-113: FB MB_ReadListParameter

ML_PLCopen.library

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Lesen des Parameters, getriggert durch die positive Flanke von "Execute"
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
	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. 6-102: Schnittstelle von FB MB_ReadListParameter

Signal-Zeit-Diagramm

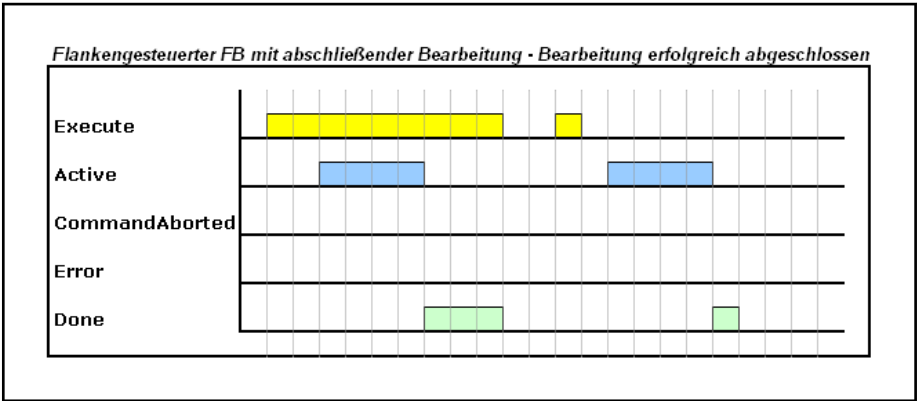


Abb. 6-114: Standard-Signal-Zeit-Diagramm

Programmbeispiel
MB_ReadListParameter

Das Programmbeispiel zeigt das Lesen des Listenparameters (C-0-2001) der MLC mit 1024 Werten (DINT) und das Abspeichern in das ReadListArray.
Programm:

```
PROGRAM ReadListParameter
VAR IN_OUT
  Axis:          AXIS_REF;          (* Antriebl ist eine Variable des AXIS_REF *)
END_VAR

VAR
  ReadListArray: ARRAY [0..1023] OF DINT;  (* Data array for the received bytes *)

  FB_ReadListParameter: MB_ReadListParameter;
  bExecute_RLP:         BOOL;          (* Start ReadListParameter *)
  diParameterNo_RLP:    MB_IDN:= FP_C_0_2001; (* Parameter ControlCAM1 der MLC *)
  uiNoOfBytes_RLP:      UINT := 4096;    (* DINT := 4 Bytes *)
```

```

bDone_RLP:          BOOL;          (* TRUE := FB ready          *)
bActive_RLP:        BOOL;          (* The FB works              *)
bError_RLP:         BOOL;          (* Error occur               *)
enErrorID_RLP:      ERROR_CODE;    (* Error class               *)
stErrorIdent_RLP:   ERROR_STRUCT;  (* Error definition          *)
uiNoOfRecBytes_RLP: UINT;          (* Number of received bytes  *)

```

END_VAR

Programmbeispiel im Funktionsplan

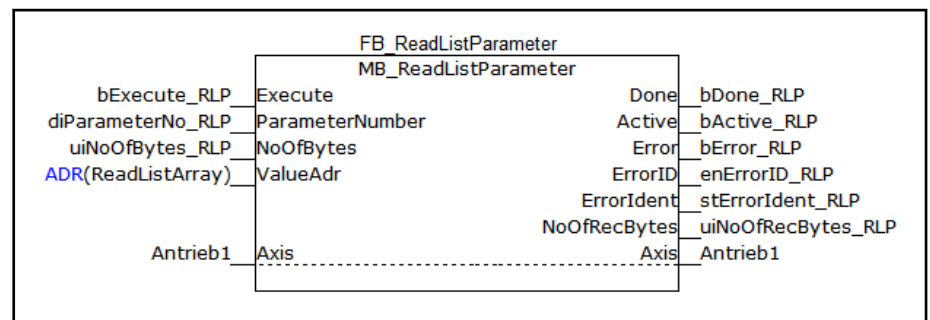


Abb. 6-115: FUP Programmbeispiel ReadListParameter

Programmbeispiel im Strukturierten Text

Programm:

```

FB_ReadListParameter(
    Execute:= bExecute_RLP,          (* Start ReadListParameter *)
    ParameterNumber:= diParameterNo_RLP, (* Parameter ControlCAM1 der MLC *)
    NoOfBytes:= uiNoOfBytes_RLP,      (* DINT := 4 Bytes *)
    ValueAdr:= ADR(ReadListArray),   (* Adresse des ReadListArray *)
    Axis:= Antrieb1,                 (* Antriebsadresse *)
    Done=> bDone_RLP,                (* TRUE := FB ready *)
    Active=> bActive_RLP,             (* The FB works *)
    Error=> bError_RLP,               (* Error occur *)
    ErrorId=> enErrorID_RLP,          (* Error class *)
    ErrorIdent=> stErrorIdent_RLP,    (* Error definition *)
    NoOfRecBytes=> uiNoOfRecBytes_RLP; (* Number of received bytes *)

```

6.10.8 MB_ReadListParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadListParameter erzeugt folgende Arten von Fehlermeldungen:

ML_PLCOpen.library

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#) , Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#) , Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.9 MB_ReadLrealListParameter

Kurzbeschreibung Der Funktionsbaustein gibt den Wert eines anwenderspezifischen 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.



In Unterschied zu MB_ReadListParameter, in dem ein ARRAY of BYTE mit n Bytes gelesen wird, wird hier ein ARRAY of LREAL mit m BYTES, also m/8 LREAL gelesen.

Bibliothek	Bereich
ML_PLCOpen.library	Parameter

Tab. 6-103: Bibliothekszuordnung

Schnittstellenbeschreibung

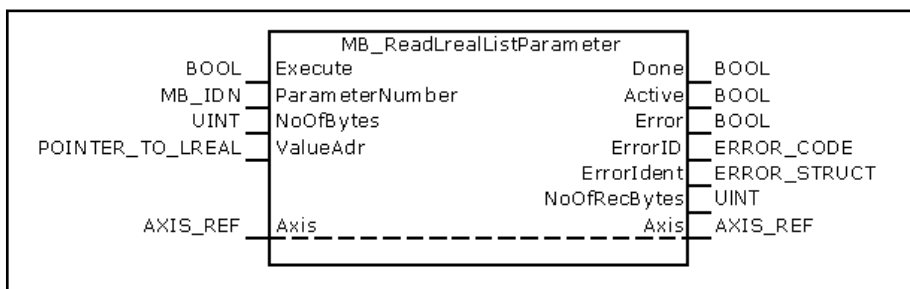


Abb. 6-116: MB_ReadLrealListParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Lesen des Parameters, getriggert durch die positive Flanke von "Execute"
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes , Seite 278)
	NoOfBytes	UINT	Maximal zu lesende Datenmenge in Byte
	ValueAdr	POINTER TO LREAL	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

	Name	Typ	Kommentar
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	NoOf RecBytes	UINT	Anzahl der tatsächlich gelesenen Bytes

Tab. 6-104: Schnittstelle von FB MB_ReadLrealListParameter

Signal-Zeit-Diagramm

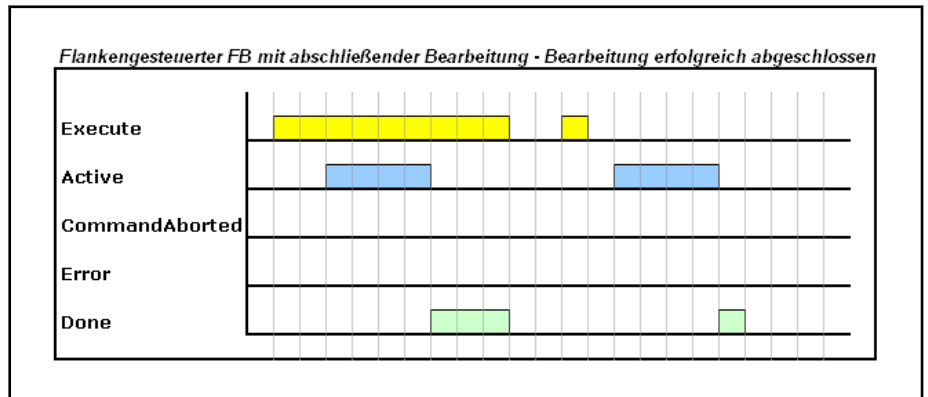


Abb. 6-117: Standard-Signal-Zeit-Diagramm

Programmbeispiel MB_ReadLrealListParameter

Das Programmbeispiel zeigt das Lesen des Listenparameters (A-0-3027) der MLC mit 1024 Werten (LREAL) und das Abspeichern in das ReadListArray.

Programm:

```

PROGRAM ReadLrealListParameter
VAR IN_OUT
  Axis:          AXIS_REF;          (* Antrieb1 ist eine Variable des AXIS_REF*)
END_VAR

VAR
  ReadListArray:  ARRAY [0..1023] OF LREAL; (* Data array for the received bytes *)

  FB_ReadLrealListParameter: MB_ReadLrealListParameter;
  bExecute_RLP:      BOOL;          (* Start ReadLrealListParameter *)
  diParameterNo_RLP: LREAL := FP_A_0_3027; (* Liste Folgeachs anfangsgeschwindigkeiten *)
  uiNoOfBytes_RLP:   UINT := 8192;  (* LREAL := 8 Bytes *)
  bDone_RLP:        BOOL;          (* TRUE := FB ready *)
  bActive_RLP:      BOOL;          (* The FB works *)
  bError_RLP:       BOOL;          (* Error occur *)
  enErrorID_RLP:    ERROR_CODE;    (* Error class *)
  stErrorIdent_RLP: ERROR_STRUCT;  (* Error definition *)
  uiNoOfRecBytes_RLP: UINT;        (* Number of received bytes *)
END_VAR

```

Programmbeispiel im Funktionsplan

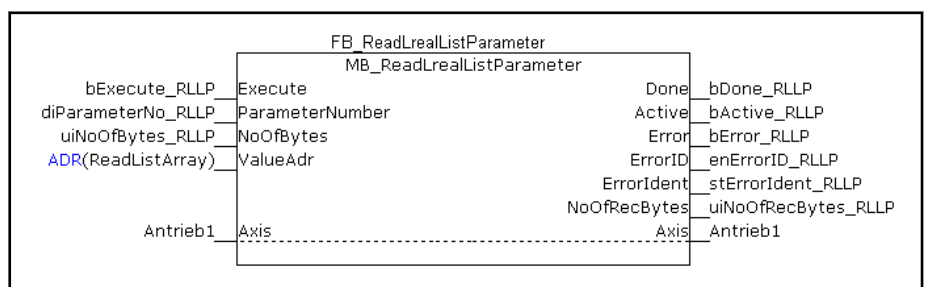


Abb. 6-118:

ML_PLCopen.library

Programmbeispiel im Strukturierten Text

Programm:

```

FB_ReadLrealListParameter(
  _Execute:=      bExecute_RLP,          (* Start ReadLrealListParameter *)
  ParameterNumber:= diParameterNo_RLP,   (* Liste Folgeachsanzfangsgeschwindigkeiten*)
  NoOfBytes:=     uiNoOfBytes_RLP,        (* LREAL:= 8 Bytes *)
  ValueAdr:=      ADR(ReadListArray),    (* Adresse des ReadListArray *)
  Axis:=          Antriebl,               (* Antriebsadresse *)
  Done=>          bDone_RLP,              (* TRUE := FB ready *)
  Active=>        bActive_RLP,            (* The FB works *)
  Error=>         bError_RLP,             (* Error occur *)
  ErrorId=>       enErrorID_RLP,          (* Error class *)
  ErrorIdent=>    stErrorIdent_RLP:      (* Error definition *)
  NoOfRecBytes=>  uiNoOfRecBytes_RLP);    (* Number of received bytes *)

```

6.10.10 MB_ReadLrealListParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen, Seite 280](#).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes, Seite 278](#).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadLrealListParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010, Seite 346](#) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.11 MB_ReadLrealParameter

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als LREAL-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-105: Bibliothekszuordnung

Schnittstellenbeschreibung

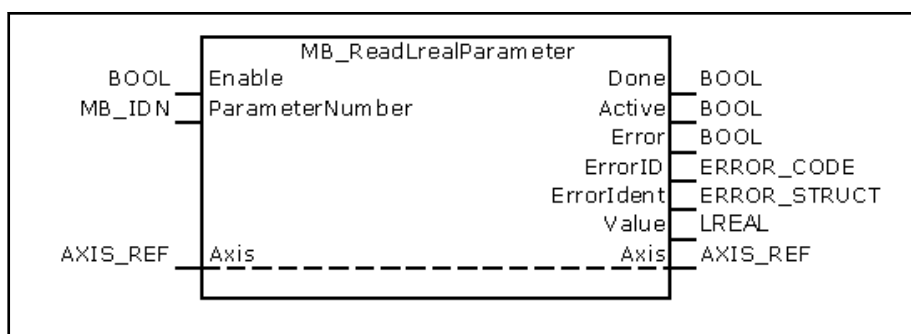


Abb. 6-119: FB MB_ReadLrealParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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	LREAL	8-Byte-Wert als LREAL-Zahl dargestellt

Tab. 6-106: Schnittstelle von FB MB_ReadLrealParameter

Signal-Zeit-Diagramm

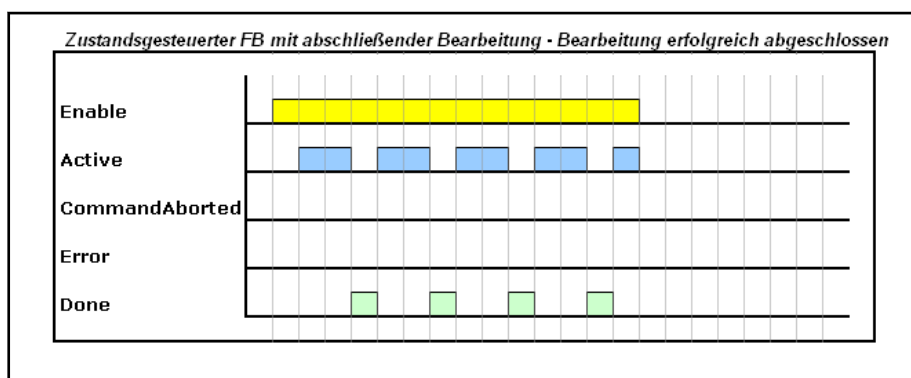


Abb. 6-120: Standard-Signal-Zeit-Diagramm

6.10.12 MB_ReadLrealParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen, Seite 280](#).

ML_PLCopen.library



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsguppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadLrealParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#) , Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#) , Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.13 MB_ReadMaxRealValue

Kurzbeschreibung

Der Funktionsbaustein gibt den zulässigen Maximalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als REAL-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-107: Bibliothekszuordnung

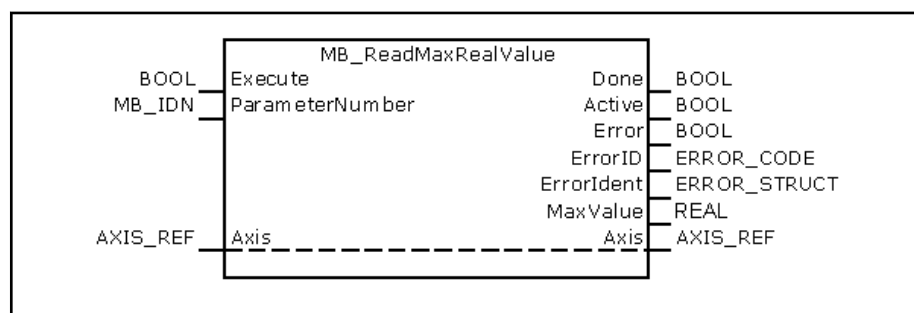
Schnittstellenbeschreibung

Abb. 6-121: FB MB_ReadMaxRealValue

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsguppe) oder eines anderen Objektes , Seite 278)
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

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	MaxValue	Real	Ist "Done"=TRUE, zeigt "MaxValue" den zulässigen maximalen Eingabewert des ausgewählten Parameters an.

Tab. 6-108: Schnittstelle von FB MB_ReadMaxRealValue

Kurzbeschreibung Signal-Zeit-Diagramm

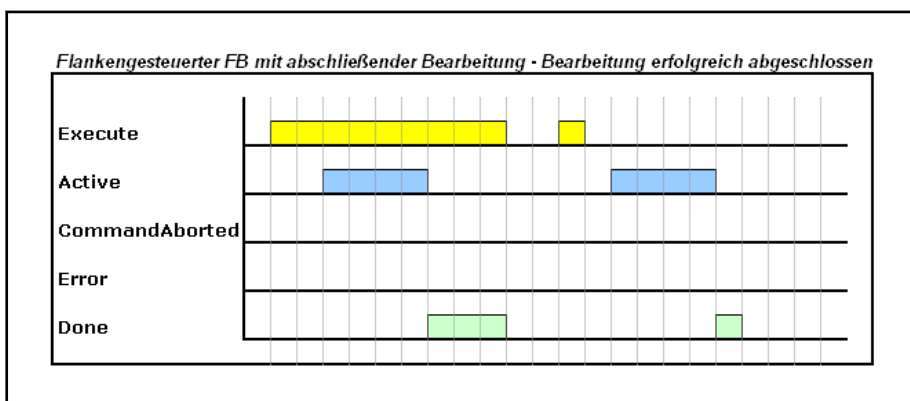


Abb. 6-122: Standard-Signal-Zeit-Diagramm

6.10.14 MB_ReadMaxRealValue, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadMaxRealValue erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.15 MB_ReadMaxValue

Kurzbeschreibung

Der Funktionsbaustein gibt den zulässigen Maximalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

ML_PLCopen.library

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-109: Bibliothekszuordnung

Schnittstellenbeschreibung

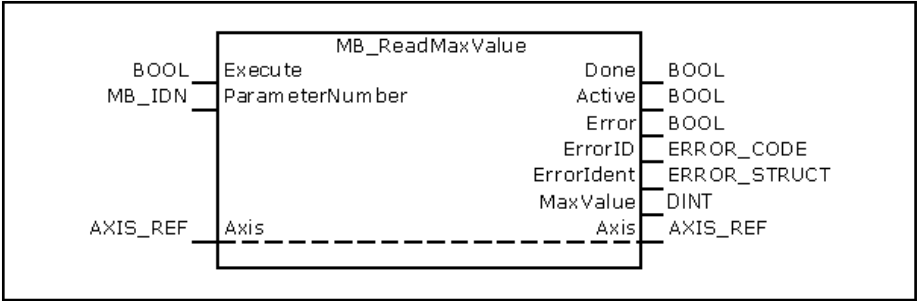


Abb. 6-123: FB MB_ReadMaxValue

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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
	MaxValue	DINT	Ist "Done"=TRUE, zeigt "MaxValue" den zulässigen maximalen Eingabewert des ausgewählten Parameters an.

Tab. 6-110: Schnittstelle von FB MB_ReadMaxValue

Kurzbeschreibung
Signal-Zeit-Diagramm

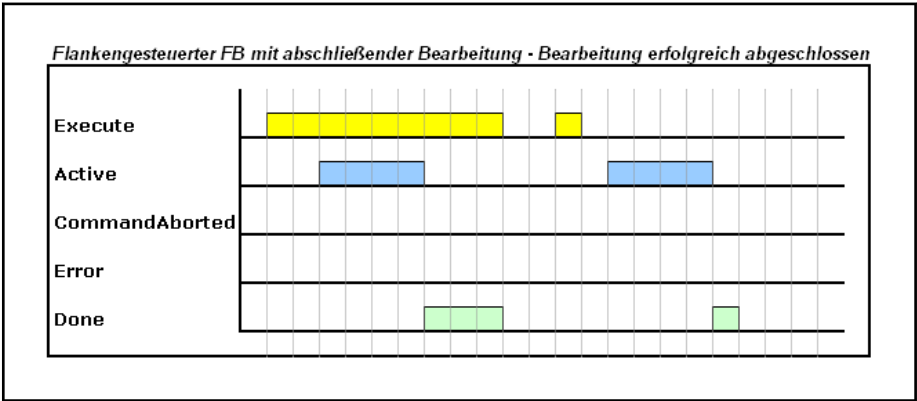


Abb. 6-124: Standard-Signal-Zeit-Diagramm

6.10.16 MB_ReadMaxValue, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadMaxValue erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.17 MB_ReadMinRealValue

Kurzbeschreibung

Der Funktionsbaustein gibt den zulässigen Minimalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als REAL-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-111: Bibliothekszuordnung

Schnittstellenbeschreibung

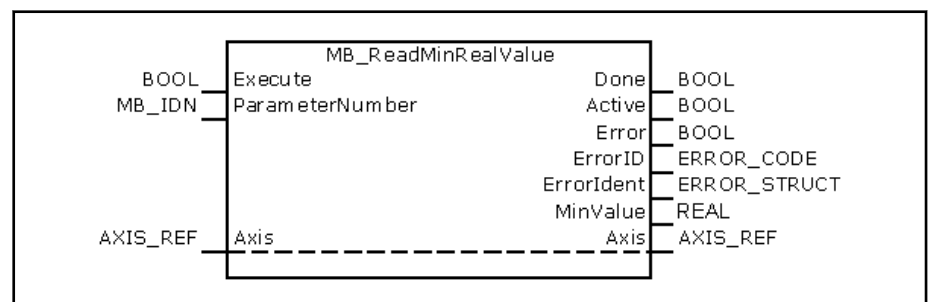


Abb. 6-125: FB MB_ReadMinRealValue

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.

ML_PLCopen.library

	Name	Typ	Kommentar
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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
	MinValue	REAL	Ist "Done"=TRUE, zeigt "MinValue" den zulässigen minimalen Eingabewert des ausgewählten Parameters an.

Tab. 6-112: Schnittstelle von FB MB_ReadMinRealValue

Kurzbeschreibung
Signal-Zeit-Diagramm

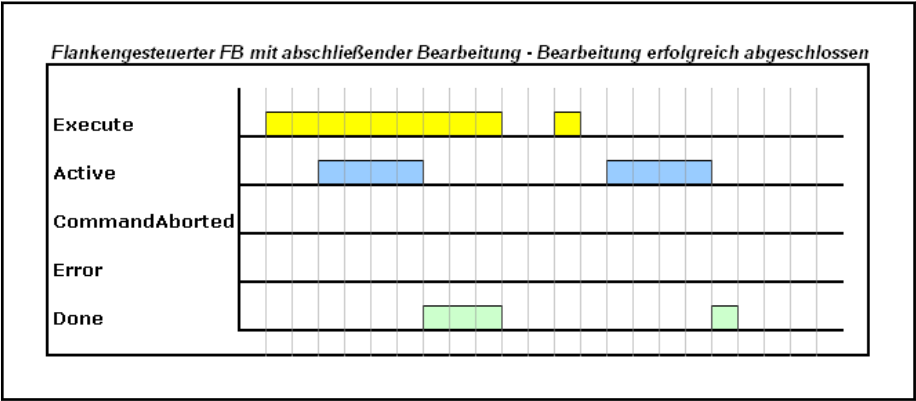


Abb. 6-126: Standard-Signal-Zeit-Diagramm

6.10.18 MB_ReadMinRealValue, Besonderheiten bei SERCOS-basierten Achsen

Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen, Seite 280](#).

Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsgruppe\) oder eines anderen Objektes, Seite 278](#).

Fehlerbehandlung Der Funktionsbaustein MB_ReadMinRealValue erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010 , Seite 346](#) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030 , Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.19 MB_ReadMinValue

Kurzbeschreibung Der Funktionsbaustein gibt den zulässigen Minimalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-113: Bibliothekszuordnung

Schnittstellenbeschreibung

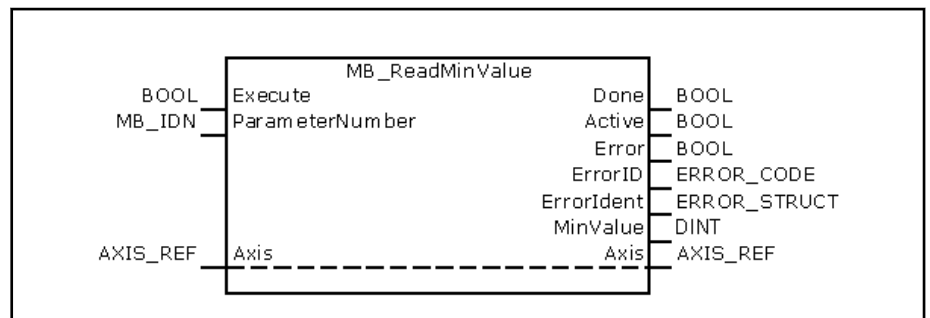


Abb. 6-127: FB MB_ReadMinValue

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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
	MinValue	DINT	Ist "Done"=TRUE, zeigt "MinValue" den zulässigen minimalen Eingabewert des ausgewählten Parameters an.

Tab. 6-114: Schnittstelle von FB MB_ReadMinValue

Kurzbeschreibung

ML_PLCopen.library

Signal-Zeit-Diagramm

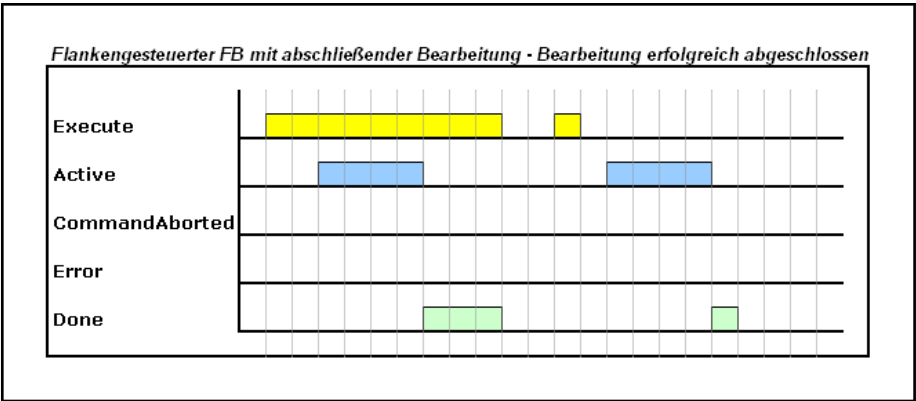


Abb. 6-128: Standard-Signal-Zeit-Diagramm

6.10.20 MB_ReadMinValue, Besonderheiten bei SERCOS-basierten Achsen

 Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.

 Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung Der Funktionsbaustein MB_ReadMinValue erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#) , Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#) , Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.21 MB_ReadName

Kurzbeschreibung Der Funktionsbaustein gibt den Namen eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als Text dargestellt.

 Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute " voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-115: Bibliothekszuordnung

Schnittstellenbeschreibung

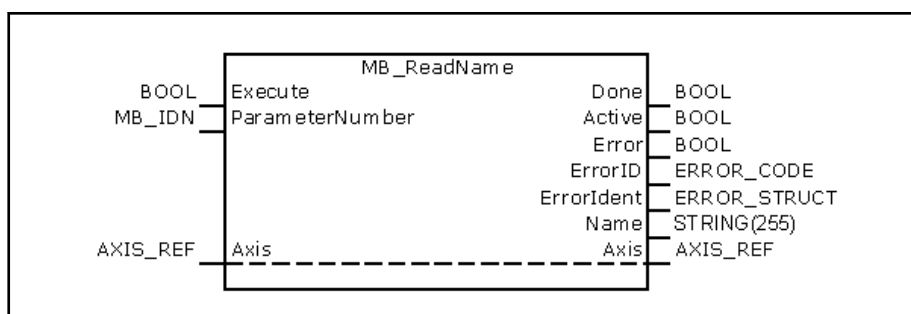


Abb. 6-129: FB MB_ReadName

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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
	Name	STRING(255)	Ist "Done"=TRUE, zeigt "MinValue" den Namen des ausgewählten Parameters an.

Tab. 6-116: Schnittstelle von FB MB_ReadName

Signal-Zeit-Diagramm

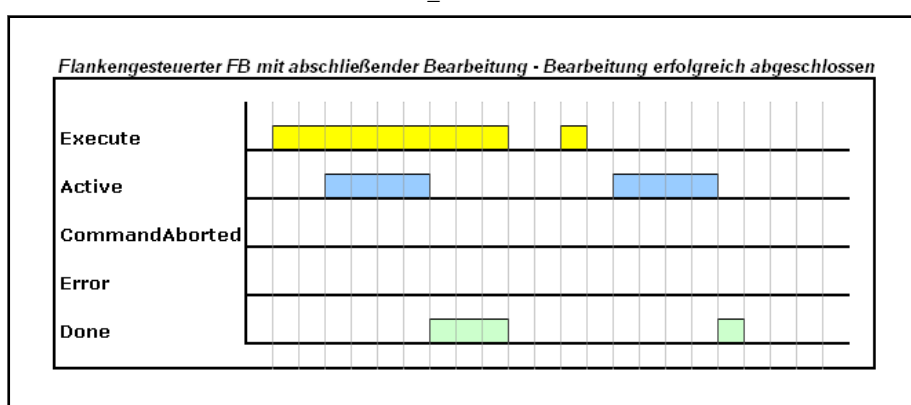


Abb. 6-130: Standard-Signal-Zeit-Diagramm

6.10.22 MB_ReadName, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen, Seite 280](#).

ML_PLCopen.library



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadName erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#) , Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#) , Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.23 MB_ReadParameter

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-117: Bibliothekszuordnung

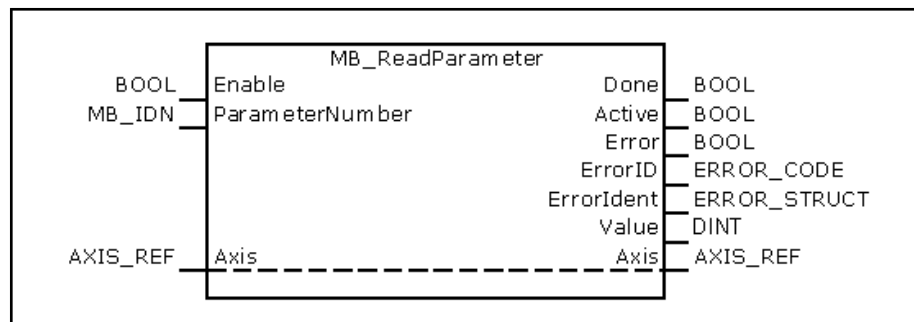
Schnittstellenbeschreibung

Abb. 6-131: FB MB_ReadParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse, bzw. zum adressierten Objekt.
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsegruppe) oder eines anderen Objektes , Seite 278)
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

	Name	Typ	Kommentar
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Value	DINT	4-Byte-Wert, als DINT dargestellt

Tab. 6-118: Schnittstelle von FB MB_ReadParameter

Signal-Zeit-Diagramm

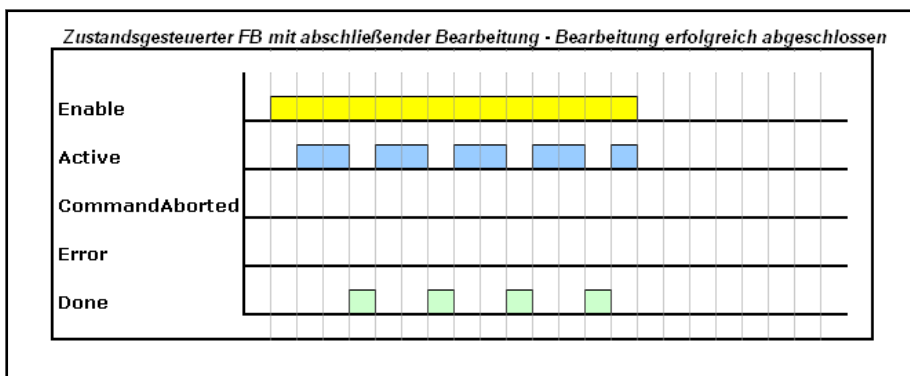


Abb. 6-132: Standard-Signal-Zeit-Diagramm

6.10.24 MB_ReadParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.25 MB_ReadRealParameter

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als REAL-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-119: Bibliotheksuzuordnung

ML_PLCOpen.library

Schnittstellenbeschreibung

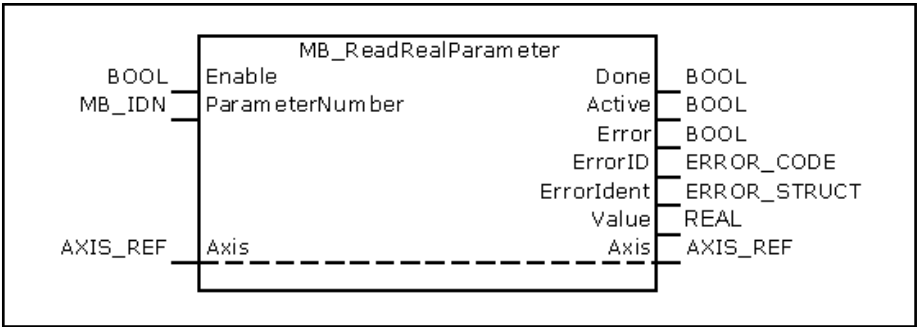


Abb. 6-133: *FB MB_ReadRealParameter*

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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. 6-120: *Schnittstelle von FB MB_ReadRealParameter*

Signal-Zeit-Diagramm

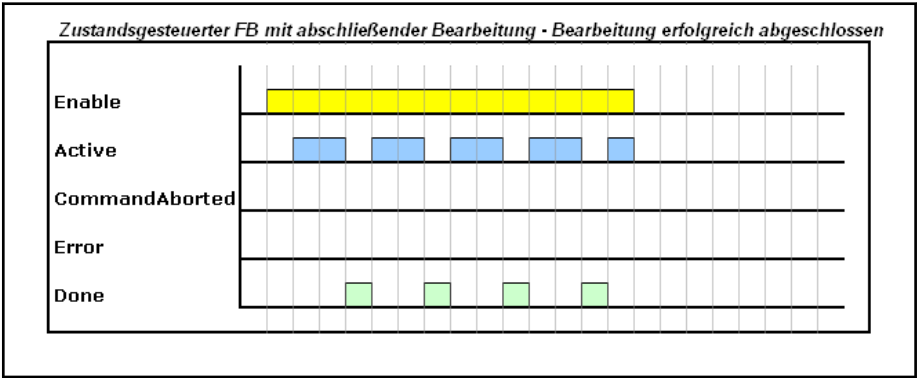


Abb. 6-134: *Standard-Signal-Zeit-Diagramm*

6.10.26 **MB_ReadRealParameter, Besonderheiten bei SERCOS-basierten Achsen**



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen, Seite 280](#).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes, Seite 278](#).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadRealParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010, Seite 346](#) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.27 MB_ReadSercosAttribute

Kurzbeschreibung

Der Funktionsbaustein erlaubt das Lesen der SERCOS-Attribute eines Parameters. Der Rückgabewert wird als DWORD dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-121: Bibliothekszuordnung

Schnittstellenbeschreibung

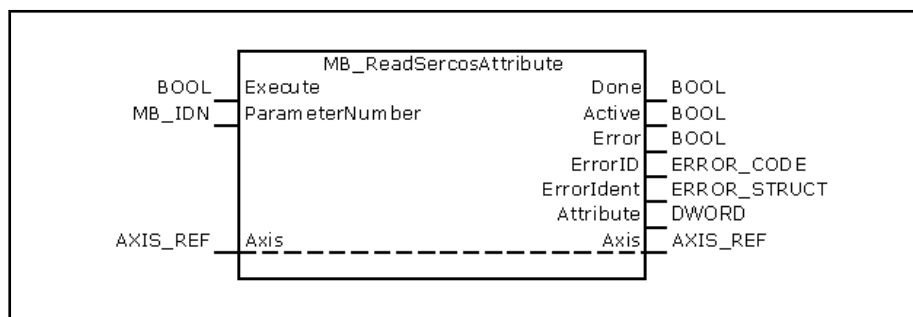


Abb. 6-135: FB MB_ReadSercosAttribute

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Lesen des Parameter-Attributes, bei 0-1-Übergang
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsegruppe) oder eines anderen Objektes, Seite 278)
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

ML_PLCopen.library

	Name	Typ	Kommentar
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Attribute	DWORD	Ist "Done"=TRUE, zeigt "Attribute" das Attribut des ausgewählten Parameters an

Tab. 6-122: Schnittstelle von FB MB_ReadSercosAttribute

Dekodierung des Attributs

Zur Dekodierung wird in der Bibliothek "ML_Base.library" der Funktionsbaustein "IL_SercosAttribute" bereitgestellt.

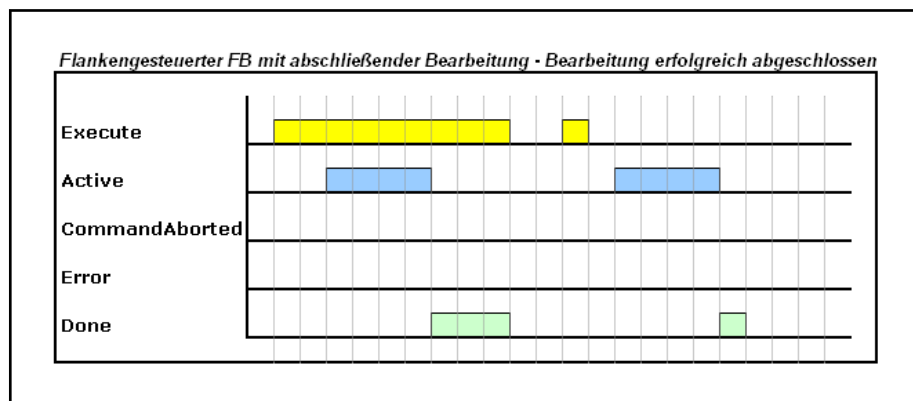
Signal-Zeit-Diagramm

Abb. 6-136: Standard-Signal-Zeit-Diagramm

6.10.28 MB_ReadSercosAttribute, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadSercosAttribute erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

Zusätzlich können auftreten:

6.10.29 MB_ReadSercosDataStatus

Kurzbeschreibung

Der Funktionsbaustein gibt den Status eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als ein ENUM-Wert (INT-Zahl) dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-123: Bibliothekszuordnung

Schnittstellenbeschreibung

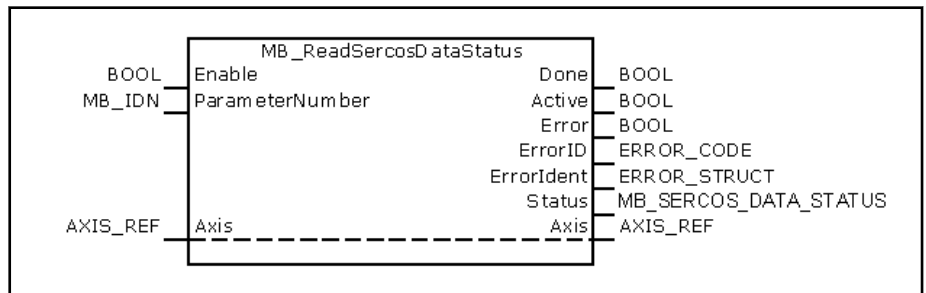


Abb. 6-137: FB MB_ReadSercosDataStatus

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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

Tab. 6-124: Schnittstelle von FB MB_ReadSercosDataStatus

Kurzbeschreibung

Mit Hilfe dieses Funktionsbausteines und eines [MB_WriteParameter, Seite 317](#), 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

ML_PLCopen.library

Signal-Zeit-Diagramm

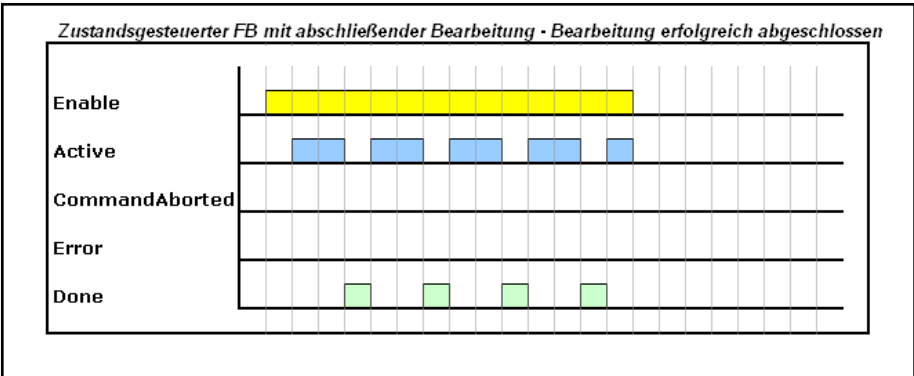


Abb. 6-138: Standard-Signal-Zeit-Diagramm

6.10.30

MB_ReadSercosDataStatus, Besonderheiten bei SERCOS-basierten Achsen

Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.

Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadSercosDataStatus erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#) , Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#) , Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.31

MB_ReadStringParameter

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als Zeichenkette (STRING) entsprechend seiner tatsächlichen Länge mit maximal 255 Zeichen dargestellt.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-125: Bibliothekszuordnung

Schnittstellenbeschreibung

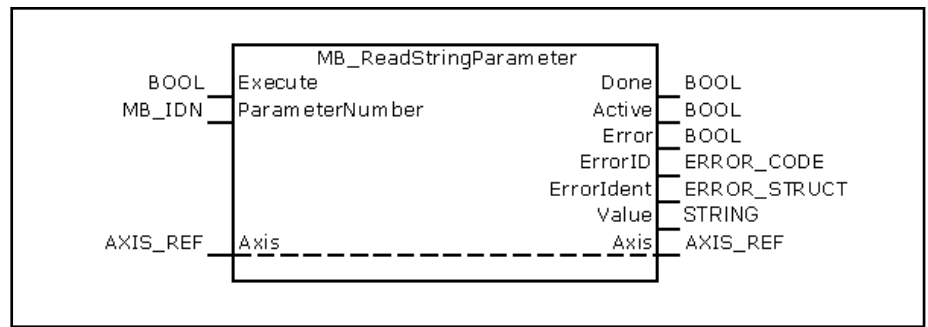


Abb. 6-139: FB MB_ReadStringParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Lesen des Textes vom Parameter mit steigender Flanke
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
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	STRING	Zeichenkette mit maximal 255 Zeichen

Tab. 6-126: Schnittstelle von FB MB_ReadStringParameter

Signal-Zeit-Diagramm

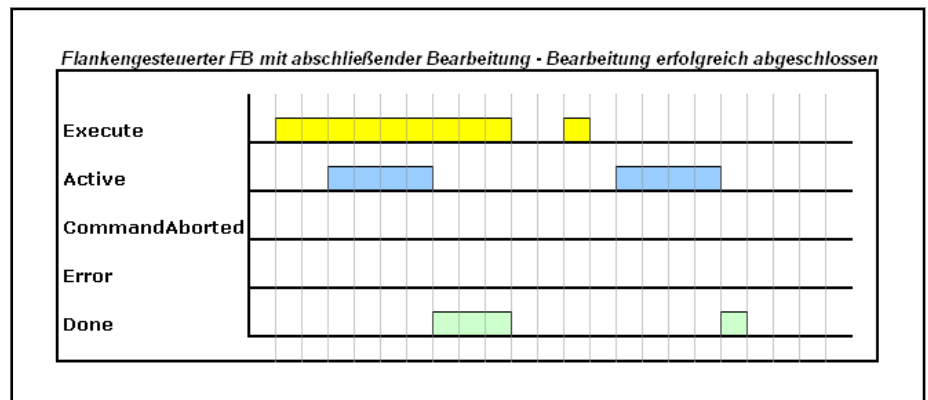


Abb. 6-140: Standard-Signal-Zeit-Diagramm

6.10.32 MB_ReadStringParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen, Seite 280](#).

ML_PLCopen.library



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes, Seite 278](#).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadStringParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010, Seite 346](#) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.33 MB_ReadUnit**Kurzbeschreibung**

Der Funktionsbaustein gibt die Maßeinheit eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als Text dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-127: Bibliothekszuordnung

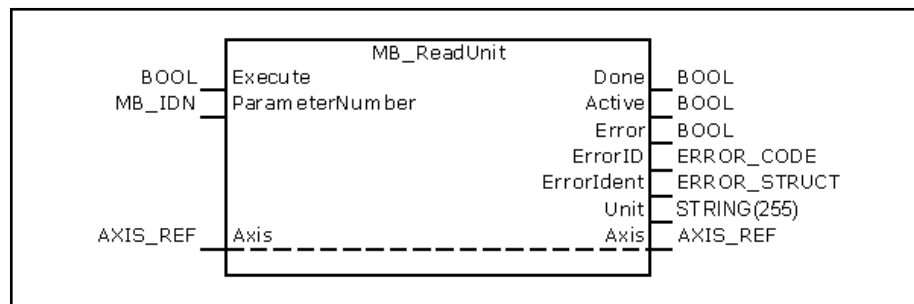
Schnittstellenbeschreibung

Abb. 6-141: FB MB_ReadUnit

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsegruppe) oder eines anderen Objektes, Seite 278)
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

	Name	Typ	Kommentar
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Unit	STRING(255)	Ist "Done"=TRUE, zeigt "Unit" die Masseinheit des ausgewählten Parameters an.

Tab. 6-128: Schnittstelle von FB MB_ReadUnit

Signal-Zeit-Diagramm

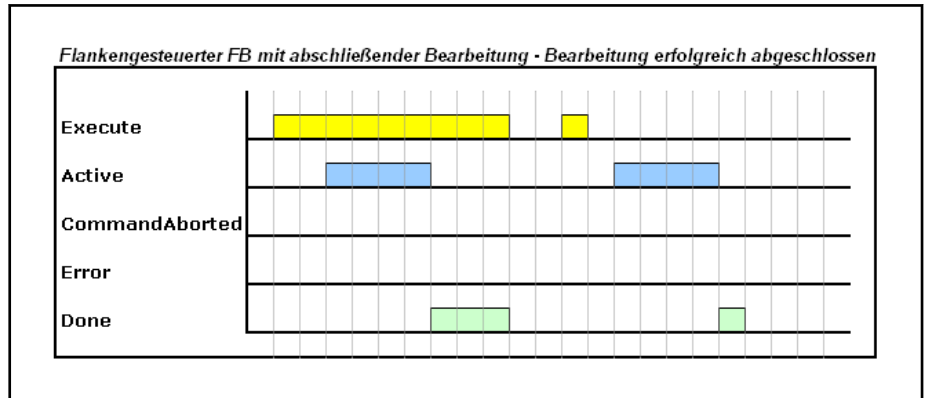


Abb. 6-142: Standard-Signal-Zeit-Diagramm

6.10.34 MB_ReadUnit, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
 - des AxisGroup-Eingangs bei K-Parametern
- siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_ReadUnit erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.35 MB_WriteLintParameter

Kurzbeschreibung

Der Funktionsbaustein schreibt einen Wert in einen durch den Anwender vorgegebenen Parameter.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-129: Bibliothekszuordnung



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsguppe\) oder eines anderen Objektes, Seite 278](#).

Fehlerbehandlung

Der Funktionsbaustein MB_WriteListParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010 , Seite 346](#) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030 , Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.37 MB_WriteListParameter**Kurzbeschreibung**

Der Funktionsbaustein schreibt Speicherinhalt definierter Länge (NoOfByte), der ab der Anfangsadresse (ValueAdr) zur Verfügung gestellt wurde, in einen anwenderspezifisierten 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 sie zu klein, fehlen Daten im Zielparameter!



Der Datentyp der Elemente des zu schreibenden Parameters und der Datentyp der Elemente des ARRAYs müssen übereinstimmen.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-131: Bibliothekszuordnung

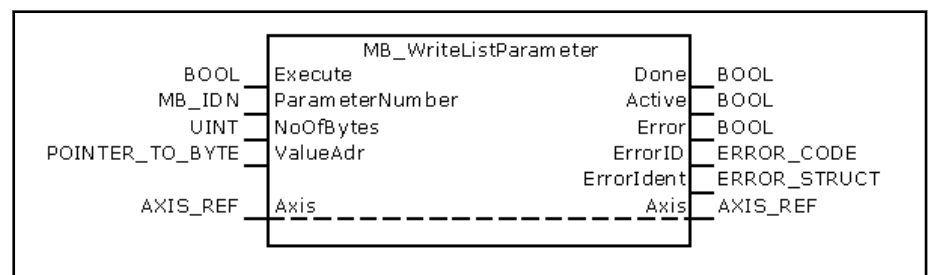
Schnittstellenbeschreibung

Abb. 6-145: FB MB_WriteListParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsguppe) oder eines anderen Objektes, Seite 278)

ML_PLCopen.library

	Name	Typ	Kommentar
	NoOfBytes	UINT	Anzahl der Bytes, die geschrieben werden sollen
	ValueAdr	POINTER TO BYTE	Pointer auf Anfangsadresse des Speichers in dem die zu schreiben- den 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. 6-132: Schnittstelle von FB MB_WriteListParameter

Signal-Zeit-Diagramm

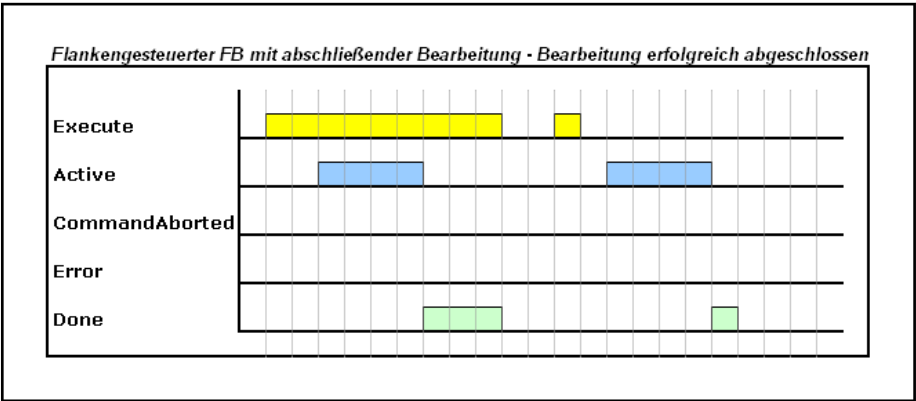


Abb. 6-146: Standard-Signal-Zeit-Diagramm

Programmbeispiel
MB_WriteListParameter

Dieses Programmbeispiel zeigt das Schreiben des Listenparameters (C-0-2001) der MLC mit 1024 Werten (DINT) aus dem WriteListArray.
Programm:

```
PROGRAM WriteListParameter
VAR IN_OUT
  Axis:          AXIS_REF;          (* Antrieb1 is a variable of AXIS_REF*)
END_VAR

VAR
  WriteListArray:  ARRAY [0..1023] OF DINT; (* Data array for the write bytes *)

  FB_WriteListParameter: MB_WriteListParameter;
  bExecute_WLP:          BOOL;          (* Start WriteListParameter *)
  diParameterNo_WLP:     MB_IDN:= FP C_0_2001; (* Parameter ControlCAM1 der MLC *)
  uiNoOfBytes_WLP:       UINT := 4096;  (* DINT := 4 Bytes *)
  bDone_WLP:             BOOL;          (* TRUE := FB ready *)
  bActive_WLP:           BOOL;          (* The FB works *)
  bError_WLP:            BOOL;          (* Error occur *)
  enErrorID_WLP:         ERROR_CODE;    (* Error class *)
  stErrorIdent_WLP:      ERROR_STRUCT;  (* Error definition *)
END_VAR
```

Programmbeispiel im Funktionsplan

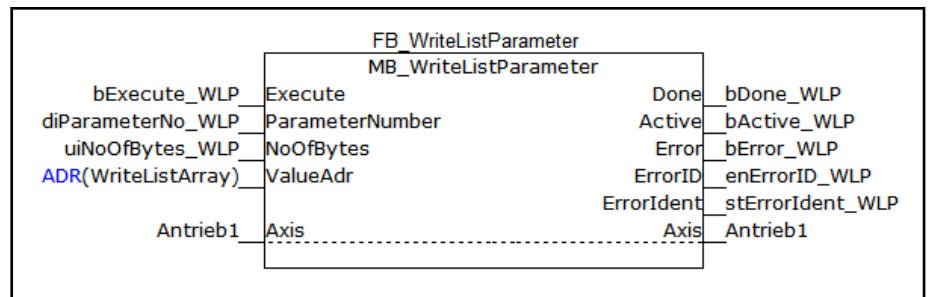


Abb. 6-147: FUP Programmbeispiel WriteListParameter

Programmbeispiel im Strukturierten Text

Programm:

```

FB_WriteListParameter(
  _Execute:=      bExecute_WLP,      (* Start WriteListParameter *)
  ParameterNumber:= diParameterNo_WLP, (* Parameter ControlCAM1 der MLC *)
  NoOfBytes:=     uiNoOfBytes_WLP,    (* DINT := 4 Bytes *)
  ValueAdr:=      ADR(WriteListArray), (* Adresse des WriteListArray *)
  Axis:=          Antrieb1,           (* Antriebsadresse *)
  Done=>          bDone_WLP,          (* TRUE := FB ready *)
  Active=>        bActive_WLP,        (* The FB works *)
  Error=>         bError_WLP,         (* Error occur *)
  ErrorId=>       enErrorID_WLP,      (* Error class *)
  ErrorIdent=>    stErrorIdent_WLP:   (* Error definition *)

```

6.10.38 MB_WriteListParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsen-Gruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_WriteListParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.39 MB_WriteLrealParameter

Kurzbeschreibung

Der Funktionsbaustein schreibt einen Wert in einen anwenderspezifischen Parameter.

ML_PLCopen.library

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-133: Bibliothekszuordnung

Schnittstellenbeschreibung

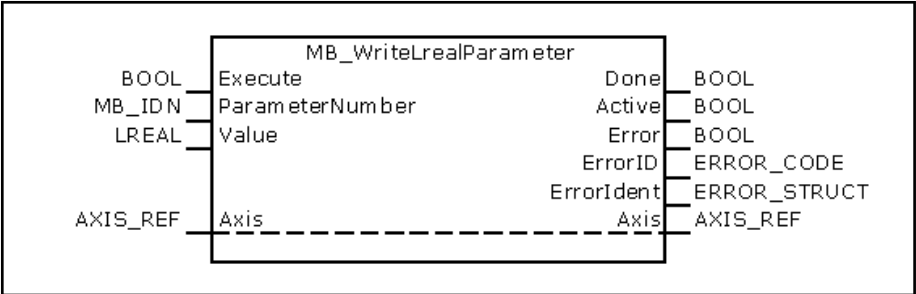


Abb. 6-148: FB MB_WriteLrealParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objektes, Seite 278)
	Value	LREAL	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. 6-134: Schnittstelle von FB MB_WriteRealParameter

Signal-Zeit-Diagramm

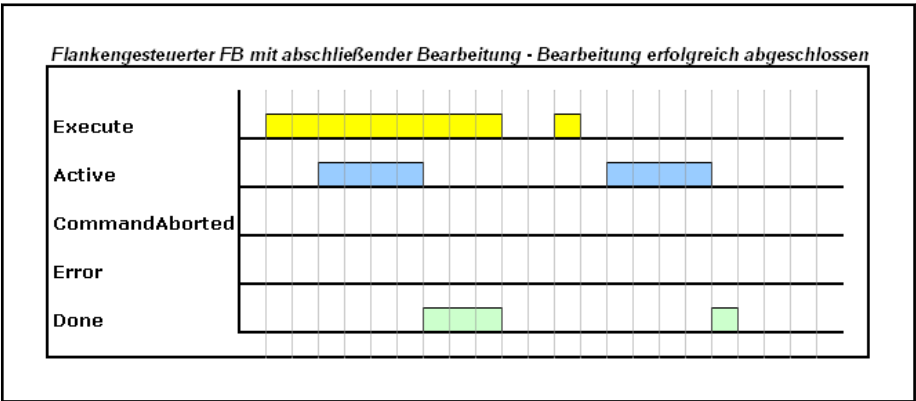


Abb. 6-149: Standard-Signal-Zeit-Diagramm

6.10.40 MB_WriteLrealParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_WriteRealParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.41 MB_WriteParameter

Kurzbeschreibung

Der Funktionsbaustein schreibt einen Wert in einen durch den Anwender vorgegebenen Parameter.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-135: Bibliothekszuordnung

Schnittstellenbeschreibung

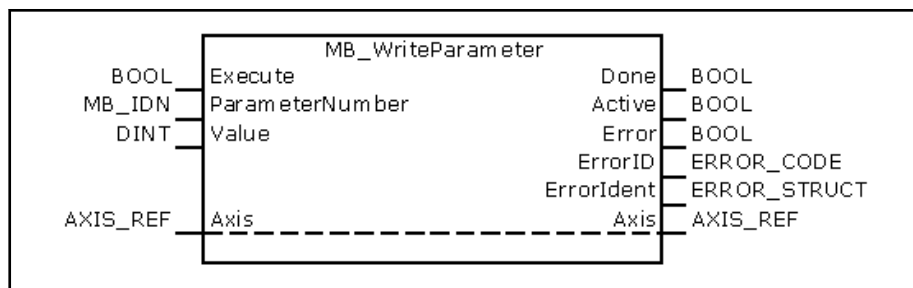


Abb. 6-150: FB MB_WriteParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsegruppe) oder eines anderen Objektes , Seite 278)
	Value	DINT	Wert, der in den Parameter geschrieben werden soll
VAR_OUTPUT	Done	BOOL	Wert erfolgreich in die Steuerung / den Antrieb geschrieben

ML_PLCOpen.library

	Name	Typ	Kommentar
	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. 6-136: Schnittstelle von FB MB_WriteParameter

Signal-Zeit-Diagramm

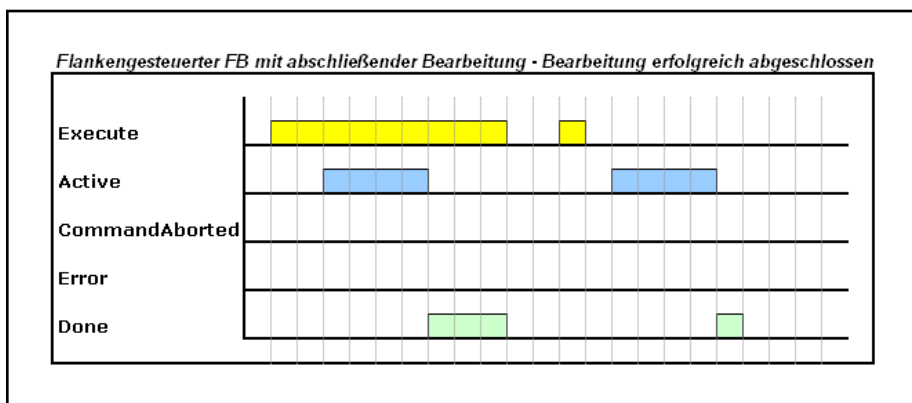


Abb. 6-151: Standard-Signal-Zeit-Diagramm

6.10.42 MB_WriteParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_WriteParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.43 MB_WriteRealParameter

Kurzbeschreibung

Der Funktionsbaustein schreibt einen Wert in einen anwenderspezifisierten Parameter.

Bibliothek	Bereich
ML_PLCOpen.library	Parameter

Tab. 6-137: Bibliothekszuordnung

Schnittstellenbeschreibung

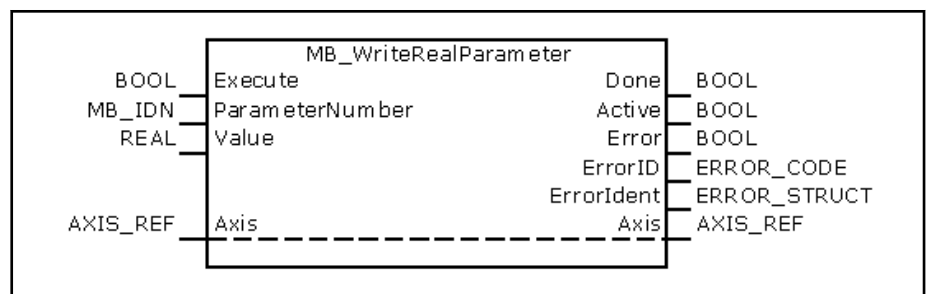


Abb. 6-152: FB MB_WriteRealParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsguppe) oder eines anderen Objektes, Seite 278)
	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. 6-138: Schnittstelle von FB MB_WriteRealParameter

Signal-Zeit-Diagramm

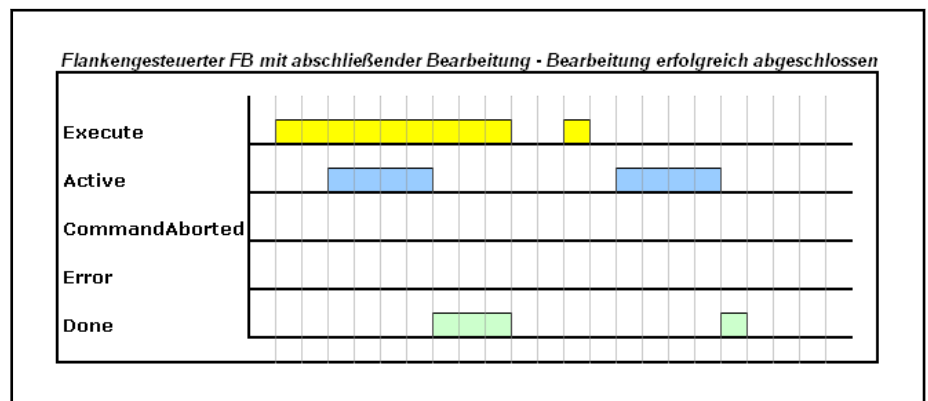


Abb. 6-153: Standard-Signal-Zeit-Diagramm

6.10.44 MB_WriteRealParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen, Seite 280](#).

ML_PLCopen.library



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsguppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung Der Funktionsbaustein MB_WriteRealParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#) , Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#) , Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.45 MB_WriteStringParameter

Kurzbeschreibung Der Funktionsbaustein schreibt eine Zeichenkette mit max. 255 Zeichen in einen anwenderspezifizierten Parameter.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-139: Bibliothekszuordnung

Schnittstellenbeschreibung

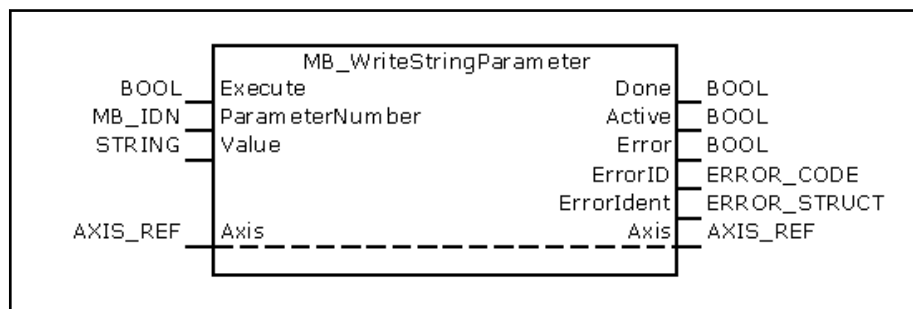


Abb. 6-154: FB MB_WriteStringParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsguppe) oder eines anderen Objektes , Seite 278)
	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. 6-140: Schnittstelle von FB MB_WriteStringParameter

Signal-Zeit-Diagramm

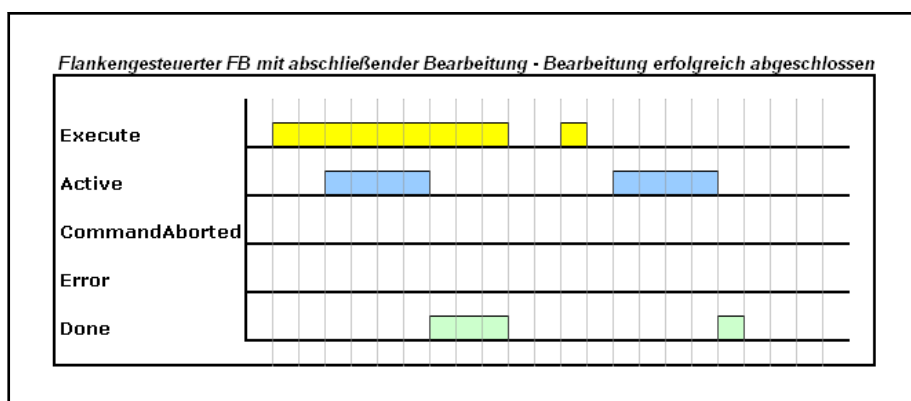


Abb. 6-155: Standard-Signal-Zeit-Diagramm

6.10.46 MB_WriteStringParameter, Besonderheiten bei SERCOS-basierten Achsen



Zum Zeitbedarf für die Datenübertragung siehe [Zeitbedarf für den Datentransport mit Read- und WriteParameter-Funktionsbausteinen](#), Seite 280.



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

siehe [Zugriff auf Daten einer Achse, Steuerung, Kinematik \(Achsegruppe\) oder eines anderen Objektes](#), Seite 278.

Fehlerbehandlung

Der Funktionsbaustein MB_WriteStringParameter erzeugt folgende Arten von Fehlermeldungen:

- [SERCOS_TABLE, ERROR_TABLE 16#0010](#), Seite 346 - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- [MLC_TABLE, ERROR_TABLE 16#0030](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.47 MC_ReadActualPosition

Kurzbeschreibung

Der Funktionsbaustein liest die aktuelle Position des Antriebes.



Eine Änderung der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-141: Bibliotheksuzuordnung

ML_PLCopen.library

Schnittstellenbeschreibung

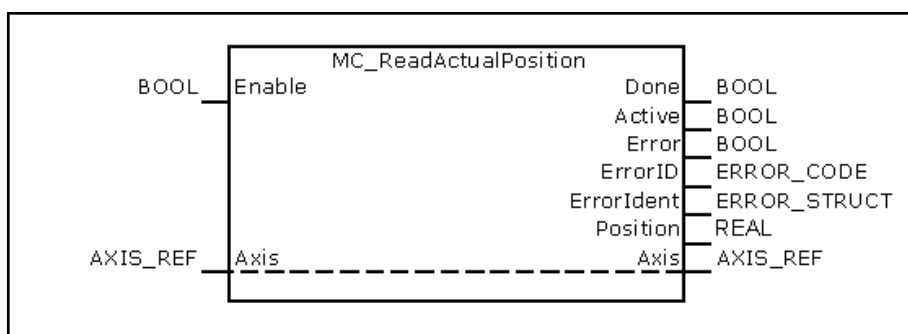


Abb. 6-156: FB MC_ReadActualPosition

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	kontinuierliches Lesen des Positionswertes, solange "Enable" TRUE ist
VAR_OUTPUT	Done	BOOL	gültiger Wert ist 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	REAL	aktuelle absolute Position des Antriebs ¹¹⁾

Tab. 6-142: Schnittstelle von FB MC_ReadActualPosition

Signal-Zeit-Diagramm

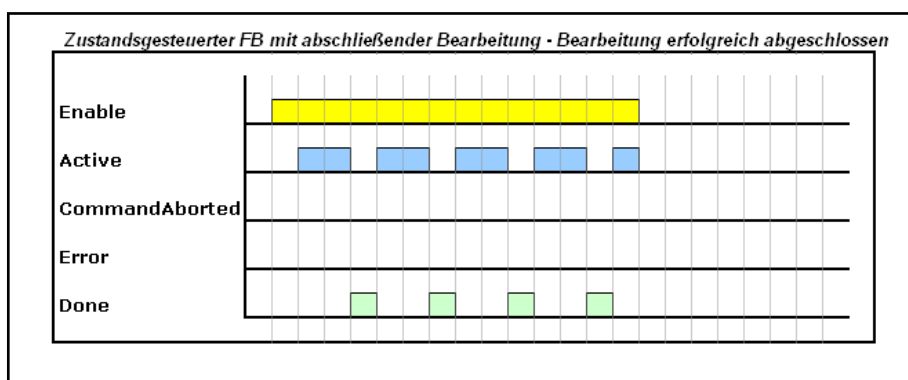


Abb. 6-157: Standard-Signal-Zeit-Diagramm

6.10.48 MC_ReadActualPosition, Besonderheiten bei SERCOS-basierten Achsen

Verwendete Parameter Der Funktionsbaustein liest die aktuelle Position des Antriebes (A-0-0100, Positionswert).

Fehlerbehandlung Der Funktionsbaustein MC_ReadActualPosition erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

¹¹⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

6.10.49 MC_ReadAxisError

Kurzbeschreibung Der Funktionsbaustein ruft den Fehlerstatus der adressierten Achse ab. Je Achse ist dazu eine Instanz des Funktionsbausteines zu verwenden.



Eine Änderung der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-143: Bibliothekszuordnung

Schnittstellenbeschreibung

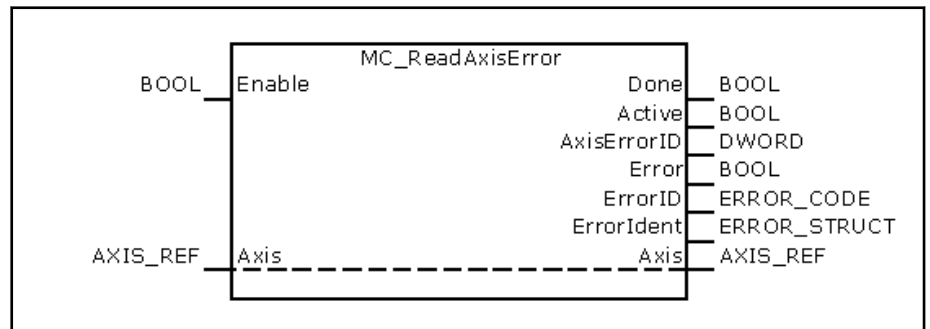


Abb. 6-158: FB MC_ReadAxisError

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	kontinuierliches Lesen der Fehlerinformation, solange "Enable" TRUE ist
VAR_OUTPUT	Done	BOOL	Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	AxisErrorID	DWORD	Zeigt den Fehler des Antriebs der Achse an
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache des Bausteinaufrufes, nicht des Achsfehlers!
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler des Bausteinaufrufes, nicht des Achsfehlers!

Tab. 6-144: Schnittstelle von FB MC_ReadActualPosition

Signal-Zeit-Diagramm

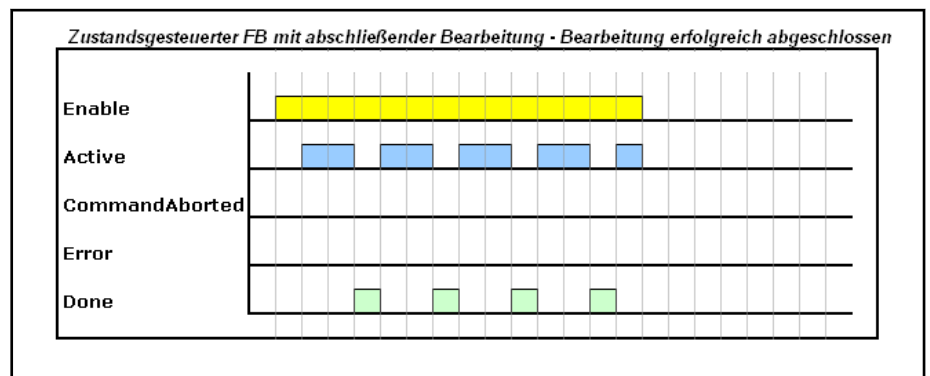


Abb. 6-159: Standard-Signal-Zeit-Diagramm

ML_PLCopen.library

6.10.50 MC_ReadAxisError, Besonderheiten bei SERCOS-basierten Achsen

Verwendete Parameter Der Funktionsbaustein ruft den Fehlerstatus (A-0-0020, Achsdiagnosemeldung) der adressierten Achse ab.

Fehlerbehandlung Der Funktionsbaustein MC_ReadAxisError erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.10.51 MC_ReadStatus

Kurzbeschreibung Dieser Funktionsbaustein liest den aktuellen Status der Achse und gibt diesen dekodiert aus.



Eine Änderung der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.



Durch die Erweiterung des Bausteins wurde dieser inkompatibel zur Vorversion geändert!

Bibliothek	Bereich
ML_PLCopen.library	Parameter

Tab. 6-145: Bibliothekszuordnung

Schnittstellenbeschreibung

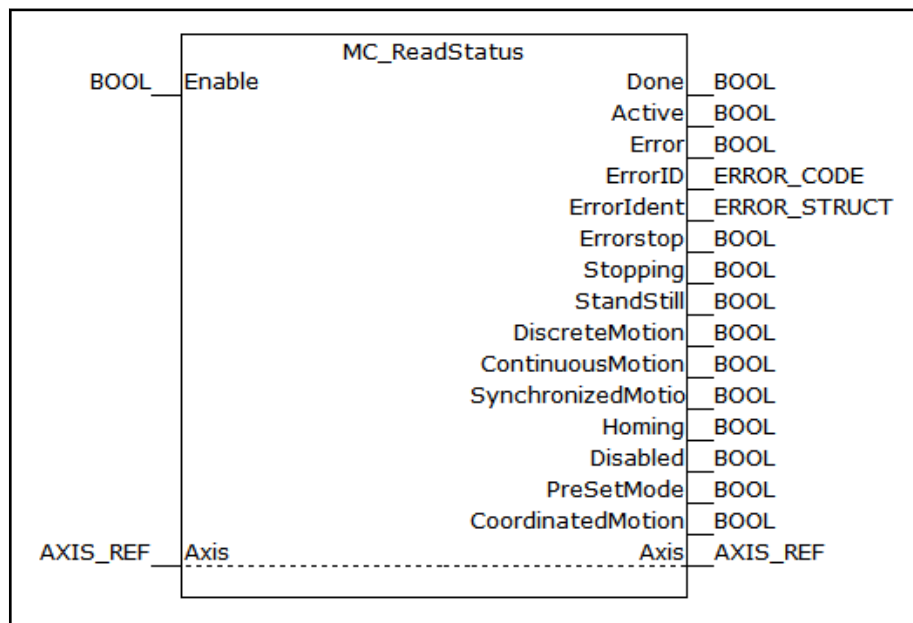


Abb. 6-160: FB MC_ReadStatus

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	kontinuierliches Lesen des Achsstatus, solange "Enable" TRUE ist
VAR_OUTPUT	Done	BOOL	gültiger Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Error	BOOL	zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	Errorstop	BOOL	Antrieb/Achse im Zustand 'Errorstop'
	Stopping	BOOL	Antrieb/Achse im Zustand 'Stopping'
	StandStill	BOOL	Antrieb/Achse im Zustand 'StandStill'
	Discrete-Motion	BOOL	Antrieb/Achse im Zustand 'DiscreteMotion'
	Continuous-Motion	BOOL	Antrieb/Achse im Zustand 'ContinuousMotion'
	SynchronizedMotion	BOOL	Antrieb/Achse im Zustand 'SynchronizedMotion'
	Homing	BOOL	Antrieb im Zustand 'Homing'
	Disabled	BOOL	Antrieb/Achse im Zustand 'Disabled'
	PreSetMode	BOOL	Antrieb im Zustand 'PreSetMode'
	Coordinated-Motion	BOOL	Antrieb/Achse im Zustand 'CoordinatedMotion'

Tab. 6-146: Schnittstelle von FB MC_ReadStatus

Signal-Zeit-Diagramm

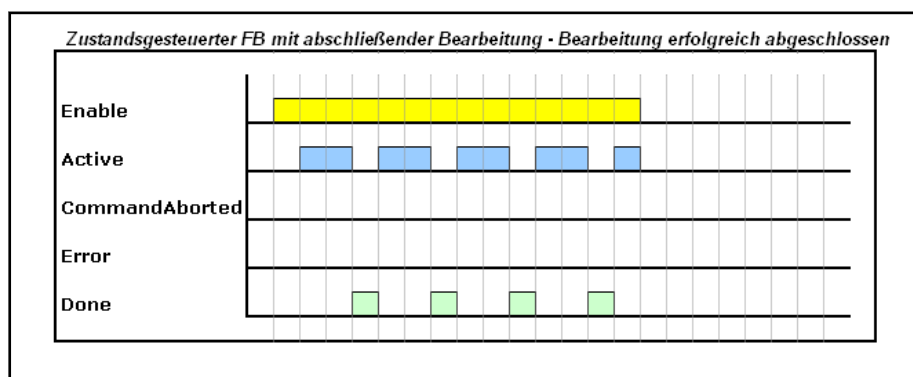


Abb. 6-161: Standard-Signal-Zeit-Diagramm

6.10.52 MC_ReadStatus, Besonderheiten bei sercos basierten Achsen

Verwendete Parameter Dieser Funktionsbaustein liest den aktuellen Status der Achse. (A-0-0021, Achsstatus; A-0-0022, Erweiterter Achsstatus) und gibt diesen dekodiert aus. (Vergleiche auch [Zustandsdiagramm einer realen Achse](#), Seite 173.)

Fehlerbehandlung Der Funktionsbaustein MC_ReadStatus erzeugt folgende Fehlermeldungen:

- [MLC_TABLE](#), Seite 348 - Fehler beim Zugriff auf die Achse/ die Steuerung.

6.11 Funktionsbausteine zur Nutzung der Messtaster-Funktion

6.11.1 Daten und Funktionsbausteine zur Nutzung der Messtaster-Funktion

Die Messtaster-Funktion erlaubt mittels eines binären Sensors (Messtaster) eine hochgenaue Positionserfassung im laufenden Fertigungsprozess.

ML_PLCopen.library

Beispielsweise können durch den Messtaster auf einer zu verarbeitenden Bahn die Druckmarken erfasst werden (positive Flanke / negative Flanke oder beide).

Ziel für den Hochlauf ist es, die Achse, der der Messtaster zugeordnet ist, in eine definierte Lage zu den Druckmarken zu bringen und im laufenden Fertigungsprozess zu halten.

Technologiebedingt pendeln die Druckmarken bezüglich der gewünschten Achsposition um einen Mittelwert.

Das Mess-Signal (positive Flanke / negative Flanke oder beide) wird zweckmässig also in einem Bereich um den gewünschten Wert erwartet. Das Fenster kann dabei für die gewünschte Flanke absolut, Startposition / Endposition des Erwartungsfensters, oder relativ, Mittelpunkt plus positive / negative Abweichung, vorgegeben werden.

Der Ausfall einer Druckmarke oder eine Abweichung der Flanke in eine Lage ausserhalb des Fensters gilt als Markenfehler, der erfasst und auf den reagiert werden muss.

Während des Fertigungsprozess ist es notwendig, die Lage und Breite des Erwartungsfensters verändern zu können.

Das Messtaster-Signal verspätet sich vom Ort der Erfassung (Messtaster) bis zum Ort der Verarbeitung (Steuerung). Diese Verspätung ist u.U. für die positive und die negative Flanke unterschiedlich und kann kompensiert werden.

Alternativ zum kontinuierlichen Betrieb soll eine Einzelmessung möglich sein. Desweiteren ist eine Zeitmessung zwischen zwei Ereignissen wünschenswert.

Auf der Basis der IndraDrive-Messtaster-Funktion wurden hardwaregestützte Messtaster-Funktionsbausteine entwickelt.

Globale Variablenliste und Datentypen

- [ProbeSelect, Seite 172](#), Globale Variablenliste, Daten der Messtaster 1..100.
- [PROBECONFIG, Seite 172](#), Datentyp, Daten der Messtaster 1..100.
- weitere Datentypen zur Nutzung der Messtaster-Funktion sind in der Bibliothek RIL_CommonTypes vereinbart.

Funktionsbausteine

- [ML_InitTouchProbe, Seite 328](#), dient zur Initialisierung und Konfiguration der Messtaster-Funktion.
- [ML_WriteExpectWindow, Seite 339](#) konfiguriert die Erwartungsfenster, ohne dass ein kompletter Initialisierungslauf mit dem ML_InitTouchProbe notwendig ist.
- [ML_TouchProbe, Seite 334](#), dient zur Aktivierung der einmaligen Position- und Zeit-Messung, außerdem bietet er eine automatisierte Zeitdifferenzmessung.
- [ML_TouchProbeContinuous, Seite 337](#), dient zur Aktivierung der kontinuierlichen Position- und Zeit-Messung, außerdem bietet er eine automatisierte Zeitdifferenzmessung.
- [ML_AbortTrigger, Seite 327](#), beendet sowohl eine Einzel- als auch eine kontinuierliche Messung.

6.11.2 ML_AbortTrigger

Kurzbeschreibung Der Funktionsbaustein ML_AbortTrigger beendet

- eine Einzelmessung, ausgelöst durch [ML_TouchProbe, Seite 334](#), oder
- eine kontinuierliche Messung, ausgelöst durch [ML_TouchProbeContinuos, Seite 337](#).

Bibliothek	Bereich
ML_PLCopen.library	TouchProbe

Tab. 6-147: Bibliothekszuordnung

Schnittstellenbeschreibung

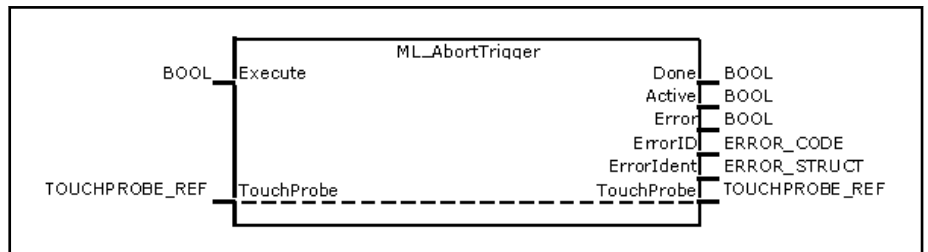


Abb. 6-162: Funktionsbaustein ML_AbortTrigger

	Name	Typ	Kommentar
VAR_IN_OUT	TouchProbe	TOUCHPROBE_REF	enthält Information zum konkreten Messtaster
VAR_INPUT	Execute	BOOL	startet den Unterbrechungsvorgang
VAR_OUTPUT	Done	BOOL	die Messung wurde erfolgreich beendet
	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

Tab. 6-148: Schnittstelle von FB ML_AbortTrigger

Signal-Zeit-Diagramm

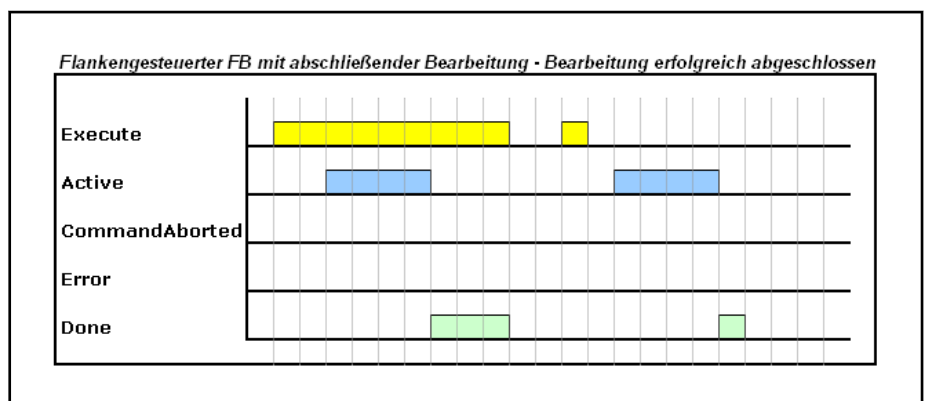


Abb. 6-163: Signal-Zeit-Diagramm: ML_AbortTrigger

Fehlerbehandlung: Der Funktionsbaustein ML_AbortTrigger erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf den Messtaster / die Steuerung.

ML_PLCopen.library

6.11.3 ML_InitTouchProbe

Kurzbeschreibung Der Funktionsbaustein ML_InitTouchProbe dient zur Initialisierung und Konfiguration der Messtaster-Funktion. Seine Abarbeitung ist Voraussetzung für den Start der Messtaster-Funktion über [ML_TouchProbe](#), [Seite 334](#), oder [ML_TouchProbeContinuous](#), [Seite 337](#).

Nach seiner Abarbeitung kann das Erwartungsfenster durch [ML_WriteExpectWindow](#), [Seite 339](#), online angepasst werden.

 Die Aktivierung der Messtaster-Funktion mit ML_InitTouchProbe ist in allen Sercos-Phasen möglich.

Die Funktionsbausteine ML_TouchProbe und ML_TouchProbeContinuous können jedoch nur in Sercos-Phase CP4 ausgeführt werden.

Ursache: der Parameter M-0-0024, Kommando: Aktivierung, ist nur noch in P4 beschreibbar. Das Ausführen der FBs in einer anderen Sercos-Phase würde zu dem Fehler F02D700C, sercos: Betriebsdatum ist schreibgeschützt (anderweitig), führen.

Bibliothek	Bereich
ML_PLCopen.library	TouchProbe

Tab. 6-149: Bibliothekszuordnung

Schnittstellenbeschreibung

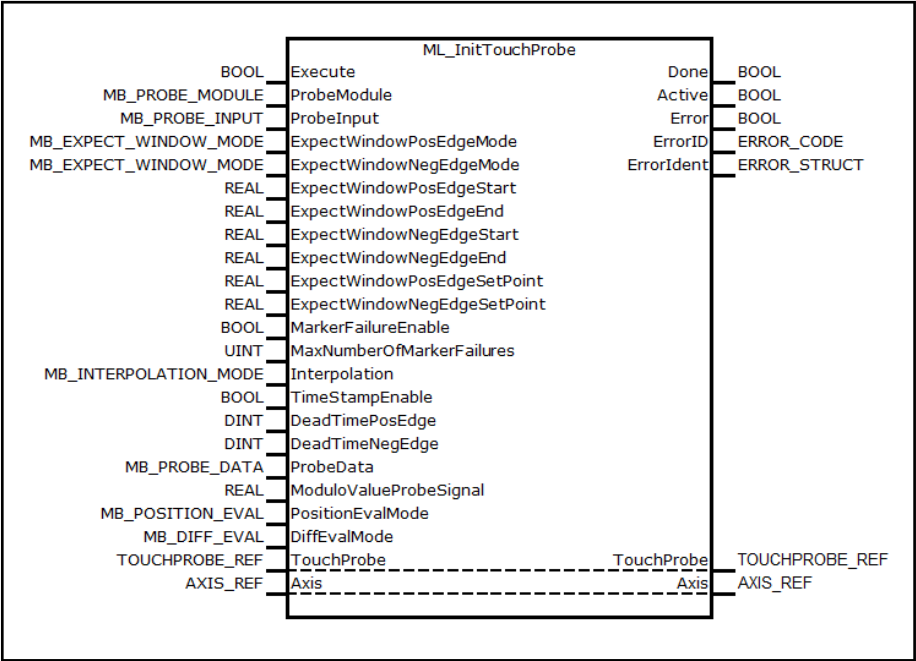


Abb. 6-164: Funktionsbaustein ML_InitTouchProbe

	Name	Typ	Kommentar ¹²⁾
VAR_IN_OUT	TouchProbe	TOUCHPROBE_REF	enthält Information zum konkreten Messtaster
	Axis	AXIS_REF	enthält Informationen zur konkreten Achse

¹²⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

	Name	Typ	Kommentar ¹²⁾
VAR_INPUT	Execute	BOOL	startet die Initialisierung bei steigenden Flanke
	ProbeModule	MB_PROBE_MODULE	Auswahl Eingangsmodule OnBoard-Modul, Funktionsmodul 1...4 M-0-0005, Auswahl Eingang
	ProbeInput	MB_PROBE_INPUT	Auswahl des Eingangs für das Triggersignal für das gewählte Modul M-0-0005, Auswahl Eingang
	ExpectWindow-PosEdgeMode	MB_EXPECT_WINDOW_MODE	Aktivierung und Konfiguration des Erwartungsfensters der positiven Flanke M-0-0007, Steuerwort
	ExpectWindow-NegEdgeMode	MB_EXPECT_WINDOW_MODE	Aktivierung und Konfiguration des Erwartungsfensters der negativen Flanke M-0-0007, Steuerwort
	ExpectWindow-PosEdgeStart	REAL	Startposition Erwartungsfenster positive Flanke M-0-0007, Steuerwort M-0-0202, Startposition Erwartungsfenster positive Flanke
	ExpectWindow-PosEdgeEnd	REAL	Endposition Erwartungsfenster positive Flanke M-0-0007, Steuerwort M-0-0203, Endposition Erwartungsfenster positive Flanke
	ExpectWindow-NegEdgeStart	REAL	Startposition Erwartungsfenster negative Flanke M-0-0007, Steuerwort M-0-0204, Startposition Erwartungsfenster negative Flanke
	ExpectWindow-NegEdgeEnd	REAL	Endposition Erwartungsfenster negative Flanke M-0-0007, Steuerwort M-0-0205, Endposition Erwartungsfenster negative Flanke
	ExpectWindow-PosEdgeSet-Point	REAL	Aufsatzpunkt des Erwartungsfensters positive Flanke M-0-0007, M-0-0202, M-0-0203
	ExpectWindow-NegEdgeSet-Point	REAL	Aufsatzpunkt des Erwartungsfensters negative Flanke M-0-0007, M-0-0204, M-0-0205
	MarkerFailureEnable	BOOL	Markenüberwachung: 0:= nicht aktiv, 1:= aktiviert M-0-0007, Steuerwort
	MaxNumberOf-MarkerFailure	UINT	Maximale Anzahl Markenausfälle M-0-0206, Maximale Anzahl Markenausfälle
	Interpolator	MB_INTERPOLATION_MODE	Interpolationsmode: 0:= keine Interpolation, 1:= lineare Interpolation M-0-0007, Steuerwort

12) Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

ML_PLCopen.library

	Name	Typ	Kommentar ¹²⁾
	TimeStampEnable	BOOL	Zeitstempel aktivieren 0:= Zeitstempel wird nicht erfasst 1:= Zeitstempel wird erfasst M-0-0007, Steuerwort
	DeadTimePosEdge	DINT	Totzeitkompensation , positive Flanke M-0-0200, Totzeitkompensation, positive Flanke
	DeadTimeNegEdge	DINT	Totzeitkompensation , negative Flanke M-0-0201, Totzeitkompensation, negative Flanke
	ProbeData	MB_PROBE_DATA	Signalauswahl (Bezugsquelle) M-0-0006, Signalauswahl
	ModuloValueProbeSignal	REAL	Modulowert des zu messenden Signals (Bezugsquelle) (siehe Tab. 6-151 "Auswahl des Messsignals (ProbeData)" auf Seite 331)
	PositionEvalMode	MB_POSITION_EVAL	Aktivierung positive /negative Flanke M-0-0007, Steuerwort
	DiffEvalMode	MB_DIFF_EVAL	Differenzmessung M-0-0007, Steuerwort
VAR_OUTPUT	Done	BOOL	Initialisierung erfolgt
	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

Tab. 6-150: Schnittstelle von FB ML_InitTouchProbe

Funktionsbeschreibung

- Zuordnung von Messtaster-Nummer und Achse,
- Konfiguration des Messtaster-Anschlusses, Modul / Moduleingang,
Zur Auswahl stehen die 8 OnBoard-Eingänge der Steuerung und die jeweils 16 Eingänge von bis zu 4 Fast-IO-Funktionsmodulen (CFL-01.1-E2). Die Hardware und Projektierung des Fast I/O- Funktionsmoduls ist in der XLC/MLC-Funktionsbeschreibung, Kapitel Funktionsmodule (Zusatzkomponenten), beschrieben



Ein SPS-Eingang lässt sich mehreren Messtastern zuordnen.

- ProbeData: Auswahl des Messsignals
Die verfügbaren Messsignale der konfigurierten Achsen sind im Listenparameter C-0-2499, "Messtaster, IDN-Liste Signalauswahl" hinterlegt.
Die Auswahl des Messsignals erfolgt über die Elemente der Enumeration "MB_PROBE_DATA"

¹²⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.



Bis zur Version 13V04 stand nur A-0-0100, Positionswert der Achse, Geber 1, zur Verfügung.

Ab Version 13V06 stehen folgende Messsignale zur Verfügung:

MB_PROBE_DATA - Element	Parameter	Achsbezug
TP_ACT_POS_FEEDBACK1	A-0-0100, Positionswert der Achse, Geber 1	Folgeachsbezug: Der Modulwert der Folgeachse (A-0-0045) und das Verarbeitungsformat (A-0-0059.7=0) wird vom Funktionsbaustein ermittelt.
TP_SYNC_CMD_POS	A-0-2653, Synchroner Lagesollwert	
TP_ELS_CMD_POS	A-0-2655, ELS, Positionssollwert	
TP_RES_MASTER_AXIS_POS	A-0-2650, Resultierende Leitachsisposition	Leitachsbezug: Der Modulwert der Leitachse kann durch den Funktionsbaustein nur während aktiver Synchronisationsbetriebsart zu einer lokalen Leitachse ermittelt werden.
TP_EFF_MASTER_AXIS_POS	A-0-2651, Effektive Leitachsisposition	
TP_CAM_SHAFT_PROF_ACCESS_ANGLE	A-0-2656, ELS, Kurvenscheibe, Zugriffswinkel	Beim Ausführen des Funktionsbausteins außerhalb des Synchronisationsbetriebs, oder bei Verwendung einer entfernten Leitachse (Querverbund C2C), muss der zugehörige Modulwert am Eingang "ModuloValueProbeSignal" angelegt werden.
TP_USER_DATA_CONTAINER_A	A-0-0521, Container des benutzerdefinierten Istwerts A	Achsbezug abhängig vom Inhalt (konfigurierter Parameter) des optionalen zyklischen Daten-Containers. Der zugehörige Modulwert muss am Eingang "ModuloValueProbeSignal" angelegt werden.
TP_USER_DATA_CONTAINER_B	A-0-0523, Container des benutzerdefinierten Istwerts B	
TP_USER_DATA_CONTAINER_C	A-0-0525, Container des benutzerdefinierten Istwerts C	
TP_USER_DATA_CONTAINER_D	A-0-0527, Container des benutzerdefinierten Istwerts D	
TP_USER_DATA_CONTAINER_E	A-0-0529, Container des benutzerdefinierten Istwerts E	
TP_USER_DATA_CONTAINER_F	A-0-0531, Container des benutzerdefinierten Istwerts F	

Tab. 6-151: Auswahl des Messsignals (ProbeData)

- ModuloValueProbeSignal: Der Modulwert des Messsignals dient zur Berechnung eines relativen Erwartungsfensters (siehe [Tab. 6-152 "Erwartungsfenster, Berechnungsvorschrift" auf Seite 333](#))

Bei absolut gewichteten Messsignalen muss der Wert "0" angelegt werden.



Der Modulwert kann immer angelegt werden.

Bei Messsignalen mit Folgeachsbezug bzw. bei Messsignalen mit Leitachsbezug und Ausführung des Funktionsbausteins während aktiver Synchronisationsbetriebsart zu einer lokalen Leitachse, wird der Modulwert durch den Funktionsbaustein ermittelt. (siehe [Tab. 6-151 "Auswahl des Messsignals \(ProbeData\)" auf Seite 331](#))

Der Eingang "ModuloValueProbeSignal" wird dann nicht berücksichtigt.

- Definition des Erwartungsfensters,
 - Getrennt für die positive und die negative Flanke sind die Entscheidungen zu treffen:
 - Erwartungsfenster aus
 - Erwartungsfenster **absolut** konfiguriert:
 - ExpectWindow***EdgeStart gibt die Anfangsposition,
 - ExpectWindow***EdgeEnd gibt die Endposition des Fensters an,
 - ExpectWindow***EdgeSetPoint wird nicht ausgewertet.
 - Erwartungsfenster **relativ** konfiguriert:
 - ExpectWindow***EdgeSetPoint gibt den "Aufsatzpunkt" absolut an,
 - ExpectWindow***EdgeStart gibt den Abstand vom "Aufsatzpunkt" zur Anfangsposition an,
 - ExpectWindow***EdgeEnd gibt den Abstand vom "Aufsatzpunkt" zur Endposition des Fensters an,

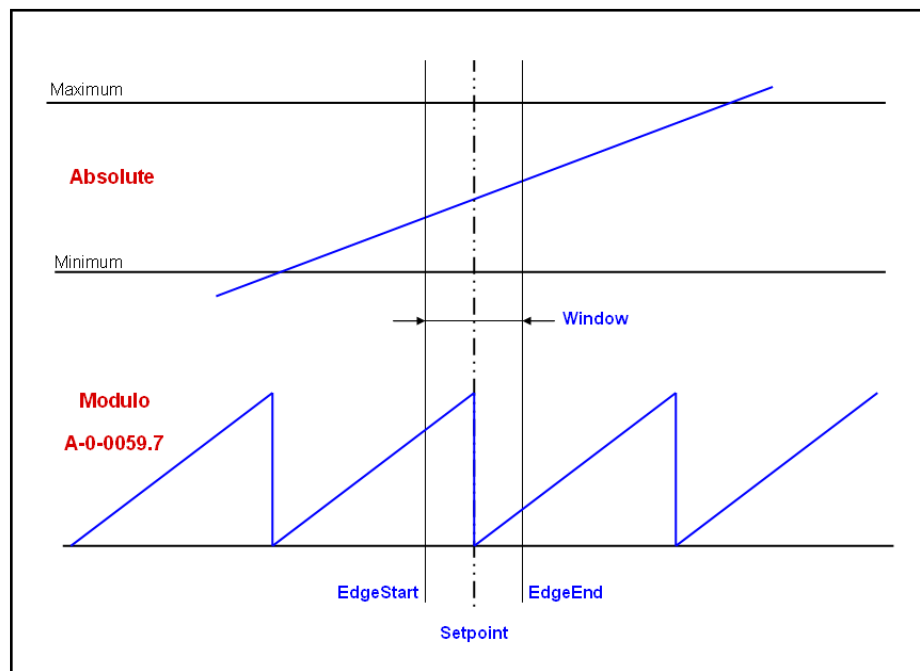


Abb. 6-165: Erwartungsfenster einer Flanke

	Moduloformat	Absolut	Relativ
Positiv	A-0-0059.7=0	Start:= ExpectWindowPosEdgeStart	Start:= ExpectWindowPosEdgeSetPoint - ExpectWindowPosEdgeStart
		End:= ExpectWindowPosEdgeEnd	End:= ExpectWindowPosEdgeSetPoint + ExpectWindowPosEdgeEnd
	A-0-0059.7=1	Start:= ExpectWindowPosEdgeStart	Start:= MOD(ExpectWindowPosEdgeSetPoint - ExpectWindowPosEdgeStart + Modulowert, Modulowert)
		End:= ExpectWindowPosEdgeEnd	End:= MOD(ExpectWindowPosEdgeSetPoint + ExpectWindowPosEdgeEnd, Modulowert)
Negativ	A-0-0059.7=0	Start:= ExpectWindowNegEdgeStart	Start:= ExpectWindowNegEdgeSetPoint - ExpectWindowNegEdgeStart
		End:= ExpectWindowNegEdgeEnd	End:= ExpectWindowNegEdgeSetPoint + ExpectWindowNegEdgeEnd
	A-0-0059.7=1	Start:= ExpectWindowNegEdgeStart	Start:= MOD(ExpectWindowNegEdgeSetPoint - ExpectWindowNegEdgeStart + Modulowert, Modulowert)
		End:= ExpectWindowNegEdgeEnd	End:= MOD(ExpectWindowNegEdgeSetPoint + ExpectWindowNegEdgeEnd, Modulowert)

Tab. 6-152: Erwartungsfenster, Berechnungsvorschrift

- Konfiguration der Markenfehler-Verarbeitung,
Die Markenfehler-Überwachung setzt ein eingeschaltetes Erwartungsfenster voraus.
Befindet sich die aktivierte Flanke außerhalb des Erwartungsfensters erfolgt keine Messwerterfassung, es wird jedoch ein Markenausfall gezählt.
Bei Überschreiten der "Maximalen Anzahl Markenausfälle" wird eine Fehlermeldung (Statusbit Messtaster) ausgelöst.
- Interpolation des Achssignals (das Achssignal steht jeweils am Ende eines sercos Zyklus zur Verfügung, das Messtaster-Signal kommt zu einem beliebigen Zeitpunkt innerhalb des sercos Zyklus, per Interpolation wird das Achssignal aus Achssignal (n-1)-sercos Zyklus, (n)-sercos Zyklus und Zeit ermittelt).
Ist die Interpolation abgeschaltet, wird das Signal am Ende des sercos Zyklus verwendet.
- Zeitstempel aktivieren: ist die Voraussetzung für die Bereitstellung der Zeitmess-Werte.
- Konfiguration der Laufzeit-Kompensation.
Die Kompensation kann für die positive und die negative Flanke des Messtaster-Signals vorgenommen werden.
- PositionEvalMode: Auswahl Aktivierung positive /negative Flanke oder beide zur Positionsermittlung.
Die Positionsermittlung kann z.B. Zeitmessung abgeschaltet werden.
- DiffEvalMode (Differenzmessung):
 - 0 - Differenzmessung ist abgeschaltet, Differenz-Signale für ML_TouchProbe bzw. ML_TouchProbe_Continuous werden nicht berechnet.

ML_PLCOpen.library

- 1 - Differenz zwischen den letzten zwei Messwerten, je nach aktivierter Flanke.
- Beispiel: beide Flanken sind aktiviert, erfasst wird Differenz pos./neg. Flanke, weiter neg./pos. Flanke, weiter pos./neg. Flanke.
- Im nachstehenden Bild sind die erweiterten Methoden 2..5 der Differenzmessung dargestellt:

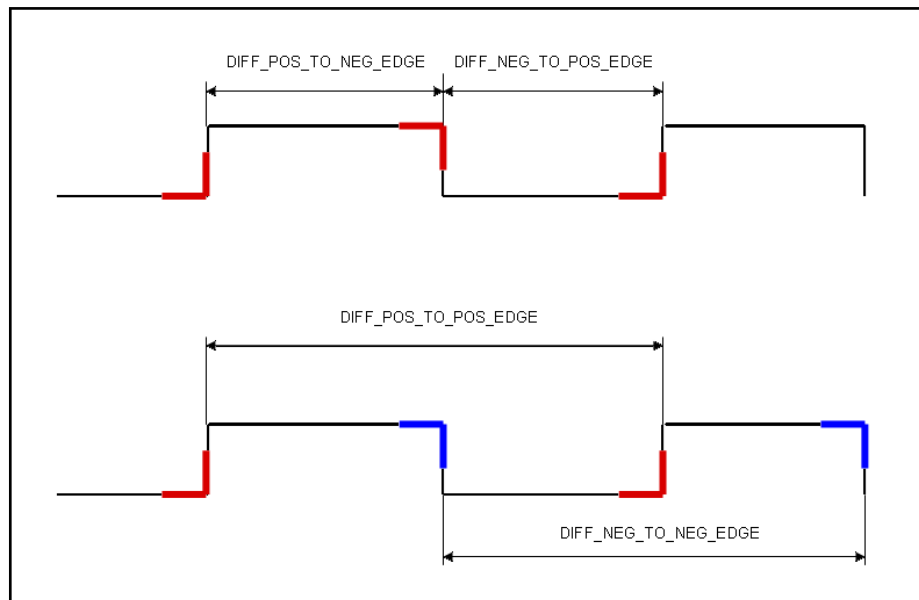


Abb. 6-166: Methoden der Differenzmessung

Signal-Zeit-Diagramm

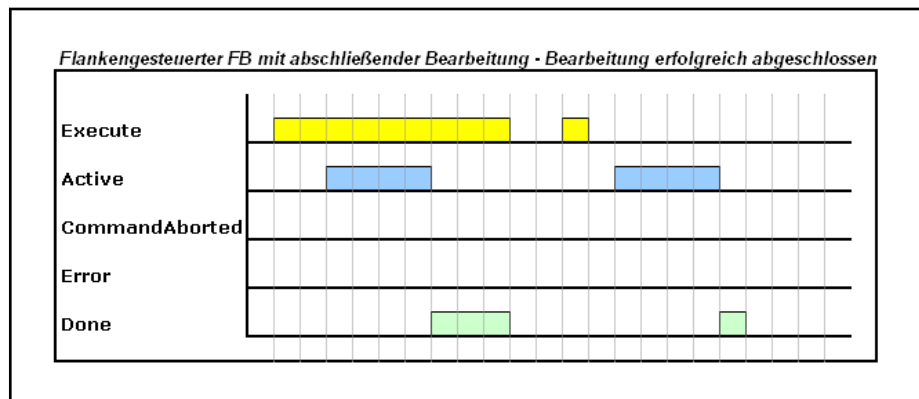


Abb. 6-167: Signal-Zeit-Diagramm: ML_InitTouchProbe

Fehlerbehandlung: Der Funktionsbaustein ML_InitTouchProbe erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf den Messtaster / die Steuerung.

6.11.4 ML_TouchProbe

Kurzbeschreibung Der Funktionsbaustein ML_TouchProbe dient zur Aktivierung der **einmaligen** Positions- und Zeit-Messung, außerdem bietet er eine automatisierte Zeitdifferenzmessung.



Die Aktivierung des Bausteins setzt die Initialisierung des Mess-tasters mittel [ML_InitTouchProbe](#), [Seite 328](#), voraus.

Das Erwartungsfenster kann nach erfolgter Initialisierung durch den Baustein [ML_WriteExpectWindow](#), [Seite 339](#), verändert werden.



Die Aktivierung der Messtaster-Funktion mit [ML_InitTouchProbe](#) ist in allen Sercos-Phasen möglich.

Die Funktionsbausteine [ML_TouchProbe](#) und [ML_TouchProbe-Continuous](#) können jedoch nur in Sercos-Phase CP4 ausgeführt werden.

Ursache: der Parameter M-0-0024, Kommando: Aktivierung, ist nur noch in P4 beschreibbar. Das Ausführen der FBs in einer anderen Sercos-Phase würde zu dem Fehler F02D700C, sercos: Betriebsdatum ist schreibgeschützt (anderweitig), führen.

Bibliothek	Bereich
ML_PLCopen.library	TouchProbe

Tab. 6-153: Bibliothekszuordnung

Schnittstellenbeschreibung

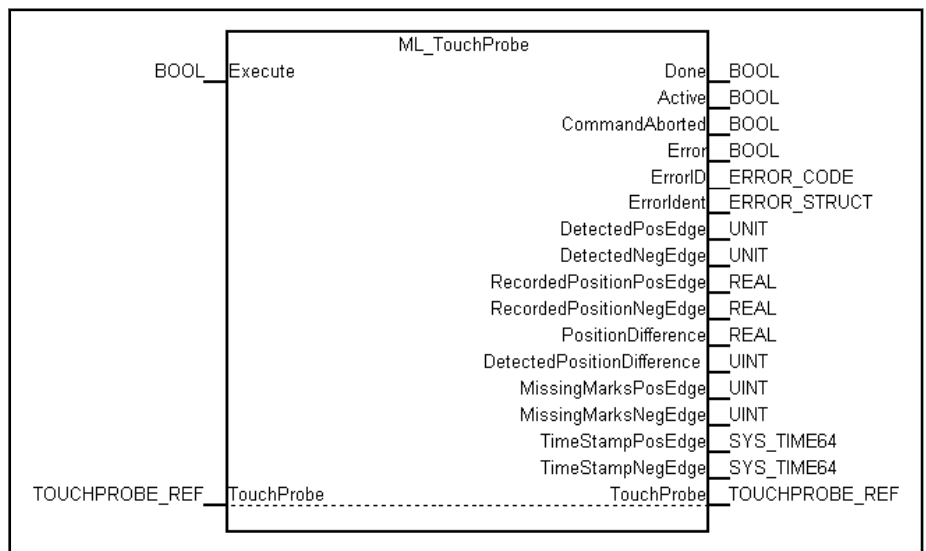


Abb. 6-168: Funktionsbaustein [ML_TouchProbe](#)

	Name	Typ	Kommentar ¹³⁾
VAR_IN_OUT	TouchProbe	TOUCHPROBE_REF	enthält Information zum konkreten Messtaster
VAR_INPUT	Execute	BOOL	startet den Messvorgang bei steigenden Flanke
VAR_OUTPUT	Done	BOOL	Messung erfolgt
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist

¹³⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

ML_PLCOpen.library

	Name	Typ	Kommentar ¹³⁾
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler
	DetectedPosEdge	UINT	Messwertzähler für positive Flanke M-0-0409, Zähler Messwerte positive Flanke
	DetectedNegEdge	UINT	Messwertzähler für negative Flanke M-0-0410, Zähler Messwerte negative Flanke
	RecordedPositionPosEdge	REAL	Messwertspeicher für positive Flanke M-0-0130, Messwert positive Flanke
	RecordedPositionNegEdge	REAL	Messwertspeicher für negative Flanke M-0-0131, Messwert negative Flanke
	PositionDifference	REAL	Differenzmesswertspeicher M-0-0132, Differenzwert
	DetectedPositionDifference	UINT	Zähler für die Erfassung der Differenzwerte M-0-0411, Zähler Differenzwerte
	MissingMarksPosEdge	UINT	Anzahl der Markenausfälle positive Flanke M-0-0224, Anzahl Markenausfälle, positive Flanke
	MissingMarksNegEdge	UINT	Anzahl der Markenausfälle negative Flanke M-0-0225, Anzahl Markenausfälle, negative Flanke
	TimeStampPosEdge	SYS_TIME64	Zeitstempelspeicher positive Flanke M-0-0140, Zeitstempel positive Flanke
	TimeStampNegEdge	SYS_TIME64	Zeitstempelspeicher negative Flanke M-0-0141, Zeitstempel negative Flanke

Tab. 6-154: Schnittstelle von FB ML_TouchProbe

Signal-Zeit-Diagramm

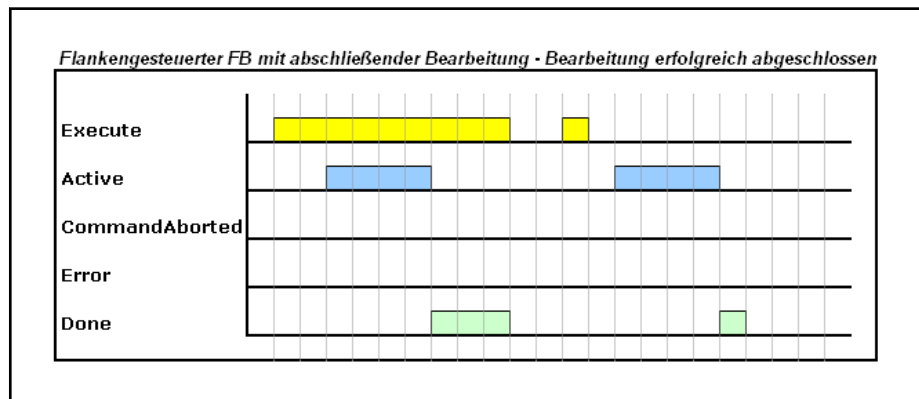


Abb. 6-169: Signal-Zeit-Diagramm: ML_TouchProbe

Funktionsbeschreibung

Modus der Messwertaufnahme

- **Einzelmessung** - Nach Freigabe des jeweiligen Messtasters (M-0-0405), ML_TouchProbe, wird nur beim ersten Messwert-Trigger ein Wert auf-

¹³⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

genommen. Vor jeder weiteren Messung ist eine erneute Messtasterfrei-
gabe erforderlich!

- **Fortlaufende Messung** - Nach Freigabe des jeweiligen Messtasters (M-0-0405), [ML_TouchProbeContinuous](#), [Seite 337](#)., wird bei jedem Messwert-Trigger ein neuer Messwert aufgenommen.

Fehlerbehandlung: Der Funktionsbaustein ML_AbortTrigger erzeugt folgende Fehlermeldungen:

- [MLC_TABLE](#), [Seite 348](#) - Fehler beim Zugriff auf den Messtaster / die Steuerung.

6.11.5 ML_TouchProbeContinuous

Kurzbeschreibung Der Funktionsbaustein ML_TouchProbeContinuous dient zur Aktivierung der **kontinuierlichen** Positions- und Zeit-Messung, außerdem bietet er eine auto-
matisierte Zeitdifferenzmessung.

 Die Aktivierung des Bausteins setzt die Initialisierung des Messtasters mittel [ML_InitTouchProbe](#), [Seite 328](#), voraus.

Das Erwartungsfenster kann nach erfolgter Initialisierung durch den Baustein [ML_WriteExpectWindow](#), [Seite 339](#), verändert werden.

 Die Aktivierung der Messtaster-Funktion mit ML_InitTouchProbe ist in allen Sercos-Phasen möglich.

Die Funktionsbausteine ML_TouchProbe und ML_TouchProbe-Continuous können jedoch nur in Sercos-Phase CP4 ausgeführt werden.

Ursache: der Parameter M-0-0024, Kommando: Aktivierung, ist nur noch in P4 beschreibbar. Das Ausführen der FBs in einer anderen Sercos-Phase würde zu dem Fehler F02D700C, sercos: Betriebsdatum ist schreibgeschützt (anderweitig), führen.

Bibliothek	Bereich
ML_PLCopen.library	TouchProbe

Tab. 6-155: Bibliothekszuordnung

Schnittstellenbeschreibung

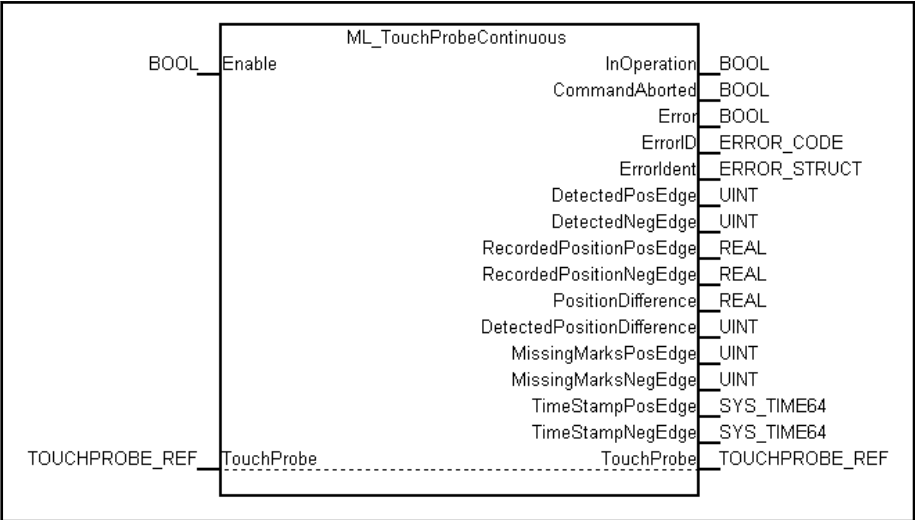


Abb. 6-170: Funktionsbaustein ML_TouchProbeContinuous

ML_PLCopen.library

	Name	Typ	Kommentar ¹⁴⁾
VAR_IN_OUT	TouchProbe	TOUCHPROBE_REF	enthält Information zum konkreten Messtaster
VAR_INPUT	Enable	BOOL	startet den Messvorgang bei steigenden Flanke
VAR_OUTPUT	InOperation	BOOL	Messvorgang läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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
	DetectedPosEdge	UINT	Messwertzähler für positive Flanke M-0-0409, Zähler Messwerte positive Flanke
	DetectedNegEdge	UINT	Messwertzähler für negative Flanke M-0-0410, Zähler Messwerte negative Flanke
	RecordedPositionPosEdge	REAL	Messwertspeicher für positive Flanke M-0-0130, Messwert positive Flanke
	RecordedPositionNegEdge	REAL	Messwertspeicher für negative Flanke M-0-0131, Messwert negative Flanke
	PositionDifference	REAL	Differenzmesswertspeicher M-0-0132, Differenzwert
	DetectedPositionDifference	UINT	Zähler für die Erfassung der Differenzwerte M-0-0411, Zähler Differenzwerte
	MissingMarksPosEdge	UINT	Anzahl der Markenausfälle positive Flanke M-0-0224, Anzahl Markenausfälle, positive Flanke
	MissingMarksNegEdge	UINT	Anzahl der Markenausfälle negative Flanke M-0-0225, Anzahl Markenausfälle, negative Flanke
	TimeStampPosEdge	SYS_TIME64	Zeitstempelspeicher positive Flanke M-0-0140, Zeitstempel positive Flanke
	TimeStampNegEdge	SYS_TIME64	Zeitstempelspeicher negative Flanke M-0-0141, Zeitstempel negative Flanke

Tab. 6-156: Schnittstelle von FB ML_TouchProbeContinuous

¹⁴⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

Signal-Zeit-Diagramm

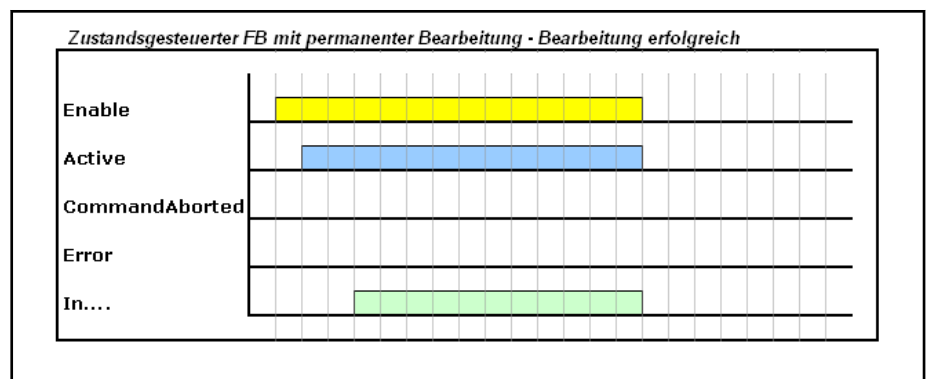


Abb. 6-171: Signal-Zeit-Diagramm: ML_TouchProbeContinuous

Funktionsbeschreibung

Modus der Messwertaufnahme

- **Einzelmessung** - Nach Freigabe des jeweiligen Messtasters (M-0-0405), [ML_TouchProbe](#), Seite 334, wird nur beim ersten Messwert-Trigger ein Wert aufgenommen. Vor jeder weiteren Messung ist eine erneute Messtasterfreigabe erforderlich!
- **Fortlaufende Messung** - Nach Freigabe des jeweiligen Messtasters (M-0-0405), [ML_TouchProbeContinuous](#), wird bei jedem Messwert-Trigger ein neuer Messwert aufgenommen.

Fehlerbehandlung:

Der Funktionsbaustein ML_AbortTrigger erzeugt folgende Fehlermeldungen:

- [MLC_TABLE](#), Seite 348 - Fehler beim Zugriff auf den Messtaster / die Steuerung.

6.11.6 ML_WriteExpectWindow

Kurzbeschreibung

Der Funktionsbaustein ML_WriteExpectWindow konfiguriert die Erwartungsfenster, ohne dass ein kompletter Initialisierungslauf mit dem [ML_InitTouchProbe](#), Seite 328, notwendig ist.



Bei aktiviertem "Enable" des Bausteines werden Änderungen an den Eingängen **sofort** übernommen.

Bibliothek	Bereich
ML_PLCopen.library	TouchProbe

Tab. 6-157: Bibliothekszuordnung

Schnittstellenbeschreibung

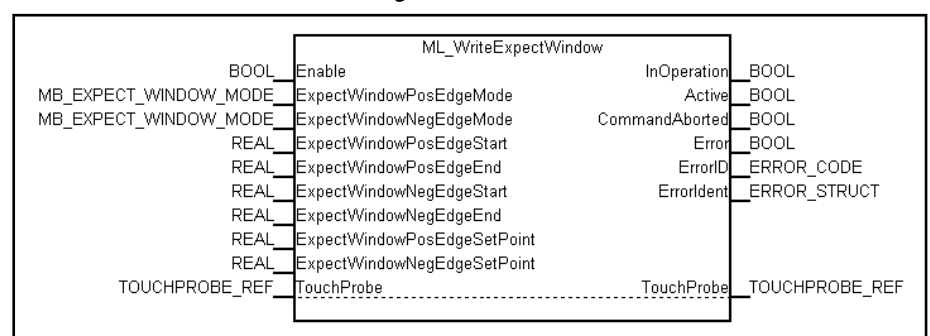


Abb. 6-172: Funktionsbaustein ML_WriteExpectWindow

ML_PLCopen.library

	Name	Typ	Kommentar ¹⁵⁾
VAR_IN_OUT	TouchProbe	TOUCHPROBE_REF	enthält Information zum konkreten Messtaster
VAR_INPUT	Enable	BOOL	startet die Bewegung bei steigenden Flanke
	ExpectWindow-PosEdgeMode	MB_EXPECT_WINDOW_MODE	Aktivierung Messtasterfenster bei positiver Flanke M-0-0007, Steuerwort
	ExpectWindow-NegEdgeMode	MB_EXPECT_WINDOW_MODE	Aktivierung Messtasterfenster bei negativer Flanke M-0-0007, Steuerwort
	ExpectWindow-PosEdgeStart	REAL	Startposition Erwartungsfenster positive Flanke M-0-0007, Steuerwort M-0-0202, Startposition Erwartungsfenster positive Flanke
	ExpectWindow-PosEdgeEnd	REAL	Endposition Erwartungsfenster positive Flanke M-0-0007, Steuerwort M-0-0203, Endposition Erwartungsfenster positive Flanke
	ExpectWindow-NegEdgeStart	REAL	Startposition Erwartungsfenster negative Flanke M-0-0007, Steuerwort M-0-0204, Startposition Erwartungsfenster negative Flanke
	ExpectWindow-NegEdgeEnd	REAL	Endposition Erwartungsfenster negative Flanke M-0-0007, Steuerwort M-0-0205, Endposition Erwartungsfenster negative Flanke
	ExpectWindow-PosEdgeSet-Point	REAL	Aufsatzpunkt des positiven Erwartungsfensters M-0-0007, M-0-0202, M-0-0203
	ExpectWindow-NegEdgeSet-Point	REAL	Aufsatzpunkt des negativen Erwartungsfensters M-0-0007, M-0-0204, M-0-0205
VAR_OUTPUT	InOperation	BOOL	gewünschte Konfiguration erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 6-158: Schnittstelle von FB ML_WriteExpectWindow

Funktionsbeschreibung

Für die angelegte Messtaster-Nummer kann mit diesem Baustein das Erwartungsfenster umdefiniert werden.

Getrennt für die positive und die negative Flanke sind die Entscheidungen zu treffen:

¹⁵⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

Definition des Erwartungsfensters

- Erwartungsfenster aus
- Erwartungsfenster **absolut** konfiguriert:
 - ExpectWindow***EdgeStart gibt die Anfangsposition,
 - ExpectWindow***EdgeEnd gibt die Endposition des Fensters an,
 - ExpectWindow***EdgeSetPoint wird nicht ausgewertet.
- Erwartungsfenster **relativ** konfiguriert:
 - ExpectWindow***EdgeSetPoint gibt den "Aufsatzpunkt" absolut an,
 - ExpectWindow***EdgeStart gibt den Abstand vom "Aufsatzpunkt" zur Anfangsposition an,
 - ExpectWindow***EdgeEnd gibt den Abstand vom "Aufsatzpunkt" zur Endposition des Fensters an,

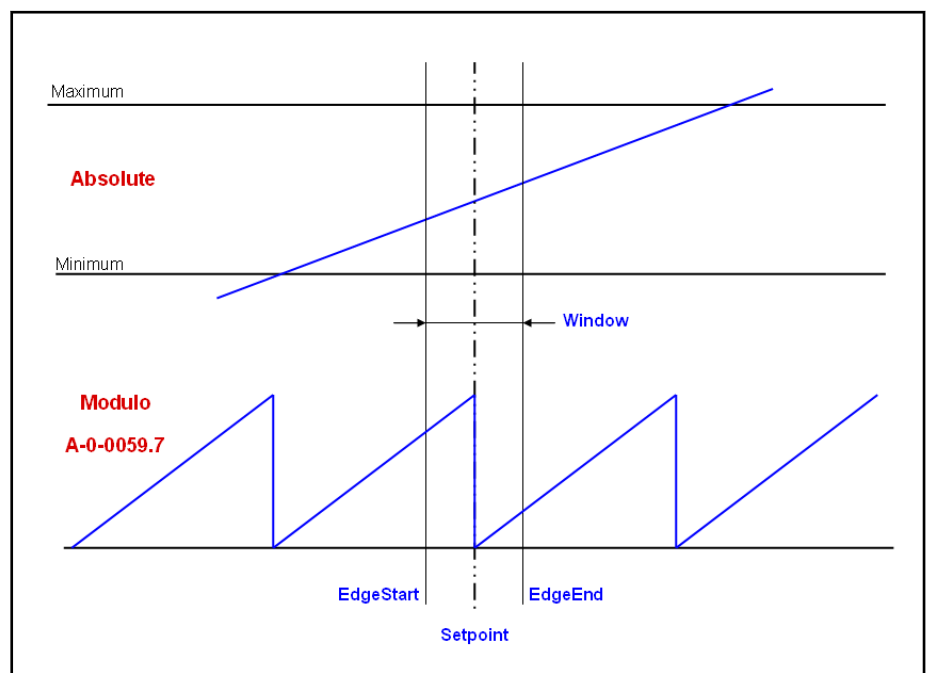


Abb. 6-173: Erwartungsfenster einer Flanke

	Moduloformat	Absolut	Relativ
Positiv	A-0-0059.7= 0	Start:=ExpectWindowPosEdgeStart	Start:= ExpectWindowPosEdgeSetPoint - ExpectWindowPosEdgeStart
		End:=ExpectWindowPosEdgeEnd	End:= ExpectWindowPosEdgeSetPoint + ExpectWindowPosEdgeEnd
	A-0-0059.7= 1	Start:=ExpectWindowPosEdgeStart	Start:= MOD(ExpectWindowPosEdgeSetPoint - ExpectWindowPosEdgeStart + A-0-0045, A-0-0045)
		End:=ExpectWindowPosEdgeEnd	End:= MOD(ExpectWindowPosEdgeSetPoint + ExpectWindowPosEdgeEnd, A-0-0045)

ML_PLCopen.library

Negativ	A-0-0059.7= 0	Start:=ExpectWindowNegEdgeStart	Start:= ExpectWindowNegEdgeSetPoint - ExpectWindowNegEdgeStart
		End:=ExpectWindowNegEdgeEnd	End:= ExpectWindowNegEdgeSetPoint + ExpectWindowNegEdgeEnd
	A-0-0059.7= 1	Start:=ExpectWindowNegEdgeStart	Start:= MOD(ExpectWindowNegEdgeSetPoint - ExpectWindowNegEdgeStart + A-0-0045, A-0-0045)
		End:=ExpectWindowNegEdgeEnd	End:= MOD(ExpectWindowNegEdgeSetPoint + ExpectWindowNegEdgeEnd, A-0-0045)

Tab. 6-159: Erwartungsfenster, Berechnungsvorschrift

Signal-Zeit-Diagramm

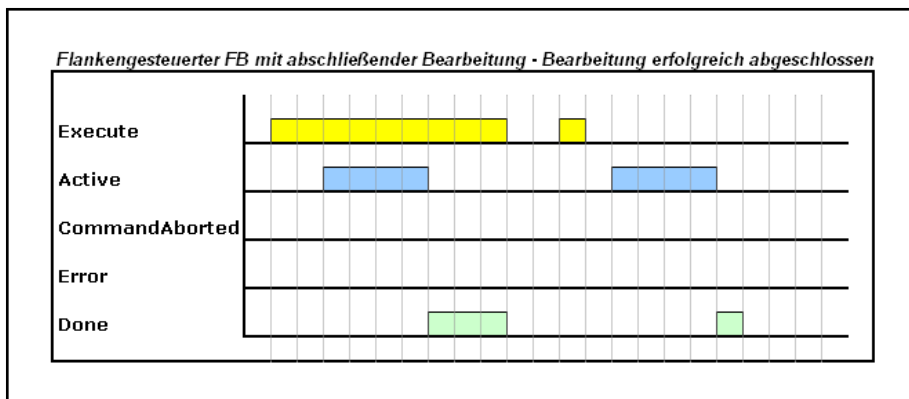


Abb. 6-174: Signal-Zeit-Diagramm: ML_WriteExpectWindow

Fehlerbehandlung: Der Funktionsbaustein ML_AbortTrigger erzeugt folgende Fehlermeldungen:

- [MLC_TABLE, Seite 348](#) - Fehler beim Zugriff auf den Messtaster / die Steuerung.

6.12 Zeitbedarf bei der Bearbeitung von Motion-Funktionsbausteinen in der IndraMotion MLC

Ausgegangen wird von einer SERCOS- und einer Motion- Zykluszeit von 2 Millisekunden.

Die motionsynchrone MotionTask, aus der die Messungen erfolgen, wird somit ebenso alle 2 Millisekunden abgearbeitet.



Der Buszyklus-Takt muss an die Motion-Task (2ms) gekoppelt werden. Per default ist der Buszyklus-Takt an die schnellste PLC Task (10ms) gekoppelt.

Die Zeiten für die Onboard I/Os sind direkt proportional zur Zykluszeit der Task, in der sie verwendet werden. Alle Messungen erfolgten in einer 2ms Motion Task. Bei einer Halbierung der Zykluszeit sind die Onboard IOs auch doppelt so schnell.

Nachstehende Tabelle zeigt durchschnittliche Bearbeitungszeiten der betreffenden Funktionsbausteine **aus der Sicht der SPS**.

Beispiel:

Zeitbedarf und Arbeitsweise des MC_Power aus Sicht der SPS

Die Bausteininstanz des MC_Power bekommt am "Enable"-Eingang einen 0→1-Übergang. Sie wird bearbeitet.

Von Erreichen bis zum Verlassen des Netzwerkes, dass den Baustein enthält, vergehen im Durchschnitt 4,6µs. Der Baustein gibt dabei einen Auftrag an die Antriebselektronik.

Die Ausführung des Auftrags in der Antriebselektronik, die nach der Leistungsfreigabe für den Antrieb die Rückmeldung für die Statusänderung am Ausgang "Status" der Bausteininstanz des MC_Power gibt, dauert natürlich wesentlich länger und benötigt viele SPS-Zyklen.

Aktion / T _{avr} [µs]	13VRS, L25	13VRS, L45	13VRS, L65
MC_Power, real axis	25,9	11,2	4,8
MC_MoveAbsolute, real axis	25,6	11,5	4,6
MC_MoveAbsolute, virtual axis	12,7	5,7	2,4
MC_MoveRelative, real axis	25,2	11,6	4,8
MC_MoveRelative, virtual axis	13,0	5,4	2,4
MC_MoveAdditive, real axis	26,8	11,8	5,4
MC_MoveAdditive, virtual axis	9,7	4,8	2
MC_MoveVelocity, real axis	26,8	11,3	4,8
MC_MoveVelocity, virtual axis	12,3	5,6	2,3
MC_Stop, real axis	29,1	11,8	5,6
MC_Stop, virtual axis	12,1	5,8	2,6
MC_GearIn	29,2	12	5
MC_GearOut	27,6	10,4	4,8
Datenzugriff, Direktvariable	18,3	5,7	2,8
Datenzugriff, Variable im Shared Memory	3,7	1,2	0,6
On Board I/Os		2000,8	2000,5
Inline I/Os	8000,1	7429,4	4000,1
Profibus I/Os	10501,9	19888	10463,5

Tab. 6-160: Durchschnittlicher Zeitbedarf zur Bearbeitung diverser Funktionsbausteine bei 50000-facher Wiederholung der Ansteuerung

Der Zeitbedarf ergibt sich aus der Summe des Zeitbedarfs der einzelnen Anweisungen, abzuarbeitenden Funktionen und Funktionsbausteininstanzen.

Werden weitere Tasks angelegt, steigt durch deren Umschaltzeit und Bildung task-lokaler Abbildspeicher der Zeitbedarf zusätzlich.

Eine besonders kritische Situation ergibt sich beim Schalten auf einen Fehler:

Beispiel:

Gleichzeitiges Schalten eines MC_Power und eines MC_MoveAbsolute der gleichen Achse

Ausgangspunkt ist die Situation

- "MC_Power.Enable"= FALSE,
- "MC_MoveAbsolute.Execute"= FALSE und
- "MC_MoveAbsolute.Error"= FALSE.

Die Eingänge "MC_Power.Enable" und "MC_MoveAbsolute.Execute" werden auf TRUE geschaltet und danach beide Bausteininstanzen abgearbeitet.

ML_PLCopen.library

"MC_MoveAbsolute" wird damit gestartet, **ohne** dass die Leistung für ihn freigegeben wurde. Er schaltet auf einen Fehler.



Der Zeitbedarf des "MC_MoveAbsolute" steigt in diesem Fall auf ca. den dreifachen Wert der normalen Bearbeitung.

6.13 Achstypen und durch sie unterstützte Funktionsbausteine

Nachstehende Tabelle gibt einen Überblick über den aktuellen Stand der Unterstützung der Funktionsbausteine durch die einzelnen Achstypen.

Handelt es sich um Bausteine, die eine Leit- und eine Folgeachse benötigen, bezieht sich die Aussage auf die Folgeachse.



Nicht unterstützte Funktionsbausteine erzeugen den Fehler: "F2229204, Kommando durch Achse nicht unterstützt."



Open-loop

Für reale Achsen ohne Positionsrückführung (open-loop) sind nur Funktionsbausteine ohne Positionsregelung möglich.



13VRS

- Wird eine Reale Achse, IndraDrive, MPx06 / MPx07 / MPx08 / MPx16 / MPx17 / MPx18, auf der Steuerung unter Nutzung der Sollwertschnittstelle zum Antrieb berechnet, siehe Spalte **IndraDrive ► IPO Control**.
- Wird eine Reale Achse, IndraDrive, MPx06 / MPx07 / MPx08 / MPx16 / MPx17 / MPx18, auf der Steuerung unter Nutzung des PackProfiles berechnet, siehe Spalte **Sercos-Drive**.

Funktionsbaustein	Reale Achsen					Virtuelle Achse	Geber-/ Verbund*- achse	Regler- achse *
	IndraDrive		HydraulicDrive *		sercos- Drive			
	Interpo- lation Drive	Interpo- lation Control	Interpo- lation Drive	Interpo- lation Control				
MB_ChangeCamData	X	X	-	X	X	X	-	X
MB_ChangeFlexEventSet	-	X	-	X	X	X	-	X
MB_ChangeFlexProfilSet	-	X	-	X	X	X	-	X
MB_ChangeProfilSet	-	X	-	X	X	X	-	X
MB_ChangeProfilStep	-	X	-	X	X	X	-	X
MB_GearInPos	X	X	-	X	X	X	-	X
MB_Home	X	X	X	X	X	X	-	X(3)
MB_MotionProfile	-	X	-	X	X	X	-	X
MB_Phasing	X(1)	X	X	X	X	X	-	X
MB_PhasingSlave	X(1)	X	X	X	X	X	-	X
MB_SetPositionControlMode (5)	X	X	-	-	-	-	-	-

Funktionsbaustein	Reale Achsen					Virtuelle Achse	Geber-/ Verbund*- achse	Regler- achse *
	IndraDrive		HydraulicDrive *		sercos- Drive			
	Interpo- lation Drive	Interpo- lation Control	Interpo- lation Drive	Interpo- lation Control				
MC_AddAxisToGroup * (4)	X	X	X	X	X	X	-	X
MC_CamIn	X	-	-	-	-	-	-	-
MC_CamOut	X	-	-	-	-	-	-	-
MC_GearIn	X(2)	X(2)	-	X	X	X	-	X
MC_GearOut	X(2)	X(2)	-	X	X	X	-	X
MC_MoveAbsolute	X	X	X	X	X	X	-	X
MC_MoveAdditive	X	X	X	X	X	X	-	X
MC_MoveRelative	X	X	X	X	X	X	-	X
MC_MoveVelocity	X(2)	X(2)	X	X	X	X	-	X
MC_Power	X(2)	X(2)	X	X	X	-	-	X
MC_RemoveAxisFromGroup * (4)	X	X	X	X	X	X	-	X
MC_Reset	X(2)	X(2)	X	X	X	X	-	X
MC_Stop	X(2)	X(2)	X	X	X	X	-	X
MC_TorqueControl	X(1)	X	X	X	X	-	-	X
ML_FlexProfile	-	X	-	X	X	X	-	X
ML_OpenCyclicAnalogChannel	-	-	X	X	-	-	-	-
ML_OpenCyclicPositionChannel	X	X	X	X	X	X	-	X
ML_OpenCyclicTorqueChannel	X	X	X	X	-	-	-	X
ML_OpenCyclicVelocityChannel	X	X	X	X	-	-	-	X
ML_WriteCyclicAnalog	-	-	X	X	-	-	-	-
ML_WriteCyclicPosition	X	X	X	X	X	X	-	X
ML_WriteCyclicTorque	X	X	X	X	-	-	-	X
ML_WriteCyclicVelocity	X	X	X	X	-	-	-	X
ML_ControlOff	X	X	X	X	X	X	-	X
ML_ControlOn	X	X	X	X	X	X	-	X
ML_ControlWriteActualData	-	-	-	-	-	-	-	X
ML_OpenCyclicAnalogChannelCtrl	-	-	X	X	-	-	-	-
ML_OpenCyclicPositionChannelCtrl	X	X	X	X	X	X	-	X
ML_OpenCyclicTorqueChannelCtrl	X	X	X	X	-	-	-	X
ML_OpenCyclicVelocityChannelCtrl	X	X	X	X	-	X	-	X
ML_WriteCyclicAnalogCtrl	-	-	X	X	-	-	-	-
ML_WriteCyclicPositionCtrl	X	X	X	X	X	X	-	X

ML_PLCopen.library

Funktionsbaustein	Reale Achsen					Virtuelle Achse	Geber-/Verbund*-achse	Regler-achse *
	IndraDrive		HydraulicDrive *		sercos-Drive			
	Interpolation Drive	Interpolation Control	Interpolation Drive	Interpolation Control				
ML_WriteCyclicTorqueCtrl	X	X	X	X	-	-	-	X
ML_WriteCyclicVelocityCtrl	X	X	X	X	-	X	-	X

* nur IndraMotion MLC

(1) nicht MPx03

(2) auch mit OpenLoop

(3) Abhängig von der Implementierung des Reglers

(4) ab 13VRS

(5) ab 13VRS und > MPx06

Tab. 6-161: Funktionsbausteine in IndraLogic XLC und IndraMotion MLC

6.14 Fehlertabellen (Error_Tables)

6.14.1 SERCOS_TABLE, ERROR_TABLE 16#0010

SERCOS_TABLE - ERROR_TABLE, Tabellenummer 16#0010

Die IndraLogic XLC / IndraMotion MLC erweitern die vierstelligen sercos Fehlercodes zu achtstelligen MLC_Fehlercodes wie folgt:

Ergänzt werden "F" für Fehler, "0" als Severity (Schwere des Fehlers) und "2D" als Kennzeichen der Herkunft. Dann folgen die vier sercos Ziffern.

Severity 0 bedeutet keine Reaktion an den Achsen/ Antrieben.

Der achtstellige Code wird an der IndraWorks-Oberfläche (Fehlerspeicher) verwendet.

An den Bausteinausgängen entscheidet die Tabellenummer "ErrorIdent/ Table", 16#0030 (MLC_TABLE) oder 16#0010 (SERCOS_TABLE), welche Darstellung verwendet wird.

SERCOS_TABLE ErrorIdent / Additional1	Beschreibung	MLC_TABLE ErrorIdent / Additional1
	Allgemeiner Fehler	
0000	sercos: Kein Fehler im Service-Kanal	F02D0000
0001	sercos: Service-Kanal nicht geöffnet	F02D0001
0003	Max. Anzahl paralleler Parameteranfragen überschritten	F02D0003
0007	Parameterzugriff auf entfernte Steuerungen verboten	F02D0007
0009	sercos: Ungültiger Zugriff beim Schließen Service-Kanal	F02D0009
00A0	sercos: Parameter Service, Kanal ist nicht geöffnet	F02D00A0
00B0	sercos: Parameter Element ist nicht vorhanden	F02D00B0
00C0	sercos: Ausgewählter Antrieb ist nicht vorhanden	F02D00C0
	Element 1 (Identnummer)	

SERCOS_ TABLE ErrorIdent / Additional1	Beschreibung	MLC_ TABLE ErrorIdent / Additional1
1001	sercos: Keine ID-Nummer	F02D1001
1009	sercos: Ungültiger Zugriff auf Element 1	F02D1009
	Element 2 (Name)	
2001	sercos: Kein Name	F02D2001
2002	sercos: Übertragungsfehler, Name zu kurz	F02D2002
2003	sercos: Übertragungsfehler, Name zu lang	F02D2003
2004	sercos: Name kann nicht geändert werden (read only)	F02D2004
2005	sercos: Name ist zur Zeit schreibgeschützt	F02D2005
	Element 3 (Attribut)	
3002	sercos: Übertragungsfehler, Attribut zu kurz	F02D3002
3003	sercos: Übertragungsfehler, Attribut zu lang	F02D3003
3004	sercos: Attribut kann nicht geändert werden (read only)	F02D3004
3005	sercos: Attribut ist zur Zeit schreibgeschützt	F02D3005
	Element 4 (Einheit)	
4001	sercos: Keine UNIT	F02D4001
4002	sercos: Übertragungsfehler, UNIT zu kurz	F02D4002
4003	sercos: Übertragungsfehler, UNIT zu lang	F02D4003
4004	sercos: UNIT kann nicht verändert werden (read only)	F02D4004
4005	sercos: UNIT ist zur Zeit schreibgeschützt	F02D4005
	Element 5 (Minimaler Eingabewert)	
5001	sercos: Kein Min-Wert	F02D5001
5002	sercos: Übertragungsfehler, Min-Wert zu kurz	F02D5002
5003	sercos: Übertragungsfehler, Min-Wert zu lang	F02D5003
5004	sercos: Min-Wert kann nicht geändert werden (read only)	F02D5004
5005	sercos: Min-Wert ist zur Zeit schreibgeschützt	F02D5005
	Element 6 (Maximaler Eingabewert)	
6001	sercos: Kein Max-Wert	F02D6001
6002	sercos: Übertragungsfehler, Max-Wert zu kurz	F02D6002
6003	sercos: Übertragungsfehler, Max-Wert zu lang	F02D6003
6004	sercos: Max-Wert kann nicht geändert werden (read only)	F02D6004
6005	sercos: Max-Wert ist zur Zeit schreibgeschützt	F02D6005
	Element 7 (Betriebsdatum)	
7001	sercos: Kein Betriebsdatum	F02D7001

ML_PLCopen.library

SERCOS_ TABLE ErrorIdent / Additional1	Beschreibung	MLC_ TABLE ErrorIdent / Additional1
7002	sercos: Übertragungsfehler, Betriebsdatum zu kurz	F02D7002
7003	sercos: Übertragungsfehler, Betriebsdatum zu lang	F02D7003
7004	sercos: Betriebsdatum nicht änderbar (read only)	F02D7004
7005	sercos: Betriebsdatum ist zur Zeit schreibgeschützt	F02D7005
7006	sercos: Betriebsdatum ist kleiner als Min-Wert	F02D7006
7007	sercos: Betriebsdatum ist größer als Max-Wert	F02D7007
7008	sercos: Ungültiges Betriebsdatum	F02D7008
7009	sercos: Betriebsdatum ist schreibgeschützt (Passwort)	F02D7009
700A	sercos: Betriebsdatum ist schreibgeschützt (zykl. konf.)	F02D700A
700B	sercos: Ungültige Adressierung (Datencontainer, Liste)	F02D700B
700C	sercos: Betriebsdatum ist schreibgeschützt (anderweitig)	F02D700C
700D	sercos: Ungültige Gleitkommazahl	F02D700D
700E	sercos: Reserviert	F02D700E
700F	sercos: Betriebsdatum ist in OM schreibgeschützt	F02D700F
7010	sercos: Befehlsfolge bereits aktiv	F02D7010
7011	sercos: Befehlsfolge nicht unterbrechbar	F02D7011
7012	sercos: Befehlsfolge zur Zeit nicht ausführbar	F02D7012
7013	sercos: Befehlsfolge nicht ausführbar, Param. ungültig	F02D7013
710C	sercos: Datum überschreitet max. Bereich	F02D710C
8001	sercos: Zugriff zeitweise nicht möglich	F02D8001
8096	sercos: Gerät deaktiviert	F02D8096
8097	sercos: Gerät nicht vorhanden	F02D8097

Tab. 6-162: *sercos Fehler und ihre Verwendung in der IndraLogic XLC /
IndraMotion MLC*

6.14.2 MLC_TABLE, ERROR_TABLE 16#0030

MLC_TABLE - ERROR_TABLE, Tabellenummer 16#0030

Die in der nachstehenden Tabelle vorhandenen Einträge sind mit den MLC-Hinweisen zur Störungsbeseitigung abgestimmt (Diagnosen).

ErrorIdent / Additional1	Diagnose
F0220200	F2229200
F0220201	
F0220202	
F0220203	F2229203
F0220204	F2229204

ErrorIdent / Additional1		Diagnose
F0220205	F2229205	Kommando zurückgewiesen, keine Leistung
F0220206	F2229206	Kommando in 'Stopping' zurückgewiesen
F0220207	F2229207	Kommando in 'Homing' zurückgewiesen
F0220208	F2229208	Ungültige Kommandonummer
F0220210	F2229210	Reset bei aktivem ErrorStop, 'state ErrorStop'
F0220211	F2229211	Kommando in 'Discrete Motion' zurückgewiesen
F0220212	F2229212	Kommando in 'Continuous Motion' zurückgewiesen
F0220213	F2229213	Kommando in 'Synchronized Motion' zurückgewiesen
F0220214	F2229214	Kommando in 'ErrorStop' zurückgewiesen
F0220216	F2229216	Keine Leistung verfügbar
F0220217	F2229217	Kommando in 'StandStill' / PowerOn zurückgewiesen
F0220218	F2229218	Kommando in 'StandStill' zurückgewiesen
F0220223	F2229223	Geschwindigkeit unter Minimalwert
F0220224	F2229224	Geschwindigkeit über Maximalwert
F0220225	F2229225	Beschleunigung unter Minimalwert
F0220226	F2229226	Beschleunigung über Maximalwert
F0220227	F2229227	Verzögerung unter Minimalwert
F0220228	F2229228	Verzögerung über Maximalwert
F0220229	F2229229	Ruck unter Minimalwert
F0220230	F2229230	Ruck über Maximalwert
F2220231		Ungültige Leitachsdaten
	F2229231	Drehrichtung ungültig
F0220232	F2229232	Ungültige Leitachssteuerungsnummer
F0220233	F2229233	Ungültige Leitachsnummer
F0220234	F2229234	Ungültige Kurventabelle
	F2229235	Fehler im MotionProfile
F0220263	F2229263	Leitachsgetriebe, Eingangsumdrehungen unter Minimalwert
F0220264	F2229264	Leitachsgetriebe, Eingangsumdrehungen über Maximalwert
F0220265	F2229265	Leitachsgetriebe, Ausgangsumdrehungen unter Minimalwert
F0220266	F2229266	Leitachsgetriebe, Ausgangsumdrehungen über Maximalwert
F0220267	F2229267	Feinabgleich der Getriebeübersetzung unter Minimalwert
F0220268	F2229268	Feinabgleich der Getriebeübersetzung über Maximalwert
F0220269	F2229269	Ungültige Aufsynchroisierrichtung
F0220270	F2229270	Ungültige Art des Aufsynchroisierens
F0220271	F2229271	Kurvenscheibenhub unter Minimalwert
F0220272	F2229272	Kurvenscheibenhub über Maximalwert

ML_PLCopen.library

ErrorIdent / Additional1		Diagnose
F0220273	F2229273	Kommando im Parametriermodus zurückgewiesen
F0220274	F2229274	Kommando zurückgewiesen, keine Positionsreferenz
F0220275	F2229275	Kommando wurde abgebrochen
F0220276	F2229276	Steuerungsfehler
F0220277	F2229277	Direkter Leitachswechsel
F0220278	F2229278	Kommando 'Deaktivierte Achse' aktiv
F0220279	F2229279	Kommando 'parkende Achse' aktiv
F0220280	F2229280	Kommando in 'Coordinated Motion' zurückgewiesen
F0220281	F2229281	Ungültige Gleitkommazahl
F0220282	F2229282	MotionProfile, SetSelection unter Minimalwert
F0220283	F2229283	MotionProfile, SetSelection über Maximalwert
	F2229284	Kurventabelle ist schreibgeschützt
	F2229285	Kurvenscheiben-Beleg-Zähler ungültig
	F2229286	RelPositioning über Max-Wert
F0220287		Keine Referenz auf Kurventabelle gefunden
	F2229287	Achse kann der Kinematik nicht zugeordnet werden
F0220288		ChangeCamTable: Tabellenwert über Maximalwert
F0220289		ChangeCamTable: Tabellenwert unter Minimalwert
F0220290		ChangeCamTable: ID ungültig
F0220291		Keine Referenz auf MotionProfile gefunden
F0220292		ChangeMotionProfileSet: SetNumber unter Minimalwert
F0220293		ChangeMotionProfileSet: SetNumber über Maximalwert
F0220294		ChangeMotionProfileSet: Profil ist schreibgeschützt
F0220295		ChangeMotionProfileStep: Ungültige StepNumber
	F2229296	Absoluter Einstiegspunkt im FlexProfile nicht gefunden
	F2229297	Absoluter Einstieg in FlexProfile nicht möglich
	F2229298	Leitachsgeschwindigkeit FlexProfile negativ

Tab. 6-163: Diagnosen der Motion FBs (Achszustand) I



Je nach vorgewählter Achsreaktion tritt der Fehlers **F*22*xxx** auf als

- F0220xxx, für den Fall keine Fehlerreaktion durch die Achse
- F2229xxx, für den Fall "Achse Halt", bestmögliche Stillsetzung.

ErrorIdent / Additional1		Diagnose
F0220300	F2229300	Ungültiger Sollwertkanal
F0220301	F2229301	Kommando in 'Controlled Motion' zurückgewiesen

Errorident / Additional1		Diagnose
	F2229310	Fehler bei der Berechnung des Bewegungsschrittes
	F2229311	FlexProfile: Ungültige Umschaltposition
	F2229312	FlexProfile: Ungültiger Profil-Startmodus
	F2229313	FlexProfile: Ungültiger Folgeachsoffset
	F2229314	Summe Hübe ungl. 0 oder modulo, cykl. Betrieb unmöglich
	F2229315	FlexProfile ungültig, ausführen nicht möglich
F0220316		Gleitkommaeinheit: Listenlänge Eigenschaften inkonsistent
F0220317		ChangeFlexProfile: Keine Bewegungsgesetze definiert
F0220318		ChangeFlexProfile: Ungültige Leitquelle
F0220319		ChangeFlexProfile: Ungültiges Bewegungsgesetz
F0220320		ChangeFlexProfile: Ungültiger Bewegungsschritt-Typ
F0220321		ChangeFlexProfile: Leitachsabschnitt 0 oder negativ
F0220322		ChangeFlexProfile: Ungültige Randwertkombination
F0220323		ChangeFlexProfile: Ungültige Randwerte
F0220324		ChangeFlexProfile: Ungültiger Hub
F0220325		ChangeFlexProfile: Ungültige Wendepunktverschiebung
F0220326		ChangeFlexProfile: Ungültige Geradensteigung
F0220327		ChangeFlexProfile: Ungültige Kurventabelle
F0220328		ChangeFlexProfile: Bewegungsgesetz + Flex-Eigenschaft
F0220329		ChangeFlexProfile: Geschwindigkeitssprung Bewegungsprofil
F0220330		ChangeFlexProfile: Beschleunigungssprung Bewegungsprofil
F0220331		ChangeFlexProfile: Rucksprung Bewegungsprofil
F0220332		ChangeFlexProfile: Max. Verfahrensgeschwindigkeit ungültig
F0220333		ChangeFlexProfile: Max. Beschleunigung ungültig
F0220334		Keine Referenz auf FlexProfile-Bewegungsprofil gefunden
F0220335	F2229335	FlexProfile: Satz ist schreibgeschützt
	F2229336	FlexProfile: Ungültige StepNumber
	F2229337	FlexProfile: SetNumber größer als Maximalwert
	F2229338	ChangeFlexProfile: Ungültige Nenn-Leitachsgeschwindigkeit
	F2229339	ChangeFlexProfile: Ungültige Op-Mode, Profilbearbeitung
F0220341	F2229341	FlexEvent: Satz ist schreibgeschützt
	F2229342	FlexEvent: SetNumber größer als Maximalwert
F0220343		ChangeFlexEvent: Listenlänge Eventeigensch. Inkonsistent
F0220344		FlexEvent: Ungültige Bewegungsschritt-Zuordnung FlexEvent
F0220345		ChangeFlexEvent: Ungült. Auslöser-Modus für FlexEvent
F0220346		ChangeFlexEvent: Ungültige Aktions-Modus für FlexEvent

ML_PLCopen.library

ErrorIdent / Additional1		Diagnose
	F2229347	FlexProfile ungültig. Einfaches Ausführen nicht möglich
	F2229348	FlexProfile ungültig. Zykl. Ausführen nicht möglich
F0220349		ChangeFlexProfile: Maximaler Ruck ungültig
F0220350		ChangeFlexProfile: Bewegungsgrenzwerte zu niedrig
	F2229351	Aufsynchronisierungsgeschwindigkeit unter Minimalwert
	F2229352	Aufsynchronisierungsgeschwindigkeit über Maximalwert
	F2229353	Aufsynchronisierungsgeschwindigkeit unter Minimalwert
	F2229354	Aufsynchronisierbeschleunigung über Maximalwert
F0220355		ChangeFlexProfile: Ungültige Abfolge von Flex-Schritten
	F2220356	Ungültige Filterzeitkonstante, Leitachsposition additiv
	F2220357	Ungültige Filterzeitkonstante, Winkelverschiebung Tabelle
	F2220358	Ungültige Filterzeitkonstante, Positionssollwert additiv
	F2229359	FlexProfile: Ungültiger Master-Offset
F0220360	F2229360	Kommando in 'Controlled Motion' zurückgewiesen
F0220361		ChangeFlexProfile: Ungültige Leitachsquelle
	F2229361	Kommando durch Achse nicht unterstützt (Konfiguration)
	F2229362	Kommando durch Achse nicht unterstützt (Funktionspaket)
	F2229363	MotionProfile ungültig

Tab. 6-164: Diagnosen der Motion FBs (Achszustand) II



Je nach vorgewählter Achsreaktion tritt der Fehlers **F*22*xxx** auf als

- F0220xxx, für den Fall keine Fehlerreaktion durch die Achse
- F2229xxx, für den Fall "Achse Halt", bestmögliche Stillsetzung.

ErrorIdent / Additional1		Diagnose
F0220501		Falscher Datentyp
F0220502		ProbeModule außerhalb des zulässigen Bereichs
F0220503		ProbeInput außerhalb des zulässigen Bereichs
F0220504		ExpectWindowPosEdgeMode außerhalb zulässiger Bereich
F0220505		ExpectWindowNegEdgeMode außerhalb zulässiger Bereich
F0220506		ExpectWindowPosEdgeStart außerhalb zulässiger Bereich
F0220507		ExpectWindowPosEdgeEnd außerhalb zulässiger Bereich
F0220508		ExpectWindowNegEdgeStart außerhalb zulässiger Bereich
F0220509		ExpectWindowNegEdgeEnd außerhalb zulässiger Bereich
F0220510		ExpectWindowPosEdgeSetPoint außerhalb des zul. Bereichs
F0220511		ExpectWindowNegEdgeSetPoint außerhalb des zul. Bereichs

ErrorIdent / Additional1	Diagnose
F0220512	MarkerFailureEnable außerhalb des zulässigen Bereichs
F0220513	MaxNumberOfMarkerFailure außerhalb des zul. Bereichs
F0220514	Interpolation liegt außerhalb des zulässigen Bereichs
F0220515	TimeStampEnable liegt außerhalb des zulässigen Bereichs
F0220516	DeadTimePosEdge liegt außerhalb des zulässigen Bereichs
F0220517	DeadTimeNegEdge liegt außerhalb des zulässigen Bereichs
F0220518	ProbeData außerhalb des zulässigen Bereichs
F0220519	PositionEvalMode liegt außerhalb des zulässigen Bereichs
F0220520	DiffEvalMode liegt außerhalb des zulässigen Bereichs
F0220521	TOUCHPROBE_REF ungültig oder kein Messtaster konfig.
F0220522	AXIS_REF nicht gültig oder keine Achse konfiguriert

Tab. 6-165: Diagnosen der Motion FBs (Achszustand) III



Je nach vorgewählter Achsreaktion tritt der Fehlers **F*22*xxx** auf als

- F0220xxx, für den Fall keine Fehlerreaktion durch die Achse
- F2229xxx, für den Fall "Achse Halt", bestmögliche Stillsetzung.

ErrorIdent / Additional1	Diagnose
F0220600	Pfad- / Dateiname unbekannt
F0220601	Nicht unterstütztes Dateisystem
F0220602	Eingang 'StartIdx' hat einen ungültigen Wert
F0220603	Ungültiger Zustand der Zustandsmaschine
F0220604	Fehlerhafte Datei
F0220605	Eingang 'ArrayDimension' hat einen ungültigen Wert
F0220606	Punkt nicht vorhanden
F0220607	Falsche Anzahl an Koordinaten je Punkt
F0220608	Falsche Kinematik Nummer
F0220609	Falscher Punkttyp, Punkt in Welt- oder Maschinen-Koord.
F2220610	Ungültige Masterachse

Tab. 6-166: Diagnosen der Motion FBs (Achszustand) IV



Je nach vorgewählter Achsreaktion tritt der Fehlers **F*22*xxx** auf als

- F0220xxx, für den Fall keine Fehlerreaktion durch die Achse
- F2229xxx, für den Fall "Achse Halt", bestmögliche Stillsetzung.

ML_PLCopen.library

ErrorIdent / Additional1		Diagnose
F00D1002		Zyklischer Sollwertkanal bereits geöffnet
F00D1003		Zyklischer Kraftsollwertkanal bereits geöffnet
F00D1004		Zyklischer Geschwindigkeitswertkanal bereits geöffnet
F00D1005		Zyklischer Analogwertkanal bereits geöffnet
F00D100A		Fehler beim Lesen C-0-0483
F00D100B		Fehler beim Lesen C-0-0484
F00D100C		Fehler beim Öffnen des zyklischen Kanals
F00D100D		Fehler beim Öffnen des zyklischen Kanals
F00D1014		Keine Referenz zum Öffnen des zyklischen Kanals
	F20D1015	Fehler beim Schließen des zyklischen Kanals
F00D1016		Falsche #-Adresse der Achsdaten
F00D1017		Falsche #-Adresse der Achsdaten
F00D1018		Falsche #-Adresse der Achsdaten
F00D1019		Fehler beim Schließen des zyklischen Kanals
	F20D1025	Fehler beim Schließen des zyklischen Kanals
F00D1027		Fehler beim Schließen des zyklischen Kanals
	F20D1029	Fehler beim Öffnen des zyklischen Kanals
F00D102A		Listenparameter angefordert, ReadParameter
	F20D102B	Fehler beim Öffnen des zykl. Kanals (READ-Access)
	F20D102C	Fehler beim Lesezugriff (DirectReadReal)
	F20D102D	Fehler beim Lesezugriff (DirectReadWord)
	F20D102E	Fehler beim Lesezugriff (DirectReadDword)
F00D1030		Fehler beim Schließen des zyklischen Kanals
	F20D1034	Fehler beim Schreibzugriff (DirectWriteWord)
	F20D1036	Achse wurde nicht gefunden (ML_AXISDATA)
	F20D1037	Ungültiger Schreib/Lesezugriff auf ML_AXISDATA
F00D103B		Ungültige Steuerungsnummer
F00D103C		Ungültige Achsnummer
F00D1205		Daten eines Singleparameters angefordert
F00D1207		Daten größer als 64KByte
F00D1210		Daten eines Singleparameters angefordert
F00D1211		Fehler ReadSERCOSDataStatus, kein Speicher
F00D1218		Kein Speicher
F00D1219		Daten eines Singleparameters angefordert
F00D1220		Nicht unterstützte Auswahl des sercos Elementes
F00D1221		Zeitüberschreitung bei Parameterzugriff

ErrorIdent / Additional1	Diagnose
F00D1222	Datengröße überschritten
F00D1224	Kein Speicher
F00D1225	Daten eines Singleparameters angefordert
F00D1227	Fehler beim Datum schreiben
F00D1230	Fehler ReadSERCOSAttribute, kein Speicher

Tab. 6-167: Diagnosen MB_ReadxxxParameter / MB_WritexxxParameter / Direktzugriff

ErrorIdent / Additional1	MB_ReadxxxParameter / MB_WritexxxParameter / Direktzugriff
F90D1021	Fehler ReadParameter, kein Speicher
F90D1023	Fehler WriteParameter
F90D1024	Fehler WriteParameter, kein Speicher
F90D1032	Fehler beim Zugriff auf die Achsen (1)
F90D1033	Fehler beim Zugriff auf die Achsen (2)
F90D1038	Fehler beim Empfangen MessageQueue
F90D1039	Thread konnte nicht gestartet werden
F90D103A	Endlosschleife verlassen
F90D1204	Kein Speicher
F90D1209	Kein Speicher

Tab. 6-168: Fatale Systemfehler von MB_ReadxxxParameter / MB_WritexxxParameter / Direktzugriff



Je nach vorgewählter Achsreaktion tritt der Fehlers **F*0Dxxxx** auf als

- F00Dxxxx, für den Fall keine Fehlerreaktion durch die Achse
- F20Dxxxx, für den Fall "Achse Halt", bestmögliche Stillsetzung
- F90Dxxxx, keine Fehlerreaktion zum Antrieb, Momentenfreischaltung

ErrorIdent / Additional1	Diagnose
F0290001	Transformation nicht gültig
F0290400	F2299400 Allgemeiner Kinematikfehler
F0290401	F2299401 Ungültige Steuerungsnummer
F0290402	F2299402 Ungültige Kinematiknummer
F0290403	F2299403 Ungültiger Kommandotyp
F0290404	F2299404 Kommando durch Kinematik nicht unterstützt
F0290405	F2299405 Kommando in 'StandStill Incomplete' zurückgewiesen
F0290406	F2299406 Kommando in 'PM Program Stopping' zurückgewiesen

ML_PLCopen.library

ErrorIdent / Additional1		Diagnose
F0290407	F2299407	Kommando in 'StandStill Complete' zurückgewiesen
F0290408	F2299408	Ungültige Kommandonummer
F0290409	F2299409	Kommando in diesem Zustand nicht unterstützt
F0290410	F2299410	Reset bei aktivem ErrorStop, 'state ErrorStop'
F0290411	F2299411	Kommando in 'PM Program Running' zurückgewiesen
F0290412	F2299412	Fehler während der RCL-Programmanwahl
F0290413	F2299413	Kommando im Parametriermodus zurückgewiesen
F0290414	F2299414	Kommando in 'ErrorStop' zurückgewiesen
F0290415	F2299415	Kommando wurde abgebrochen
F0290416	F2299416	Umschaltung in den Zustand 'StandStill' nicht möglich
F0290417	F2299417	RCL-Prozess existiert nicht
F0290501	F2299501	Zwei SPS-Bewegungskommandos pro Motion-Zyklus nicht erlaubt
F0290502	F2299502	Kommando zurückgewiesen - Buffer voll
F0290503	F2299503	Ungültiges Motion Kommando
F0290505	F2299505	Synchronisation auf mehrere Bänder nicht erlaubt

Tab. 6-169: Kinematikzustands-Diagnosen der MLC_Table (1)



Je nach vorgewählter Achsreaktion tritt der Fehlers **F*29*xxx** auf als

- F0290xxx, für den Fall keine Fehlerreaktion durch die Achse
- F2299xxx, für den Fall "Achse Halt", bestmögliche Stillsetzung
- F9299xxx, keine Fehlerreaktion zum Antrieb, Momentenfreischaltung

ErrorIdent / Additional1		Diagnose
F0290501	F2299501	Zwei SPS-Bewegungskommandos pro Motion-Zyklus nicht erlaubt
F0290502	F2299502	Kommando zurückgewiesen - Buffer voll
F0290503	F2299503	Ungültiges Motion Kommando
F0290505	F2299505	Synchronisation auf mehrere Bänder nicht erlaubt
	F2299506	Nicht unterstützte Überschleifart
	F2299507	Ungültiger Überschleifmodus
	F2299508	Ungültiges Koordinatensystem
	F2299509	Bandkonfiguration zur Zeit nicht erlaubt
	F2299510	Band nicht konfiguriert
	F2299511	Band-Synchronisationsfehler
	F2299512	Ungültige Band-Synchronisierart
	F2299513	Ungültige Band-Fehlerreaktionsart

ErrorIdent / Additional1	Diagnose
F2299514	Band-Startlänge ungültig
F2299515	Band-Startlänge größer Gesamtlänge
F2299516	Falscher PCS-Index
F2299517	Falsches Koordinatensystem ausgewählt
F2299518	TCP nicht im Bandarbeitsbereich
F2299528	TeachIn: Kein Name angegeben
F2299529	TeachIn: Name zu lang
F229952A	TeachIn: Name falsch
F229952B	TeachIn: Punkt existiert bereits
F229952C	TeachIn: Punkt existiert nicht
F229952D	TeachIn: Falsche Option
F229952E	TeachIn: Datei fehlerhaft
F229952F	TeachIn: Falsche Koordinate
F2299530	TeachIn: Falsches Koordinatensystem
F2299531	TeachIn: Datei kann nicht geöffnet werden
F2299532	Werte für relatives Absynchronisieren ungültig
F2299533	Kommandopuffer voll
F2299534	Transformation unterstützt Freiheitsgrad nicht

Tab. 6-170: Kinematikzustands-Diagnosen der MLC_Table (2)



Je nach vorgewählter Achsreaktion tritt der Fehlers **F*29*xxx** auf als

- F0290xxx, für den Fall keine Fehlerreaktion durch die Achse
- F2299xxx, für den Fall "Achse Halt", bestmögliche Stillsetzung
- F9299xxx, keine Fehlerreaktion zum Antrieb, Momentenfreischaltung

ErrorIdent / Additional1	Diagnose
F9299519	JCS ist nicht gesetzt
F929951A	BCS ist nicht gesetzt
F929951B	WCS ist nicht gesetzt
F929951C	PCS ist nicht gesetzt
F929951D	Trafo-Manager nicht gesetzt
F929951E	Preshape execute error
F929951F	Shape execute error
F9299520	Postshape execute error
F9299521	Bewegung bei diesem Kommando nicht erlaubt

ML_PLCopen.library

ErrorIdent / Additional1	Diagnose
F9299522	Robot nicht gesetzt
F9299523	Bandzeit nicht gesetzt
F9299524	Initialisierungsfehler Absynchronisation
F9299525	Ipo: Zykluszeit Null
F9299526	Ipo: Distanz kleiner Null
F9299527	Ipo: Zu viele Elemente

Tab. 6-171: *Fatale Systemfehler von Kinematikzustands-Diagnosen*

Je nach vorgewählter Achsreaktion tritt der Fehlers **F*29*xxx** auf als

- F0290xxx, für den Fall keine Fehlerreaktion durch die Achse
- F2299xxx, für den Fall "Achse Halt", bestmögliche Stillsetzung
- F9299xxx, keine Fehlerreaktion zum Antrieb, Momentenfreischaltung

7 RMB_PLCopenFieldBus.library

7.1 RMB_PLCopenFieldBus.library, Allgemeines

Die RMB_PLCopenFieldBus.library enthält die Funktionsbausteine, die mit den Feldbusachsen kommunizieren.

Bedeutung der Funktionsbaustein-Präfixe

- **Präfix MC_**
Dieses Präfix bedeutet, dass dieser Baustein zu 100% PLCopen-konform ist.
- **Präfix MB_**
Dieses Präfix bedeutet, dass dieser Baustein zwar nach der Systematik der PLCopen ausgeführt ist, von dieser abweicht, bzw. diese ergänzt, bzw. nicht definiert ist.
- **Präfix MP_**
Dieses Präfix wird bei systemintern genutzten Funktionsbausteinen verwendet.

Die Funktionsbausteine, Funktionen, Datentypen und Konstanten sind in nachstehenden Ordnern und Globalen Variablenlisten enthalten:

- [Parameter-Konstanten, Seite 359,](#)
- [Datatypes, Seite 360,](#)
- [Motion_FBs, Seite 363,](#)
- [Parameter_FBs, Seite 383, und](#)
- [System_FBs, Seite 420.](#)

7.2 Parameter-Adresskonstanten

7.2.1 Zugriff auf Daten eines (Feldbus-)Antriebs

Die Daten eines Antriebs sind über Parameterzugriff lesend und teilweise auch schreibend erreichbar. Folgende Parameterarten werden für Feldbusantriebe am Profibus unterschieden:

S-Parameter Die S-Parameter entsprechen den in der sercos interface-Norm festgelegten Standard-Parametern.

P-Parameter Die P-Parameter sind vom Antriebshersteller definierte hersteller- oder produktspezifische Parameter im Sinne der sercos interface-Norm.

Jeder Systemparameter wird durch seine Ident-Nr. eindeutig identifiziert.

Dem Anwender stehen dazu für P- und S-Parameter Vereinbarungsdateien für Globale Konstanten zur Verfügung.

Vereinbarungsdateien für Globale Konstanten

- [GVL_S_Param_ID, Seite 359,](#)
- [GVL_P_Param_ID, Seite 360](#)

7.2.2 GVL_S_Param_ID

Kurzbeschreibung Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.

Der Parameter-Konstantenname ergibt sich zu

S-0-0001 ⇒ FP_S_0_0001 usw..

Nummer und Parameter-Konstantenname können alternativ verwendet werden.

RMB_PLCopenFieldBus.library

Bibliothek	Bereich
RMB_PLCopenFieldbus	Global Variables Lists/ Parameter ID

Tab. 7-1: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_S_Param_ID

```

VAR_GLOBAL CONSTANT
  FP_S_0_0000: MB_IDN:= 0 ; (* Dummy S parameter *)
  FP_S_0_0001: MB_IDN:= 1 ; (* NC cycle time (TNcyc) *)
  FP_S_0_0002: MB_IDN:= 2 ; (* SERCOS cycle time (TScyc) *)
  (* ..... *)
END_VAR

```

Siehe auch:

- [MB_IDN, Seite 361.](#)

7.2.3 GVL_P_Param_ID

Kurzbeschreibung Die Globale Konstantenliste enthält die Zuordnung der Adresskonstanten zum jeweiligen Parameter.

Der Parameter-Konstantenname ergibt sich zu

P-0-0001 ⇒ FP_P_0_0001 usw..

Nummer und Parameter-Konstantenname können alternativ verwendet werden.

Bibliothek	Bereich
RMB_PLCopenFieldbus	Global Variables Lists/ Parameter ID

Tab. 7-2: Bibliothekszuordnung

Deklaration der Globalen Konstantenliste GVL_P_Param_ID

```

VAR_GLOBAL CONSTANT
  FP_P_0_0001: MB_IDN:= 32769 ; (* Switching frequency of the power output stage *)
  FP_P_0_0002: MB_IDN:= 32770 ; (* SPI flash aging counter *)
  FP_P_0_0003: MB_IDN:= 32771 ; (* Status of parameter buffering *)
  FP_P_0_0004: MB_IDN:= 32772 ; (* Velocity loop smoothing time constant *)
  (* ..... *)
END_VAR

```

Siehe auch:

- [MB_IDN, Seite 361.](#)

7.3 Datatypes

7.3.1 Datatypes, Allgemeines

Der Ordner enthält Datentypen, die die Arbeit der Bibliothekselemente der "RMB_PLCopenFieldbus.library" unterstützen.

Für den Anwender sind lediglich die nachstehenden beiden Datentypen von Interesse:

- [MB_IDN \(Feldbusantriebe\), Seite 361,](#) und

- [AXIS_REF \(Feldbusantriebe\), Seite 361,](#)



Die Verwendung von Elementen, außer der hier beschriebenen, ist zu unterlassen, da Nebenwirkungen, einschließlich Störungen im Bewegungsablauf der Achsen, nicht ausgeschlossen werden können!

7.3.2 MB_IDN

Kurzbeschreibung

Der Datentyp erlaubt die Definition einer systemspezifischen Parameterlänge.

Für die Parameter der IndraLogic XLC bzw. IndraMotion MLC sind dies 64 Bit.

S- und P-Parameter werden automatisch auf 64 Bit aufgefüllt.

Bibliothek	Bereich
RMB_PLCopenFieldbus	Datatypes

Tab. 7-3: Bibliothekszuordnung

Programm:

```
TYPE
  MB_IDN : LINT ;
END_TYPE
```

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
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. 7-4: Parameter-Offsets

7.3.3 AXIS_REF

Kurzbeschreibung

Die Struktur enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt (Achse).

Die Informationsübergabe erfolgt per VAR_IN_OUT an den jeweiligen in der "RMB_PLCopenFieldBus.library" definierten PLCopen-Funktionsbaustein.



Nur für Feldbusantriebe/ -achsen, die am PROFIBUS DP angeschlossen sind.

RMB_PLCopenFieldBus.library

Bibliothek	Bereich
RMB_PLCopenFieldBus	Datatypes

Tab. 7-5: Bibliothekszuordnung

Programm:

```

(* Structure of PLCopen element AXIS_REF. The AXIS_REF is a structure that contains *)
(* information on the corresponding Object. *)
(* It is used as a VAR_IN_OUT in all Motion Control Function Blocks. *)
TYPE AXIS_REF :
STRUCT
  CntrlNo: CONTROLS := LOCAL_CNTRL; (* control reference number *)
  AxisNo: OBJECTS := FieldBusAxis_1; (* Steuerungsnummer, Init: lokale Steuerung *)
  IndraDrive: MP_Drive; (* object reference number *)
  IndraDriveParameter: MP_DriveParameter; (* Achsreferenznummer, Init: FieldBusAxis_1 *)
END_STRUCT
END_TYPE

```

Funktionsbeschreibung

Die Struktur AXIS_REF (RIL_CommonTypes) enthält Adress-Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt (Achsnnummer).

Im Unterschied zur AXIS_REF-Struktur (RIL_CommonTypes), die die Adressierung von SERCOS-basierten IndraDrives unterstützt, wird die AXIS_REF-Struktur (RMB_PLCopenFieldbus) durch zwei zusätzliche Elemente (MP_Drive und MP_DriveParameter) erweitert. Diese beiden Elemente werden nur durch das System genutzt.

Die Achsnnummer kann sich von der Stationsnummer, die am Antrieb eingestellt ist, unterscheiden.

Für SERCOS-basierte Achsen wird die globale Variablenliste "MLCVarGlobal" mit allen SERCOS-basierten Achsen automatisch angelegt und gepflegt.

Für die Feldbusantriebe ist eine globale Variablenliste hier "GVL_FieldBusDrives" mit allen Feldbusantrieben manuell anzulegen und zu pflegen.

Die ersten beiden Elemente jedes Antriebs "CntrlNo" und "AxisNo" werden in ihr vorbelegt. Die Elemente "MP_Drive" und "MP_DriveParameter" bleiben unverändert. Sie müssen deshalb nicht in die Liste aufgenommen werden.

Globale Variablenliste "GVL_FieldBusDrives", Beispiel

```

VAR GLOBAL
Drive_X: FieldBusDrives.AXIS_REF :=
  (CntrlNo:=LOCAL_CNTRL,
   AxisNo:=FieldBusAxis_11); // IndraDrive X - example
Drive_Y: FieldBusDrives.AXIS_REF :=
  (CntrlNo:=LOCAL_CNTRL,
   AxisNo:=FieldBusAxis_12); // IndraDrive Y - example
Drive_Z: FieldBusDrives.AXIS_REF :=
  (CntrlNo:=LOCAL_CNTRL,
   AxisNo:=FieldBusAxis_13); // IndraDrive Z - example
Drive_n: FieldBusDrives.AXIS_REF :=
  (CntrlNo:=LOCAL_CNTRL,
   AxisNo:=FieldBusAxis_14); // IndraDrive n - example
END_VAR

```

Namensraumangabe FieldBusDrives

Die Namensraumangabe "FieldBusDrives." muss bei einem Feldbusantrieb zwingend vorangestellt werden.

Deklaration AXIS_REF

RMB_PLCopenFieldBus.library

```
Drive_A: FieldBusDrives.AXIS_REF: =
```

Die Datentypen hierfür sind in der RMB_PLCopenFieldBus.library, Ordner "Datatypes", Element "AXIS_REF" enthalten.

Zuweisung CntrlNo

```
(CntrlNo: = LOCAL_CNTRL
```

Der Enumerationswert LOCAL_CNTRL ist in der RIL_CommonTypes.lib, Ordner "Datatypes of logical addresses", CONTROLS enthalten.

Die Feldbusbausteine beziehen sich immer auf die lokale SPS, also LOCAL_CNTRL.

Zuweisung AxisNo

```
AxisNo: = FieldBusAxis_1);
```

Der Enumerationswert FieldBusAxis_1 ist in der RIL_CommonTypes.lib, Ordner "Datatypes of logical addresses", OBJECTS enthalten.

Um Doppeldeutigkeiten zu vermeiden, ist hier für die Feldbusachsen ein eigener Nummernbereich

FieldBusAxis_1 (6001) bis FieldBusAxis_99 (6099)

hinterlegt.

Bei Feldbusantrieben werden die Angaben zur Steuerung und die Achsnummer zur Fehlerausgabe (ErrorIdent) verwendet.

Eine Funktionalität ist mit der Vergabe der Achsnummer nicht verbunden.



Es ist sinnvoll, PROFIBUS-Stationsnummer und Achsnummer abzugleichen...

7.4 Funktionsbausteine zur Einzelachssteuerung (Motion_FBs)

7.4.1 Motion_FBs, Allgemeines



Die nachstehend genannten Bausteine dienen zur Bewegungssteuerung.

Sie sind nur für IndraDrive C/M ab MPx2VRS, IndraDrive Cs ab MPx16VRS gekoppelt über Profibus, verwendbar.

Zustandsdiagramm

Das nachstehende Diagramm definiert normativ das Verhalten von Motion-Control-Achsen aus der Sicht des Anwenders und bei der Reaktion auf Ausnahmesituationen in Programmen.

Grundsätzlich werden Motion-Kommandos sequentiell gegeben und abgearbeitet, selbst wenn die SPS in der Lage ist, eine parallele Verarbeitung vorzunehmen. Diese Kommandos folgen dabei dem Zustandsdiagramm der MotionControl-Achsen.

Jede Achse ist dabei in genau einem der definierten Zustände (siehe nachstehendes Zustandsdiagramm). Jedes Bewegungskommando stellt dabei einen Zustandsübergang für die konkrete Achse dar. Die Folge dieser Übergänge beschreibt das gesamte Verhalten der Achse.

Das Diagramm konzentriert sich auf die Einzelachse.



Für Feldbusantriebe funktionieren nur Einzelachs-Bausteine!

Die folgenden Funktionsbausteine mit "Verwaltungscharakter" haben keinen Einfluss auf das Zustandsdiagramm:

- [MC_ReadActualPosition](#), Seite 415,
- [MC_ReadAxisError](#), Seite 417
- [MC_ReadStatus](#), Seite 418,

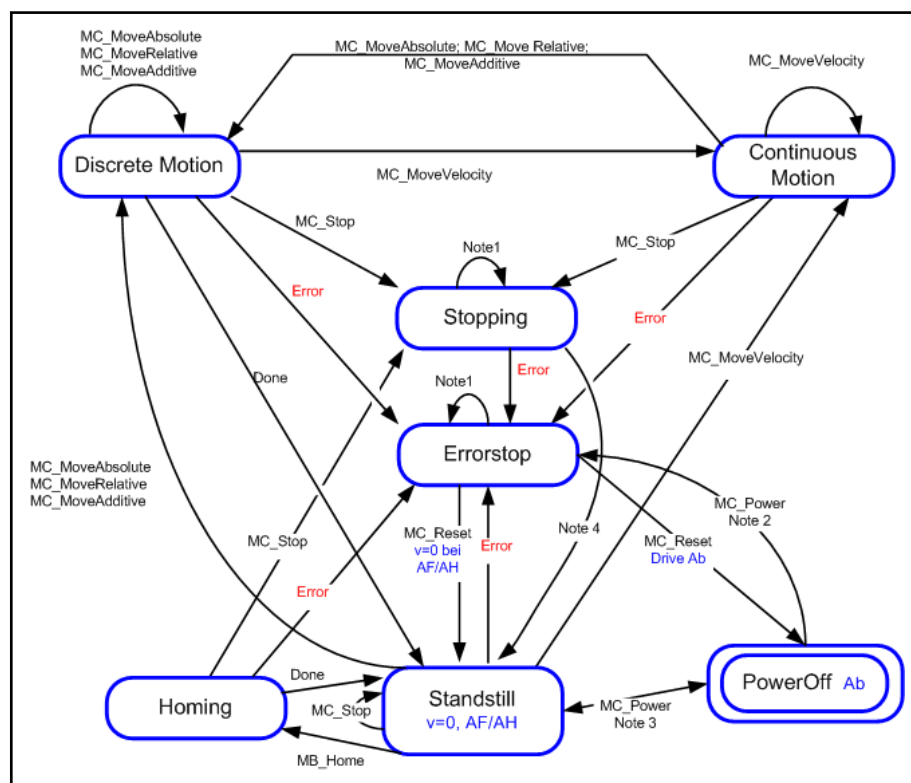


Abb. 7-1: MotionControl-Zustandsdiagramm einer Feldbusachse

Legende

(Note 1) - Alle Funktionsbausteine können gerufen werden, ohne dass sie bearbeitet werden, mit Ausnahme von MC_Reset und Error - diese erzeugen jeweils den Übergang zu StandStill oder ErrorStop.

(Note 2) - Power.Enable= TRUE und Fehler in der Achse.

(Note 3) - Power.Enable= TRUE und fehlerfreie Achse.

(Note 4) - MC_Stop.Done AND NOT MC_Stop.Execute

Ein Fehler im Antrieb führt abhängig von der programmierten Fehlerreaktion irgendwann zur Leistungsabschaltung. Die Zustandsmaschine bleibt aber im ErrorStop, MC_Reset führt bei abgeschalteter Leistung in den Power-Off-Zustand.

	Funktion
MC_Power , Seite 365	steuert die Leistungszuschaltung (ein oder aus).
MC_Reset , Seite 366	realisiert den Übergang vom Zustand 'ErrorStop' in den Zustand 'StandStill' durch Rücksetzen aller achsinternen Fehler.

	Funktion
MC_Stop, Seite 368	realisiert einen gesteuerten Bewegungsstopp und überführt die Achse in den Zustand 'Stopping'. Dies stoppt jede beliebige Funktionsbausteinabarbeitung. Mit Erreichen des Stillstandes wird der "Done"-Ausgang gesetzt. Bevor die Achse nicht den Zustand "Stopping" erreicht hat, kann kein anderes Kommando gestartet werden.
MB_Stop, Seite 369	realisiert einen gesteuerten Bewegungsstopp und überführt die Achse in den Zustand 'Stopping'. Dies stoppt jede beliebige Funktionsbausteinabarbeitung. Mit Erreichen des Stillstandes wird der "Done"-Ausgang gesetzt. Solange "ExecuteLock" TRUE ist, verbleibt die Achse im Zustand 'Stopping' und kann durch kein anderes Kommando bewegt werden.
MB_Home, Seite 371	steuert Achsen mit Inkrementalgeber bei der Ausführung der "search home"-Sequenz. Die Details der Sequenz sind über Achsparameter voreinzustellen. Das "Homing" endet im Stillstand.
MC_MoveAbsolute, Seite 372	Der Funktionsbaustein kopiert die absolut vorgegebenen "Position" in die "Zielposition" und fährt diese an.
MC_MoveAdditive, Seite 375	Der Funktionsbaustein addiert die "Distanz" zur "Zielposition" und fährt diese an.
MC_MoveRelative, Seite 377	Der Funktionsbaustein ermittelt die neue "Zielposition" durch Addition der "Distanz" und der "Istposition" und fährt diese an.
MC_MoveVelocity, Seite 380	steuert eine "endlose" Bewegung mit der jeweils festgelegten Geschwindigkeit. Um die Bewegung zu beenden, muss die FB-Instanz durch eine andere FB-Instanz, die ein neues Kommando ausgibt, unterbrochen werden.

Tab. 7-6: Funktionsbausteine des Ordners "Motion_FBs"

7.4.2 MC_Power

Kurzbeschreibung

Der Funktionsbaustein steuert die Leistungszuschaltung (ein oder aus).

Sie benötigen **für jede reale Achse eine separate Instanz** dieses Bausteines.

Das Aktivieren des Bausteins ist Voraussetzung für jede Bewegung, für Beschleunigen oder Bremsen.

Eine Sondersituation ergibt sich beim Baustein [MC_Stop, Seite 368](#). Hier wird die Position durch den Antrieb aktiv gehalten. Damit kann bei aktivem MC_Stop der Baustein MC_Power nicht abgeschaltet werden.



Der Baustein führt in Verbindung mit einer virtuellen Achse zu einer Fehlermeldung.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Motion_FBs

Tab. 7-7: Bibliothekszuordnung

Schnittstelle

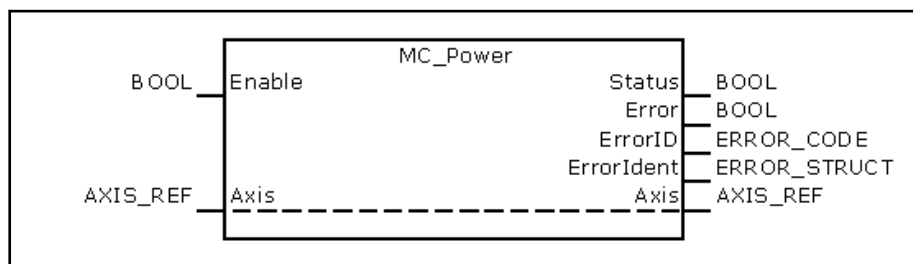


Abb. 7-2: FB MC_Power

RMB_PLCopenFieldBus.library

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Solange "Enable" TRUE ist, ist die Leistung zugeschaltet
VAR_OUTPUT	Status	BOOL	tatsächlicher Zustand der Leistungszuschaltung
	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

Tab. 7-8: Schnittstelle des FB MC_Power

7.4.3 MC_Power, Besonderheiten der Feldbusantriebe

- Fehlerbehandlung:
- Eine Fehlermeldung wird abgesetzt, wenn bei gesetztem "Enable", nach Ablauf einer Wartezeit, der "Status" immer noch FALSE ist. Dies zeigt an, dass ein Hardwareproblem bei der Leistungszuschaltung aufgetreten.
 - Eine Fehlermeldung wird ferner dann abgesetzt, wenn versucht wird, die Leistung in einem Achszustand abzuschalten, in dem dies nicht erlaubt ist.
 - Falls die Leistungszuschaltung fehlschlägt oder während einer Bewegung ausfällt, wird für die Achse "ErrorStop" generiert.

Im Falle eines Konfigurationsfehlers erscheint

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.4.4 MC_Reset

Kurzbeschreibung Der Funktionsbaustein realisiert den Übergang vom Zustand 'ErrorStop' in den Zustand 'StandStill' durch Rücksetzen aller achsinternen Fehler.

 Die Ausgänge der Funktionsbausteininstanzen, die den/die Fehler angezeigt haben, werden nicht beeinflusst.

Steht kein Fehler an, bleibt das Auslösen des MC_Resets ohne Wirkung.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Motion_FBs

Tab. 7-9: Bibliothekszuordnung

Schnittstellenbeschreibung

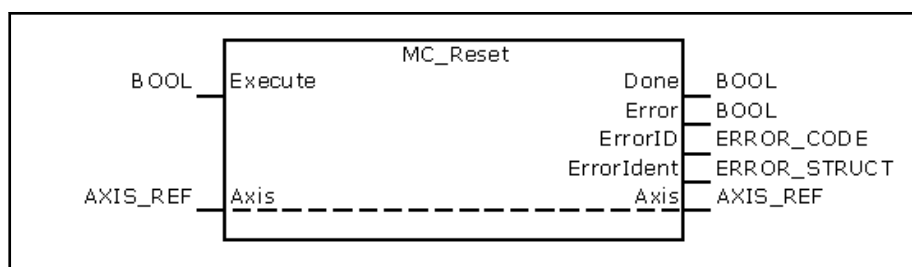


Abb. 7-3: FB MC_Reset

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigenden Flanke
VAR_OUTPUT	Done	BOOL	Fehler sind zurückgesetzt
	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

Tab. 7-10: Schnittstelle des FB MC_Reset

Signal-Zeit-Diagramm

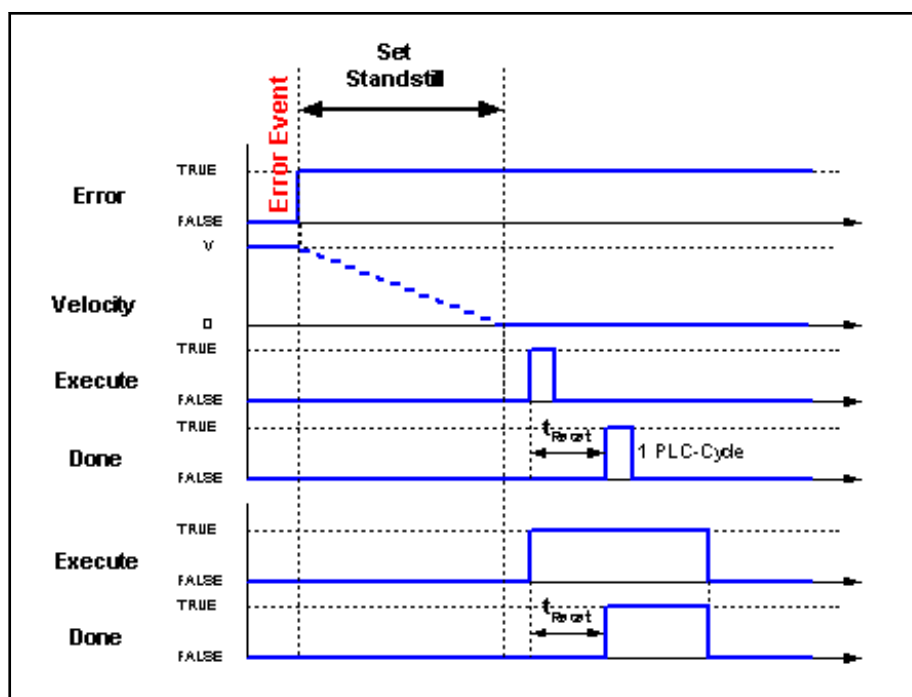


Abb. 7-4: Zeitdiagramm des MC_Reset

7.4.5 MC_Reset, Besonderheiten der Feldbusantriebe

Der Baustein benutzt den Parameter "S-0-0099", C0500 Reset Zustandsklasse 1.

Steht kein Fehler an, bleibt das Auslösen des MC_Resets ohne Wirkung.

Ein Fehlerlöscher-Kommando "C5" wird jedoch in jedem Fall an den Antrieb abgesetzt.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

RMB_PLCopenFieldBus.library

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.4.6 MC_Stop

Kurzbeschreibung

Der Funktionsbaustein realisiert einen gesteuerten Bewegungsstop und überführt die Achse in den Zustand 'Stopping'. Dies stoppt jede beliebige Funktionsbausteinabarbeitung. Mit Erreichen des Stillstandes wird der "Done"-Ausgang gesetzt.
Die Position wird aktiv gehalten.

 Solange "Execute" TRUE ist, verbleibt die Achse im Zustand 'Stopping'.
Bevor die Achse nicht den Zustand "Stopping" erreicht hat, kann kein anderes Kommando gestartet werden.

Im Zustand 'Stopping' ist es nicht möglich, die Leistung ([MC_Power, Seite 365,](#)) der Achse abzuschalten. Wird das "Enable" der MC_Power-Instanz deaktiviert, bleibt dessen "Status" trotzdem, bis das "Execute" der MC_Stop-Instanz gleichfalls deaktiviert wird, eingeschaltet.

 Der MC_Stop wird nicht für einen IndraDrive mit aktivierten Betriebshalt (P-0-0558 Bit 0 = 1) unterstützt.
Der Funktionsbaustein setzt den Done-Ausgang nie auf TRUE.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Motion_FBs

Tab. 7-11: Bibliothekszuordnung

Schnittstellenbeschreibung

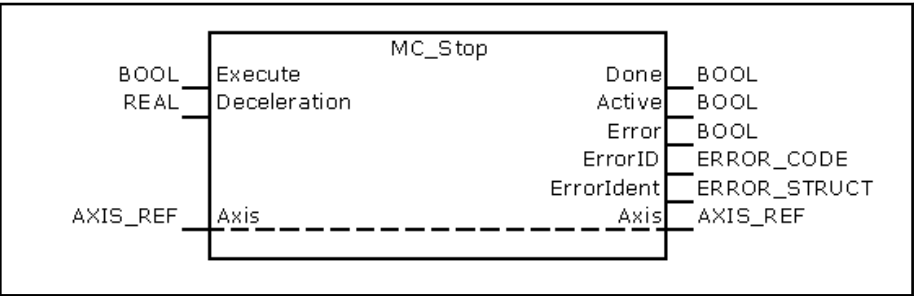


Abb. 7-5: FB MC_Stop

	Name	Typ	Kommentar ¹⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Startet die Aktion bei steigenden Flanke. Baustein ist nicht unterbrechbar, bis Zustand "Stopping" erreicht ist.

¹⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

	Name	Typ	Kommentar ¹⁾
	Deceleration	REAL	Verzögerung (stets größer 0, sonst kein "Stoppen").
VAR_OUTPUT	Done	BOOL	Stillstand erreicht
	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

Tab. 7-12: Schnittstelle des FB MC_Stop

Signal-Zeit-Diagramm

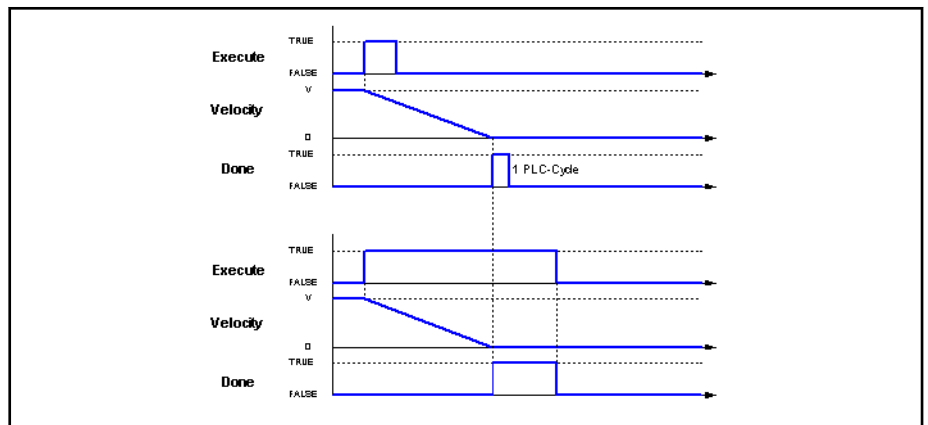


Abb. 7-6: Zeitdiagramm von MC_Stop

7.4.7 MC_Stop, Besonderheiten der Feldbusantriebe

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.4.8 MB_Stop

Kurzbeschreibung

Der Funktionsbaustein realisiert einen gesteuerten Bewegungsstopp und überführt die Achse in den Zustand 'Stopping'. Dies stoppt jede beliebige Funktionsbausteinabarbeitung. Mit Erreichen des Stillstandes wird der "Done"-Ausgang gesetzt.

Die Position wird aktiv gehalten.

¹⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

RMB_PLCopenFieldBus.library



Solange "ExecuteLock" TRUE ist, verbleibt die Achse im Zustand 'Stopping' und kann durch kein anderes Kommando bewegt werden.

Im Gegensatz zum "MC_Stop, Seite 368," kann der Baustein, sobald "ExecuteLock" FALSE ist, durch einen anderen Baustein unterbrochen werden.

Im Zustand 'Stopping' ist es nicht möglich, die Leistung (MC_Power, Seite 365,) der Achse abzuschalten. Wird das "Enable" der MC_Power-Instanz deaktiviert, bleibt dessen "Status" trotzdem, bis das "ExecuteLock" der MB_Stop-Instanz gleichfalls deaktiviert wird, eingeschaltet.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Motion_PLCopen

Tab. 7-13: Bibliothekszuordnung

Schnittstellenbeschreibung

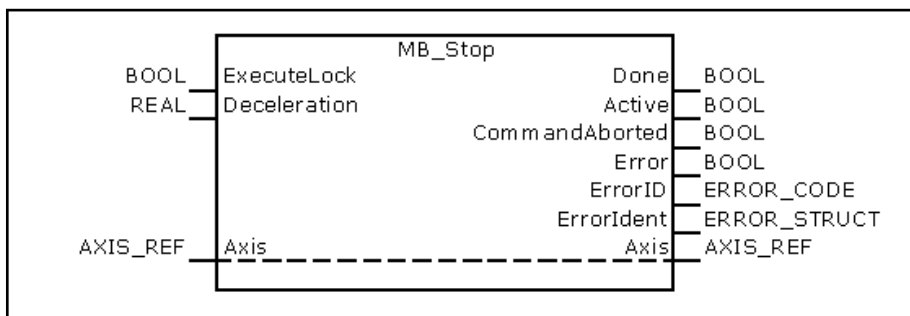


Abb. 7-7: FB MB_Stop

	Name	Typ	Kommentar ²⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	ExecuteLock	BOOL	Startet das Stoppen der Achse mit der steigenden Flanke; solange "ExecuteLock" TRUE ist, ist kein Start einer Bewegung möglich.
	Deceleration	REAL	Verzögerung (stets größer 0, sonst kein "Stoppen").
VAR_OUTPUT	Done	BOOL	Der Stillstand ist erreicht.
	Active	BOOL	Das Kommando ist aktiv. Die Bearbeitung der Daten nach Abschluss der Vorverarbeitung läuft.
	Command-Aborted	BOOL	Das Kommando wird durch ein anderes Kommando abgebrochen.
	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

Tab. 7-14: Schnittstelle des FB MB_Stop

²⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

Signal-Zeit-Diagramm

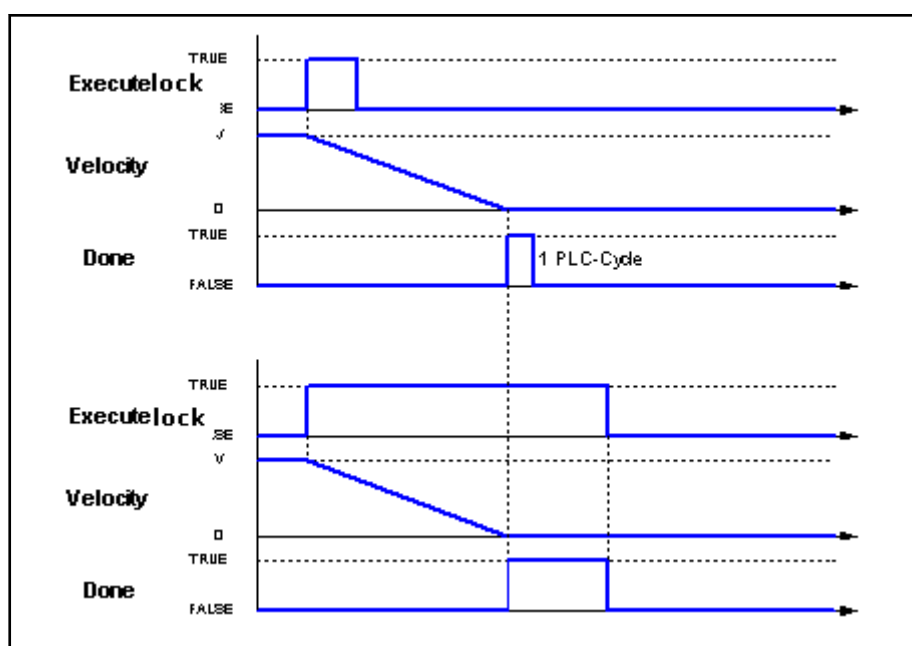


Abb. 7-8: Zeitdiagramm von MB_Stop

7.4.9 MB_Stop, Besonderheiten der Feldbusantriebe

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "ExecuteLock".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.4.10 MB_Home

Kurzbeschreibung

Der Funktionsbaustein steuert **Achsen mit Inkrementalgeber** bei der Ausführung der "search home"-Sequenz. Die Details der Sequenz sind über Achsparameter voreinzustellen. Das "Homing" endet im Stillstand.

Bibliothek	Bereich
RMB_PLCOpenFieldBus	Motion_FBs

Tab. 7-15: Bibliothekszuordnung

Schnittstellenbeschreibung

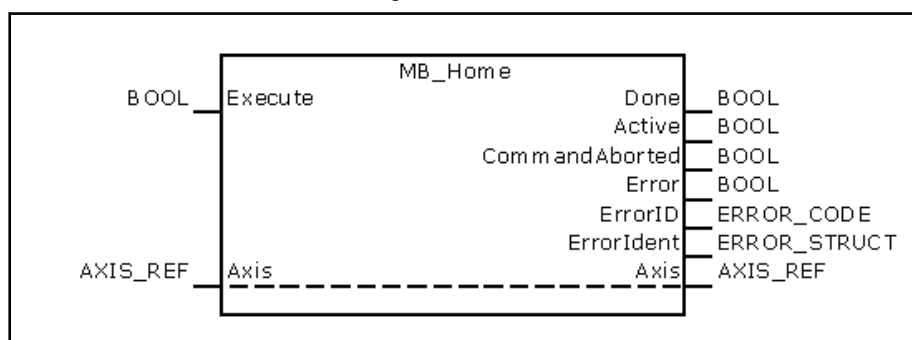


Abb. 7-9: FB MB_Home

RMB_PLCopenFieldBus.library

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
VAR_OUTPUT	Done	BOOL	Homing abgeschlossen, Geschwindigkeit 0
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 7-16: Schnittstelle von FB MB_Home

Signal-Zeit-Diagramm

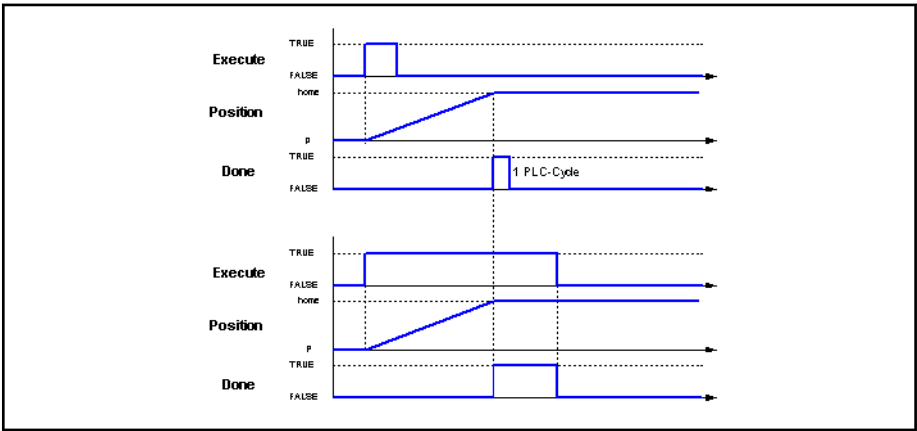


Abb. 7-10: Zeitdiagramm von MB_Home

7.4.11 MB_Home, Besonderheiten der Feldbusantriebe

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.4.12 MC_MoveAbsolute

Kurzbeschreibung Der Funktionsbaustein kopiert die absolut vorgegebenen "Position" in die "Zielposition" und fährt diese an.



Die Anwendung des Bausteins setzt die vorherige Leistungsschaltung über eine Instanz des [MC_Power](#), [Seite 365](#), (bei realen Achsen) voraus.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Motion_FBs

Tab. 7-17: Bibliothekszuordnung

Schnittstellenbeschreibung

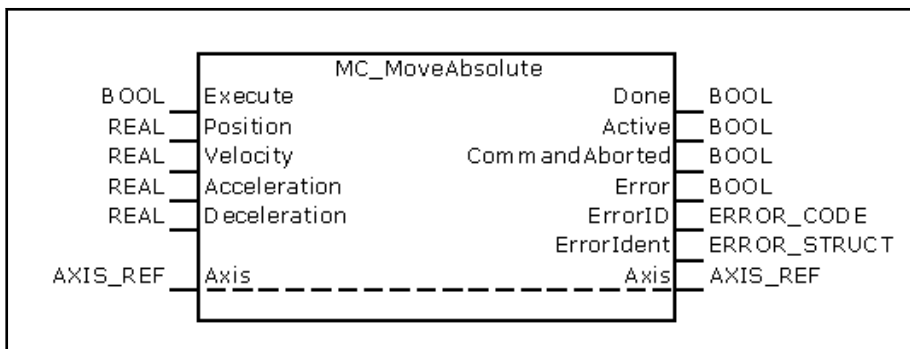


Abb. 7-11: FB MC_MoveAbsolute

	Name	Typ	Kommentar ³⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Position	REAL	Zielposition für die Bewegung (+/-)
	Velocity	REAL	Maximalwert der Geschwindigkeit (stets +) (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).
	Deceleration	REAL	Verzögerung (stets +).
VAR_OUTPUT	Done	BOOL	gewünschte Position erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 7-18: Schnittstelle von FB MC_MoveAbsolute

Implementationsbeispiel:

In folgenden Bild sind zwei Beispiele für die Zusammenarbeit von zwei FB-Instanzen des "MC_MoveAbsolute" dargestellt:

1. Der linke Teil des Zeitdiagramms zeigt den Fall, dass Instance2 nach Instance1 gerufen wird. Nachdem Instance1 die gewünschte Position 6000 (Geschwindigkeit 0) realisiert hat, aktiviert sein Ausgang "Done" Instance2 mit der Zielposition 10000.
2. Die rechte Seite des Zeitdiagramms zeigt wie die Instance2 bereits aktiviert wird, während die Instance1 noch arbeitet. In diesem Fall wird die Bewegung von Instance1 durch das "Test"-Signal gestoppt und abgebrochen. Instance2 fährt seine Zielposition 10000 direkt an, obwohl die Position 6000 noch nicht erreicht wurde.

³⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

RMB_PLCOpenFieldBus.library

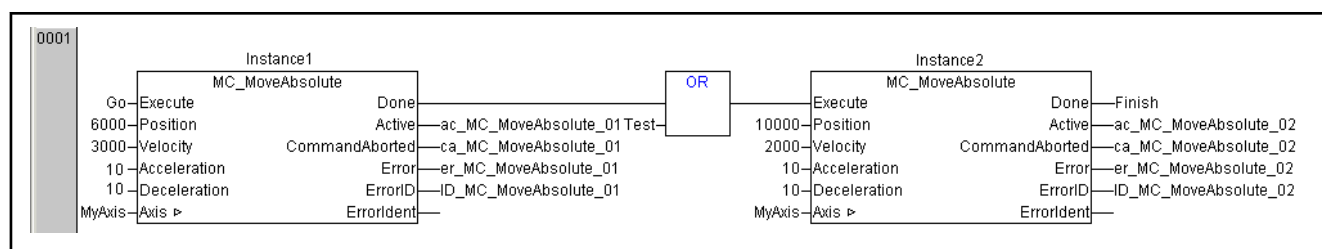


Abb. 7-12: Zwei Instanzen von MC_MoveAbsolute

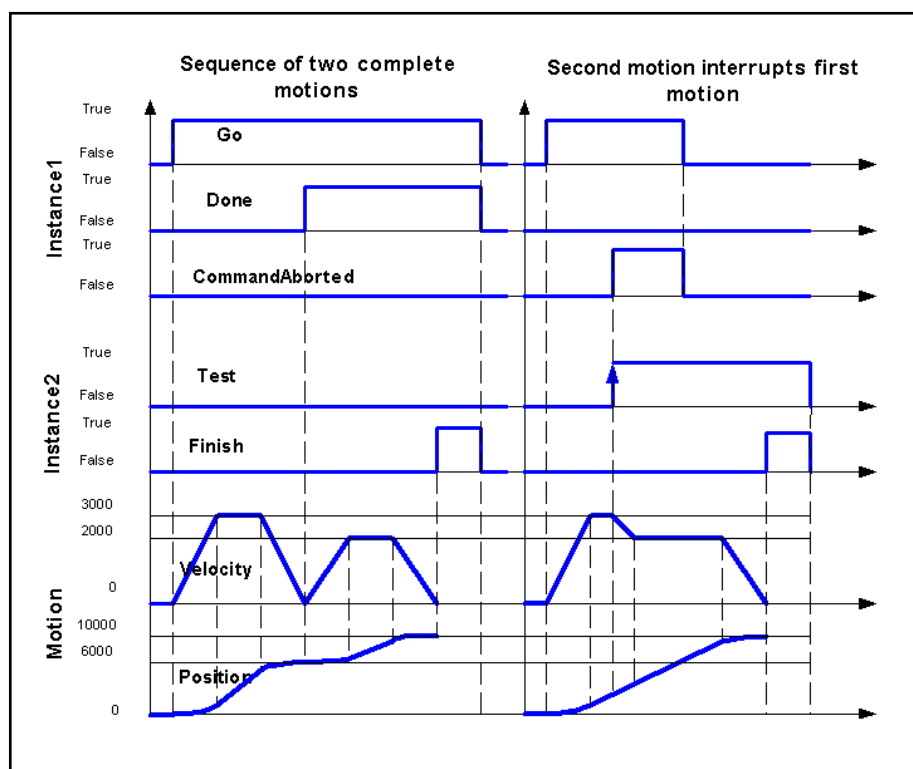


Abb. 7-13: Zeitdiagramm: zwei Instanzen von MC_MoveAbsolute

7.4.13 MC_MoveAbsolute, Besonderheiten der Feldbusantriebe

Verwendete Parameter Der Funktionsbaustein ermittelt die neue Zielposition durch Übernahme der Sollposition und fährt diese an.

Sonderfunktion: Die FB's MC_MoveAbsolute, MC_MoveAdditive und MC_MoveRelative akzeptieren eine Geschwindigkeitsvorgabe von $V_{soll} = 0$.

Mit Aufruf dieser FB's ($V_{soll} = 0$) wird die Achse angehalten. Die Positionierung wird lediglich unterbrochen. Die Zielposition in der Achse bleibt erhalten.

Folgt anschließend der Aufruf eines MC_MoveAdditive ($V_{soll} \neq 0$) wird der Restweg verfahren. Erst mit Aufruf eines anderen Motion-FB's wird die Zielposition überschrieben.

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

Programm:

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F RELATED TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.4.14 MC_MoveAdditive

Kurzbeschreibung Der Funktionsbaustein addiert die "Distanz" zur "Zielposition" und fährt diese an.



Die Anwendung des Bausteins setzt die vorherige Leistungszuschaltung über eine Instanz des [MC_Power](#), [Seite 365](#), (bei realen Achsen) voraus.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Motion_FBs

Tab. 7-19: Bibliothekszuordnung

Schnittstellenbeschreibung

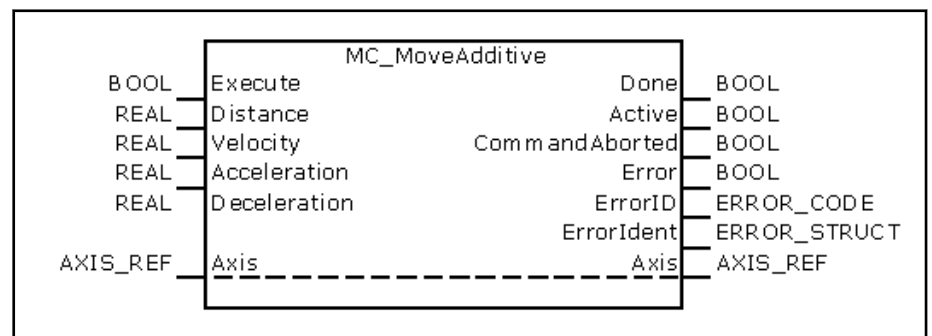


Abb. 7-14: FB MC_MoveAdditive

	Name	Typ	Kommentar ⁴⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Distance	REAL	Relativentfernung für die Bewegung (+/-).
	Velocity	REAL	Maximalwert der Geschwindigkeit (stets +) (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).
	Deceleration	REAL	Verzögerung (stets +).
VAR_OUTPUT	Done	BOOL	gewünschte Position erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 7-20: Schnittstelle des FBs MC_MoveAdditive

Implementationsbeispiel: Das nachfolgende Bild zeigt zwei Beispiele der Kombination von zwei FB-Instanzen während die Achse im "Discrete Motion State" ist:

⁴⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

RMB_PLCopenFieldBus.library



Hier sind zwei unterschiedliche FB-Typen in Benutzung:

- Instance1: MC_MoveAbsolute
- Instance2: MC_MoveAdditive.

1. Der linke Teil des Zeitdiagramms zeigt den Fall, dass der zweite FB nach dem ersten gestartet wird. Der erste erreicht die gewünschte Entfernung 6000 (bei Geschwindigkeit 0). Danach bewirkt der Ausgang "Done", dass der zweite FB die Distanz bis 10000 überwindet.
2. Der rechte Teil des Zeitdiagramms zeigt den Fall, bei dem der zweite FB bereits gestartet wird, bevor der erste seine Arbeit abschließen konnte. In diesem Fall wird die erste Bewegung bei konstanter Geschwindigkeit durch das Signal "Test" gestoppt und abgebrochen. Der zweite FB addiert seine Distanz von 4000 zur ursprünglich befohlenen Position 6000 und bewegt die Achse auf die resultierende Position 10000 mit der am zweiten FB vorgegebenen Geschwindigkeit.

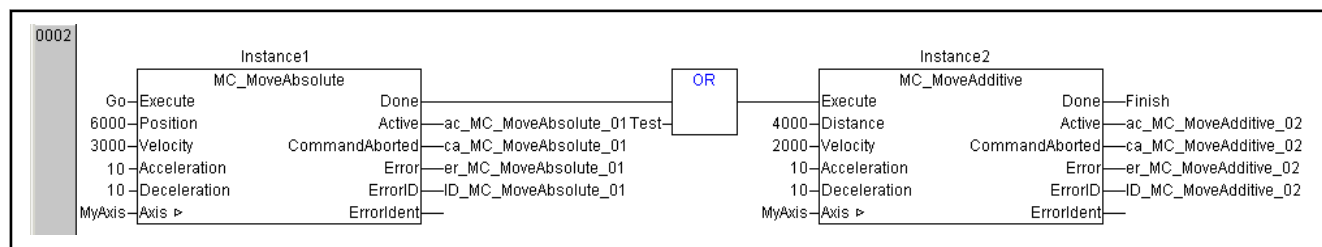


Abb. 7-15: Instanzen von MC_MoveAbsolute und MC_MoveAdditive

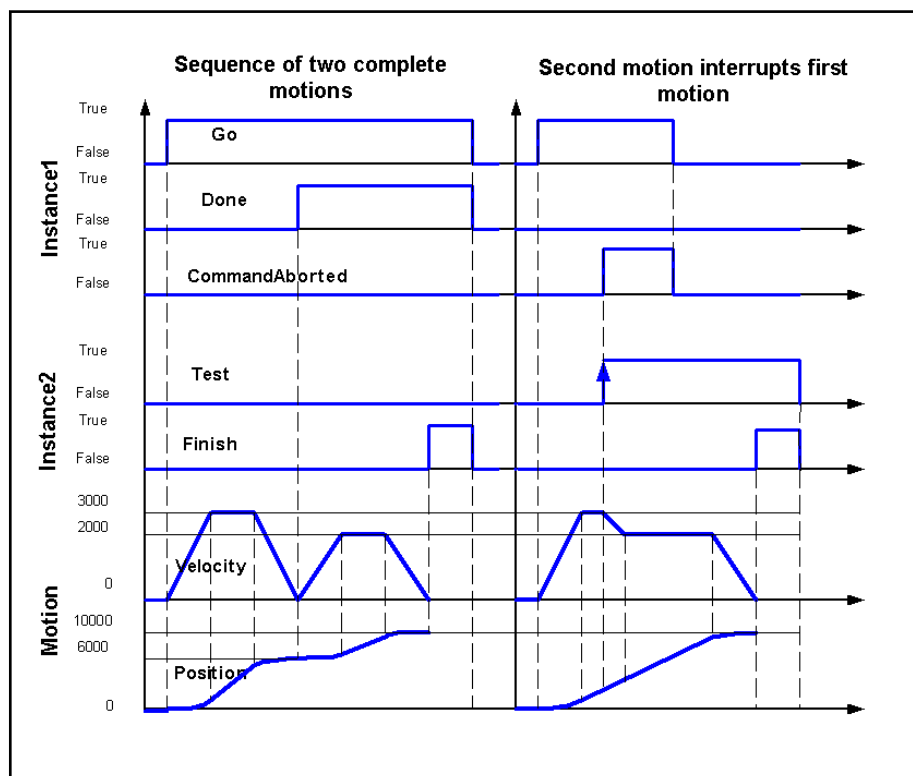


Abb. 7-16: Zeitdiagramm: Instanzen von MC_MoveAbsolute und MC_MoveAdditive

7.4.15 MC_MoveAdditive, Besonderheiten der Feldbusantriebe

Verwendete Parameter

Der Funktionsbaustein ermittelt die neue Zielposition durch Addition der Distanz zur vorherigen Zielposition und fährt diese an.



Die Distanz wird relativ zur Istposition angegeben.

Sonderfunktion: Die FB's MC_MoveAbsolute, MC_MoveAdditive und MC_MoveRelative akzeptieren eine Geschwindigkeitsvorgabe von Vsoll = 0.

Mit Aufruf dieser FB's (Vsoll = 0) wird die Achse angehalten. Die Positionierung wird lediglich unterbrochen. Die Zielposition in der Achse bleibt erhalten. Folgt anschließend der Aufruf eines MC_MoveAdditive (Vsoll ≠ 0) wird der Restweg verfahren. Erst mit Aufruf eines anderen Motion-FB's wird die Zielposition überschrieben.

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

Programm:

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F RELATED TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.4.16 MC_MoveRelative

Kurzbeschreibung Der Funktionsbaustein ermittelt die neue "Zielposition" durch Addition der "Distanz" und der "Istposition" und fährt diese an.



Die Anwendung des Bausteins setzt die vorherige Leistungszuschaltung über eine Instanz des [MC_Power](#), [Seite 365](#), (pro realer Achse) voraus.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Motion_FBs

Tab. 7-21: Bibliothekszuordnung

Schnittstellenbeschreibung

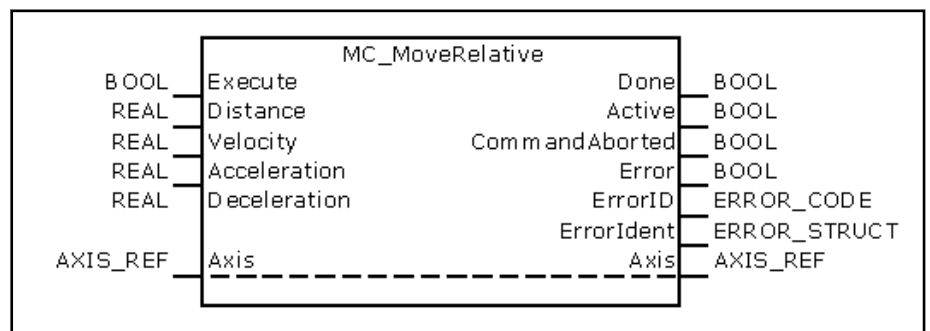


Abb. 7-17: FB MC_MoveRelative

	Name	Typ	Kommentar ⁵⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Distance	REAL	Relativentfernung für die Bewegung (+/-).

⁵⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

RMB_PLCopenFieldBus.library

	Name	Typ	Kommentar ⁵⁾
	Velocity	REAL	Maximalwert der Geschwindigkeit (stets +) (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).
	Deceleration	REAL	Verzögerung (stets +).
VAR_OUTPUT	Done	BOOL	Gewünschte Position erreicht
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 7-22: Schnittstelle von FB MC_MoveRelative

Signal-Zeit-Diagramm

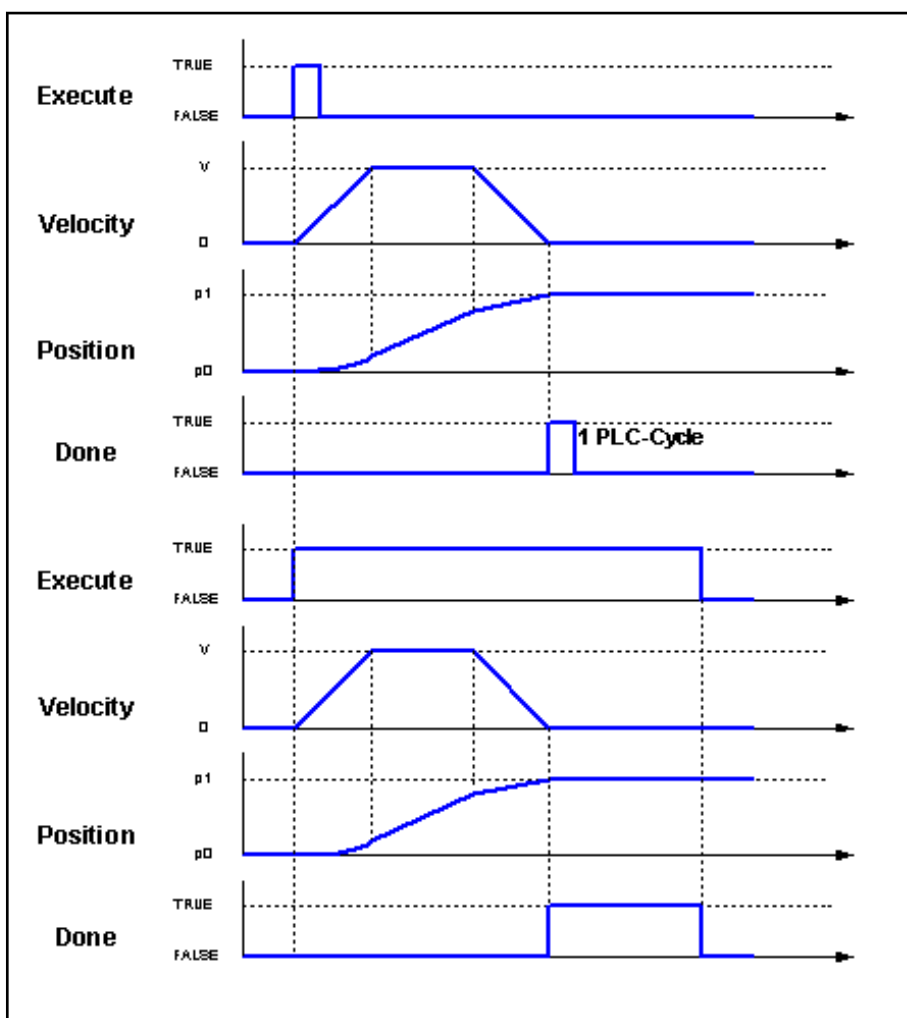


Abb. 7-18: Zeitdiagramme zu MC_MoveRelative

⁵⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

Implementationbeispiel

Das folgende Beispiel zeigt die Kombination von zwei FB-Instanzen MC_MoveRelative.

1. Der linke Teil des Zeitdiagramms zeigt den Fall, in dem Instance2 nach Abarbeitung von Instance1 gerufen wird. Nachdem Instance1 die gewünschte Distanz 6000 (Geschwindigkeit 0) erreicht hat, verursacht sein Ausgang "Done" über Instance2 eine weitere Bewegung um 4000 auf die Gesamtentfernung 10000.
2. Der rechte Teil zeigt, wie Instance2 bereits während der Arbeit von Instance1 durch das Signal "Test" aktiviert wird. In diesem Fall wird die erste Bewegung während der konstanten Geschwindigkeit von Instance1 abgebrochen. Instance1 addiert zur aktuellen Position (3250) noch seine Entfernung von 4000. Die Achse bewegt sich zur resultierenden Position von 7250.

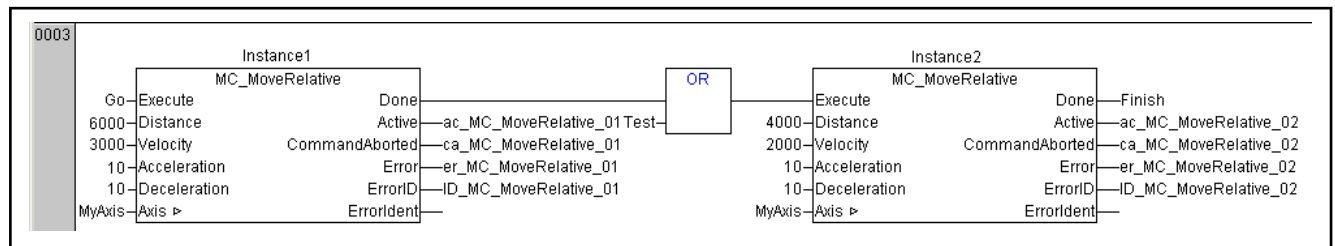


Abb. 7-19: Zwei Instanzen von MC_MoveRelative

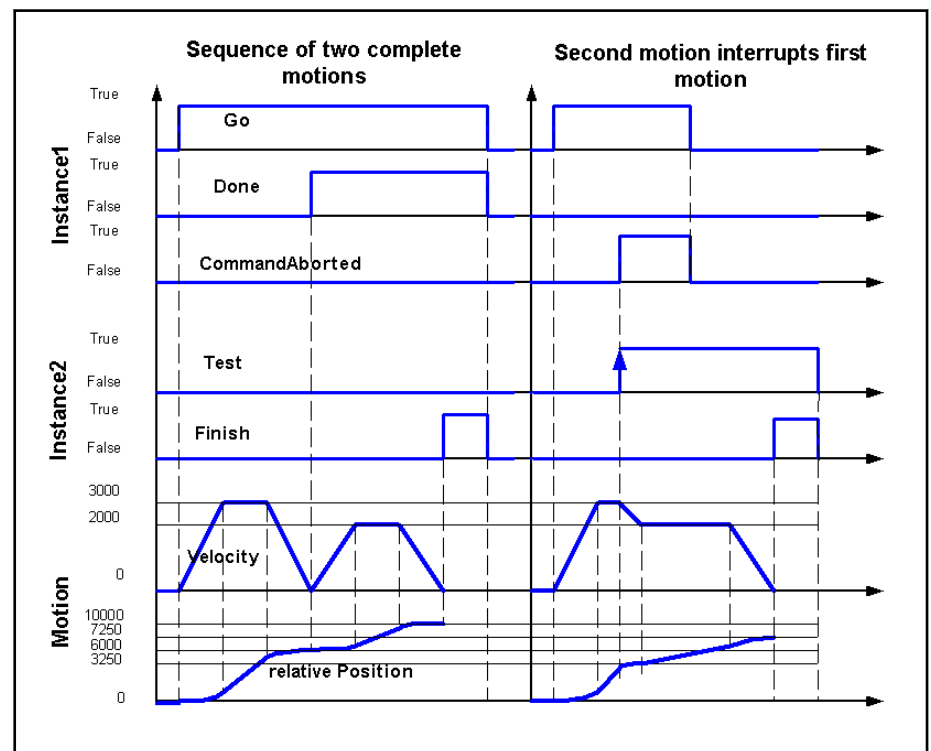


Abb. 7-20: Zeitdiagramm: zwei Instanzen von MC_MoveRelative

7.4.17 MC_MoveRelative, Besonderheiten der Feldbusantriebe

Verwendete Parameter

Der Funktionsbaustein ermittelt die neue "Zielposition" durch Addition der "Distanz" und der "Istposition" und fährt diese an.



Die Distanz wird relativ zur Istposition angegeben.

RMB_PLCopenFieldBus.library

Sonderfunktion: Die FB's MC_MoveAbsolute, MC_MoveAdditive und MC_MoveRelative akzeptieren eine Geschwindigkeitsvorgabe von Vsoll = 0.

Mit Aufruf dieser FB's (Vsoll = 0) wird die Achse angehalten. Die Positionierung wird lediglich unterbrochen. Die Zielposition in der Achse bleibt erhalten. Folgt anschließend der Aufruf eines MC_MoveAdditive (Vsoll ≠ 0) wird der Restweg verfahren. Erst mit Aufruf eines anderen Motion-FB's wird die Zielposition überschrieben.

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

Programm:

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.4.18 MC_MoveVelocity

Kurzbeschreibung Der Funktionsbaustein steuert eine "endlose" Bewegung mit der jeweils festgelegten Geschwindigkeit. Um die Bewegung zu beenden, muss die FB-Instanz durch eine andere FB-Instanz, die ein neues Kommando ausgibt, unterbrochen werden.

Die Anwendung des Bausteins setzt die vorherige Leistungszuschaltung über eine Instanz des MC_Power, Seite 365, (pro realer Achse) voraus.

Das Signal "InVelocity" wird zurückgesetzt, wenn die FB-Instanz durch einen andere FB-Instanz unterbrochen wird (CommandAborted) oder bei fallender Flanke seines "Execute"-Eingangs.

Bei Abschalten des "Execute" läuft der Antrieb mit der gleichen Geschwindigkeit weiter.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Motion_FBs

Tab. 7-23: Bibliothekszuordnung

Schnittstellenbeschreibung

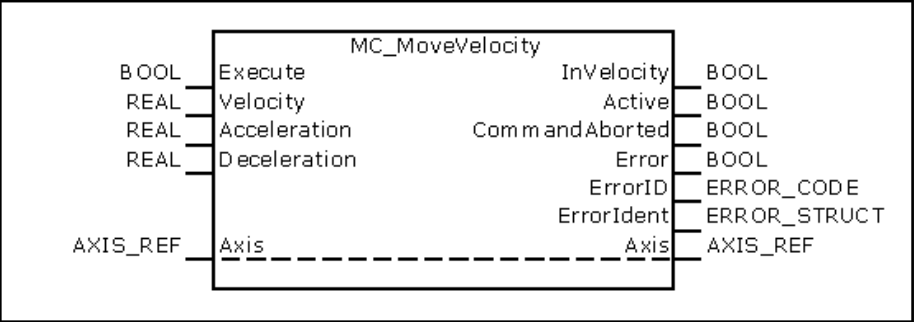


Abb. 7-21: FB MC_MoveVelocity

	Name	Typ	Kommentar ⁶⁾
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	startet die Bewegung bei steigenden Flanke
	Velocity	REAL	Maximalwert der Geschwindigkeit (muss nicht unbedingt erreicht werden).
	Acceleration	REAL	Beschleunigung (stets +).
	Deceleration	REAL	Abbremsen (stets +).
VAR_OUTPUT	InVelocity	BOOL	Geschwindigkeit erreicht (erstmalig erreicht)
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	Command-Aborted	BOOL	Kommandoabbruch durch nachfolgendes Kommando
	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

Tab. 7-24: Schnittstelle von FB MC_MoveVelocity

Signal-Zeit-Diagramm

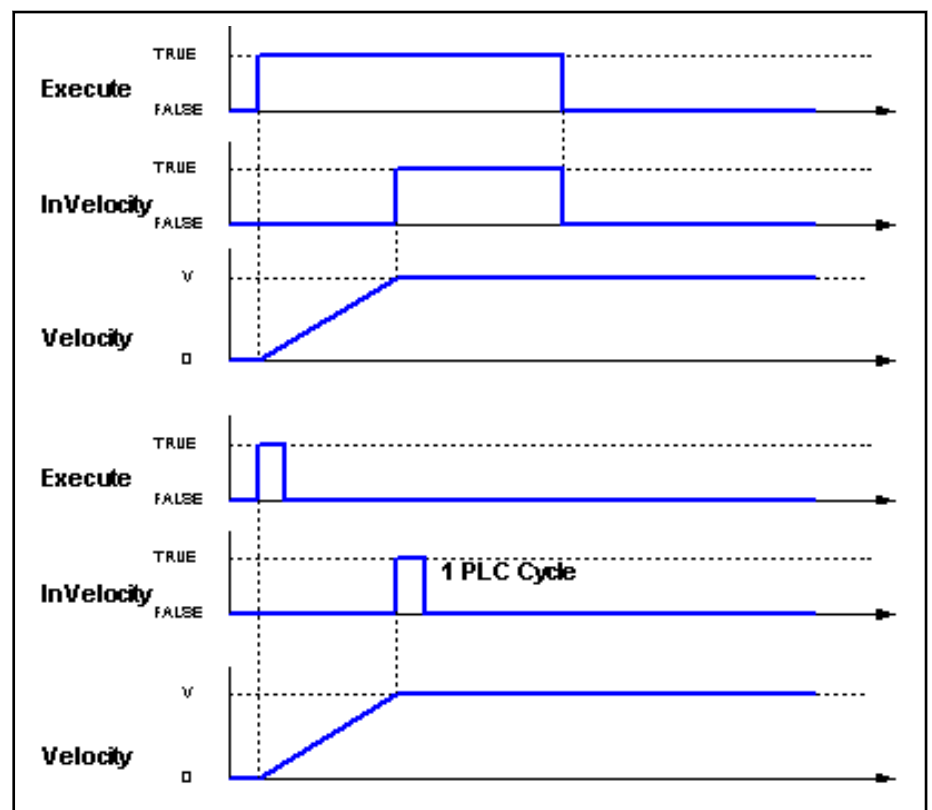


Abb. 7-22: Zeitdiagramm zu MC_MoveVelocity (ohne Kommandoabbruch)

⁶⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

RMB_PLcOpenFieldBus.library

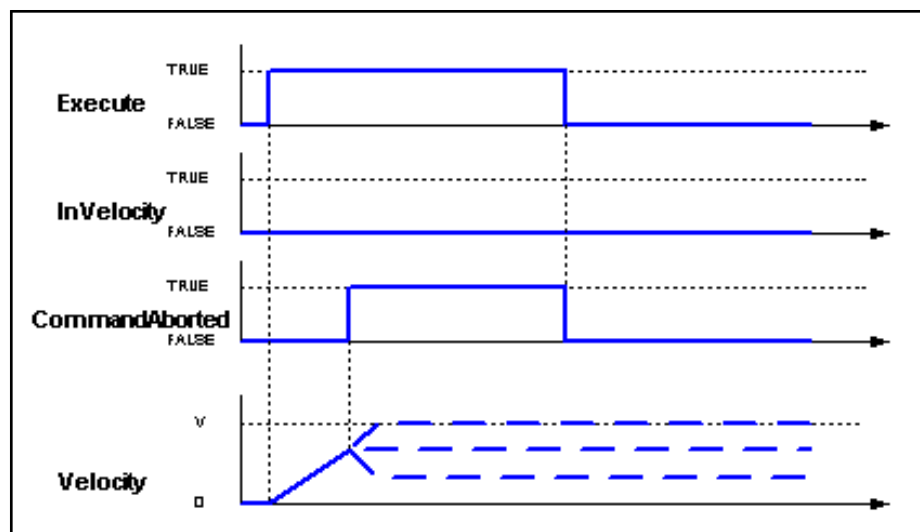


Abb. 7-23: Zeitdiagramm zu MC_MoveVelocity (mit Kommandoabbruch)

Implementationsbeispiel:

Das nachstehende Bild zeigt zwei Beispiele der Kombination von zwei Instanzen der FBs MC_MoveVelocity:

1. Der linke Teil des Zeitdiagramms zeigt den Fall, bei dem die zweite Instanz gerufen wird, nachdem die Arbeit der ersten abgeschlossen ist. Nachdem der erste die geforderte Geschwindigkeit 3000 erreicht hat, bewirken der Ausgang "Instance1.InVelocity" **UND** das Signal "Next", dass der zweite FB die Bewegung bei Geschwindigkeit 2000 fortsetzt.
2. Der rechte Teil des Zeitdiagramms zeigt den Fall, bei dem der erste FB noch nicht die gewünschte Geschwindigkeit erreicht hat, wenn der zweite gestartet wird. Nachfolgend ist gezeigt: Die erste Bewegung wird erneut gestartet durch "Go" am Eingang "Instance1.Execute". Während der erste FB noch beschleunigt, um die Geschwindigkeit 3000 zu erreichen, wird er durch das Signal "Test" unterbrochen, dass den zweiten FB startet. Jetzt läuft der zweite FB und vermindert die Geschwindigkeit auf 2000.

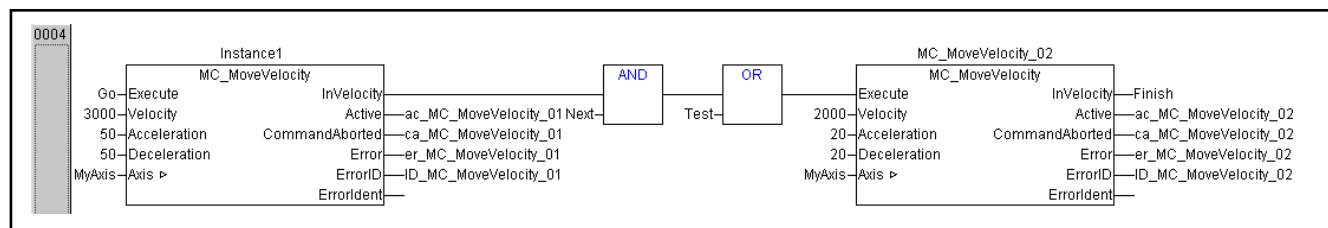


Abb. 7-24: Zwei Instanzen von MC_MoveVelocity

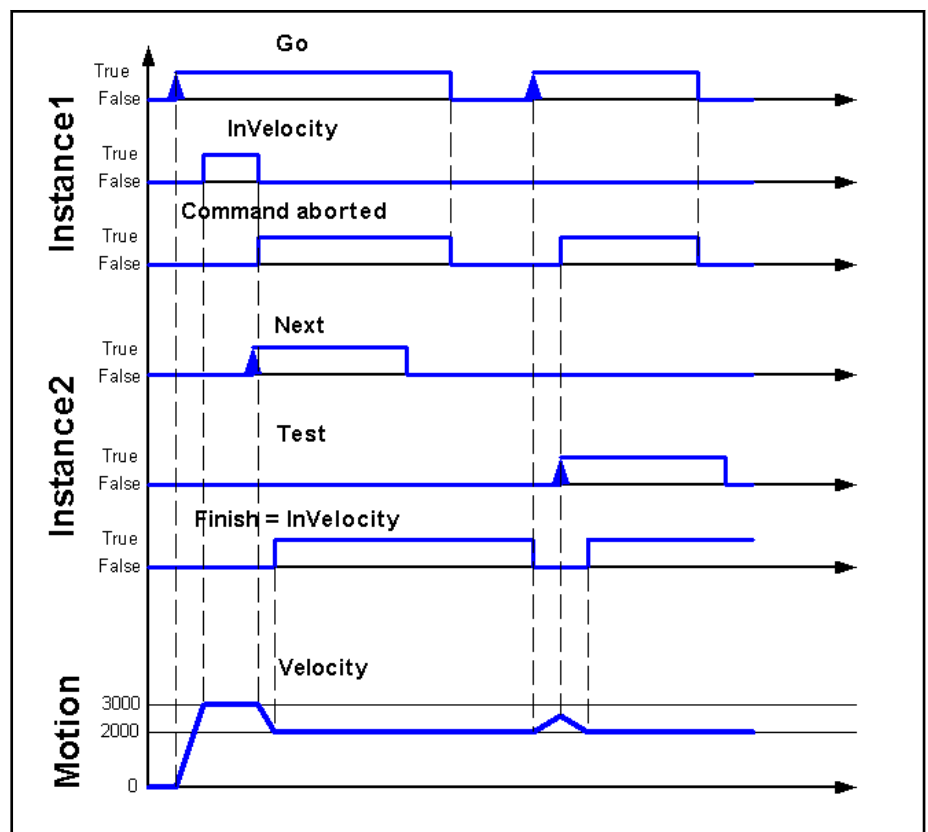


Abb. 7-25: Zeitdiagramm: zwei Instanzen von MC_MoveVelocity

7.4.19 MC_MoveVelocity, Besonderheiten der Feldbusantriebe

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

Programm:

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5 Funktionsbausteine zum Parameterzugriff (Parameter_FBs)

7.5.1 Parameter_FBs, Allgemeines

Die Bibliothek RMB_PLCOpenFieldBus.library für IndraDrives nutzt für die Funktionsbausteine MB_Read/WriteXXX den Parameterkanal der zyklischen Kommunikation (im zyklischen I/O-Austausch).

Die Parametrierung des Antriebes über den Feldbus hat im zyklischen Kanal des IndraDrive-Antriebs ein konfigurierbarer Parameterkanal implementiert, dessen Größe auf 4...16 Byte eingestellt werden kann. Der Parameterkanal besteht aus einem Steuer- bzw. Statuswort und bis maximal 7 Worten für Daten.

Der Parameterkanal steht immer am Anfang des zyklischen Datenkanals. Die Länge des Parameterkanals wird im Master, also der XLC/MLC konfiguriert

RMB_PLCopenFieldBus.library

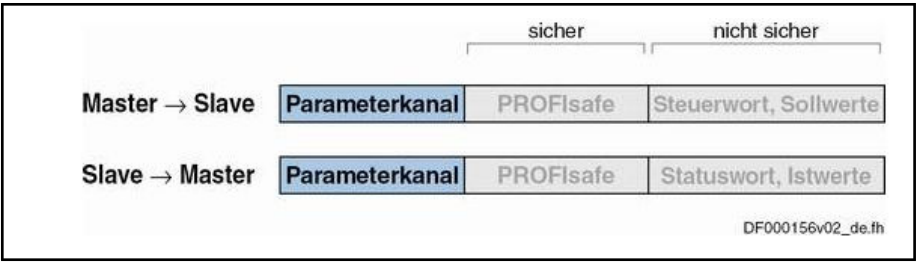


Abb. 7-26: Parameterkanal im zyklischen Datenkanal

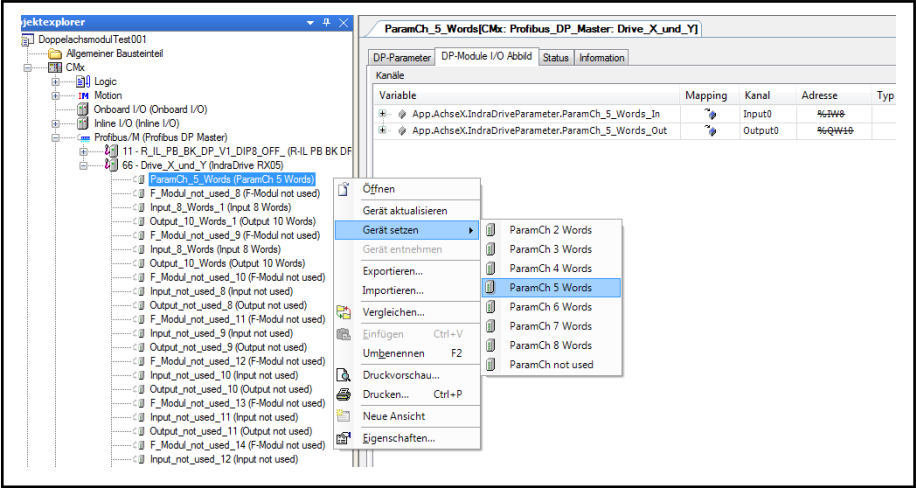


Abb. 7-27: Konfiguration des Parameterkanals

 Die nachstehend genannten Bausteine dienen zur Parameterübertragung zwischen Steuerung und Antrieb.
Sie sind nur für IndraDrive C/M ab MPx2VRS, IndraDrive Cs ab MPx16VRS gekoppelt über Profibus, verwendbar.

	Funktion
MB_ReadListParameter, Seite 387	gibt den Wert eines anwenderspezifischen Parameters zurück. Dem Baustein wird ein Pointer übergeben, der einen in Anwenderverantwortung festgelegten Speicher- raum von maximal 64KByte adressiert. Ferner wird die gewünschte Nutzdatenlänge übergeben (maximal lesbare Anzahl an Bytes). Als Rückgabewert erhält man ferner die Anzahl der tatsächlich gelesenen Bytes.
MB_ReadListSegParameterPtr, Seite 390	gibt die Werte eines Abschnittes eines anwenderspezifischen Parameters zurück. Dem Baustein wird ein Pointer übergeben, der einen in Anwenderverantwortung fest- gelegten 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.
MB_ReadMaxRealValue, Seite 393	Der Funktionsbaustein gibt den zulässigen Maximalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als REAL-Zahl darge- stellt.
MB_ReadMaxValue, Seite 395	Der Funktionsbaustein gibt den zulässigen Maximalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.
MB_ReadMinRealValue, Seite 396	Der Funktionsbaustein gibt den zulässigen Minimalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als REAL-Zahl darge- stellt.

	Funktion
MB_ReadMinValue, Seite 398	Der Funktionsbaustein gibt den zulässigen Minimalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.
MB_ReadRealParameter, Seite 401	gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als REAL-Zahl dargestellt.
MB_ReadParameter, Seite 399	gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.
MB_ReadSercosAttribute, Seite 403	erlaubt das Lesen der sercos Attribute eines Parameters. Der Rückgabewert wird als DWORD dargestellt.
MB_ReadStringParameter, Seite 404	gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als Zeichenkette (STRING) entsprechend seiner tatsächlichen Länge mit maximal 255 Zeichen dargestellt.
MB_WriteListParameter, Seite 406	schreibt Speicherinhalt definierter Länge (NoOfByte), der ab der Anfangsadresse (ValueAdr) zur Verfügung gestellt wurde, in einen anwenderspezifizierten Parameter.
MB_WriteListSegParameterPtr, Seite 408	schreibt Speicherinhalt definierter Länge (NoOfByte), der ab der Anfangsadresse (ValueAdr) zur Verfügung gestellt wurde, in einen vorgegebenen Abschnitt eines anwenderspezifizierten Parameter.
MB_WriteParameter, Seite 411	schreibt einen DINT-Wert in einen durch den Anwender vorgegebenen Parameter.
MB_WriteRealParameter, Seite 412	schreibt einen REAL-Wert in einen anwenderspezifizierten Parameter.
MB_WriteStringParameter, Seite 414	schreibt eine Zeichenkette mit max. 255 Zeichen in einen anwenderspezifizierten Parameter.
MC_ReadActualPosition, Seite 415	liest die aktuelle Position des Antriebes.
MC_ReadAxisError, Seite 417	ruft den Fehlerstatus der adressierten Achse ab. Je Achse ist dazu eine Instanz des Funktionsbausteines zu verwenden.
MC_ReadStatus, Seite 418	liest den aktuellen Status der Achse und gibt diesen dekodiert aus.

Tab. 7-25: Funktionsbausteine des Ordners "Parameter_FBs"

7.5.2 Zugriff auf Daten eines Antriebs (IndraDrive)

Aufbau und Zugriff auf Parameter aus dem SPS-Programm

Die Daten eines Antriebes sind über Parameterzugriff lesend oder teilweise auch schreibend erreichbar.

Über den Feldbus können Zugriffe auf die S- und P-Parametern vorgenommen werden.

Alle MotionControl-Parameter werden nach dem gleichen Bildungsgesetz aufgebaut:

S, P als Parametertyp-Kennzeichner gefolgt von der Gruppennummer und der vierstelligen Parameternummer.

Dem Anwender stehen in dieser Bibliothek für P- und S-Parameter Vereinbarungsdateien für Globale Konstanten zur Verfügung.

Parameter-Konstanten

- [GVL_S_Param_ID, Feldbus, Seite 359,](#)
- [GVL_P_Param_ID, Feldbus, Seite 360,](#)

Nummer des Parameters:

S-Parameter

(*Constants to address parameters via parameter ident*)

RMB_PLCopenFieldBus.library

```

VAR GLOBAL CONSTANT
  FP_S_0_0000 : MB_IDN := 0; (* Dummy parameter *)
  FP_S_0_0001 : MB_IDN := 1; (* NC cycle time TNcyc *)
  FP_S_0_0002 : MB_IDN := 2; (* SERCOS cycle time TScyc *)
  FP_S_0_0003 : MB_IDN := 3; (* Minimum AT transmit starting...
  (*.....*)
END_VAR

```

P-Parameter

(*Constants to address parameters via parameter ident*)

```

VAR GLOBAL CONSTANT
  FP_P_0_0001 : MB_IDN := 32769; (* Switching frequency...
  FP_P_0_0004 : MB_IDN := 32772; (* Speed loop smoothing...
  FP_P_0_0007 : MB_IDN := 32775; (* Display text of diagnostic...
  FP_P_0_0008 : MB_IDN := 32776; (* Activation EStop function *)
  (*.....*)
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

Tab. 7-26: Parameter-Offsets

7.5.3 Besonderheiten bei Doppel- bzw. Mehrachsmodulen

Doppel- bzw. Mehrachsmodul und CDD-Verbundachsen

Der Parameterkanal ist bei Doppelachs- bzw. Mehrachsmodulen und CDD-Verbundachsen nur für die erste Achse vorhanden.

Von der ersten Achse ausgehend, können über einen Subindex die Folgeachsen ausgewählt werden.

In der Datenstruktur "AXIS_REF" der RMB_PLCopenFieldBus-Bibliothek hat der Anwender die Möglichkeit diesen Subindex einzustellen:

Deklaration des Mehrachsmoduls:

```
<AxisName>: <LibraryNamespace>.AXIS_REF:= ...;
```

```
Drive_X: FieldBusDrives.AXIS_REF:= ...;
```

Festlegen des Subindex der Achse eines Mehrachsmoduls bzw. der Achse im CDD-Verbund:

```
Drive_X.IndraDriveParameter.Subindex := Achs_Index;
```

Zugriff auf unterschiedliche Achsen eines Mehrachsmoduls

```

PROGRAM prParameterLesenST
VAR
  Drive_X:      FieldBusDrives.AXIS_REF:=
                (AxisNo:= OBJECTS.FieldBusAxis 11,
                 CntrlNo:= CONTROLS.LOCAL_CNTRL);
  ReadParameter: FieldBusDrives.MB_ReadRealParameter;
  bReadEnable:   BOOL; // Anforderung Lesen des Parameters
  bSlaveAxis:    BOOL; // Lesen/Schreiben von und zur Folgeachse
  rValue_X:      REAL; // gelesener Wert der X-Achse
  rValue_Y:      REAL; // gelesener Wert der Y-Achse, die Folgeachse der X-Achse ist
  bReadDone:     BOOL; // Baustein meldet erfolgreiches Lesen
  rTmpValue:     REAL; // gelesener Wert
END_VAR

//bSlaveAxis = TRUE -> Lesen vom 2. Achsmodul eines Mehrachsmoduls
IF NOT bSlaveAxis THEN
  // Subindex des 1. Achsmoduls eines Mehrachsmoduls

```

```

    Drive_X.IndraDriveParameter.Subindex := 0;
ELSE
    // Subindex des 2. Achsmoduls eines Mehrachsmoduls
    Drive_X.IndraDriveParameter.Subindex := 1;
END_IF
ReadParameter(
    Enable:= bReadEnable,
    ParameterNumber:= FieldBusDrives.FP_S_0_0051,
    // Bei Mehrachsmodulen immer Referenz (AXIS_REF) der ersten Achse
    Axis:= Drive_X,
    Done=> bReadDone,
    ...
    Value=> rTmpValue);
// Auswertung des Paramater-Inhaltes in Abhängigkeit vom Subindex
IF bReadDone
    // erstes Achsmodul des Mehrachsmoduls
    AND Drive_X.IndraDriveParameter.Subindex = 0
    THEN
        rValue_X := rTmpValue;
    ELSIF bReadDone
        // zweites Achsmodul des Mehrachsmoduls
        AND Drive_X.IndraDriveParameter.Subindex = 1
        THEN
            rValue_Y := rTmpValue;
        END_IF
    END_IF

```

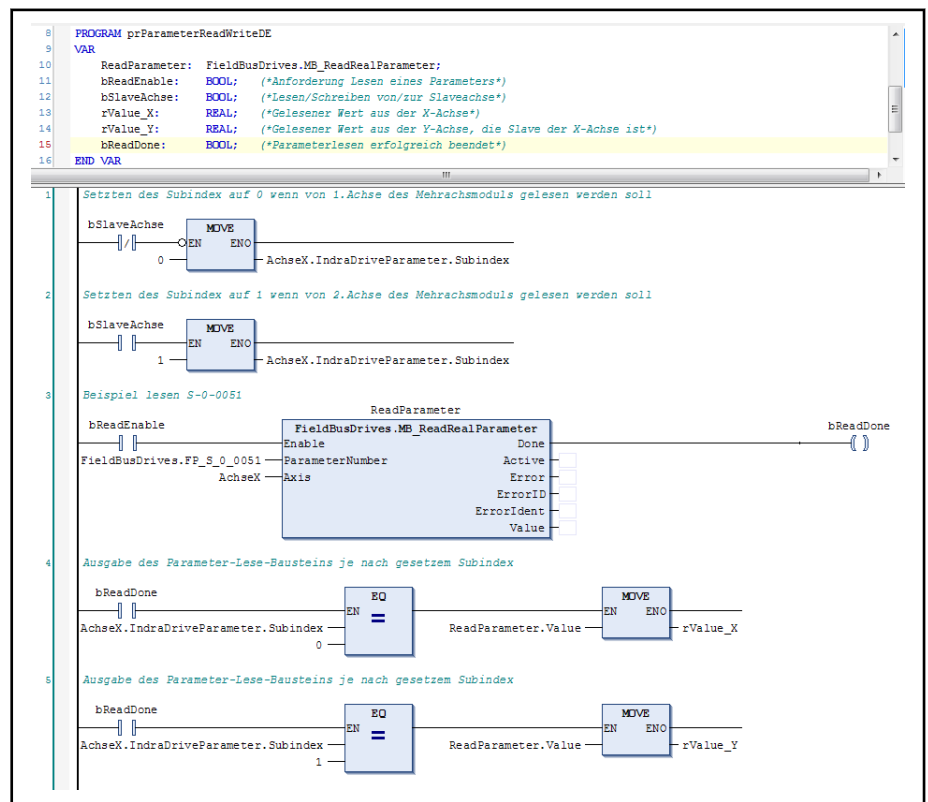


Abb. 7-28: Programmbeispiel Parameterlesen Doppel- bzw. Mehrachsmodul

Für alle PLCopen-Parameterbausteine muss dem AXIS_REF der ersten Achse eines Mehrachsmoduls dieser Subindex mitgegeben werden (default = 0).

Für den Doppelachser bzw. CDD-Verbund bedeutet dies, dass jeweils nur auf eine Achse zugegriffen werden kann.

Sollen Parameter von mehreren Achsen gelesen oder geschrieben werden, muss dieses nacheinander geschehen.

7.5.4 MB_ReadListParameter

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Dem Baustein wird ein Pointer übergeben, der einen in Anwen-

RMB_PLCopenFieldBus.library

derverantwortung 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.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-27: Bibliothekszuordnung

Schnittstellenbeschreibung

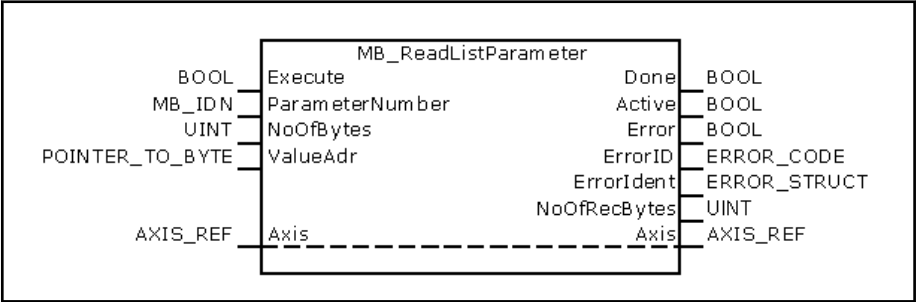


Abb. 7-29: FB MB_ReadListParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Lesen des Parameters, getriggert durch die positive Flanke von "Execute"
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Parameter-Konstanten , Seite 385)
	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. 7-28: Schnittstelle von FB MB_ReadListParameter

Signal-Zeit-Diagramm

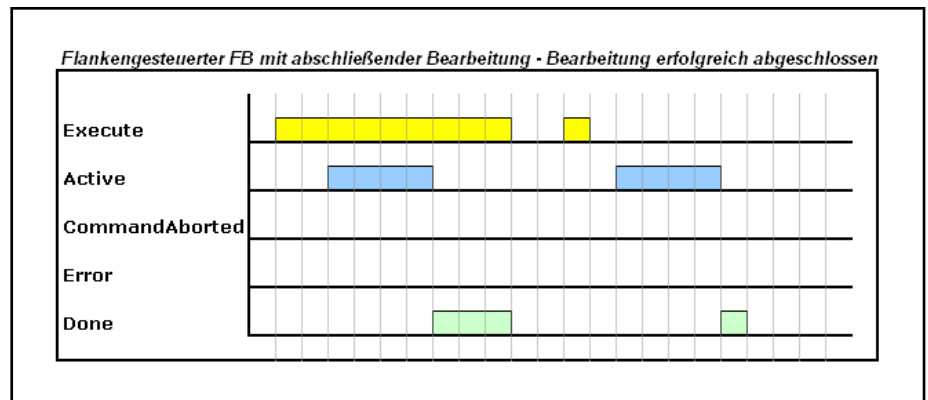


Abb. 7-30: Standard-Signal-Zeit-Diagramm

Programmbeispiel
MB_ReadListParameter

Das Programmbeispiel zeigt das Lesen des Listenparameters (P-0-0072) mit 1024 Werten (DINT) und das Abspeichern in das ReadListArray.

Programmbeispiel, Deklarationsteil

```

PROGRAM ReadListParameter
VAR IN_OUT
    RA1:          AXIS_REF;          (* Fieldbus axis RA1 *)
END_VAR

VAR
    ReadListArray:    ARRAY [0..1023] OF DINT; (* Data array for the received bytes *)

    FB_ReadListParameter: MB_ReadListParameter;
    bExecute_RLP:        BOOL;          (* Start FB *)
    diParameterNo_RLP:   MB_IDN := FP_P_0_0072; (* CAM parameter P-0-0072 *)
    uiNoOfBytes_RLP:     UINT := 4096;  (* DINT := 4 Bytes *)
    bDone_RLP:          BOOL;          (* TRUE := FB ready *)
    bActive_RLP:        BOOL;          (* FB works *)
    bError_RLP:         BOOL;          (* Error occurred *)
    enErrorID_RLP:      ERROR_CODE;    (* Error class *)
    stErrorIdent_RLP:   ERROR_STRUCT;  (* Error definition *)
    uiNoOfRecBytes_RLP:  UINT;         (* Number of received bytes *)
END_VAR

```

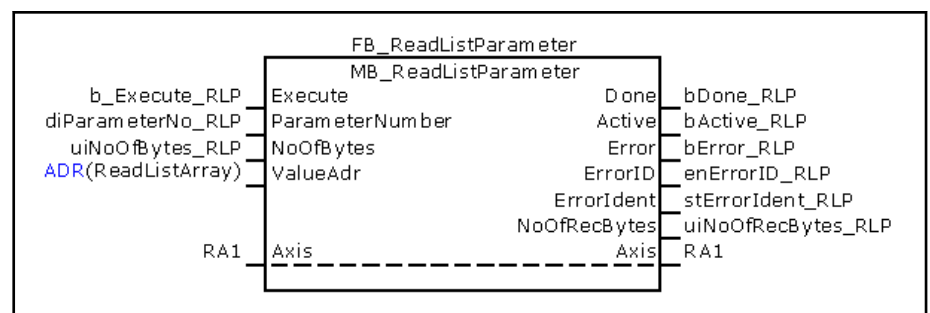
Programmbeispiel im Funktionsplan

Abb. 7-31: FUP Programmbeispiel ReadListParameter

Programmbeispiel im Strukturierten Text

```

FB_ReadListParameter(
    _Execute:= bExecute_RLP,          (* Start FB*)
    ParameterNumber:= diParameterNo_RLP, (* CAM parameter P-0-0072 *)
    NoOfBytes:= uiNoOfBytes_RLP,      (* DINT := 4 Bytes *)

```

RMB_PLCopenFieldBus.library

```

ValueAdr:=      ADR(ReadListArray),    (* Address of ReadListArray *)
Axis:=          RA1,                   (* Fieldbus axis *)
Done=>          bDone_RLP,              (* TRUE := FB ready *)
Active=>        bActive_RLP,           (* FB works *)
Error=>         bError_RLP,            (* Error occurred *)
ErrorId=>       enErrorID_RLP,         (* Error class *)
ErrorIdent=>    stErrorIdent_RLP:      (* Error definition *)
NoOfRecBytes=>  uiNoOfRecBytes_RLP;    (* Number of received bytes *)

```

7.5.5 MB_ReadListParameter, Besonderheiten der Feldbusantriebe



Der Baustein MB_ReadListParameter funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```

Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;

```

oder

F_RELATED_TABLE

```

Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;

```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.6 MB_ReadListSegParameterPtr

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines Bereiches eines anwenderspezifischen Parameters zurück.

Dem Baustein wird ein Pointer übergeben, der einen in **Anwenderverantwortung** festgelegten Speicherraum von maximal 64KByte adressiert (Speicher in der SPS).

Ferner werden ein Offset (Ort ab dem die Daten gelesen werden sollen in Bezug auf den Anfang im Parameter) und die gewünschte Nutzdatenlänge (NoOfBytes) übergeben (NoOfBytes ≤ maximal lesbare Anzahl in Bytes des Parameters - Offset).

Als Rückgabewert erhält man die Anzahl der tatsächlich gelesenen Bytes.



1. Der Offset ist so zu wählen, dass das erste zu lesende Datenelement vollständig gelesen werden kann.
2. Der Datentyp der Elemente des zu lesenden Parameters und der Datentyp der Elemente des ARRAYS in der SPS müssen übereinstimmen.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-29: Bibliothekszuordnung

Schnittstellenbeschreibung

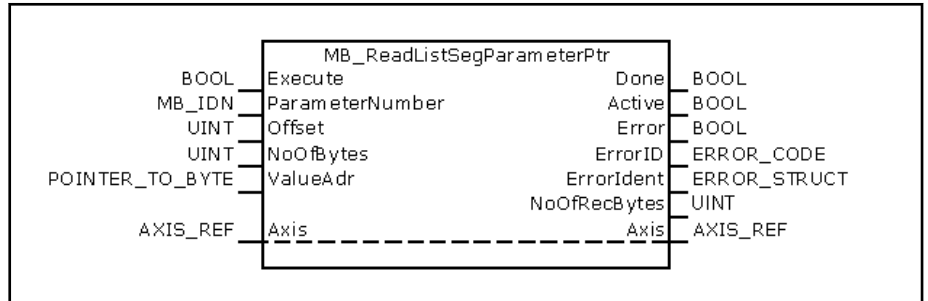


Abb. 7-32: FB MB_ReadListSegParameterPtr

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Lesen des Parameters, getriggert durch die positive Flanke von "Execute"
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Zugriff auf Daten eines Antriebs (IndraDrive) , Seite 385)
	Offset	UINT	Offset in Bytes, bezogen auf Anfang des Parameters, ab dem gelesen werden soll
	NoOfBytes	UINT	maximal zu lesende Datenmenge in Byte
	ValueAdr	POINTER_TO_BYTE	Zeigt auf Anfangsadresse des Speicherraums in der SPS, 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. 7-30: Schnittstelle von FB MB_ReadListSegParameterPtr

RMB_PLCOpenFieldBus.library

Signal-Zeit-Diagramm

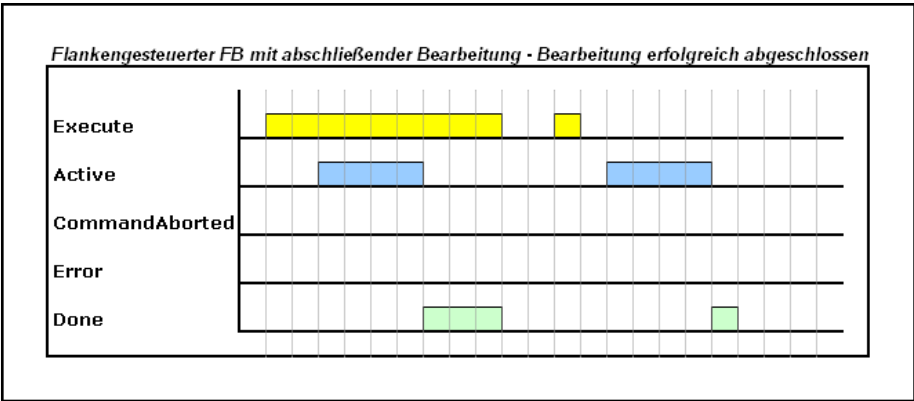


Abb. 7-33: Standard-Signal-Zeit-Diagramm

Programmbeispiel MB_ReadListSegParameterPtr

Das Programmbeispiel zeigt das teilweise Lesen des Listenparameters (P-0-0072) ab Byte 128 mit der Länge von 512 Byte und das Abspeichern in das ReadListArray.

Programmbeispiel, Deklarationsteil

```
PROGRAM ReadListSegParameterPtr
VAR IN_OUT
    RA1:          AXIS_REF;          // Fieldbus axis RA1
END_VAR
VAR
    ReadListArray:  ARRAY [0..1023] OF DINT; // Data array for the received bytes

    FB_ReadListSegParameterPtr: MB_ReadListSegParameterPtr;
    bExecute_RLSPP:  BOOL;           // Start FB
    diParameterNo_RLSPP: MB_IDN := FP_P_0_0072; // CAM parameter P-0-0072
    uiOffset_RLSPP:  UINT := 128;    // beginning with byte 128
    uiNoOfBytes_RLSPP:  UINT := 512; // 512 bytes (128 DINT)
    bDone_RLSPP:      BOOL;          // TRUE := FB ready
    bActive_RLSPP:    BOOL;          //
    bError_RLSPP:     BOOL;          // Error occurred
    enErrorID_RLSPP:  ERROR_CODE;    // Error class
    stErrorIdent_RLSPP: ERROR_STRUCT; // Error definition
    uiNoOfRecBytes_RLSPP:  UINT;     // Number of received bytes
END_VAR
```

Programmbeispiel im Funktionsplan

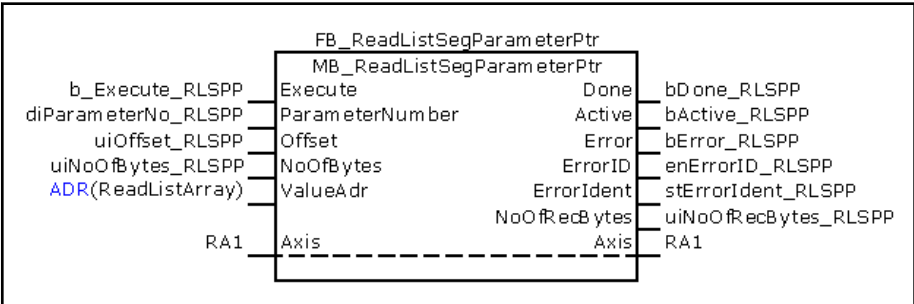


Abb. 7-34: FUP Programmbeispiel MB_ReadListSegParameterPtr

Programmbeispiel im Strukturierten Text

```
FB_ReadListSegParameterPtr(
    Execute:= bExecute_RLSPP, // Start FB
    ParameterNumber:= diParameterNo_RLSPP, // Parameter Cam data
    Offset:= uiOffset_RLSPP, // beginning with Byte 128
    NoOfBytes:= uiNoOfBytes_RLSPP, // 512 bytes (128 DINT)
```

```

ValueAdr:=      ADR(ReadListArray),      // Address of ReadListArray
Axis:=          Antriebl,                // Fieldbus axis RA1
Done=>          bDone_RLSPP,              // TRUE := FB ready
Active=>        bActive_RLSPP,            //
Error=>         bError_RLSPP,             // Error occurred
ErrorId=>       enErrorID_RLSPP,          // Error class
ErrorIdent=>    stErrorIdent_RLSPP:      // Error definition
NoOfRecBytes=>  uiNoOfRecBytes_RLSPP);    // Number of received bytes

```

7.5.7 MB_ReadListSegParameterPtr, Besonderheiten der Feldbusantriebe



Der Baustein MB_ReadListSegParameterPtr funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```

Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;

```

oder

F_RELATED_TABLE

```

Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;

```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.8 MB_ReadMaxRealValue

Kurzbeschreibung

Der Funktionsbaustein gibt den zulässigen Maximalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als REAL-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Bibliothek	Bereich
RMB_PlcOpenFieldBus	Parameter_FBs

Tab. 7-31: Bibliothekszuordnung

Schnittstellenbeschreibung

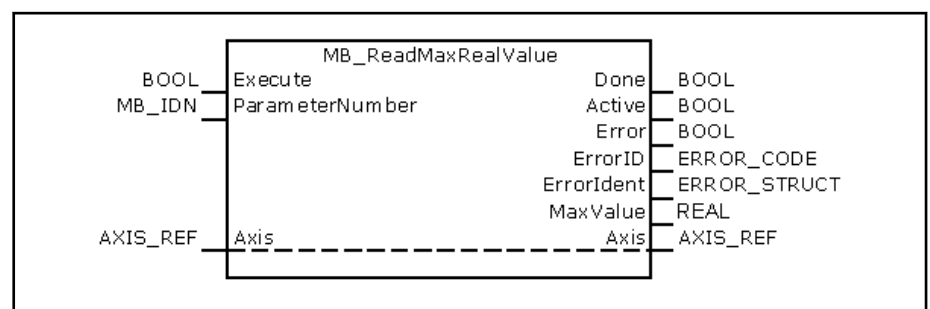


Abb. 7-35: FB MB_ReadMaxRealValue

RMB_PLCopenFieldBus.library

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Parameter-Konstanten, Seite 385)
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
	MaxValue	REAL	Ist "Done"=TRUE, zeigt "MaxValue" den zulässigen maximalen Eingabewert des ausgewählten Parameters an.

Tab. 7-32: Schnittstelle von FB MB_ReadMaxRealValue

Signal-Zeit-Diagramm

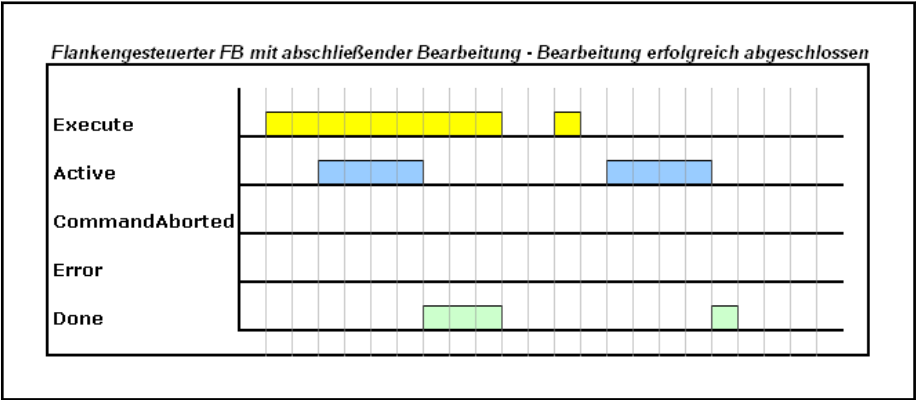


Abb. 7-36: Standard-Signal-Zeit-Diagramm

7.5.9 MB_ReadMaxRealValue, Besonderheiten der Feldbusantriebe

Der Baustein MB_ReadMaxRealValue aus der RMB_PLCopen-FieldBus.Library funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
```

```
ErrorIdent.Table := F RELATED TABLE;  
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)  
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.10 MB_ReadMaxValue

Kurzbeschreibung Der Funktionsbaustein gibt den zulässigen Maximalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute " voraus.

Bibliothek	Bereich
RMB_PlcOpenFieldBus	Parameter_FBs

Tab. 7-33: Bibliothekszuordnung

Schnittstellenbeschreibung

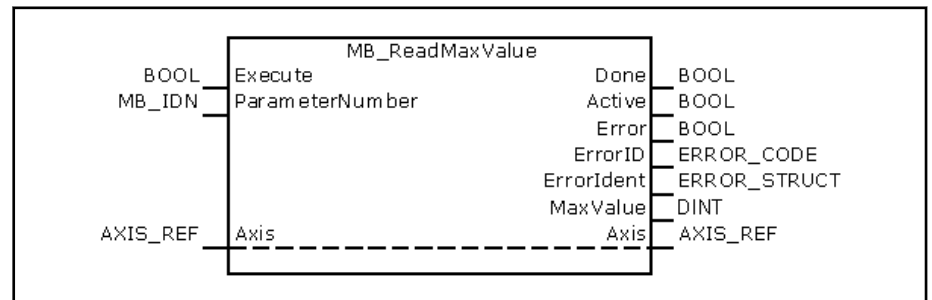


Abb. 7-37: FB MB_ReadMaxValue

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Parameter-Konstanten , Seite 385)
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
	MaxValue	DINT	Ist "Done"=TRUE, zeigt "MaxValue" den zulässigen maximalen Eingabewert des ausgewählten Parameters an.

Tab. 7-34: Schnittstelle von FB MB_ReadMaxValue

Kurzbeschreibung

RMB_PLcOpenFieldBus.library

Signal-Zeit-Diagramm

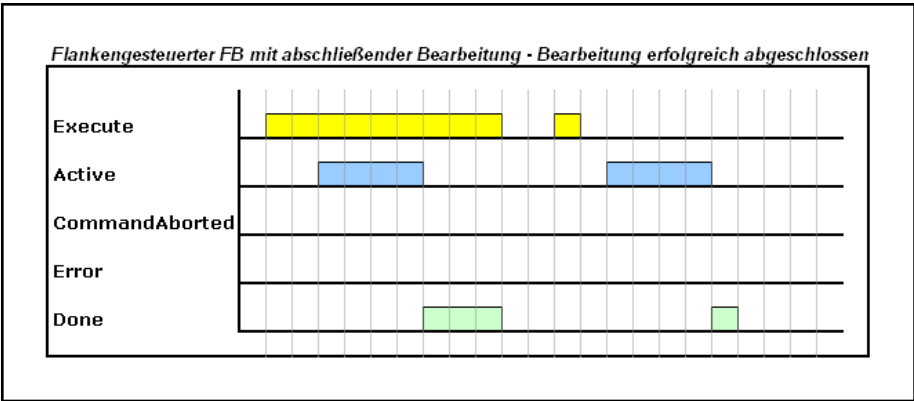


Abb. 7-38: Standard-Signal-Zeit-Diagramm

7.5.11 MB_ReadMaxValue, Besonderheiten der Feldbusantriebe

Der Baustein MB_ReadMaxValue aus der RMB_PLcOpenField-Bus.Library funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.12 MB_ReadMinRealValue

Kurzbeschreibung Der Funktionsbaustein gibt den zulässigen Minimalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als REAL-Zahl dargestellt.

Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute " voraus.

Bibliothek	Bereich
RMB_PlcOpenFieldBus	Parameter_FBs

Tab. 7-35: Bibliothekszuordnung

Schnittstellenbeschreibung

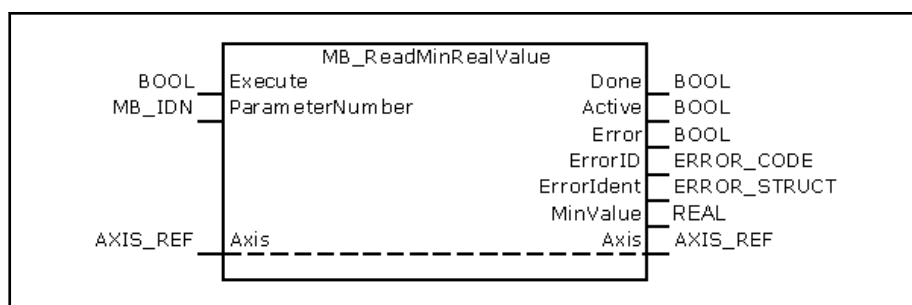


Abb. 7-39: FB MB_ReadMinRealValue

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Parameter-Konstanten , Seite 385)
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
	MinValue	REAL	Ist "Done"=TRUE, zeigt "MinValue" den zulässigen minimalen Eingabewert des ausgewählten Parameters an.

Tab. 7-36: Schnittstelle von FB MB_ReadMinRealValue

Signal-Zeit-Diagramm

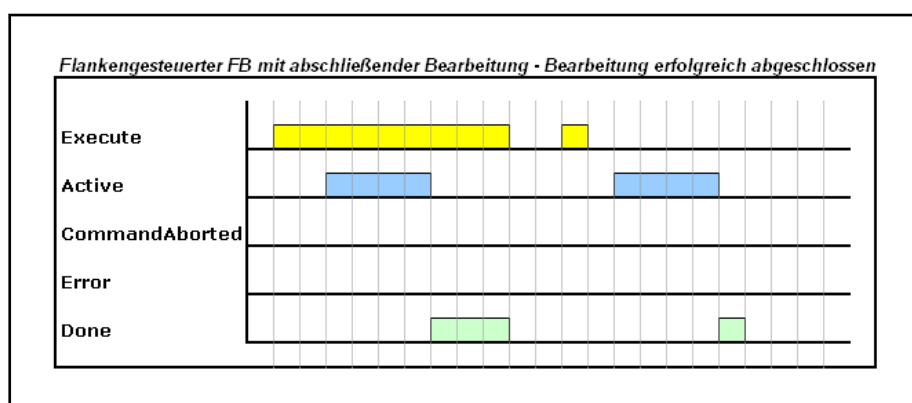


Abb. 7-40: Standard-Signal-Zeit-Diagramm

7.5.13 MB_ReadMinRealValue, Besonderheiten der Feldbusantriebe



Der Baustein MB_ReadMinRealValue aus der RMB_PLCopenFieldBus.Library funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

RMB_PLCopenFieldBus.library

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.14 MB_ReadMinValue

Kurzbeschreibung Der Funktionsbaustein gibt den zulässigen Minimalwert eines durch den Anwender vorgegebenen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.

 Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Execute " voraus.

Bibliothek	Bereich
RMB_PlcOpenFieldBus	Parameter_FBs

Tab. 7-37: Bibliothekszuordnung

Schnittstellenbeschreibung

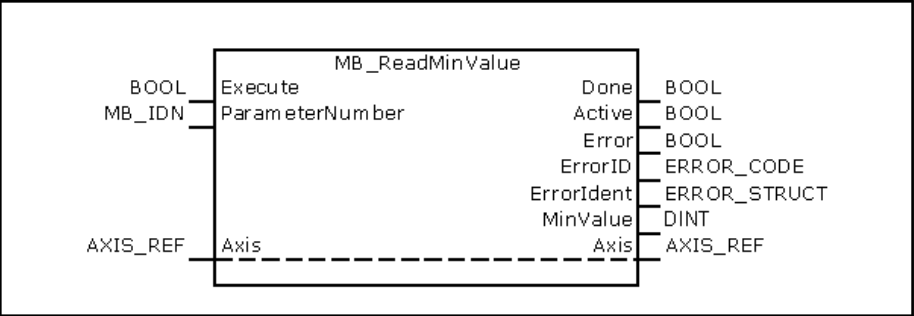


Abb. 7-41: FB MB_ReadMinValue

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Mit einer positiven Flanke (0/1-Übergang) von „Execute“ werden die Eingangsvariablen erfasst. Danach läuft der Lesevorgang unabhängig vom Wert der Variablen ab.
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (siehe Parameter-Konstanten, Seite 385)
VAR_OUTPUT	Done	BOOL	gültiger Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft

	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
	MinValue	DINT	Ist "Done"=TRUE, zeigt "MinValue" den zulässigen minimalen Eingabewert des ausgewählten Parameters an.

Tab. 7-38: Schnittstelle von FB MB_ReadMinValue

Signal-Zeit-Diagramm

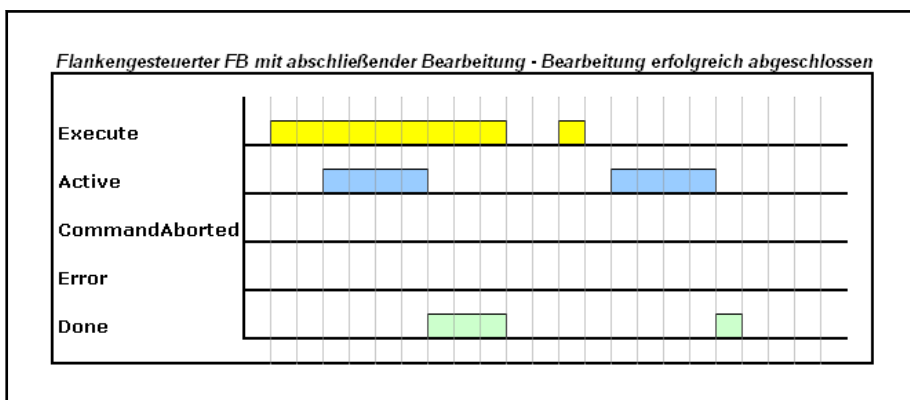


Abb. 7-42: Standard-Signal-Zeit-Diagramm

7.5.15 MB_ReadMinValue, Besonderheiten der Feldbusantriebe



Der Baustein MB_ReadMinValue aus der RMB_PLCOpenFieldBus.Library funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.16 MB_ReadParameter

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als DINT-Zahl dargestellt.

RMB_PLcOpenFieldBus.library



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
RMB_PlcOpenFieldBus	Parameter_FBs

Tab. 7-39: Bibliothekszuordnung

Schnittstellenbeschreibung

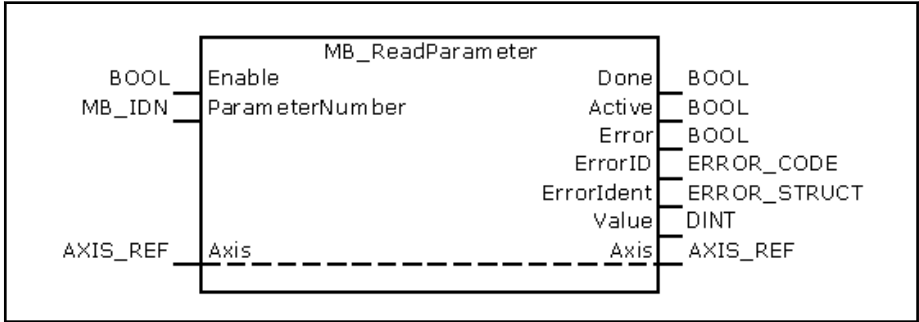


Abb. 7-43: FB MB_ReadParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse, bzw. zum adressierten Objekt.
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Parameter-Konstanten, Seite 385)
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	Ist "Done"=TRUE, zeigt "Value" den Wert des ausgewählten Parameters als DINT an.

Tab. 7-40: Schnittstelle von FB MB_ReadParameter

Signal-Zeit-Diagramm

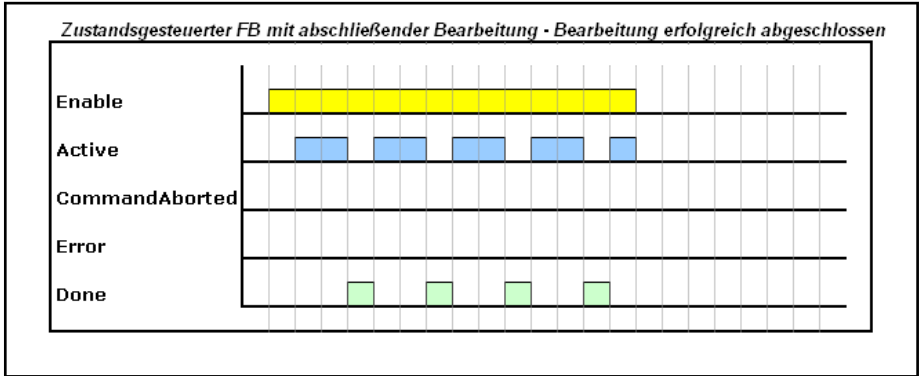


Abb. 7-44: Standard-Signal-Zeit-Diagramm

7.5.17 MB_ReadParameter, Besonderheiten der Feldbusantriebe



Der Baustein MB_ReadRealParameter funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Enable".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.18 MB_ReadRealParameter

Kurzbeschreibung

Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als REAL-Zahl dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-41: Bibliothekszuordnung

Schnittstellenbeschreibung

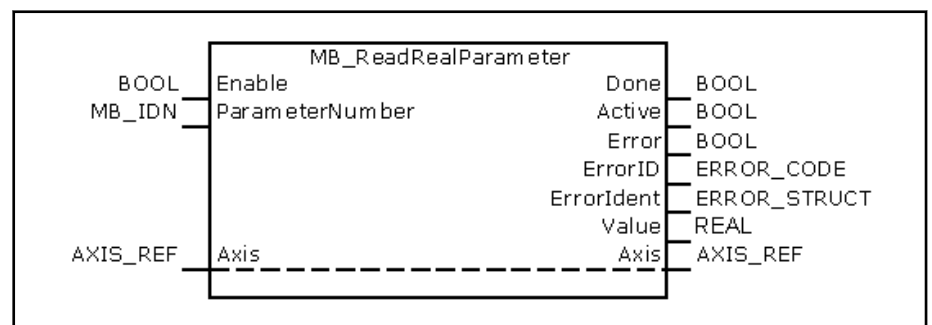


Abb. 7-45: FB MB_ReadRealParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist

RMB_PLCopenFieldBus.library

	Name	Typ	Kommentar
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Zugriff auf Daten eines Antriebs (IndraDrive) , Seite 385)
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	Ist "Done"=TRUE, zeigt "Value" den 4-Byte-Wert als REAL-Zahl

Tab. 7-42: Schnittstelle von FB MB_ReadRealParameter

Signal-Zeit-Diagramm

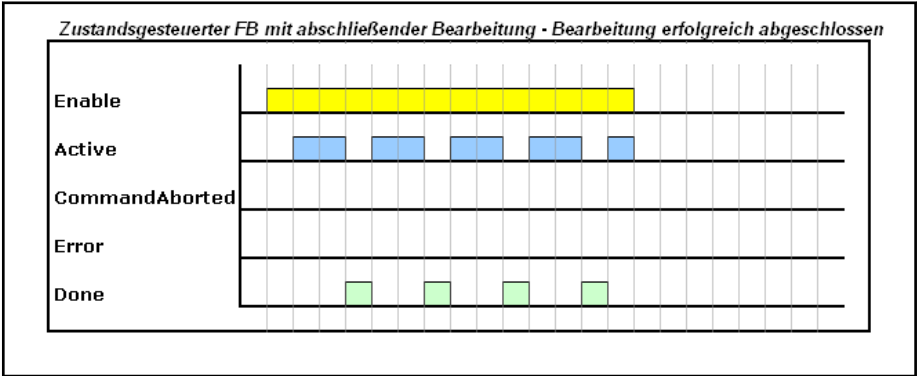


Abb. 7-46: Standard-Signal-Zeit-Diagramm

7.5.19 MB_ReadRealParameter, Besonderheiten der Feldbusantriebe

Der Baustein MB_ReadRealParameter funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Enable".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.20 MB_ReadSercosAttribute

Kurzbeschreibung Der Funktionsbaustein erlaubt das Lesen der SERCOS-Attribute eines Parameters. Der Rückgabewert wird als DWORD dargestellt.



Eine Änderung, z.B. der Parameternummer oder der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
RMB_PLCPopenFieldBus	Parameter_FBs

Tab. 7-43: Bibliothekszuordnung

Schnittstellenbeschreibung

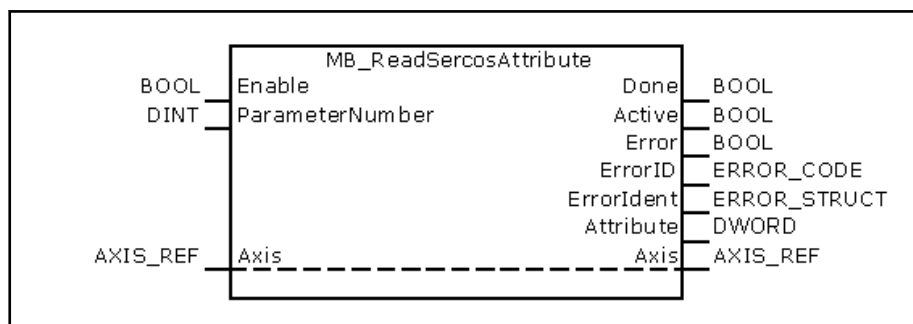


Abb. 7-47: FB MB_ReadSercosAttribute

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameter-Attributes, solange "Enable" TRUE ist
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Zugriff auf Daten eines Antriebs (IndraDrive) , Seite 385)
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
	Attribute	DWORD	Ist "Done"=TRUE, zeigt "Attribute" das Attribut des ausgewählten Parameters an

Tab. 7-44: Schnittstelle von FB MB_ReadSercosAttribute

Dekodierung des Attributs Zur Dekodierung wird in der Bibliothek "RIL_Utilitys.library" der Funktionsbaustein "IL_SercosAttribute" bereitgestellt.

RMB_PLCopenFieldBus.library

Signal-Zeit-Diagramm

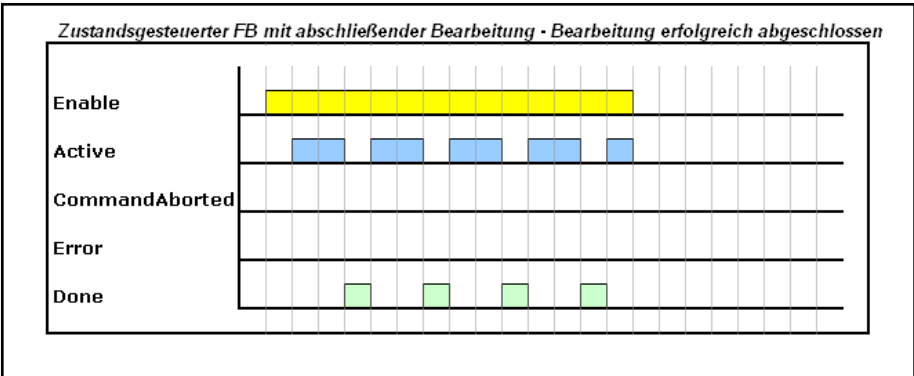


Abb. 7-48: Standard-Signal-Zeit-Diagramm

7.5.21 MB_ReadSercosAttribute, Besonderheiten der Feldbusantriebe

 Der Baustein MB_ReadSercosAttribute funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Enable".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.22 MB_ReadStringParameter

Kurzbeschreibung Der Funktionsbaustein gibt den Wert eines anwenderspezifischen Parameters zurück. Der Rückgabewert wird als Zeichenkette (STRING) entsprechend seiner tatsächlichen Länge mit maximal 255 Zeichen dargestellt.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-45: Bibliothekszuordnung

Schnittstellenbeschreibung

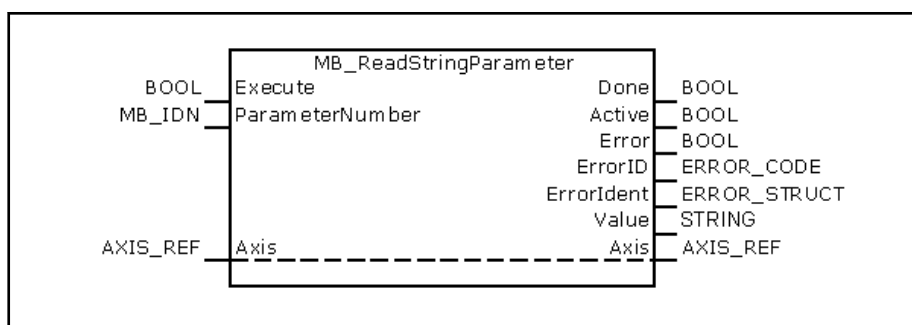


Abb. 7-49: FB MB_ReadStringParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Lesen des Textes vom Parameter mit steigender Flanke
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Zugriff auf Daten eines Antriebs (IndraDrive) , Seite 385)
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	STRING	Ist "Done"-TRUE, zeigt "Value" den Wert des ausgewählten Parameters als Zeichenkette mit maximal 255 Zeichen an.

Tab. 7-46: Schnittstelle von FB MB_ReadStringParameter

Signal-Zeit-Diagramm

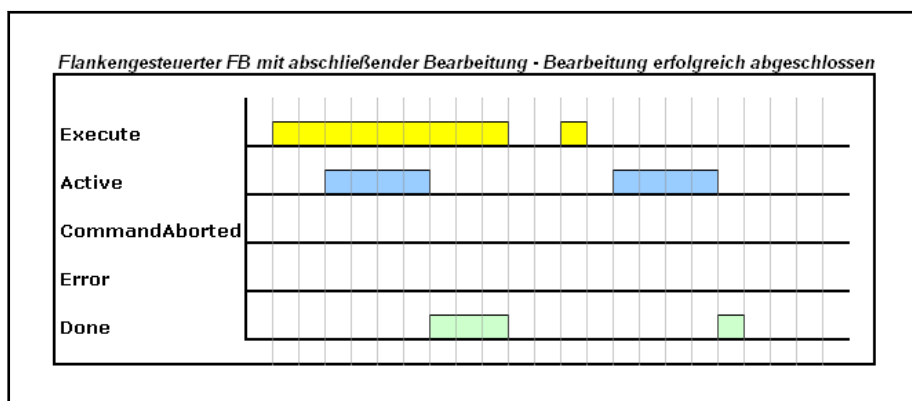


Abb. 7-50: Standard-Signal-Zeit-Diagramm

7.5.23 MB_ReadStringParameter, Besonderheiten der Feldbusantriebe



Der Baustein MB_ReadStringParameter funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

RMB_PLCopenFieldBus.library

F_RELATED_TABLE

```

Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;

```

oder

F_RELATED_TABLE

```

Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;

```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.24 MB_WriteListParameter

Kurzbeschreibung Der Funktionsbaustein schreibt einen Speicherinhalt definierter Länge (NoOf-Byte), der ab der Anfangsadresse (ValueAdr) zur Verfügung gestellt wurde, von der SPS in einen anwenderspezifischen 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 sie zu klein, fehlen Daten im Zielparameter!



Der Datentyp der Elemente des zu schreibenden Parameters und der Datentyp der Elemente des ARRAYs müssen übereinstimmen.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-47: Bibliothekszuordnung

Schnittstellenbeschreibung

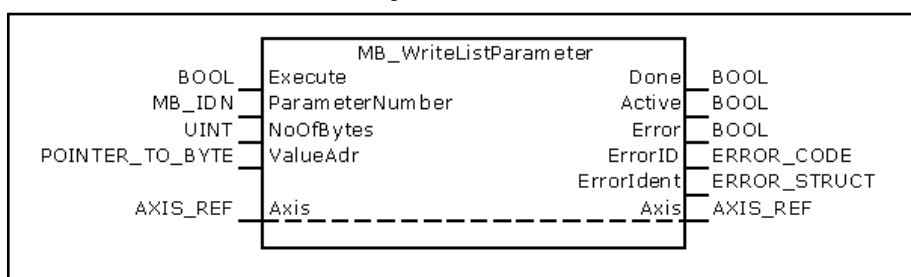


Abb. 7-51: FB MB_WriteListParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Parameter-Konstanten , Seite 385)

	Name	Typ	Kommentar
	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. 7-48: Schnittstelle von FB MB_WriteListParameter

Signal-Zeit-Diagramm

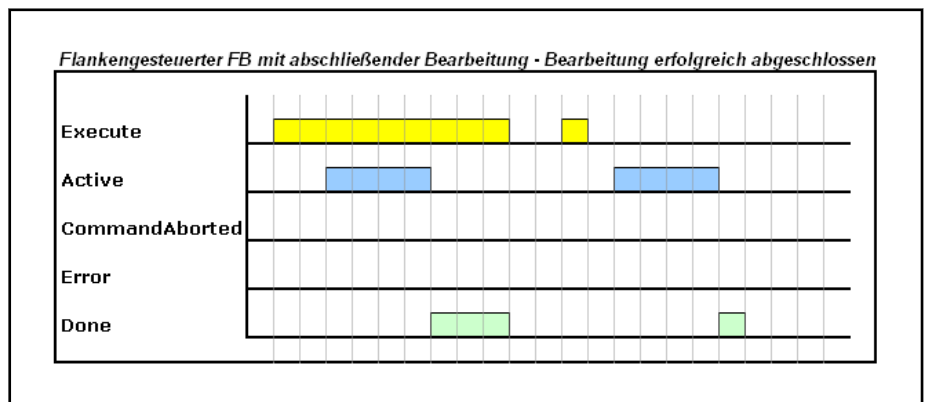


Abb. 7-52: Standard-Signal-Zeit-Diagramm

**Programmbeispiel
MB_WriteListParameter**

Dieses Programmbeispiel zeigt das Schreiben des Listenparameters (P-0-0072) mit 1024 Werten (DINT) aus dem WriteListArray.

Programmbeispiel, Deklarationsteil

```

PROGRAM WriteListParameter
VAR IN_OUT
    RA1:          AXIS_REF;           // Fieldbus axis RA1
END_VAR
VAR
    WriteListArray:  ARRAY [0..1023] OF DINT; // Data source

    FB WriteListParameter: MB WriteListParameter;
    bExecute_WLP:        BOOL;           // Start FB
    diParameterNo_WLP:   MB_IDN:= FP_P_0_0072; // CAM parameter P-0-0072
    uiNoOfBytes_WLP:     UINT:= 4096;    // DINT := 4 Bytes
    bDone_WLP:           BOOL;           // TRUE := FB ready
    bActive_WLP:         BOOL;           // FB works
    bError_WLP:          BOOL;           // Error occurred
    enErrorID_WLP:       ERROR_CODE;     // Error class
    stErrorIdent_WLP:    ERROR_STRUCT;   // Error definition
END_VAR

```

Programmbeispiel im Funktionsplan

RMB_PLCOpenFieldBus.library

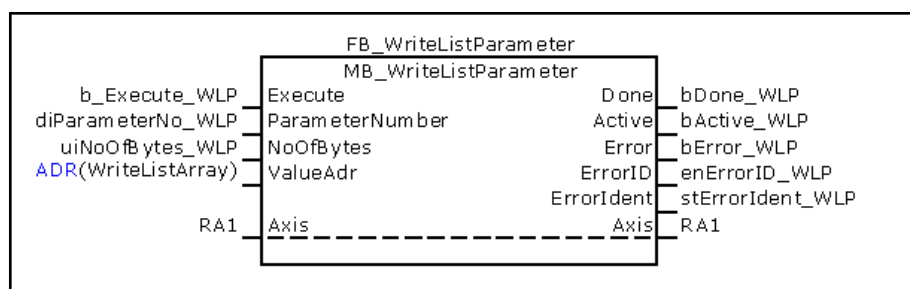


Abb. 7-53: FUP Programmbeispiel WriteListParameter

Programmbeispiel im Strukturierten Text

```

FB_WriteListParameter(
  Execute:=      bExecute_WLP,           // Start FB
  ParameterNumber:= diParameterNo_WLP,   // CAM parameter P-0-0072
  NoOfBytes:=    uiNoOfBytes_WLP,        // DINT := 4 Bytes
  ValueAdr:=     ADR(WriteListArray),    // Address of WriteListArray
  Axis:=         RA1,                    // Fieldbus axis
  Done=>         bDone_WLP,              // TRUE := FB ready
  Active=>       bActive_WLP,            // FB works
  Error=>        bError_WLP,             // Error occurred
  ErrorId=>      enErrorID_WLP,          // Error class
  ErrorIdent=>   stErrorIdent_WLP);      // Error definition

```

7.5.25 MB_WriteListParameter, Besonderheiten der Feldbusantriebe



Der Baustein MB_WriteListParameter funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```

Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;

```

oder

F_RELATED_TABLE

```

Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;

```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.26 MB_WriteListSegParameterPtr

Kurzbeschreibung

Der Funktionsbaustein schreibt einen Speicherinhalt definierter Länge (NoOf-Byte), der ab der Anfangsadresse (ValueAdr) in der SPS zur Verfügung gestellt wurde, in einen anwenderspezifisierten Parameter um den Offset verschoben.



Die vorgegebene Anzahl an Bytes muss genau mit der Datenmenge, die übertragen werden soll, übereinstimmen!

Ist die Anzahl zu groß, wird Datenmüll übertragen, ist sie zu klein, fehlen Daten im Zielparameter!



1. Der Offset ist so zu wählen, dass das erste zu schreibende Datenelement vollständig geschrieben werden kann.
2. Der Datentyp der Elemente des zu beschreibenden Parameters und der Datentyp der Elemente des ARRAYS in der SPS müssen übereinstimmen.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-49: Bibliothekszuordnung

Schnittstellenbeschreibung

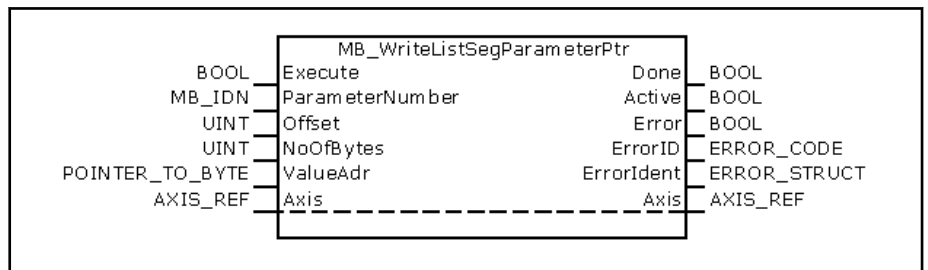


Abb. 7-54: FB MB_WriteListSegParameterPtr

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Parameter-Konstanten , Seite 385)
	Offset	UINT	Offset in Bytes, bezogen auf Anfang des Parameters, ab der beschrieben werden soll
	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. 7-50: Schnittstelle von FB MB_WriteListSegParameterPtr

RMB_PLCopenFieldBus.library

Signal-Zeit-Diagramm

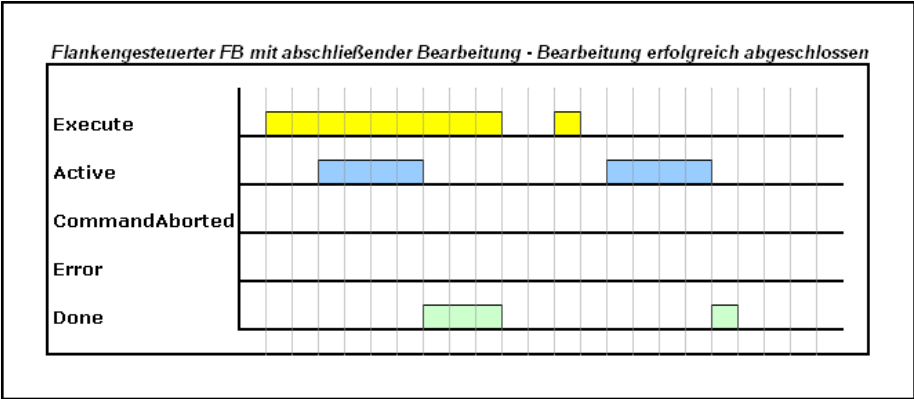


Abb. 7-55: Standard-Signal-Zeit-Diagramm

Programmbeispiel MB_WriteListSegParameterPtr

Dieses Programmbeispiel zeigt das Schreiben von 512 Byte auf den Listenparameter (P-0-0072) ab Byte 128 aus dem WriteListArray.
Programmbeispiel, Deklarationsteil

```
PROGRAM WriteListSegParameterPtr
VAR IN_OUT
    RA1:                AXIS_REF;                // Fieldbus axis RA1
END_VAR
VAR
    WriteListArray:     ARRAY [0..1023] OF DINT; // Data source

FB WriteListSegParameterPtr: MB_WriteListSegParameterPtr;
bExecute_WLSPP:             BOOL;                // Start FB
diParameterNo_WLSPP:        MB_IDN:= FP_P_0_0072; // CAM parameter P-0-0072
uiOffset_WLSPP:             UINT:= 128;          // beginning with Byte 128
uiNoOfBytes_WLSPP:          UINT:= 512;          // 512 bytes (128 DINT)
bDone_WLSPP:                BOOL;                // TRUE := FB ready
bActive_WLSPP:              BOOL;                // FB works
bError_WLSPP:               BOOL;                // Error occurred
enErrorID_WLSPP:            ERROR_CODE;          // Error class
stErrorIdent_WLSPP:         ERROR_STRUCT;        // Error definition
END_VAR
```

Programmbeispiel im Funktionsplan

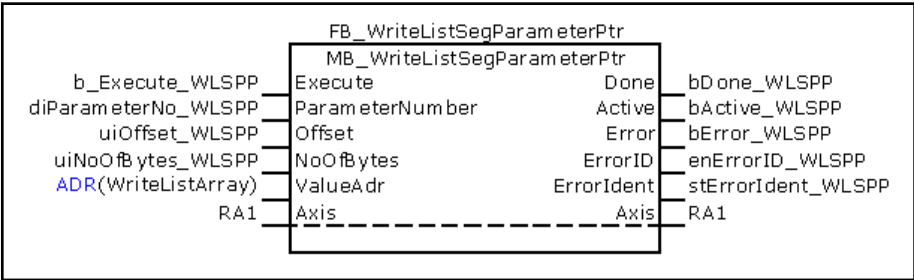


Abb. 7-56: FUP Programmbeispiel WriteListSegParameterPtr

Programmbeispiel im Strukturierten Text

```
FB_WriteListSegParameterPtr(
    _Execute:=          bExecute_WLSPP,                // Start FB
    ParameterNumber:=   diParameterNo_WLSPP,           // CAM parameter P-0-0072
    Offset:=            uiOffset_WLSPP,                // beginning with Byte 128
    NoOfBytes:=         uiNoOfBytes_WLSPP,             // 512 bytes (128 DINT)
    ValueAdr:=          ADR(WriteListArray),           // Address of WriteListArray
    Axis:=              RA1,                            // Drive address
    Done=>              bDone_WLSPP,                   // TRUE := FB ready
    Active=>            bActive_WLSPP,                 // FB works
    Error=>             bError_WLSPP,                  // Error occurred
```

```
ErrorId=>          enErrorID_WLSPP,          // Error class
ErrorIdent=>       stErrorIdent_WLSPP);      // Error definition
```

7.5.27 MB_WriteListSegParameterPtr, Besonderheiten der Feldbusantriebe



Der Baustein MB_WriteListSegParameterPtr funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.28 MB_WriteParameter

Kurzbeschreibung

Der Funktionsbaustein schreibt einen Wert in einen durch den Anwender vorgegebenen Parameter.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-51: Bibliothekszuordnung

Schnittstellenbeschreibung

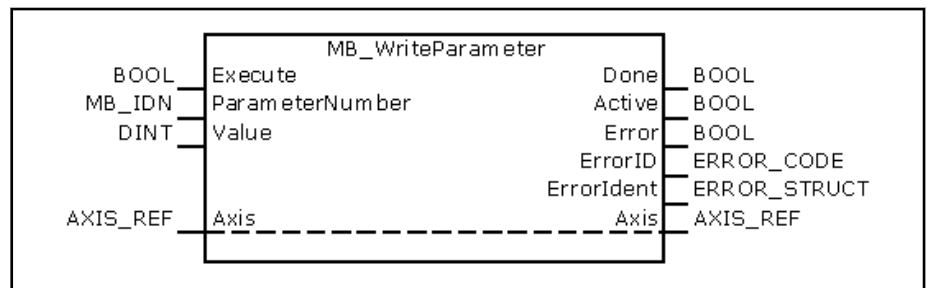


Abb. 7-57: FB MB_WriteParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Parameter-Konstanten , Seite 385)

RMB_PLCopenFieldBus.library

	Name	Typ	Kommentar
	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. 7-52: Schnittstelle von FB MB_WriteParameter

Signal-Zeit-Diagramm

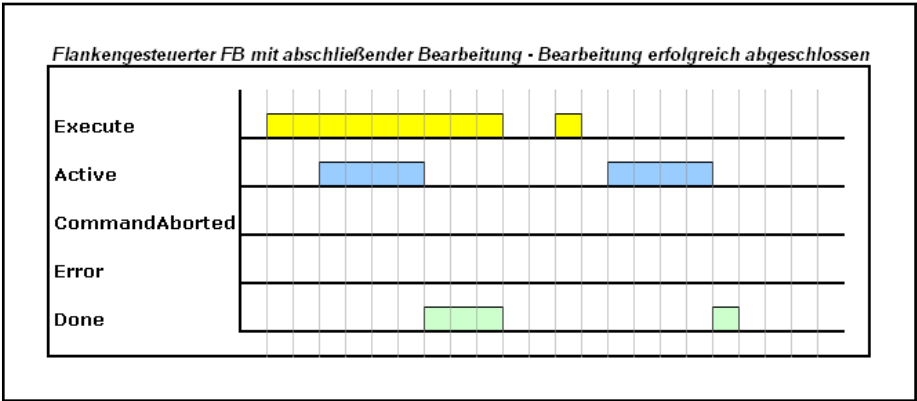


Abb. 7-58: Standard-Signal-Zeit-Diagramm

7.5.29 MB_WriteParameter, Besonderheiten der Feldbusantriebe

Der Baustein MB_WriteParameter funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.30 MB_WriteRealParameter

Kurzbeschreibung Der Funktionsbaustein schreibt einen Wert in einen anwenderspezifisierten Parameter.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-53: Bibliothekszuordnung

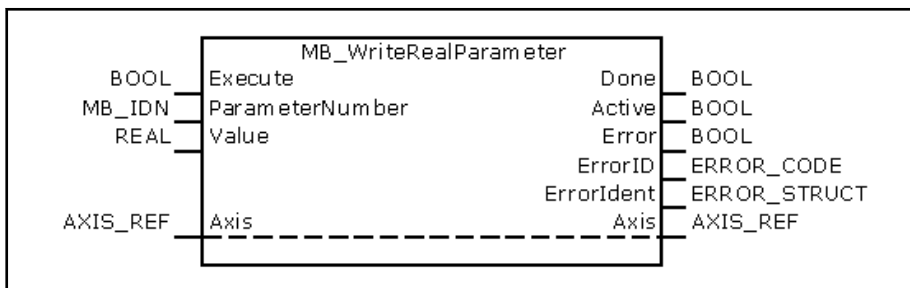


Abb. 7-59: FB MB_WriteRealParameter

Schnittstellenbeschreibung

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Parameter-Konstanten , Seite 385)
	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. 7-54: Schnittstelle von FB MB_WriteRealParameter

Signal-Zeit-Diagramm

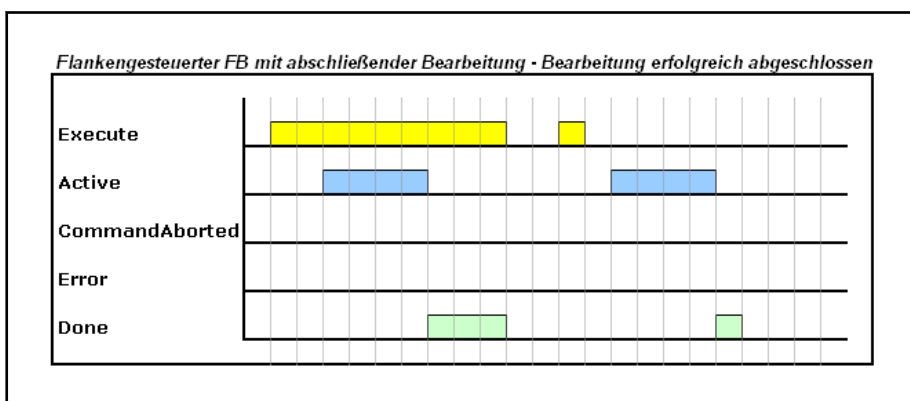


Abb. 7-60: Standard-Signal-Zeit-Diagramm

7.5.31 MB_WriteRealParameter, Besonderheiten der Feldbusantriebe



Der Baustein MB_WriteRealParameter funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

RMB_PLCopenFieldBus.library

Fehlerbehandlung Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.32 MB_WriteStringParameter

Kurzbeschreibung Der Funktionsbaustein schreibt eine Zeichenkette mit max. 255 Zeichen in einen anwenderspezifizierten Parameter.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-55: Bibliothekszuordnung

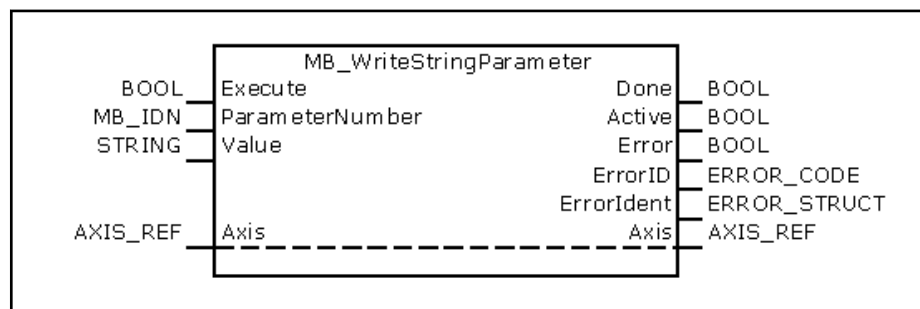
Schnittstellenbeschreibung

Abb. 7-61: FB MB_WriteStringParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Wertes, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, (z.B. FP_S_0_0000, siehe auch Parameter-Konstanten , Seite 385)
	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

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 7-56: Schnittstelle von FB MB_WriteStringParameter

Signal-Zeit-Diagramm

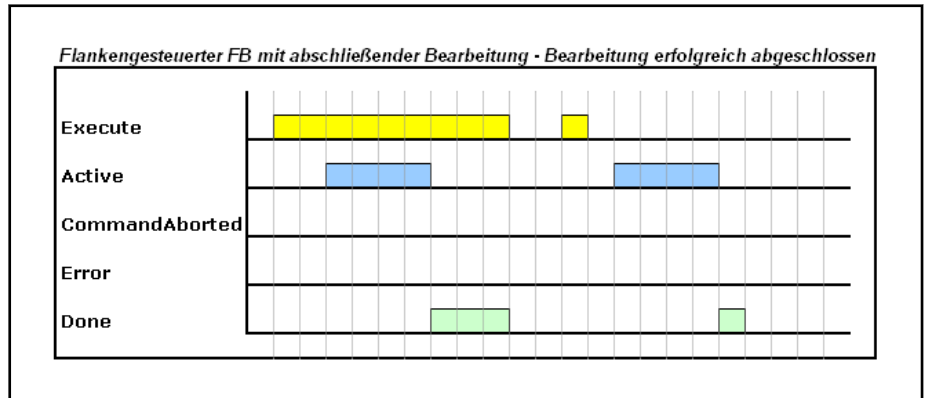


Abb. 7-62: Standard-Signal-Zeit-Diagramm

7.5.33 MB_WriteStringParameter, Besonderheiten der Feldbusantriebe



Der Baustein MB_WriteStringParameter funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Execute".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := ACCESS_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#F60B;
(*Command execution not possible (transfer still active)*)
ErrorIdent.Additional2 := 16#0000;
```

oder

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.5.34 MC_ReadActualPosition

Kurzbeschreibung

Der Funktionsbaustein liest die aktuelle Position des Antriebes.



Eine Änderung der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-57: Bibliotheksuzuordnung

RMB_PLCopenFieldBus.library

Schnittstellenbeschreibung

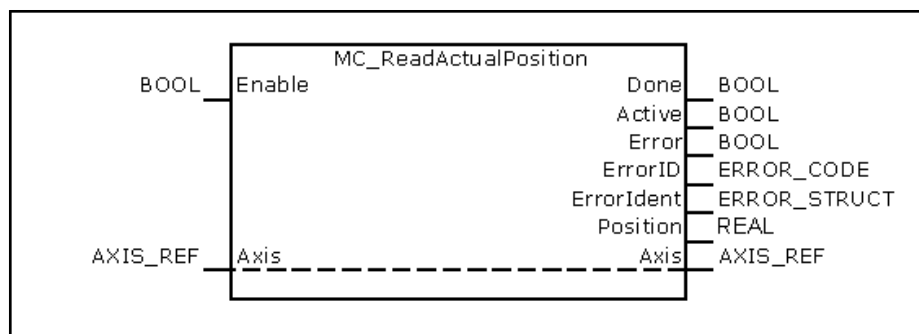


Abb. 7-63: FB MC_ReadActualPosition

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	kontinuierliches Lesen des Positionswertes, solange "Enable" TRUE ist
VAR_OUTPUT	Done	BOOL	gültiger Wert ist 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	REAL	aktuelle absolute Position des Antriebs ⁷⁾

Tab. 7-58: Schnittstelle von FB MC_ReadActualPosition

Signal-Zeit-Diagramm

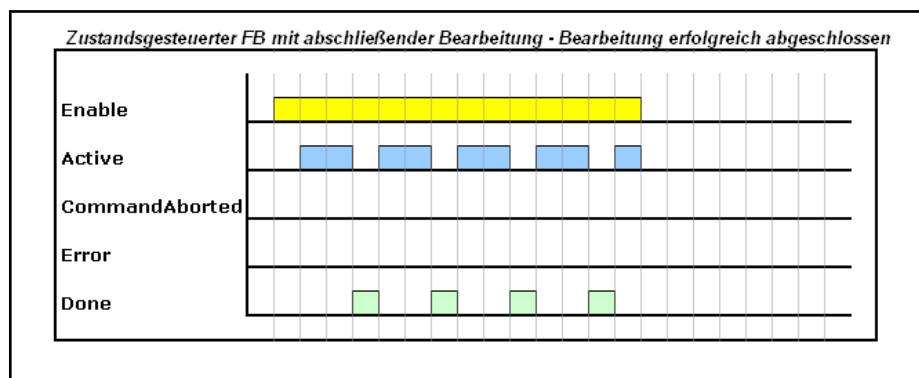


Abb. 7-64: Standard-Signal-Zeit-Diagramm

7.5.35 MC_ReadActualPosition, Besonderheiten der Feldbusantriebe



Der Baustein MC_ReadActualPosition funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Verwendete Parameter

Der Funktionsbaustein liest die aktuelle Position des Antriebes "Positionswert" S-0-0051, bzw. den Parameter, der im Antrieb auf der Feldbus-Prozessdaten-Eingangsbeschreibung angegeben ist.

S-0-0051, Lage-Istwert Geber 1 zeigt die aktuelle Position des Motorgebers.

⁷⁾ Die Maßeinheiten der Zeit, der Lage, der Geschwindigkeit, der Beschleunigung und der Verzögerung sind von der wirksamen Wichtung abhängig.

Fehlerbehandlung Der Funktionsbaustein MC_ReadActualPosition erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Enable".

Im Falle von Konfigurationsfehlern am Baustein erscheint

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

ansonsten wird der Auftrag an den Antrieb gegeben und dieser reagiert gegebenenfalls mit eigener Fehlermeldung.

7.5.36 MC_ReadAxisError

Kurzbeschreibung Der Funktionsbaustein ruft den Fehlerstatus der adressierten Achse ab. Je Achse ist dazu eine Instanz des Funktionsbausteines zu verwenden.



Eine Änderung der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-59: Bibliothekszuordnung

Schnittstellenbeschreibung

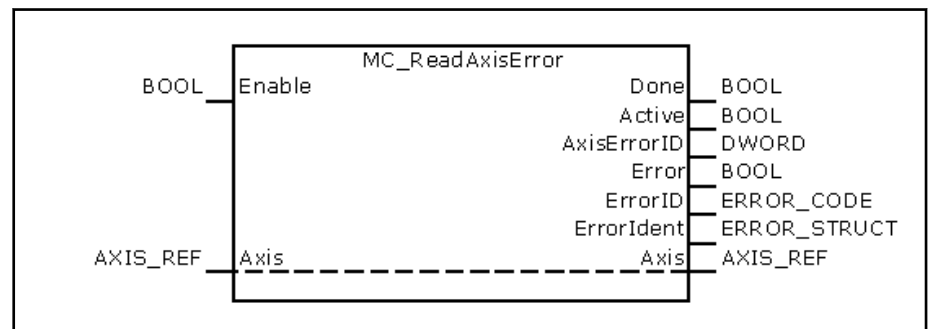


Abb. 7-65: FB MC_ReadAxisError

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	kontinuierliches Lesen der Fehlerinformation, solange "Enable" TRUE ist
VAR_OUTPUT	Done	BOOL	Wert ist verfügbar
	Active	BOOL	Bearbeiten der Daten nach Abschluss der Vorverarbeitung läuft
	AxisErrorID	DWORD	Zeigt den Fehler des Antriebs der Achse an
	Error	BOOL	Zeigt an, dass beim Aufruf der FB-Instanz ein Fehler aufgetreten ist
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache des Bausteinaufrufes, nicht des Achsfehlers!
	ErrorIdent	ERROR_STRUCT	Detailed information about the error of the Bausteinaufrufes, not the axis error!

Tab. 7-60: Schnittstelle von FB MC_ReadActualPosition

RMB_PLCopenFieldBus.library

Signal-Zeit-Diagramm

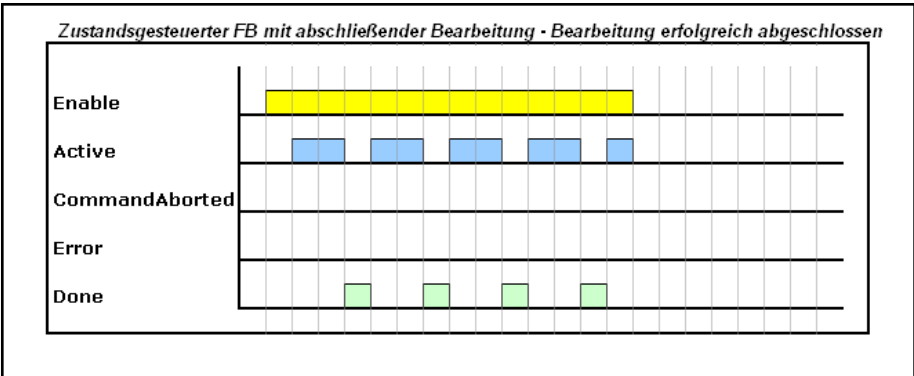


Abb. 7-66: Standard-Signal-Zeit-Diagramm

7.5.37 MC_ReadAxisError, Besonderheiten der Feldbusantriebe

Der Baustein MC_ReadAxisError funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Verwendete Parameter Der Funktionsbaustein ruft den Fehlerstatus (S-0-0390, Diagnose-Nummer) der adressierten Achse ab.
In diesem Parameter wird die im Display angezeigte Diagnosenummer abgelegt. Dadurch ist es der Steuerung möglich, eigene Diagnosen anhand dieser Nummer zu generieren (z.B. in Sprachen, in denen die Diagnosetexte nicht im Antrieb abgelegt sind).

Fehlerbehandlung Der Funktionsbaustein MC_ReadAxisError erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Enable".
Im Falle von Konfigurationsfehlern am Baustein erscheint
F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

7.5.38 MC_ReadStatus

Kurzbeschreibung Dieser Funktionsbaustein liest den aktuellen Status der Achse und gibt diesen dekodiert aus.

Eine Änderung der Achse am Eingang des Funktionsbausteines, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Durch die Erweiterung des Bausteins wurde dieser inkompatibel zur Vorversion geändert!

Bibliothek	Bereich
RMB_PLCopenFieldBus	Parameter_FBs

Tab. 7-61: Bibliothekszuordnung

Schnittstellenbeschreibung

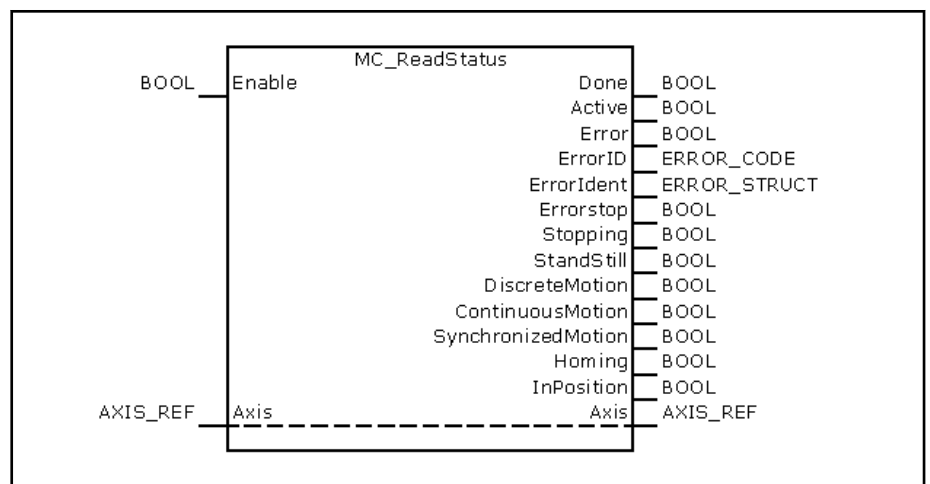


Abb. 7-67: FB MC_ReadStatus

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	kontinuierliches Lesen des Achsstatus, solange "Enable" TRUE ist
VAR_OUTPUT	Done	BOOL	gültiger Wert ist 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
	Errorstop	BOOL	Antrieb/Achse im Zustand 'Errorstop'
	Stopping	BOOL	Antrieb/Achse im Zustand 'Stopping'
	StandStill	BOOL	Antrieb/Achse im Zustand 'StandStill'
	DiscreteMotion	BOOL	Antrieb/Achse im Zustand 'DiscreteMotion'
	Continuous-Motion	BOOL	Antrieb/Achse im Zustand 'ContinuousMotion'
	Synchronized-Motion	BOOL	Antrieb/Achse im Zustand 'SynchronizedMotion'
	Homing	BOOL	Antrieb im Zustand 'Homing'
	InPosition	BOOL	Antrieb im Zustand 'InPosition'

Tab. 7-62: Schnittstelle von FB MC_ReadStatus

RMB_PLCopenFieldBus.library

Signal-Zeit-Diagramm

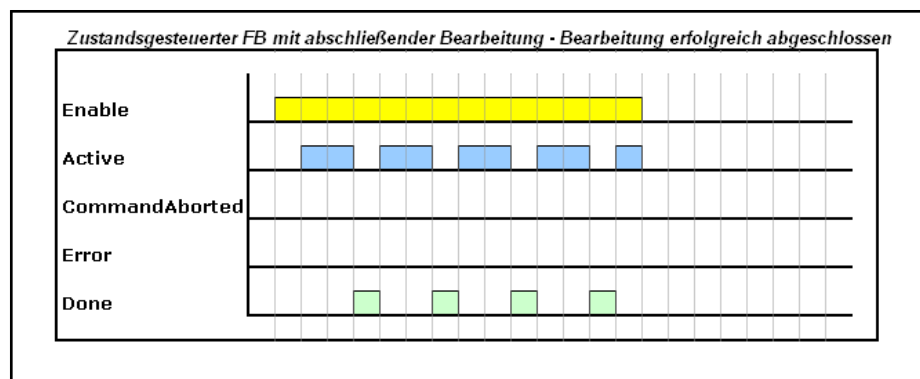


Abb. 7-68: Standard-Signal-Zeit-Diagramm

7.5.39 MC_ReadStatus, Besonderheiten der Feldbusantriebe



Der Baustein MC_ReadStatus funktioniert nur im Zusammenhang mit IndraDrive-Antrieben.

Verwendete Parameter

Dieser Funktionsbaustein liest den aktuellen Status der Achse (S-0-0144, Signal-Statuswort, bzw. den Parameter, der im Antrieb auf der Feldbus-Prozessdaten-Eingangsbeschreibung angegeben ist) und gibt diesen dekodiert aus.

Fehlerbehandlung

Der Funktionsbaustein erzeugt Fehlermeldungen nur bei einer positiven Flanke oder TRUE am Eingang "Enable".

F_RELATED_TABLE

```
Error := TRUE;
ErrorID := RESOURCE_ERROR;
ErrorIdent.Table := F_RELATED_TABLE;
ErrorIdent.Additional1 := 16#000A; (*Configuration error*)
ErrorIdent.Additional2 := 0;
```

Wurde der Auftrag an den Antrieb übergeben und kann dort nicht ausgeführt werden, antwortet der Antrieb mit einer entsprechenden Fehlermeldung.

7.6 System_FBs

Im Ordner "System_FBs" befinden sich die Funktionsbausteine

- MP_Drive
- MP_DriveParameter und
- MP_FB_IndraDrv_AcyclicCommunication.

Die Funktionsbausteine und ihre Methoden realisieren die Kommunikation zwischen der Steuerung und dem jeweiligen Feldbusantrieb.



Die Verwendung dieser Elemente ist zu unterlassen, da Nebenwirkungen, einschließlich Störungen im Bewegungsablauf der Achsen, nicht ausgeschlossen werden können!

7.7 SPS steuert Antriebe über eine Feldbusverbindung (Feldbusachsen)

7.7.1 Allgemeines

Antriebe (IndraDrive ab MPx2) können in Abhängigkeit von ihrer Schnittstelle über den entsprechenden Feldbus von einer Rexroth IndraLogic XLC oder einer IndraMotion MLC gesteuert werden.



Die Antriebe sind dabei über einen PROFIBUS mit der Steuerung verbunden.

Die Kopplung über einen sercos II- oder sercos III-Bus ist nicht Gegenstand dieser Beschreibung.

Die Ansteuerung erfolgt aus dem SPS-Programm unter Nutzung von MotionLogic-Bausteinen nach dem PLCopen-Konzept.

Bevor eine solche Feldbusachse am PROFIBUS betrieben werden kann, sind

- ein IndraWorks-Projekt anzulegen,
- die gewünschte SPS aus der Gerätebibliothek in das Projekt zu übernehmen,
- eine PROFIBUS-Schnittstelle (onboard oder Funktionsmodul) als PROFIBUS DP Master zu aktivieren,
- [Feldbusachsen der SPS bekannt zu machen, Seite 422](#), und
- [kommunikationsnotwendige Antriebseinstellungen, Seite 424](#), vorzunehmen.

Die SPS-Programme nutzen MotionLogic-Bausteine, Funktionen und Datentypen, die in mehreren [Bibliotheken, Seite 426](#), zusammengefasst werden.

Diese Bibliotheken werden gemeinsam mit dem/den SPS-Programm(en) in das IndraWorks-Projekt eingebunden.

Zusätzlich zur Steuerung des Feldbusantriebes über das SPS-Programm ist es notwendig, mit dem Antrieb Werte auszutauschen, die in seinen Parametern gespeichert sind.

Für **IndraDrive** kann dies über die PLCopen-Funktionsbausteine (MB_ReadParameter, MB_ReadRealParameter, MB_WriteParameter, MB_WriteRealParameter...) der **RMB_PLCopenFieldBus.library**, erfolgen.

7.7.2 Weiterführende Dokumentationen zu IndraDrives

Titel Rexroth IndraDrive ...	Dokumentationsart	Dokumentations-Type ¹⁾ DOK-INDRV*-...	Materialnummer R911...
MPx-17 Funktionen	Anwendungsbeschreibung	MP*-17VRS**-APxx-DE-P	331235
MPx-16 Funktionen	Anwendungsbeschreibung	MP*-16VRS**-APxx-DE-P	326484
Firmware für Antriebsregelgeräte MPH-08, MPB-08, MPD-08, MPC-08	Funktionsbeschreibung	MP*-08VRS**-APxx-DE-P	332642
Firmware für Antriebsregelgeräte MPH-07, MPB-07, MPD-07, MPC-07	Funktionsbeschreibung	MP*-07VRS**-FKxx-DE-P	328669

RMB_PLCopenFieldBus.library

Titel Rexroth IndraDrive ...	Dokumentationsart	Dokumentations-Type ¹⁾ DOK-INDRV*-...	Materialnummer R911...
Firmware für Antriebsregelgeräte MPH-06, MPB-06, MPD-06, MPC-06	Funktionsbeschreibung	MP*-06VRS**-FKxx-DE-P	326079
Firmware für Antriebsregelgeräte MPH-05, MPB-05, MPD-05	Funktionsbeschreibung	MP*-05VRS**-FKxx-DE-P	320181
Firmware für Antriebsregelgeräte MPH-04, MPB-04, MPD-04	Funktionsbeschreibung	MP*-04VRS**-FKxx-DE-P	315484
Firmware für Antriebsregelgeräte MPH-03, MPB-03, MPD-03	Funktionsbeschreibung	MP*-03VRS**-FKxx-DE-P	308328
Firmware für Antriebsregelgeräte MPH-02, MPB-02, MPD-02	Funktionsbeschreibung	MP*-02VRS**-FKxx-DE-P	299224
MPx-16 und MPx-17 Parameter	Referenz	GEN1-PARA**-RExx-DE-P	328650
Antriebsregelgeräte MPx-02 bis MPx-08	Parameterbeschreibung	GEN-**VRS**-PAxx-DE-P	297316
MPx-16 und MPx-17 Diagnosen	Referenz	GEN1-DIAG**-RExx-DE-P	326539
MPx-02 bis MPx-08 und HMV	Hinweise zur Störungsbeseitigung	GEN-**VRS**-WAxx-DE-P	297318

1) In den Dokumentations-Typen ist "xx" ein Platzhalter für den aktuellen Ausgabestand der Dokumentation (Beispiel: FK02 bedeutet die zweite Ausgabe einer Funktionsbeschreibung)

Tab. 7-63: Dokumentationen – Übersicht

7.7.3 Feldbusachsen im Projekt anlegen

Damit die Antriebe korrekt mit der SPS kommunizieren können, sind System- und Antriebseinstellungen im IndraWorks-Projekt an der SPS notwendig.

Die Feldbusachsen sind am PROFIBUS normale Feldbusteilnehmer.

Die Beschreibungsdateien für mögliche IndraDrives sind in der IndraWorks-Gerätebibliothek enthalten:

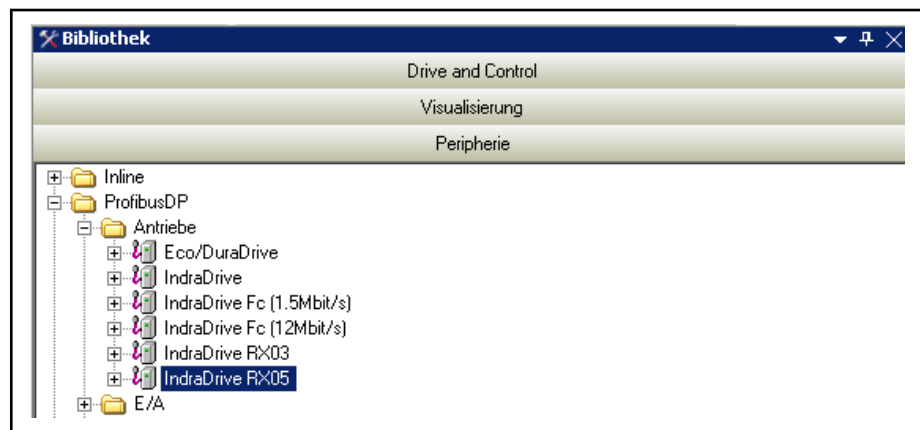


Abb. 7-69: Beschreibungsdateien der Antriebe in der IndraWorks Bibliothek

Ziehen Sie aus der Bibliothek die benötigten Antriebe auf das PROFIBUS DP Master-Objekt.

Antriebe lassen sich im Projektextplorer so auch zwischen bereits vorhandene Antriebe einfügen.

Alternativ können Sie Antriebe im Kontextmenü über **Hinzufügen ► Slave ► Antriebe** des PROFIBUS DP Master-Objekts hinzufügen.

Hierbei wird der neue Antrieb als letzter Antrieb unter das PROFIBUS DP Master-Objekt eingefügt.

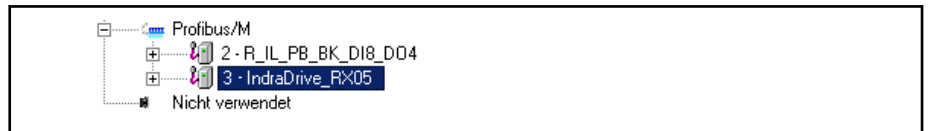


Abb. 7-70: IndraDrive als PROFIBUS-Teilnehmer eintragen

Der Antrieb kann dabei umbenannt werden:

Kontextmenü, **Umbenennen F2**, z.B. "Drive_X" statt "IndraDrive_RX05".

Der Antrieb kann eine neue Stationsadresse erhalten,:

Doppelklick auf das Objekt, ändern der Stationsadresse z.B. auf "11" statt "3".

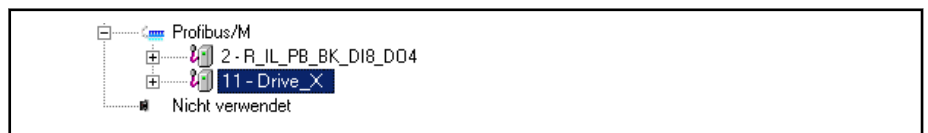


Abb. 7-71: IndraDrive als PROFIBUS-Teilnehmer umbenannt

Der PROFIBUS-Teilnehmer besitzt mehrere Subteilnehmer, deren Inhalt durch die Gerätebeschreibungsdatei vorgegeben ist.



Beim IndraDrive sind folgende Einstellungen erforderlich:

- ParamCh_5_Words,
- F-Modul_not_used, (**wenn das Modul nicht benutzt wird**)
- Input_8_Words,
- Output_10_Words.

Die Antriebe (Gerätedateien) im Urzustand haben nicht die notwendigen Einstellungen, um über den PROFIBUS in Zusammenhang mit den PLCopen-Bausteinen aus der RMB_PLCopenFieldBus.Library zu kommunizieren.

Zunächst müssen die im Hinweis enthaltenen Einstellungen durch "Gerät setzen" der Subteilnehmer gemacht werden.

RMB_PLCopenFieldBus.library

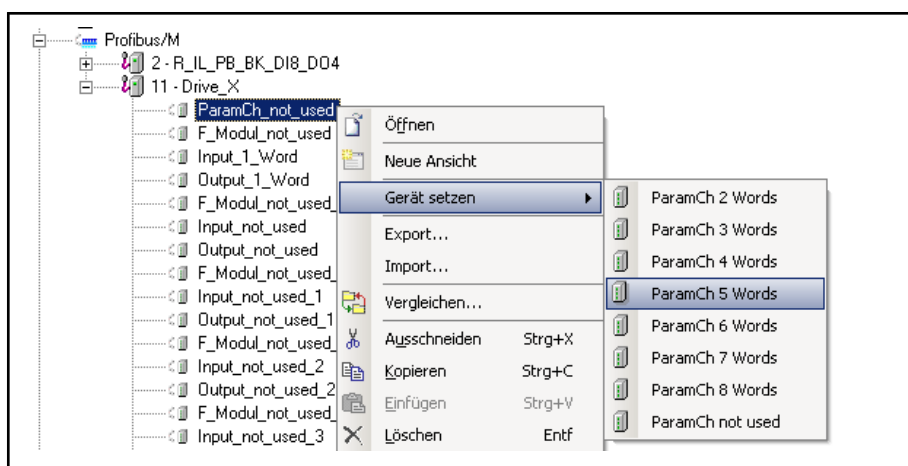


Abb. 7-72: IndraDrive, Einstellen der Subteilnehmer

Auch die Antriebe selbst sind in ihrem Urzustand nicht für eine solche Kommunikation mit einer SPS über einen PROFIBUS konfiguriert.

Aus diesem Grund müssen Antriebseinstellungen für die SPS-Kommunikation vorgenommen werden.

7.7.4 Antriebseinstellungen für die SPS-Kommunikation

Allgemeines

Um Feldbusantriebe mit den PLCopen-Bausteinen ansteuern zu können, werden einige grundlegende Einstellungen benötigt, die die Kommunikation der PLCopen-Bausteine mit den Feldbusantrieben betreffen.



Die hier beschriebenen Einstellungen beziehen sich nur auf die Kommunikation des Antriebes mit der SPS und beschreiben NICHT die allgemeine Antriebsinbetriebnahme.

Die dazu notwendigen Einstellungen können mit dem "IndraWorks Engineering"-Desktop oder mit "IndraWorks DS" vorgenommen werden.



Die Parametrierung kann zur Zeit **nicht** über die PROFIBUS-Anschaltung der SPS erfolgen.

Vom PC, auf dem der IndraWorks Engineering Desktop gestartet ist, ist die Kommunikation zur Parametrierung über RS232 seriell oder einen eigens dafür vorhanden PROFIBUS DP Master herzustellen.

Im folgenden wird die Parametrierung über eine serielle Kommunikation beschrieben.

Herstellen der Kommunikation zur Antriebsparametrierung

Mittels der Funktion **Projekt ► Nach Geräten suchen** im IndraWorks Engineering-Desktop wird die Parametrierkommunikation zum Antrieb hergestellt.

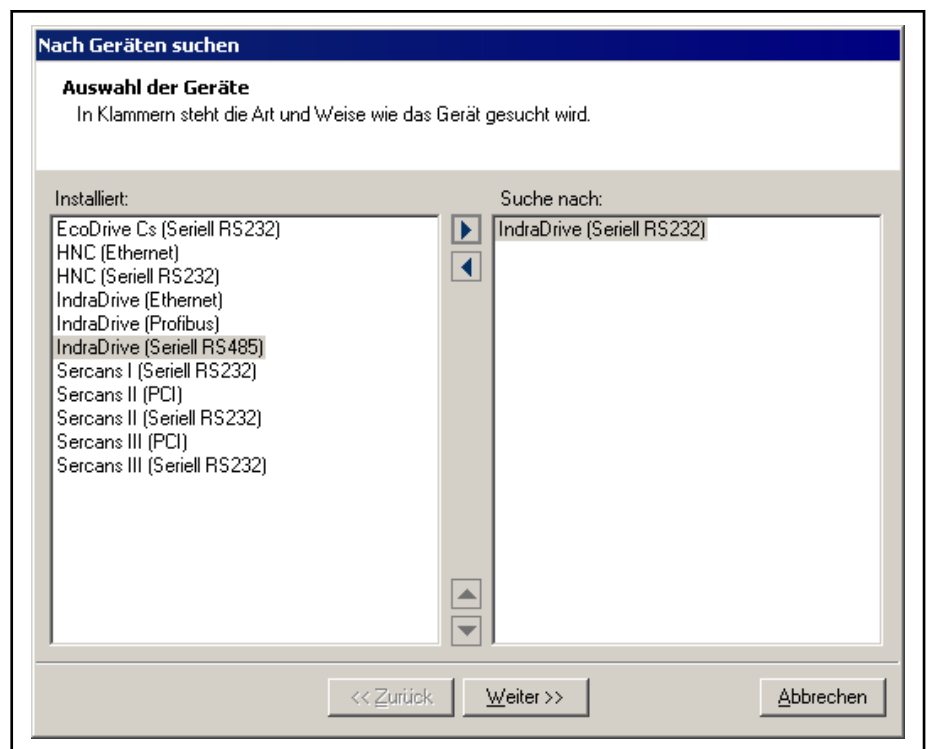


Abb. 7-73: IndraWorks Parametrierkommunikation zum Antrieb herstellen

Ist die Verbindung zum Antrieb hergestellt, können die Einstellungen vorgenommen werden.

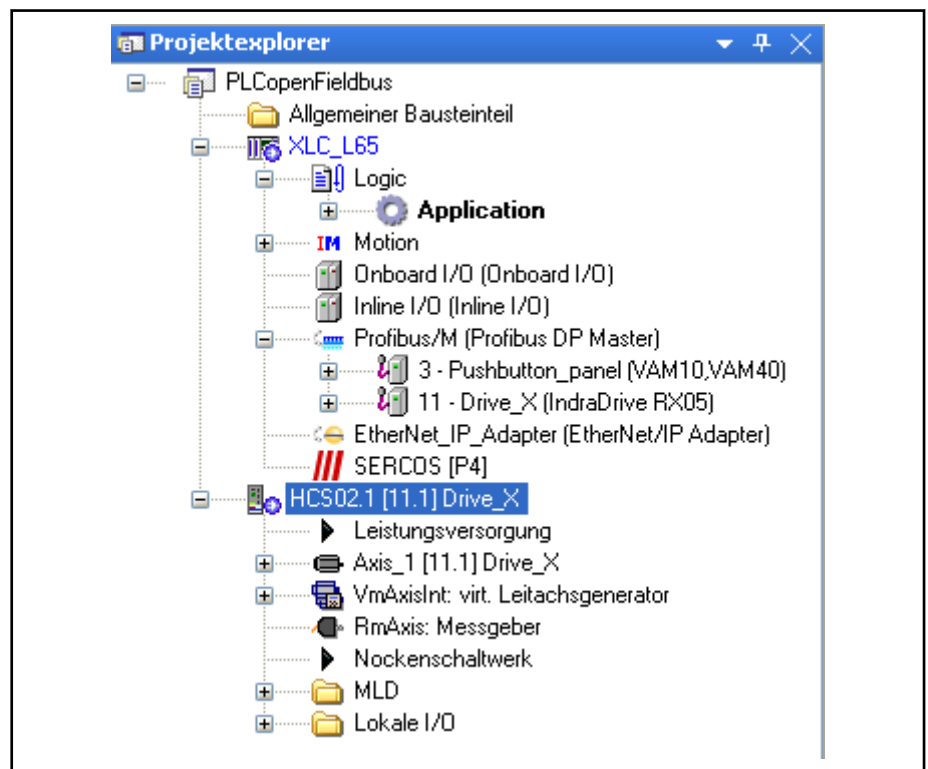


Abb. 7-74: Beispiel IW-Dialog zur Parametrierung eines IndraDrives bis MPx05

RMB_PLCopenFieldBus.library

Kommunikationseinstellungen IndraDrive

Allgemeines	Im folgenden werden die zur Kommunikation notwendigen Einstellungen eines IndraDrives beschrieben.
Einstellung der Busteilnehmeradresse	In dem Knoten "Führungskommunikation" muss die Busteilnehmeradresse des Antriebs eingestellt werden in Abhängigkeit der gesetzten Stationsadresse.
Länge des Parameterkanals	Mittels der direkten Eingabe des Parameters P-0-4083 mit dem Parameter-Editor, wird die Länge des Parameterkanals auf 10 Bytes eingestellt. Eine neuere IndraDrive-Firmware (ab MPH-03VRS) stellt sich automatisch auf 5 Worte (ParamCH_5_Worte) ein.
Einstellungen Prozessdaten und Profiltyp	Hier werden der Profiltyp "Frei konfigurierter Modus" und die Parameter des zyklischen Datenaustauschs mit der SPS eingestellt: <i>Prozessdaten-Eingänge (AT)</i> • P-0-4078 Feldbus-Statuswort • S-0-0051 Lage-Istwert Geber 1 • S-0-0040 Geschwindigkeits-Istwert • S-0-0390 Diagnose-Nummer • S-0-0144 Signal-Statuswort <i>Prozessdaten-Ausgänge (MDT)</i> • P-0-4077 Feldbus Steuerwort • S-0-0282 Positioniersollwert • S-0-0259 Positioniergeschwindigkeit • S-0-0260 Positionierbeschleunigung • S-0-0359 Positionierverzögerung • S-0-0145 Signal-Steuerwort
Signalsteuerwort	Bit 0: S-0-0346: Positioniersteuerwort, Bitnummer 4
Signalstatuswort	Bit 0: S-0-0343: Meldung Interpolator angehalten, Bitnummer 0
Betriebsarteneinstellungen	Als Hauptbetriebsart muss der Betriebsmodus "Antriebsgeführtes Positionieren" eingestellt werden. Als erste Nebenbetriebsart muss "Geschwindigkeitsregelung" zugewiesen werden. Die zweite und folgende Nebenbetriebsarten sind nicht relevant; diese werden von den PLCopen-Bausteinen nicht angewählt.



Der Funktionsbaustein MB_Home führt das Referenzieren nur aus, wenn der Antrieb für das Referenzieren parametrier ist.

7.7.5 Einbinden der notwendigen Bibliotheken

Das PLCopen-Konzept wurde für IndraWorks Engineering für zwei Feldbus-typen realisiert und durch nachstehende Bibliotheken unterstützt:

sercos II und sercos III

- ML_PLCopen.library, Funktionsbausteine der PLCopen für sercos Antriebe.
- RIL_CommonTypes.library, Datentypen systemübergreifend, Adressierung, Fehlermanagement.
- ML_Base_xxx_yyy.library⁸⁾, ergänzende Datentypen zum System ML*,

- ergänzende Bibliotheken z.B. RIL_Uilities.library.

PROFIBUS DP

- RMB_PLCopenFieldBus.library, Funktionsbausteine und Datentypen der PLCopen für PROFIBUS-Antriebe.
- RIL_CommonTypes.library, Fehlermanagement.
- ergänzende Bibliotheken z.B., RIL_Uilities.library.

Es wurde versucht, das Konzept für beide Feldbustypen weitgehend einheitlich zu halten. Dies betrifft gleiche Namen für funktionell gleiche Funktionsbausteine und Datentypen.

Mit Ausnahme der RMB_PLCopenFieldBus.library gehören die übrigen Bibliotheken zur Standard-Installation des IndraWorks Engineering. Sie werden bei der Installation mit installiert und benötigen **keine** Angabe zum Namensraum.

RMB_PLCopenFieldBus.Library

Die RMB_PLCopenFieldBus.library wird standardmäßig nicht mit installiert.

Wird sie benötigt, ist sie über den Bibliotheksverwalter, **Bibliothek hinzufügen...** nachzuinstallieren.

Der Menüpunkt öffnet ein Auswahlfenster, in dem unter **IndraLogic ► Base** die gewünschte Bibliothek zu finden ist.

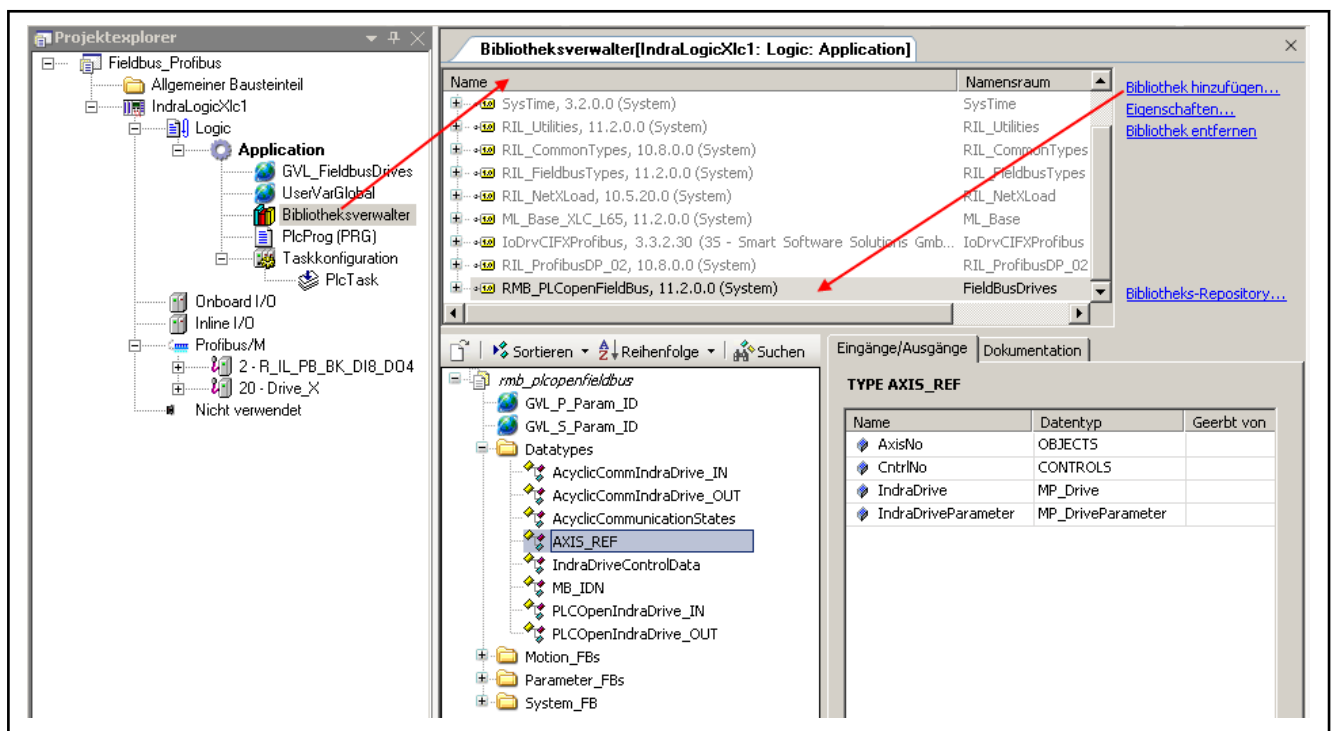


Abb. 7-75: Beispiel Bibliotheksverwalter

Die RMB_PLCopenFieldBus.library ist auf allen IndraWorks-Zielsystemen lauffähig.

Die Elemente der RMB_PLCopenFieldBus.library sind grundsätzlich mit einer Angabe des Namensraumes "FieldBusDrives." zu verwenden.

Sollen Feldbusachsen und sercos Achsen gemeinsam betrieben werden, müssen die beiden Bibliotheken (RMB_PLCopenFieldBus.library und ML_PLCopen.library) im Projekt vorhanden sein. Mit Angabe des Namens-

⁸⁾ xxx: MLC oder XLC; yyy: L25, L45; L65, VEP

RMB_PLCopenFieldBus.library

Namensraumangabe "FieldBusDrives."

raums werden dann die Funktionsbausteine und die Achs-Referenz (AXIS_REF) unterschieden.

Die Namensraumangabe "FieldBusDrives." muss bei einem Feldbusantrieb zwingend vorangestellt werden.

Die nachfolgende Abbildung zeigt die Verwendung des Namenraums.

Die Deklaration einer Achsreferenz (AXIS_REF) unter Angabe des Namenraums ist immer notwendig.

Die Angabe des Namenraums bei der Deklaration eines PLCopen-Funktionsbausteins ist nur notwendig, wenn die Bibliothek (ML_PLCopen) mit PLCopen-Funktionsbausteinen im Projekt vorhanden ist.

The screenshot displays two windows from the STEP 7 LAD editor:

- GVL_FieldBusDrives[XLC_L65: Logic: Application]**: Shows a global variable declaration.


```

1  VAR_GLOBAL
2      Drive_X : FieldBusDrives.AXIS_REF :=
3          (AxisNo:= OBJECTS.FieldBusAxis_11,
4           CntrlNo:= CONTROLS.LOCAL_CNTRL); // IndraDrive X - example
5      Drive_Y : FieldBusDrives.AXIS_REF :=
6          (AxisNo:= OBJECTS.FieldBusAxis_12,
7           CntrlNo:= CONTROLS.LOCAL_CNTRL); // IndraDrive Y - example
8  END_VAR
      
```
- PR_IndraDrive[XLC_L65: Logic: Application]**: Shows a program variable declaration and a function block call.


```

1  PROGRAM PR_IndraDrive
2  VAR
3      MCPower: FieldBusDrives.MC_Power; // Example Power ON/OFF
4      bPowerOn_q: BOOL; // PowerOn LED
5      bPowerOff_i: BOOL; // Power OFF pushbutton
6      bPowerOn_i: BOOL; // Power ON pushbutton
7      bPowerError: BOOL; // Power-FB error
8      iPowerErrorId: ERROR_CODE; // Power-FB error code
9      stPowerErrorIdent: ERROR_STRUCT; // Power-FB error ident
10 END_VAR
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
      
```

Below the code windows, a function block call for **FieldbusDrives.MC Power** is shown. The inputs are:

- Enable**: Connected to **bPowerOn_i**.
- Axis**: Connected to **GVL_FieldBusDrives.Drive_X**.
- Status**: Connected to **bPowerOn_q**.
- Error**: Connected to **bPowerError**.
- ErrorID**: Connected to **iPowerErrorId**.
- ErrorIdent**: Connected to **stPowerErrorIdent**.

- (1) AXIS_REF entstammt der RMB_PLCopenFieldBus.library, nicht der RIL_CommonTypes.library
- (2) MC_Power entstammt der RMB_PLCopenFieldBus.library, nicht der ML_PLCopen.library (Deklaration und Implementation)
- (3) ERROR_STRUCT entstammt der RIL_CommonTypes.library
- (4) Drive_X entstammt der globalen Variablenliste "GVL_FieldBusDrives"

Abb. 7-76: Beispiele für Namensraumangaben

7.7.6 Adressierung von Feldbusantrieben im SPS-Projekt

AXIS_REF, Achsadressen der Feldbusantriebe

Die Struktur `AXIS_REF` (`RIL_CommonTypes`) enthält Adress-Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt (Achsnnummer).

Im Unterschied zur `AXIS_REF`-Struktur (`RIL_CommonTypes`), die die Adressierung von SERCOS-basierten IndraDrives unterstützt, wird die `AXIS_REF`-Struktur (`RMB_PLCopenFieldbus`) durch zwei zusätzliche Elemente (`MP_Drive` und `MP_DriveParameter`) erweitert. Diese beiden Elemente werden nur durch das System genutzt.

Die Achsnnummer kann sich von der Stationsnummer, die am Antrieb eingestellt ist, unterscheiden.

Für SERCOS-basierte Achsen wird die globale Variablenliste "MLCVarGlobal" mit allen SERCOS-basierten Achsen automatisch angelegt und gepflegt.

Für die Feldbusantriebe ist eine globale Variablenliste hier "GVL_FieldBusDrives" mit allen Feldbusantrieben manuell anzulegen und zu pflegen.

Die ersten beiden Elemente jedes Antriebs "CntrlNo" und "AxisNo" werden in ihr vorbelegt. Die Elemente "MP_Drive" und "MP_DriveParameter" bleiben unverändert. Sie müssen deshalb nicht mit in die Liste aufgenommen werden.

Globale Variablenliste "GVL_FieldBusDrives", Beispiel

```
VAR GLOBAL
Drive_X: FieldBusDrives.AXIS_REF :=
    (CntrlNo:=LOCAL_CNTRL,
     AxisNo:=FieldBusAxis_11); // IndraDrive X - example
Drive_Y: FieldBusDrives.AXIS_REF :=
    (CntrlNo:=LOCAL_CNTRL,
     AxisNo:=FieldBusAxis_12); // IndraDrive Y - example
Drive_Z: FieldBusDrives.AXIS_REF :=
    (CntrlNo:=LOCAL_CNTRL,
     AxisNo:=FieldBusAxis_13); // IndraDrive Z - example
Drive_n: FieldBusDrives.AXIS_REF :=
    (CntrlNo:=LOCAL_CNTRL,
     AxisNo:=FieldBusAxis_14); // IndraDrive n - example
END_VAR
```

Namensraumangabe FieldBusDrives

Die Namensraumangabe "FieldBusDrives." muss bei einem Feldbusantrieb zwingend vorangestellt werden.

Deklaration AXIS_REF

```
Drive_A: FieldBusDrives.AXIS_REF :=
```

Die Datentypen hierfür sind in der `RMB_PLCopenFieldBus.library`, Ordner "Datatypes", Element "AXIS_REF" enthalten.

Zuweisung CntrlNo

```
(CntrlNo: = LOCAL_CNTRL
```

Der Enumerationswert `LOCAL_CNTRL` ist in der `RIL_CommonTypes.lib`, Ordner "Datatypes of logical addresses", `CONTROLS` enthalten.

Die Feldbusbausteine beziehen sich immer auf die lokale SPS, also `LOCAL_CNTRL`.

RMB_PLCopenFieldBus.library

Zuweisung AxisNo

```
AxisNo: = FieldBusAxis_1);
```

Der Enumerationswert FieldBusAxis_1 ist in der RIL_CommonTypes.lib, Ordner "Datatypes of logical addresses", OBJECTS enthalten.

Um Doppeldeutigkeiten zu vermeiden, ist hier für die Feldbusachsen ein eigener Nummernbereich

FieldBusAxis_1 (6001) bis FieldBusAxis_99 (6099)

hinterlegt.

Bei Feldbusantrieben werden die Angaben zur Steuerung und die Achsnummer zur Fehlerausgabe (ErrorIdent) verwendet.

Eine Funktionalität ist mit der Vergabe der Achsnummer nicht verbunden.



Es ist sinnvoll, PROFIBUS-Stationsnummer und Achsnummer abzugleichen...

AxisData[], Achsdaten

Das AxisData- Array, bekannt von sercos Achsen, wird bei Feldbusachsen nicht unterstützt.

7.7.7 E/A-Adressierung der Antriebe im SPS-Projekt

Allgemeines Auf dem PROFIBUS erscheinen die Feldbusantriebe wie normale E/A-Module, eine E/A-Adresszuweisung muss durchgeführt werden.

Um die E/A-Adresszuweisung so einfach wie möglich zu gestalten, wird sie als Variablenmapping direkt auf dem PROFIBUS-Modul des IndraDrives durchgeführt.

E/A-Adresszuweisung IndraDrive Die Parameterkommunikation, das F-Modul (Sicherheitstechnik) und die zyklische Kommunikation, sind bei IndraDrives jeweils eigene Submodule des Teilnehmers.

Die Kommunikation "Antrieb zur SPS" setzt sich so aus folgenden Teilen zusammen:

- Subteilnehmer (ParamCh_5_Words), 5 Worte, zur azyklischen Parameterkommunikation,
- Subteilnehmer (F-Modul), Verwendung optionalen, hier nicht genutzt,
- Subteilnehmern des zyklischen Datenaustauschs (8 Worte Input, bzw. 10 Worte Output)

Das Mapping der Adressen wird am jeweiligen Subteilnehmer-Kanäle des Antriebes durchgeführt.

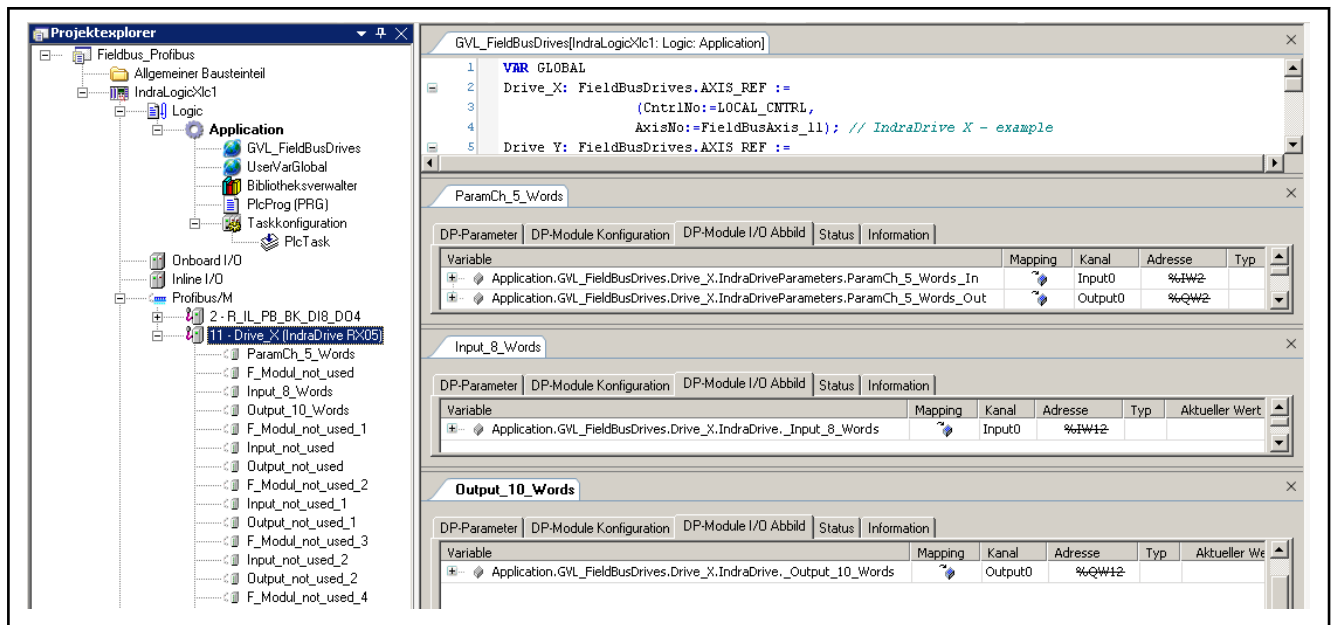


Abb. 7-77: Beispiel: I/O-Mapping für einen IndraDrive-Feldbusantrieb

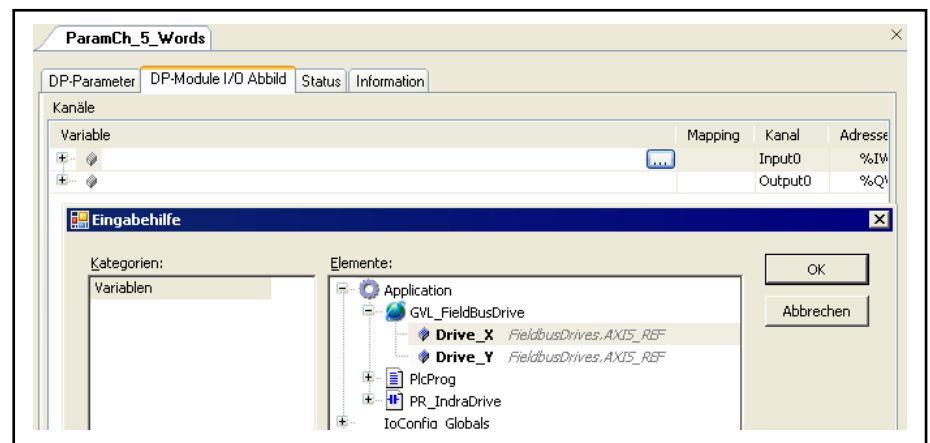


Abb. 7-78: Beispiel: Eingabebeihilfe für das Mapping zur Auswahl des Antriebs

Im AXIS_REF der Feldbusantriebe wird unterschieden nach "IndraDrive" für den zyklischen Parameterraustausch und "IndraDriveParameter" für den azyklischen Parameterraustausch

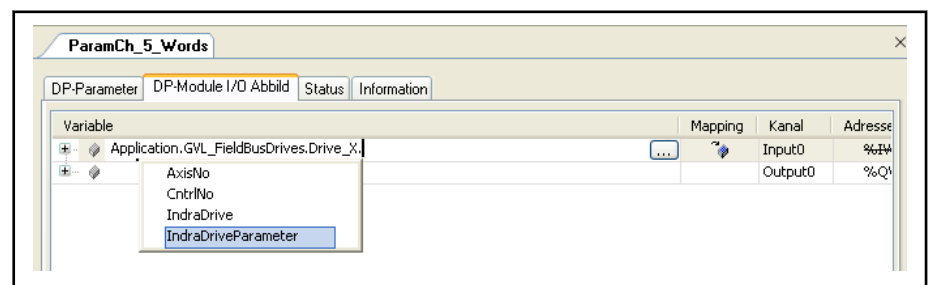


Abb. 7-79: Beispiel: Mapping für den Parameterraustausch

RMB_PLCopenFieldBus.library

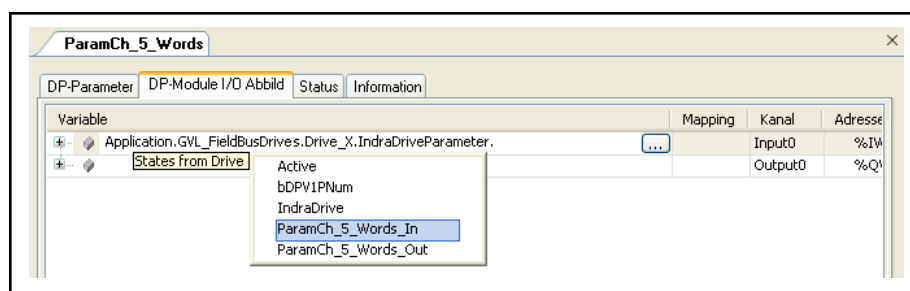


Abb. 7-80: Beispiel: Mapping für die Eingänge des Parameteraustauschs

Durch das direkte Mapping brauchen die Anfangsadressen für die Parameterkommunikation, des zyklischen Datenaustausches und eines ggf. vorhandenen F-Moduls in Bezug auf den Antrieb nicht berücksichtigt zu werden.

8 ML_ControllerBase.library

8.1 Einleitung

Die Bibliothek bietet die Möglichkeit auf der Basis vorgefertigter Module einen Regler in der SPS zu implementieren.

Durch die Nutzung der Bibliothek wird nicht nur die Implementierung vereinfacht, sondern auch das Verhalten der Regler vereinheitlicht. Die Bibliothek ist zweigeteilt.

- [Basisklasse des Reglers, Seite 433](#), Funktionalität des Reglers,
- [Reglerverwaltung, Seite 437](#), Verhalten des Reglers.

8.2 Basisklasse des Reglers

8.2.1 Basisklasse

Die Basisklasse aller Regler in der SPS heißt "ML_ControllerBase".

Von dieser Klasse können eigene Regler abgeleitet werden.

Damit die Reglerverwaltung den eigenen Regler aufrufen kann, muss dieser einige vorgegebene Methoden realisieren. Die unten aufgeführten Interfaces bilden eine Sammlung dieser Methoden ab und sind nach Funktionalität gruppiert.

Abhängig vom jeweiligen Systemzustand der Steuerung ruft die Reglerverwaltung die Interface-Methoden des eigenen Reglers auf, um den Zustand auch im Regler einzustellen. Beispiele hierfür sind das Phasenschalten, Betriebszustände (bb, AB, AH...) oder Regelzustände (Position, Geschwindigkeit, Kraft...).

IActualValue, IModuleControl, IOperationModeHandler, IOutputData mit ihren gekapselten Methoden müssen im eigenem Regler realisiert werden.

IRetainHandler, IDigitalIO, IAdditionalSetValue sind optional.

Interface-Name	Funktion
IActualValue	Istwerterfassung
IModuleControl	Initialisierung des Reglers Phasenschalten
IOperationModeHandler	Betriebsarten eines Reglers siehe: Reglerverwaltung, Seite 437
IOutputData	Sollwertgenerierung des Reglers

Tab. 8-1: Implementierbare Interfaces eines Reglers in der SPS

Implementiert ein Regler ein Interface, muss die entsprechende Get-Methode aus der Basisklasse überladen werden.

In der Implementierung muss zurückgegeben werden, welcher Funktionsbaustein dieses Interface implementiert.

Implementiert die abgeleitete Klasse von "ML_ControllerBase" die Methode selbst, so muss `THIS^` zurückgegeben werden.

Zugriffsmethoden auf die Interfaces

Methodenname	Implementiertes Interface
getIActualValue	IActualValue

ML_ControllerBase.library

getModuleControl	IModuleControl
getOperationModeHandler	IOperationModeHandler
getOutputData	IOutputData
getRetainHandler	IRetainHandler
getDigitalIO	IDigitalIO
getAdditionalSetValue	IAdditionalSetValue

Tab. 8-2: Zugriffsmethoden auf die Interfaces

Implementiert die abgeleitete Klasse von "ML_ControllerBase" die Methode selbst, so muss `THIS^` zurückgegeben werden.

Programm:

```

METHOD getIActualValue : IActualValue
VAR_INPUT
END_VAR
VAR
END_VAR

getIActualValue := THIS^;

```

8.2.2 Daten eines Reglers

Alle Regler besitzen Daten, die in jedem Regler gleich sind. Zu diesen Daten gehören z.B. Ist- und Sollwerte.

Um diese Daten nicht in jedem Regler neu implementieren zu müssen, wurden diese Daten in einer Struktur zusammengefasst, die jeder Regler nutzt.

Jeder Methoden der einzelnen Interfaces wird ein Zeiger auf diese Struktur übergeben, so dass die Methoden auf diese Daten zugreifen können. Im folgenden wird eine Übersicht über die wichtigsten Elemente in dieser Struktur gegeben.

Elemente der Struktur ML_OORV_CONTROLLERDATA

Elementname	Typ	Bedeutung
stControlAxisRef	AXIS_REF	Achsreferenz auf die Reglerachse des Reglers
IrActualPos	LREAL	Istposition des Reglers
IrActualVel	LREAL	Istgeschwindigkeit des Reglers
IrActualAcceleration	LREAL	Istbeschleunigung des Reglers
IrActualForce	LREAL	Istkraft des Reglers
IrTargetPos	LREAL	Zyklischer Lagesollwert des Reglers
IrTargetVel	LREAL	Zyklischer Geschwindigkeitssollwert des Reglers
IrTargetAcceleration	LREAL	Zyklischer Beschleunigungssollwert des Reglers
IrTargetForce	LREAL	Zyklischer Kraftsollwert des Reglers
fbConverterCAxis	ML_Convert	Konvertierungsfunktionen zum Umrechnen zwischen internen und externen Einheiten (4 Byte)

fbConverterCA-axis_DoublePrecision	ML_Convert_DoublePrecision	Konvertierungsfunktionen zum Umrechnen zwischen internen und externen Einheiten (8 Byte)
IrCycleTime	LREAL	Zykluszeit des Motion-Kernels in μs . In dieser Zeit werden neue Sollwerte generiert.

Tab. 8-3: Zyklische Ist- und Soll-Werte

Die internen Einheiten des Reglers wurden festgelegt. Das bedeutet, dass innerhalb des Reglers alle Daten die entsprechende Einheit besitzen.

Typ	rotatorische Wichtung	translatorische Wichtung
Lage	grad	mm
Geschwindigkeit	grad/s	mm/s
Beschleunigung	grad/s ²	mm/s ²
Kraft	Nm	KN
Analogwert	%	%

Tab. 8-4: Einheiten der Daten

8.2.3 Umrechnung der Daten eines Reglers

An den Schnittstellen zum Anwender wird von den Einheiten des Anwenders in die internen Einheiten umgerechnet. Hierfür stehen die zwei Funktionsbausteine "ML_Convert" und "ML_Convert_DoublePrecision" zur Verfügung. Die bereitgestellten Methoden zur Konvertierung sind:

ML_ControllerBase.library

Umrechnungsmethoden zwischen interner und externer Wichtung

Methode	Bedeutung
convFromAxis_Position	Rechnet einen externen Positionswert in einen internen Positionswert um
convFromAxis_Velocity	Rechnet einen externen Geschwindigkeitswert in einen internen Geschwindigkeitswert um
convFromAxis_Acceleration	Rechnet einen externen Beschleunigungswert in einen internen Beschleunigungswert um
convFromAxis_Force	Rechnet einen externen Kraftwert in einen internen Kraftwert um
convFromAxis_Analog	Rechnet einen externen Analogwert in einen internen Analogwert um
convToAxis_Position	Rechnet einen internen Positionswert in einen externen Positionswert um
convToAxis_Velocity	Rechnet einen internen Geschwindigkeitswert in einen externen Geschwindigkeitswert um
convToAxis_Acceleration	Rechnet einen internen Beschleunigungswert in einen externen Beschleunigungswert um
convToAxis_Force	Rechnet einen internen Kraftwert in einen externen Kraftwert um
convToAxis_Analog	Rechnet einen internen Analogwert in einen externen Analogwert um

Tab. 8-5: Konvertierungsfunktionen zwischen interner und externer Wichtung



Beide Funktionsbausteine unterscheiden sich hinsichtlich ihres Rückgabewertes in der darstellbaren Genauigkeit. Der Baustein "ML_Convert" liefert als Ergebnis einen 4-Bytewert, während der Baustein "ML_Convert_DoublePrecision" ein Ergebnis mit 8-Byte Datenbreite liefert. Werden in die Bausteine Werte eingelesen, die von deren jeweiliger Genauigkeit abweichen, so führt dies zu Ungenauigkeiten (Rundungsfehler) im konvertierten Wert.

8.2.4 Interface IActualValue

Dieses Interface enthält die Methode für die Istwerterfassung.

runActualValue Die Methode `runActualValue` wird für die Istwerterfassung des Reglers aufgerufen. Die Aufgabe dieser Methode ist es die Istwerte der Achse von den Sensoren zu bestimmen. Diese Werte müssen in den Daten des Reglers im Element `lrActualPos` (Positionswert) bzw. `lrActualForce` (Kraftwert) eingetragen werden.

8.2.5 Interface IModuleControl

Dieses obligatorische Interface beinhaltet administrative Methoden.

InitModule Diese Methode `InitModule` wird zum Start des gesamten Reglers aufgerufen. In dieser Methode können Initialisierungen des Reglers vorgenommen werden. Die Methode wird solange aufgerufen, bis die Methode `TRUE` zurück liefert

SwitchToParamMode Diese Methode `SwitchToParamMode` wird beim Schalten in den Parametriermodus aufgerufen. Die Methode wird solange aufgerufen, bis die Methode `TRUE` zurück liefert

SwitchToOperationMode Diese Methode `SwitchToOperationMode` wird beim Schalten in den Betriebsmodus aufgerufen. Die Methode wird solange aufgerufen, bis die Methode `TRUE` zurück liefert

8.2.6 Interface IOperationModeHandler

Dieses Interface ist die Schnittstelle zu den Betriebsarten des Reglers. Die Betriebsarten werden von der Reglerverwaltung vorgegeben. Folgende Betriebsarten sind vorgesehen.

bInitOperationCard Diese Methode `bInitOperationCard` wird aufgerufen, wenn in eine neue Betriebsart gewechselt werden soll. Die Nummer der neuen Betriebsart steht in den Reglerdaten im Element `byActOpCardIndex`.

bCyclic Diese Methode `bCyclic` wird in jedem Zyklus aufgerufen. Die Nummer der Betriebsart steht in den Reglerdaten im Element `byActOpCardIndex`. In dieser Methode wird der Regelalgorithmus für diese Betriebsart implementiert.

8.2.7 Interface IOutputData

Dieses Interface enthält die Methode für die Stellgrößenausgabe an den Aktor.

runOutputData Die Methode `runOutputData` wird in jedem Zyklus aufgerufen. In dieser Methode wird die Ausgabe der Stellgrößen an den Aktor implementiert.

8.3 Reglerverwaltung

8.3.1 Verbinden des Reglers mit der Reglerverwaltung

Die Reglerverwaltung bildet die Laufzeitumgebung eines Reglers.

Um den Regler mit der Reglerverwaltung zu verbinden, müssen die Instanz eines Reglerfunktionsbausteins mit dem Eingang `aritfModuleInterface` einer Instanz des Funktionsbausteins `ML_OORV` verbunden werden.

Der Aufruf der Instanz des Funktionsbausteins muss in einem Programm erfolgen, das mit einer Task verbunden ist, die die Zykluszeit des Reglers definiert.

Als Übergabeparameter muss die Achsreferenz der Reglerachse an den Eingang `AXIS` angelegt werden, für die der Regler arbeiten soll.

`MY_CONTROLLER_FB` ist ein implementierter Regler, der von `ML_ControllerBase` abgeleitet ist.

Programm:

```

Program MotionProg
VAR
  myController: MY_CONTROLLER_FB;
  myAdministration: ML_OORV := (aritfModuleInterface := myController);
END_VAR

myAdministration(AXIS := ControlAxis);

```

8.3.2 Zyklischer Ablauf der Reglerverwaltung

In jedem Zyklus der Reglerverwaltung findet folgender Ablauf nacheinander statt:

ML_ControllerBase.library

Erfassen der Steuerinformation	Die Steuerinformationen der Reglerachse werden in den Datenbestand des Reglers eingelesen. Diese Informationen bestimmen im weiteren zyklischen Verlauf das Verhalten.
Zustandsmaschine Betriebszustand	Die Zustandsmaschine des Betriebszustands, Seite 438 , wird durchlaufen.
Schreiben von Statusinformationen	Statusinformationen werden vom Regler in die Achse geschrieben. Folgende Werte werden umgerechnet und synchronisiert:

Element in der Struktur der Reglerdaten	Parameter in der Reglerachse
lrActualPos	A-0-0100, Positionswert
lrActualVel	A-0-0102, Geschwindigkeitswert
lrActualTorque	A-0-0110, Drehmoment/Kraftwert
wCStatusWord	A-0-0019, Reglerstatuswort

Tab. 8-6: Synchronisierte Statusinformationen

8.3.3 Zustandsmaschine Betriebszustand

In dieser Zustandsmaschine wird die erstmalige Initialisierung des Reglers und das Phasenschalten realisiert. In dieser Zustandsmaschine werden Methoden des Interfaces `IModuleControl` verwendet. Die folgende Tabelle zeigt die Zustände und die Aufrufe der Interface-Methoden.

Zustand	Bedeutung	Übergang	Aufruf der Interfacemethode
IModuleControl	Initialzustand	Erfolgreiches Lesen aller Interfaces und erfolgreiche Initialisierung des Reglers	getModuleControl getActualValue getOutputData getOperationModeHandler InitModule
PreParamState	Warten auf Parametriermodus	ParamState, wenn Anwender in den Parametriermodus schaltet	SwitchToParamMode
ParamState	Warten auf Betriebsmodus	OperationState, wenn Anwender in den Betriebsmodus schaltet	SwitchToOperationMode
OperationState	Betriebszustand In diesem Zustand wird der zyklische Ablauf im Regler durchgeführt.	PreParamState, wenn der Betriebsmodus verlassen wird	siehe: Zyklischer Ablauf des Reglers, Seite 438

Tab. 8-7: Zustandsmaschine Betriebszustand

8.3.4 Zyklischer Ablauf des Reglers

Ist der Betriebszustand erreicht, findet in jedem Takt folgender Ablauf statt.

Istwerterfassung	Es wird über die Methode <code>runActualValue</code> des Interfaces <code>IActualValue</code> des Reglers ausgeführt.
Betriebsartenhandling	Es wird die aktuelle ausgewählte Betriebsart, die von der Steuerung vorgegeben wird, ausgewertet und die entsprechende Betriebsart des Reglers ausgeführt. Siehe Betriebsartenhandling, Seite 439 .
Stellwertausgabe	Es wird über die Methode <code>runOutputData</code> des Interfaces <code>IOutputData</code> des Reglers ausgeführt.

8.3.5 Betriebsartenhandling

Ist ein Regler im Betriebsmodus, wirkt die Zustandsmaschine der Betriebsarten. In ihr sind die Betriebsartenzustände und die Übergänge definiert.

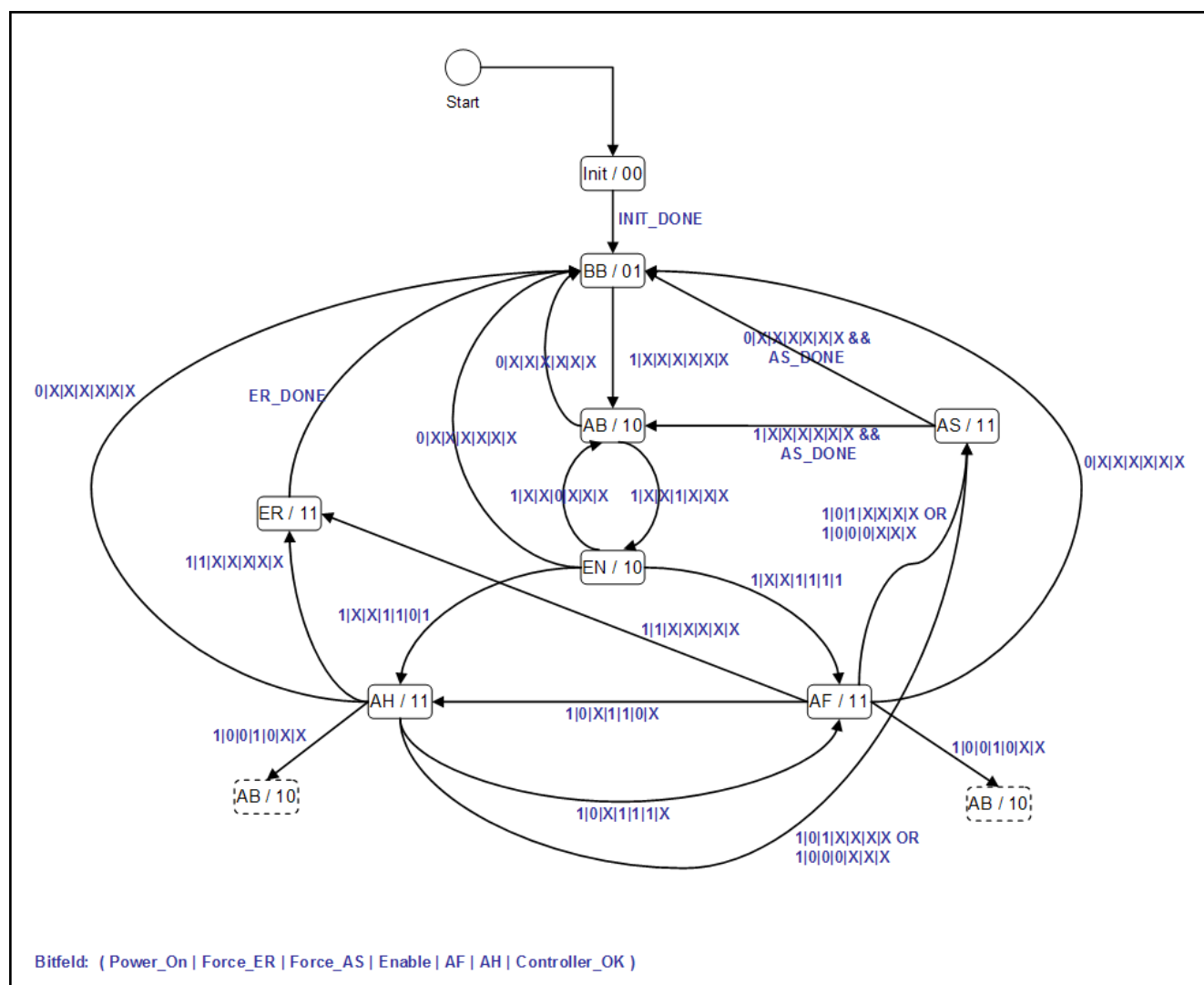


Abb. 8-1: Zustandsmaschine der Betriebsarten

Übergänge von einem Zustand zum anderen werden vom Anwender über Motion-Funktionsbausteine und vom Regler selbst ausgeführt.

Für die Übergänge sind jeweils die Bedingungen für die jeweiligen Steuerbits angegeben. Falls ein Bit für den Zustandsübergang ignoriert wird, steht in der entsprechenden Spalte ein x.

Die folgende Tabelle zeigt die Bedeutung der Steuerbits und wer diese Steuerbits bedient.

Steuerbit	Bedeutung	Handling des Bits
Power_On	Status Leistungszuschaltung	Dieses Bit wird vom Regler vorgegeben. Es zeigt an, ob alle Bedingungen erfüllt sind, die Leistung zuzuschalten.
Force_ER	Fehler erzwingen	Diese Bit wird vom Regler gesetzt, um anzuzeigen, dass ein Fehler ansteht, und um der Übergang zum Fehlerzustand zu erzwingen.

ML_ControllerBase.library

Force_AS	Antrieb stop erzwingen	Dieses Bit wird vom Regler gesetzt, um einen Antrieb Stop zu erzwingen.
Enable	Leistungszuschaltung	Dieses Bit wird vom Anwender gesetzt. Der Regler soll die Leistung zuschalten(z.B. Spannungsversorgung des Motors oder einer Pumpe) .
AF	Reglerfreigabe	Diese Bit wird vom Anwender gesetzt. Die Reglerfreigabe wird erteilt.
AH	Antrieb Halt	Diese Bit wird vom Anwender gesetzt. Der Regler soll den Antrieb anhalten und regeln.
Controller_OK	Regler bereit zur Regelung.	Dieses Bit wird vom Regler gesetzt, wenn der Regler bereit zur Regelung ist.
AS_DONE	Antrieb Stop ausgeführt. Antrieb steht.	Dieses Bit wird vom Regler gesetzt, wenn die Betriebsart K_BAK_IND_INT_AS angewählt wurde und der Antrieb "Stillstand" erreicht hat.
ER_DONE	Fehlerreaktion ausgeführt.	Dieses Bit wird vom Regler gesetzt, wenn die Betriebsart K_BAK_IND_INT_ER angewählt wurde und der Antrieb die Fehlerreaktion ausgeführt hat.
INIT_DONE	Initialisierung abgeschlossen	Dieses Bit wird vom Regler gesetzt, wenn die Initialisierung abgeschlossen ist.

Tab. 8-8: Bedeutung der Steuerbits der Betriebsartenumschaltung

In jedem Zustand wird eine Betriebsart des Reglers aufgerufen.

Folgende Tabelle zeigt, in welchem Zustand welche Betriebsart aufgerufen wird.

Zustand	Betriebsart
INIT	K_BAK_IND_INT_NO_RF
BB	K_BAK_IND_INT_NO_RF
AB	K_BAK_IND_INT_RF_INIT
EN	K_BAK_IND_INT_RF_CHECK
AH	K_BAK_IND_INT_AH
ER	K_BAK_IND_INT_ER
AS	K_BAK_IND_INT_AS
AF	Vom Anwender wählbare Betriebsarten: K_OPCARD_PRIM, K_OPCARD_SCND_1, K_OPCARD_SCND_2, K_OPCARD_SCND_3

Tab. 8-9: Betriebsartenaufruf im Zustand...

Im Zustand AF wird die "Kommandierte Betriebsart" vom Anwender aufgerufen. In alle anderen Zuständen wird eine feste Betriebsarten mit einer festgelegten Aufgabe aufgerufen.

Diese Aufgabe wird im folgenden erklärt:

Betriebsart	Bedeutung
K_BAK_IND_INT_AH	Antrieb Halt: Bestmögliche Stillsetzung. Bei Anwahl dieser Betriebsart soll der Regler die Achse bestmöglich anhalten und weiter regeln.
K_BAK_IND_INT_ER	Error: Diese Betriebsart wird angewählt, wenn die Achse in den Fehlerzustand überführt wird. Der Regler soll die Fehlerreaktion ausführen.
K_BAK_IND_INT_AS	Antrieb Stop: Der Regler soll die Achse anhalten. Es muss das Steuerbit AS_DONE bedient werden, um zu signalisieren, dass der Antrieb stillstand erreicht hat.
K_BAK_IND_INT_RF_CHECK	Reglerfreigabe Check: Der Regler soll überprüfen, ob die Freigabe des Reglers erteilt werden darf d.h. ob alle Bedingungen erfüllt sind, den Regelkreis zu schließen.
K_BAK_IND_INT_RF_INIT	Der Regler hat keine Freigabe. Es darf keine Regelung stattfinden. In dieser Betriebsart muss der Regler überprüfen, ob die Bedingungen zur Leistungszuschaltung gegeben sind. Entsprechend muss das Steuerbit Power_On bedient werden.
K_BAK_IND_INT_NO_RF	Der Regler hat keine Freigabe. Es darf keine Regelung stattfinden. In dieser Betriebsart muss der Regler überprüfen, ob die Bedingungen zur Leistungszuschaltung gegeben sind. Entsprechend muss das Steuerbit Power_On bedient werden.
K_OPCARD_PRIM	Primäre Betriebsart: Der Regelkreis ist geschlossen. Der Regler befindet sich in zyklischer Lageregelung.
K_OPCARD_SCND_1	1. Nebenbetriebsart: Der Regelkreis ist geschlossen. Der Regler befindet sich in zyklischer Geschwindigkeitsregelung.
K_OPCARD_SCND_2	2. Nebenbetriebsart: Der Regelkreis ist geschlossen. Der Regler befindet sich in zyklischer Kraftregelung.
K_OPCARD_SCND_3	3. Nebenbetriebsart: Der Regelkreis ist geschlossen. Der Regler befindet sich in zyklischer Analogwertregelung.

Tab. 8-10: Betriebsarten eines Reglers

8.3.6 Vordefinierte Generatoren der Betriebsarten

In den Betriebsarten stehen vordefinierte Generatoren zur Verfügung. Diese haben die Aufgabe die Sollwerte des Reglers zu ermitteln und in die Datenstruktur des Reglers `ML_OORV_CONTROLLERDATA` (siehe [ML_OORV_CONTROLLERDATA](#), Seite 434) zu kopieren.

Dazu stellt der Funktionsbaustein "ML_Generator" folgende Methoden zur Verfügung.

GenerateCopyActualPos	Diese Methode kopiert die aktuelle Position des Reglers in den Positionssollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .
GenerateCopyActualVel	Diese Methode kopiert die aktuelle Geschwindigkeit des Reglers in den Geschwindigkeitssollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .
GenerateCopyActualAcc	Diese Methode kopiert die aktuelle Beschleunigung des Reglers in den Beschleunigungssollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .
GenerateCopyActualTrq	Diese Methode kopiert die aktuelle Kraft des Reglers in den Kraftsollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .
GenerateCopyActualAna	Diese Methode kopiert die aktuelle Analogwert des Reglers in den Analogsollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .
GenerateCopyCyclicPosition	Diese Methode kopiert die zyklische Sollposition des Interpolators der Steuerung in den Positionssollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .

ML_ControllerBase.library

GenerateCopyCyclicVelocity	Diese Methode kopiert die zyklische Geschwindigkeitssollwert des Interpolators der Steuerung in den Geschwindigkeitssollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .
GenerateCopyCyclicAcceleration	Diese Methode kopiert die zyklische Beschleunigungssollwert des Interpolators der Steuerung in den Beschleunigungssollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .
GenerateCopyCyclicTorque	Diese Methode kopiert die zyklische Kraftsollwert des Interpolators der Steuerung in den Kraftsollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .
GenerateCopyCyclicAnalog	Diese Methode kopiert die zyklische Anlogsollwert des Interpolators der Steuerung in den Anlogsollwert der Datenstruktur <code>ML_OORV_CONTROLLERDATA</code> .

8.3.7 Diagnosehandling

Die Reglerverwaltung bietet die Möglichkeit Meldungen, Warnungen und Fehler abzusetzen. Diese werden automatisch im Diagnoselogbuch protokolliert. Für das Handling steht in der Datenstruktur des Reglers `ML_OORV_CONTROLLERDATA` (siehe [ML_OORV_CONTROLLERDATA](#), Seite 434) der Funktionsbaustein `ML_MessageHandler` zur Verfügung.

Folgende Methoden stehen zur Verfügung.

run_MSGH_SetMessagecode Über diese Methode können Diagnosen abgesetzt werden. Diagnosen bestehen aus vier Byte und sind hexadezimal kodiert. Das oberste Nibbel legt die Kategorie (severity) fest:

Diagnosecode	Bedeutung
Axxxxxx	Diagnose ist eine Meldung
Exxxxxx	Diagnose ist eine Warnung
Fxxxxxx	Diagnose ist ein Fehler

Tab. 8-11: Codierung der Diagnose

Diagnosen werden vom System priorisiert:

- Steht keine Warnung oder kein Fehler an, so wird die letzte Meldung ausgegeben.
- Sobald eine Warnung abgesetzt wird und kein Fehler ansteht, wird die Warnung ausgegeben.
- Sobald ein Fehler abgesetzt wird und der Fehler eine höhere Priorität als alle vorher abgesetzten Fehler hat, wird dieser ausgegeben.

Die Priorität (class) ist im zweiten Nibbel eines Fehlers codiert.

run_MSGH_ClearWarning Mit dieser Methode kann eine Warnung gelöscht werden, die vorher abgesetzt wurde. Steht kein Fehler an, so wird die letzte Meldung ausgegeben.

run_MSGH_ClearErrors Mit dieser Methode können alle anstehenden Fehler und Warnungen gelöscht werden. Diese Methode wird von der Reglerverwaltung automatisch ausgeführt, wenn der Anwender ein "Fehler löschen"-Kommando startet.

9 ML_IPOImpactControl.library

9.1 Einführung und Übersicht

Ziel Für die Realisierung der ablösenden Regelung wird eine Funktionalität bereitgestellt, die einen direkten Eingriff in die Interpolation der MLC-Steuerung ermöglicht. Ziel ist es, die Sollwertvorgabe des Interpolators zu manipulieren, um ein vom SPS-Regler gewünschtes Verhalten einstellbar zu machen.

Technisches Konzept Im Normalfall wird bei der Ausführung eines Bewegungsbefehls vom Interpolator ein neuer Sollwert zur Verfügung gestellt. Diese Sollwertvorgabe erfolgt in jedem Motion-Takt. Das Konzept zur Manipulation dieser Vorgabe sieht vor, dem eigentlichen Aufruf in die Interpolation einen Aufruf in eine zusätzliche Zustandsmaschine voranzustellen. Diese Zustandsmaschine monitort Parameter, mit denen der SPS-Regler ein gewünschtes Verhalten einstellen kann und beeinflusst damit die eigentliche Interpolation.

Man kann sich folgendes vereinfachtes Beispiel vorstellen:

Um die Sollwertgenerierung auf einen bestehenden Wert „einzufrieren“, könnte die vorangestellte Zustandsmaschine den Interpolator auffordern, keine weiteren Berechnungen durchzuführen. Damit würde der Interpolator ständig den letzten bekannten Wert liefern. Dies wiederum hätte zur Folge, dass, obwohl der eigentliche Bewegungsbefehl noch aktiv ist, die Bewegung steht. Das SPS-Programm erkennt diesen Zustand aber nicht, da weder das Done- noch das CommandAborted-Flag gesetzt werden.

Das Konzept sieht folgende Manipulationsmöglichkeiten vor:

Manipulationsmöglichkeiten

Eingriff	Kommentar
Richtungsabhängiges Anhalten der Interpolation	Dieser Eingriff wird auch als "Break" bezeichnet.
Richtungsabhängiges Springen in den Interpolationsendwert.	Dieser Eingriff wird auch als "Jump" bezeichnet.
Setzen von individuellen Positions- und Geschwindigkeitswerten im Zustand "Break" oder am Ende eines Bewegungsbefehls.	Diese Funktionalität wird benötigt, um Werte aus der SPS zurück in das aktive Bewegungskommando zu schreiben.

Der Übergang in den Zustand "Break" kann mit unterschiedlichen Beschleunigungswerten geschehen.



Die Manipulation wirkt nur auf folgende Befehle:

- MC_MoveAbsolute,
- MC_MoveAdditive,
- MC_MoveRelative und
- MC_MoveVelocity.

Die unten stehende Grafik zeigt die interpolierten Werte für Position, Geschwindigkeit und Beschleunigung während eines MC_MoveAbsolute-Befehls und den Eingriff in die Interpolation bei 100 Grad.

Zwischen (1) und (2) ist die Interpolation gestoppt (Zustand "Break"). Es werden keine neuen Sollwerte generiert.

ML_IPOImpactControl.library

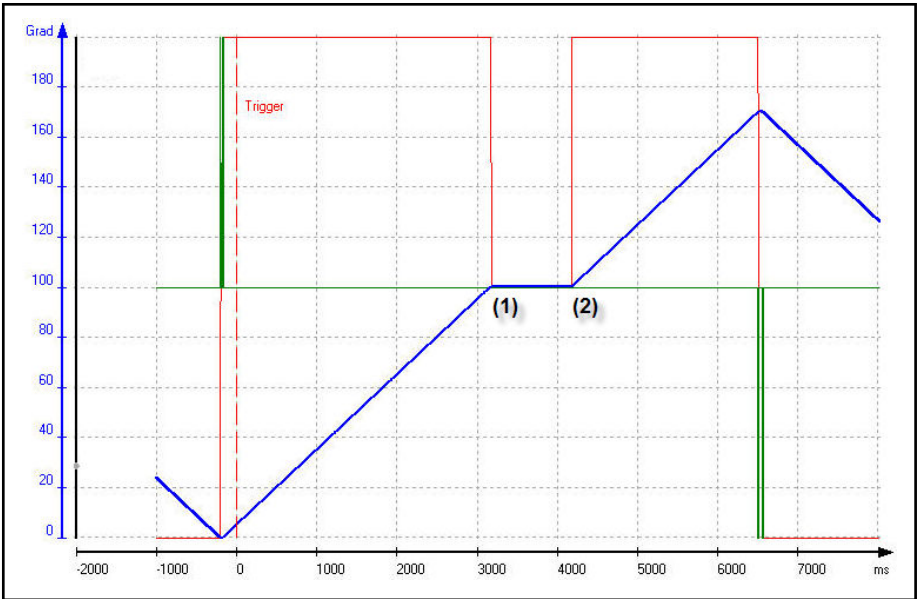


Abb. 9-1: blau = interpolierte Position, rot = interpolierte Geschwindigkeit, grün = interpolierte Beschleunigung

9.2 ML_IPOImpactControl

Kurzbeschreibung Der FB ML_IPOImpactControl und seine Methoden dienen dazu Manipulationen an der Interpolation vorzunehmen.
Der Funktionsbaustein muss zyklisch aufgerufen werden, da er zusätzlich den Fehlerzustand der Achse überwacht. Sollte die Achse einen Fehler melden, werden alle eingestellten Manipulationen automatisch zurückgenommen.

Bibliothek	Bereich
ML_IPOImpactControl	Motion

Tab. 9-1: Bibliothekszuordnung

Schnittstelle

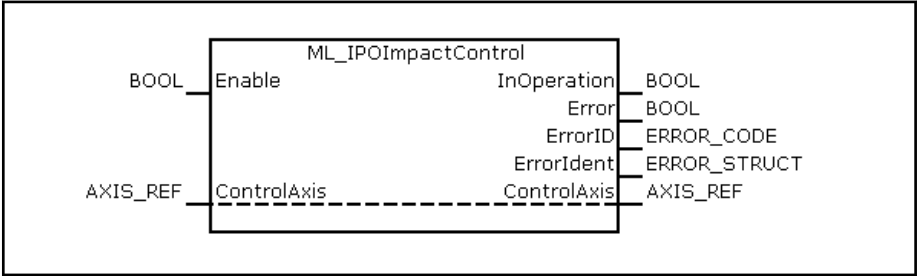


Abb. 9-2: FB ML_IPOImpactControl

	Name	Typ	Kommentar
VAR_IN_OUT	ControlAxis	AXIS_REF	Enthält Informationen zur konkreten Achse
VAR_INPUT	Enable	BOOL	Solange "Enable" TRUE ist, ist die Beeinflussung der Interpolation möglich
VAR_OUTPUT	InOperation	BOOL	FB ist aktiviert, Beeinflussung der Interpolation möglich
	Error	BOOL	Zeigt an, dass in der FB-Instanz ein Fehler aufgetreten ist

	Name	Typ	Kommentar
	ErrorID	ERROR_CODE	Kurzhinweis auf die Fehlerursache
	ErrorIdent	ERROR_STRUCT	Detailinformation über den Fehler

Tab. 9-2: Schnittstelle des FB ML_IPOImpactControl

9.3 ML_IPOImpactControl-Methoden

9.3.1 mSetNewState

Kurzbeschreibung Diese Methode setzt den neuen Zustand für die Interpolationsbeeinflussung.

Signatur

```
METHOD mSetNewState : DINT
VAR_INPUT
    eNewState: eIPOCONTROL_STATE;
END_VAR
```

Mögliche Werte für eNewState:

Wert	Kommentar
IPOCONTROL_STATE_NEUTRAL	Interpolation soll in den Zustand "neutral" übergehen oder bleiben.
IPOCONTROL_STATE_BREAK	Interpolation soll in den Zustand "break" übergehen.
IPOCONTROL_STATE_JUMP	Interpolation soll in den Zustand "jump" übergehen.

9.3.2 mGetNewState

Kurzbeschreibung Diese Methode liefert den administrierten Zustand, der mit mSetNewState eingestellt wurde.

Signatur

```
METHOD mGetNewState : DINT
VAR_INPUT
    peNewState: POINTER TO eIPOCONTROL_STATE;
END_VAR
```

Mögliche Werte für peNewState:

Wert	Kommentar
IPOCONTROL_STATE_NEUTRAL	Interpolation soll in den Zustand "neutral" übergehen oder bleiben.
IPOCONTROL_STATE_BREAK	Interpolation soll in den Zustand "break" übergehen.
IPOCONTROL_STATE_JUMP	Interpolation soll in den Zustand "jump" übergehen.



Diese Funktion liefert nicht den aktuellen Zustand der Interpolation.

Diese Information liefert die Methode mGetAxisControlledBy.

ML_IPOImpactControl.library

9.3.3 mSetDirection

Kurzbeschreibung Diese Methode setzt die Richtung, in der die Interpolation beeinflusst werden soll.

Signatur

```
METHOD mSetDirection : DINT
VAR_INPUT
    eDirection: eIPOCONTROL_DIRECTION;
END_VAR
```

Mögliche Werte für eDirection:

Wert	Kommentar
IPOCONTROL_DIRECTION_NONE	Interpolation wird in keine Richtung beeinträchtigt.
IPOCONTROL_DIRECTION_FORWARD	Interpolation soll in Vorwärtsrichtung beeinflusst werden.
IPOCONTROL_DIRECTION_BACKWARD	Interpolation soll in Rückwärtsrichtung beeinflusst werden.
IPOCONTROL_DIRECTION_BACKWARD OR IPOCONTROL_DIRECTION_FORWARD	Interpolation soll in beide Richtungen beeinflusst werden.

9.3.4 mGetDirection

Kurzbeschreibung Diese Methode liefert die Richtung, in der die Interpolation beeinflusst werden soll.

Signatur

```
METHOD mGetDirection : DINT
VAR_INPUT
    peDirection: POINTER TO eIPOCONTROL_DIRECTION;
END_VAR
```

Mögliche Werte für peDirection:

Wert	Kommentar
IPOCONTROL_DIRECTION_NONE	Interpolation wird in keine Richtung beeinträchtigt.
IPOCONTROL_DIRECTION_FORWARD	Interpolation soll in Vorwärtsrichtung beeinflusst werden.
IPOCONTROL_DIRECTION_BACKWARD	Interpolation soll in Rückwärtsrichtung beeinflusst werden.
IPOCONTROL_DIRECTION_BACKWARD OR IPOCONTROL_DIRECTION_FORWARD	Interpolation soll in beide Richtungen beeinflusst werden.

9.3.5 mSetAcceleration

Kurzbeschreibung Diese Methode setzt die individuelle Beschleunigung, die von der Interpolation beim Zustandsübergang

"neutral" ↔ "break"

berücksichtigt werden soll.

Signatur

```
METHOD mSetAcceleration : DINT
VAR_INPUT
    rAcceleration: REAL ;
END_VAR
```

9.3.6 mGetAcceleration

Kurzbeschreibung Diese Methode liefert die individuelle Beschleunigung, die von der Interpolation beim Zustandsübergang berücksichtigt werden soll.

Signatur

```
METHOD mGetAcceleration : DINT
VAR_INPUT
    prAcceleration: POINTER TO REAL;
END_VAR
```

9.3.7 mSetBreakBehaviour

Kurzbeschreibung Diese Methode setzt das Break-Verhalten, das beim Übergang "neutral" ↔ "break" eingestellt werden soll.

Signatur

```
METHOD mSetBreakBehaviour : DINT
VAR_INPUT
    eBreakBehaviour : eIPOCONTROL_BREAKBEHAVIOUR;
END_VAR
```

Mögliche Werte für eBreakBehaviour:

Wert	Kommentar
IPOCONTROL_BREAKBEHAVIOUR_DIRECT	Der Zustandsübergang erfolgt direkt.
IPOCONTROL_BREAKBEHAVIOUR_MOVECMD	Der Zustandsübergang erfolgt mit dem Beschleunigungswert des aktuellen Bewegungskommandos.
IPOCONTROL_BREAKBEHAVIOUR_INDIVIDUAL	Der Zustandsübergang erfolgt mit einem individuellen Beschleunigungswert.

9.3.8 mGetBreakBehaviour

Kurzbeschreibung Diese Methode liefert das "Break"-Verhalten, das für den Übergang "neutral" ↔ "break" eingestellt wurde.

Signatur

```
METHOD mGetBreakBehaviour : DINT
VAR_INPUT
    peBreakBehaviour: POINTER TO eIPOCONTROL_BREAKBEHAVIOUR;
END_VAR
```

ML_IPOImpactControl.library

Mögliche Werte für peBreakBehaviour:

Wert	Kommentar
IPOCONTROL_BREAKBEHAVIOUR_DIRECT	Der Zustandsübergang erfolgt direkt.
IPOCONTROL_BREAKBEHAVIOUR_MOVECMD	Der Zustandsübergang erfolgt mit dem Beschleunigungswert des aktuellen Bewegungskommandos.
IPOCONTROL_BREAKBEHAVIOUR_INDIVIDUAL	Der Zustandsübergang erfolgt mit einem individuellen Beschleunigungswert.

9.3.9 mSetPosition

Kurzbeschreibung Diese Methode setzt die neue Position, die von der Interpolation berücksichtigt werden soll.

Signatur

```
METHOD mSetPosition : DINT
VAR_INPUT
    lrPosition: LREAL;
END_VAR
```

9.3.10 mGetPosition

Kurzbeschreibung Diese Methode liefert die neue Position, die von der Interpolation berücksichtigt werden soll.

Signatur

```
METHOD mGetPosition : DINT
VAR_OUTPUT
    lrPosition: LREAL;
END_VAR
```

9.3.11 mSetValidValues

Kurzbeschreibung Diese Methode setzt die Einstellungen, welche Werte bezüglich Geschwindigkeit und Lage von der Interpolation berücksichtigt werden sollen.

Signatur

```
METHOD mSetValidValues : DINT
VAR_INPUT
    eValidValues: eIPOCONTROL_RESPECT;
END_VAR
```

Mögliche Werte für eValidValues:

Wert	Kommentar
IPOCONTROL_RESPECT_NONE	Die Interpolation berücksichtigt keinen Wert.
IPOCONTROL_RESPECT_POSITION	Die Interpolation berücksichtigt den Lagewert.
IPOCONTROL_RESPECT_VELOCITY	Die Interpolation berücksichtigt den Geschwindigkeitswert.
IPOCONTROL_RESPECT_POSITION OR IPOCONTROL_RESPECT_VELOCITY	Die Interpolation berücksichtigt beide Werte.

9.3.12 mGetValidValues

Kurzbeschreibung Diese Methode liefert die Einstellungen, welche Werte bezüglich Geschwindigkeit und Lage von der Interpolation aktuell berücksichtigt werden sollen.

Signatur

```
METHOD mGetValidValues : DINT
VAR_INPUT
    peValidValues: POINTER TO eIPOCONTROL_RESPECT;
END_VAR
```

Mögliche Werte für peValidValues:

Wert	Kommentar
IPOCONTROL_RESPECT_NONE	Die Interpolation berücksichtigt keinen Wert.
IPOCONTROL_RESPECT_POSITION	Die Interpolation berücksichtigt den Lagewert.
IPOCONTROL_RESPECT_VELOCITY	Die Interpolation berücksichtigt den Geschwindigkeitswert.
IPOCONTROL_RESPECT_POSITION OR IPOCONTROL_RESPECT_VELOCITY	Die Interpolation berücksichtigt beide Werte.

9.3.13 mSetVelocity

Kurzbeschreibung Diese Methode setzt die neue Geschwindigkeit, die von der Interpolation berücksichtigt werden soll.

Signatur

```
METHOD mSetVelocity : DINT
VAR_INPUT
    rVelocity: REAL;
END_VAR
```

9.3.14 mGetVelocity

Kurzbeschreibung Diese Methode liefert die neue Geschwindigkeit, die von der Interpolation berücksichtigt werden soll.

Signatur

```
METHOD mGetVelocity : DINT
VAR_INPUT
    prVelocity: POINTER TO REAL;
END_VAR
```

9.3.15 mGetAxisControlledBy

Kurzbeschreibung Diese Methode liefert die Information, ob die Achsinterpolation von dem SPS-Programm beeinflusst wird.

Signatur

```
METHOD mGetAxisControlledBy : DINT
VAR_INPUT
    peAxisControlledBy: POINTER TO eIPOCONTROL_CONTROLLED_BY;
END_VAR
```


ML_IPOImpactControl.library

Mögliche Werte für peAxisControlledBy:

Wert	Kommentar
IPOCONTROL_CONTROLED_BY_DEFAULT	Die Interpolation unbeeinflusst.
IPOCONTROL_CONTROLED_BY_PLC	Die Interpolation wird vom SPS-Programm beeinflusst.

9.4 Anwendung der Methoden

9.4.1 Einleitung

Anhand von zwei konkreten Beispielen - Zustand "Break" und "Jump" - wird der Aufruf der Methoden demonstriert.

9.4.2 Beispiel 1: Zustand "Break" aktivieren

Erläuterung Beim Zustand "Break" wird die Interpolation ausgesetzt, d.h. im Motionzyklus werden keine neuen Sollwerte mehr berechnet.

Der Motion-Baustein läuft normal weiter und setzt trotz des Stillstandes weder den Zustand "Done" noch den Zustand "CommandAborted".

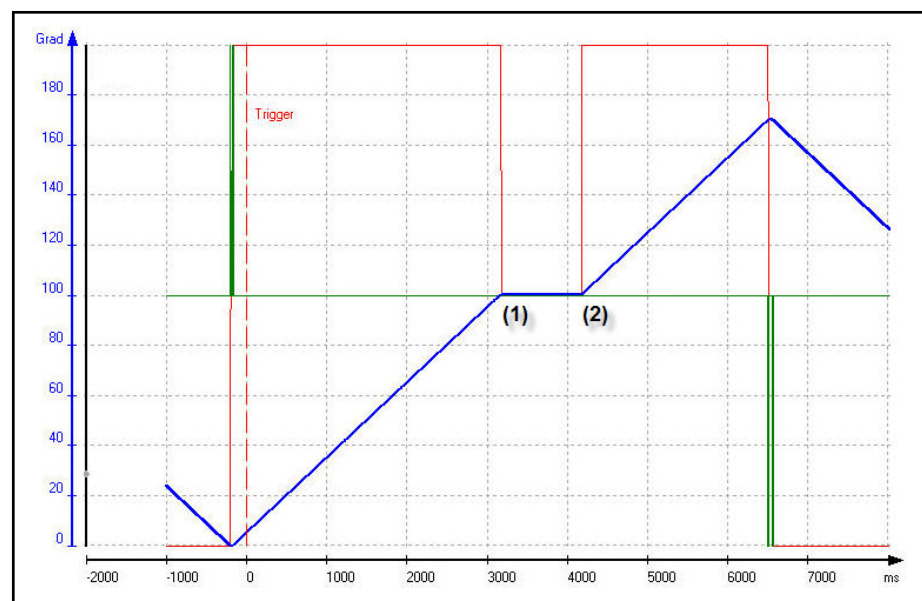


Abb. 9-3: blau = interpolierte Position, rot = interpolierte Geschwindigkeit, grün = interpolierte Beschleunigung

Methodenaufrufe

```
//Create an instance of ML IPOImpactControl, here MyIpoImpactControl,
//within the variable section of the PLC
MyIPOImpactControl: ML_IPOImpactControl;

//Enable MyIPOImpactControl and wait for InOperation becomes "TRUE"
MyIPOImpactControl(...);

//Set all values to default. Just to start from a neutral situation
//Set state to neutral
MyIPOImpactControl.mSetNewState(eNewState:=IPOCONTROL_STATE_NEUTRAL);
//Set direction to neutral.
MyIPOImpactControl.mSetDirection(eDirection:=IPOCONTROL_DIRECTION_NONE);
//Set valid values to neutral.
MyIPOImpactControl.mSetValidValues(eValidValues:=IPOCONTROL_RESPECT_NONE);

//Now start a MC_MoveAbsolute.
```

```

//For this example the move starts at 0° and ends at 170° degree
MC_MoveAbsolute(...);

//Wait for position 100° degree reached (See picture above, mark:(1))
//Set the impact direction to forward
MyIPOImpactControl.mSetDirection(eDirection:=IPOCONTROL_DIRECTION_FORWARD);
//Set state to break
MyIPOImpactControl.mSetNewState(eNewState:=IPOCONTROL_STATE_BREAK);

//Optional it is possible to set a invidual position and velocity between
//the marks:(1) and (2) (not shown in the picture above)
//These values will be used as a new start condition for the interpolation
//Set a new velocity
MyIPOImpactControl.mSetVelocity(rVelocity:= 1.0);
//Set a new position
MyIPOImpactControl.mSetPosition(lrPosition:= 105.0);
//Activates the values
MyIPOImpactControl.mSetValidValues(eValidValues:=IPOCONTROL_RESPECT_POSITION
                                     OR IPOCONTROL_RESPECT_VELOCITY);

//Check if the axis is under PLC control
// ^peAxisControlledBy has to be IPOCONTROL_CONTROLEDBY_PLC
MyIPOImpactControl.mGetAxisControlledBy(peAxisControlledBy);

//Wait 1000ms (See picture above, mark:(2))
//Set state to neutral, movement continues
MyIPOImpactControl.mSetNewState(eNewState:=IPOCONTROL_STATE_NEUTRAL);

```

9.4.3 Beispiel 2: Zustand "Jump" aktivieren

Erläuterung

Beim Zustand "Jump" springt die Interpolation in den Endzustand des aktuellen Motion-Befehls.

Der Motion-Baustein nimmt automatisch den Zustand "Done" ein.



Dieser Zustandsübergang ist für MC_MoveVelocity nicht möglich.

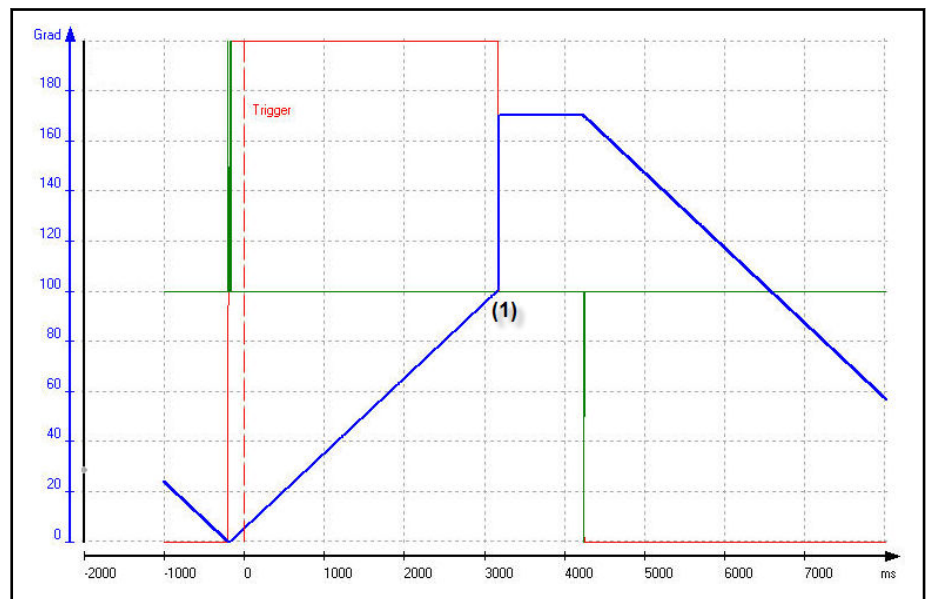


Abb. 9-4: blau = interpolierte Position, rot = interpolierte Geschwindigkeit, grün = interpolierte Beschleunigung

Methodenaufrufe

```

//Create an instance of ML_IPOImpactControl, here MyIpoImpactControl,
//within the variable section of the PLC
MyIPOImpactControl: ML_IPOImpactControl;

//Enable MyIPOImpactControl and wait for InOperation becomes "TRUE"
MyIPOImpactControl(...);

//Set all values to default. Just to start from a neutral situation

```

ML_IPOImpactControl.library

```

//Set state to neutral
MyIPOImpactControl.mSetNewState(eNewState:=IPOCONTROL_STATE_NEUTRAL);
//Set direction to neutral.
MyIPOImpactControl.mSetDirection(eDirection:=IPOCONTROL_DIRECTION_NONE);
//Set valid values to neutral.
MyIPOImpactControl.mSetValidValues(eValidValues:=IPOCONTROL_RESPECT_NONE);

//Now start a MC_MoveAbsolute.
//For this example the move starts at 0° and ends at 170° degree
MC_MoveAbsolute(...);

//Wait for position 100° degree reached (See picture above, mark:(1))
//Set the impact direction to forward
MyIPOImpactControl.mSetDirection(eDirection:=IPOCONTROL_DIRECTION_FORWARD);
//Set state to jump
MyIPOImpactControl.mSetNewState(eNewState:=IPOCONTROL_STATE_JUMP);

//Now the MC_MoveAbsolute is done immediately, position reaches 170° degree.

//Optional it is possible to set a individual position and velocity at mark:(1)
//(not shown in the picture above)
//These values will be used as a new start condition for the next MC_MoveCmd.
//Set a new velocity
MyIPOImpactControl.mSetVelocity(rVelocity:= 1.0);
//Set a new position
MyIPOImpactControl.mSetPosition(lrPosition:= 180.0);
//Activates the values
MyIPOImpactControl.mSetValidValues(eValidValues:=IPOCONTROL_RESPECT_POSITION
OR IPOCONTROL_RESPECT_VELOCITY);

```

10 ML_ParamBase.library

10.1 Verwendung

Mit dieser Bibliothek ist es möglich, die Reglerachse mit einer erweiterten Parametrierbarkeit auszustatten, d.h. SPS-Funktionalität wird über S- und P-Parameter gesteuert und deren Status ermittelt.

Durch diese Funktionalität wird erreicht, dass ein in der SPS realisierter Regler, ähnlich einem Antrieb, parametrisiert werden kann. Der Zusammenhang ist in der folgenden Grafik erläutert.

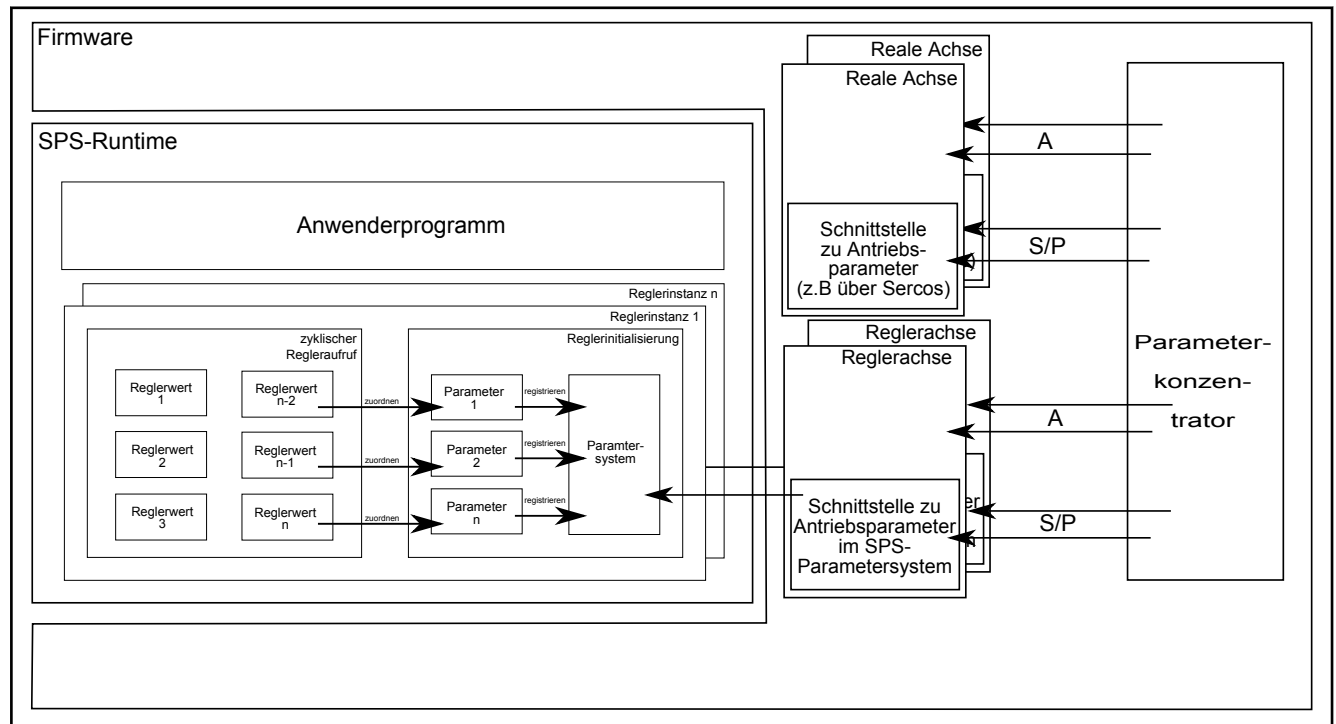


Abb. 10-1: Verwendung Reglerachse und SPS

Ebenso wie eine reale Achse kann eine Reglerachse S- und P-Parameter besitzen. Da die Reglerachse in der SPS gerechnet wird, müssen sich diese S- und P-Parameter in der SPS befinden. Aus diesem Grund wird in der SPS ein Parametersystem angelegt, welches sich mit der Firmware verbindet. Es werden in der SPS Parameter angelegt, denen die Daten zugeordnet werden. Damit die Parameter verwendet werden können, müssen sich diese am Parametersystem anmelden.

Diese Bibliothek enthält das [Parametersystem auf Seite 454](#) und zwei unterschiedliche Parameterdefinitionen:

- [PARAM_SercosParameter auf Seite 458](#),
- [PARAM_ExtSercosParameter auf Seite 463](#).

Komplettiert wird die Bibliothek durch einen Funktionsbaustein zur Abarbeitung von Kommandofolgen ([PARAM_CmdExecuter auf Seite 465](#)).

ML_ParamBase.library

10.2 Funktionsbausteine

10.2.1 PARAM_Parametersystem

Initialisierung

Kurzbeschreibung Das Parametersystem muss bei jeden Download oder Reset der SPS initialisiert werden. Um das Parametersystem einer Reglerachse zuzuweisen, muss bei der Initialisierung eine Achsreferenz angegeben werden.

Signatur

```
METHOD Init
VAR_INPUT
    Axis : AXIS_REF;
END_VAR
```

Anmelden von Parametern

Kurzbeschreibung Damit das Parametersystem weiß, welche Parameter der Reglerachse zugeordnet sind, müssen diese am Parametersystem angemeldet werden.

Dazu muss die Methode **addParameter** mit dessen Adresse aufgerufen werden. Auf diese Weise muss jeder zu verwendende Parameter registriert werden.

Wenn ein Parameter mit der einer bereits registrierten EIDN angemeldet wird, schlägt die Anmeldung fehl, der alte Parameter bleibt erhalten.

Signatur

```
METHOD addParameter : DWORD
VAR_INPUT
    _Parameter : POINTER TO PARAM_SercosParameter;
END_VAR
```

Funktionalitäten

Allgemeines

Die nachfolgenden Funktionen werden vom Parametersystem zur Verfügung gestellt. Zudem implementiert das Parametersystem einige Interfaces ([PARAM_SercosParameter](#) , [Seite 458](#)).

In der Beschreibung der Methode wird die entsprechende Parameternummer angegeben. Die Parameter müssen allerdings extern angelegt und dann angemeldet werden.

Dazu muss die Funktionsweise der oben genannten Interfaces beachtet werden.

Die hier genannten Methodeen sind als Low-Level-Methodeen zu betrachten.

Sprache umschalten

Das Parametersystem unterstützt die Verwendung von unterschiedlichen Sprachen eines Parameters. Somit werden die Anzeigesprache des Parameternamens und seine Einheit umgestellt. Es können die in [PARAM_Language](#) ([Sprache](#), [Seite 468](#)) definierten Sprachen verwendet werden.

Nähere Informationen sind im Abschnitt Datentypen vorhanden ([Datentypen](#), [Seite 468](#)).

In der SPS lässt sich die Sprache über die Property Language umschalten. Außerdem reagiert das Parametersystem auf die EIDN S-0-0265.

Parameter lesen

Kurzbeschreibung

In der SPS können Parameter über die Methode **ParamRead** gelesen werden. Dabei erfolgt ein direkter Zugriff auf den Parameter. Neben dem Wert des Parameters können auch Name, Einheit, Minwert, Maxwert und Attribut gelesen werden.

Diese Methode ist als systemnahe Funktion zu betrachten, so dass sie einige, im nachfolgenden erläuterte, Eigenheiten besitzt. Der Vorteil ist allerdings, dass die Parameterdaten im selben Zyklus zur Verfügung stehen.

Der Pointer **length** zeigt auf eine Variable, die die Anzahl der gültigen Bytes angibt, die in **data** geschrieben worden sind.

Bei Listen, Einheiten und Namen ist diese Angabe um die Größe des sercos Headers erhöht (4 Byte).

Signatur

```
METHOD ParamRead : SERCOS_ERROR
VAR_INPUT
  element: SERCOS_ELEMENT; // Element to read (Name, Unit, Value,...)
  eidn:   DWORD;           // EIDN to Read, e.g. 265 (S-0-0265)
  length: POINTER TO UINT; // Pointer to Value for written size in data
  data:   PARAM_upData;    // Union to get different typesm, of pointer
END_VAR
```

VORSICHT

Speicher überschreiben

Reservieren Sie genügend Speicher, da es sonst zu ungewolltem Überschreiben im Speicher kommen kann.

Parameter schreiben

Kurzbeschreibung

Analog der Methode **ParamRead** können mit **ParamWrite** Parameterdaten beschrieben werden.

In Abhängigkeit von seinem Attribut können nicht zu jeden Zeitpunkt die Werte geschrieben werden. So kann in den Parametern ein Schreibschutz für bestimmte sercos Phasen definiert werden.

Signatur

```
METHOD ParamWrite : SERCOS_ERROR
VAR_INPUT
  element: SERCOS_ELEMENT; // Element to write (Name, ...)
  eidn:   DWORD;           // EIDN to write
  length: POINTER TO UINT; // Pointer to Value to write size
  data:   PARAM_upData;    // Pointer to Value to write
END_VAR
```

Parameter sichern

Kurzbeschreibung

Alle Parameter, die am Parametersystem angemeldet sind, können auf CF-Karte gesichert werden.

Dieses Kommando wird über den Parameter **S-0-0264** ausgeführt. Der Methode kann ein Dateiname mitgegeben werden.

Wird ein leerer String übergeben wird ein Dateiname in der Form "ReglerCF<Achsnnummer>.bin" erzeugt.

Gesichert werden folgende Elemente:

- EIDN
- Highword des Attributes

ML_ParamBase.library

- Daten
- CRC16 Checksumme

Signatur

```
METHOD dwSaveToCF : DWORD
VAR_INPUT
    strFilename : STRING := '';
END_VAR
```



Diese Funktionalität kann extrem lange dauern. Aus diesem Grund kann ein Watchdog auftreten.

Sie sollten die Funktion asynchron mit Parameterzugriff verwenden.

Parameter wiederherstellen**Kurzbeschreibung**

Die über das Kommando des S-0-0264 gesicherten Parameter werden mit dem Kommando **S-0-0263** wieder hergestellt.

Beim Aufruf kann eine Datei angegeben werden. Bei einem leeren String wird ein generischer String verwendet.



Es können nur Parameter wiederhergestellt werden, die bereits am Parametersystem angemeldet sind.

Signatur

```
METHOD dwLoadFromCF : DWORD
VAR_INPUT
    strFilename : STRING := '';
END_VAR
```



Diese Funktionalität kann extrem lange dauern. Aus diesem Grund kann ein Watchdog auftreten.

Sie sollten die Funktion asynchron mit Parameterzugriff verwenden.

Defaultwerte laden

Mit Hilfe des Kommandos **S-0-0262** werden die Defaultwerte wiederhergestellt. Die Konfiguration erfolgt über den Parameter P-0-4090.



Der Defaultwert ist nicht für Listenparameter definiert.

Die Konfiguration kann innerhalb der SPS durch die Property **LoadDefault-Settings** erfolgen. Dabei ist aktuell nur der Wert LOAD_DEFAULT (:= 1) zulässig.

Mit Hilfe der Methode **dwLoadDefaultParam()** lässt sich der Vorgang auch in der SPS ausführen. Es wird versucht zuerst die Wichtungparameter zu schreiben.

Dazu wird die Methode **LoadDefaultParam** des Parameters zuerst mit der Option **K_LOADDEFAULT_WEIGHTING_ONLY** und danach mit **K_LOADDEFAULT_NO_OPTIONS** gerufen ([Administration, Seite 462](#)).



Diese Funktionalität kann extrem lange dauern. Aus diesem Grund kann ein Watchdog auftreten.

Sie sollten die Funktion asynchron mit Parameterzugriff verwenden.

Parameterlisten erstellen

Liste aller Parameter

Kurzbeschreibung

Über den Parameter **S-0-0017** stellt das Parametersystem eine Liste mit allen angemeldeten Parametern zur Verfügung.

Über die Methode **dwCreateListAllParam** kann die Liste auch über die SPS ausgelesen werden.

Signatur

```
METHOD dwCreateListAllParam : DWORD
VAR_INPUT
    upData:    PARAM upData;
    puiLength: POINTER TO UINT;
END_VAR
```



Diese Funktionalität kann extrem lange dauern. Aus diesem Grund kann ein Watchdog auftreten.

Sie sollten die Funktion asynchron mit Parameterzugriff verwenden.

Liste aller Kommandos

Kurzbeschreibung

Eine Liste der angemeldeten Kommando erhält man über den Parameter **S-0-0025**.

Über die Methode **dwCreateListAllCommands** kann die Liste auch über die SPS ausgelesen werden.

Signatur

```
METHOD dwCreateListAllCommands : DWORD
VAR_INPUT
    upData:    PARAM upData;
    puiLength: POINTER TO UINT;
END_VAR
```



Diese Funktionalität kann extrem lange dauern. Aus diesem Grund kann ein Watchdog auftreten.

Sie sollten die Funktion asynchron mit Parameterzugriff verwenden.

Liste aller auf der CF-Karte gesicherten Parameter

Kurzbeschreibung

Die Liste mit den Parametern, die auf der CF-Karte gespeichert werden, erhält man über den Parameter **S-0-0192**.

Über die Methode **dwCreateListSaveOnCF** kann die Liste auch über die SPS ausgelesen werden.

Signatur

```
METHOD dwCreateListSaveOnCF : DWORD
VAR_INPUT
    upData:    PARAM upData;
    puiLength: POINTER TO UINT;
END_VAR
```



Diese Funktionalität kann extrem lange dauern. Aus diesem Grund kann ein Watchdog auftreten.

Sie sollten die Funktion asynchron mit Parameterzugriff verwenden.

ML_ParamBase.library

Parameterwerte synchronisieren**Kurzbeschreibung**

Verfügbar ab ML_ParamBase 12.4.0.0

Mit dieser Methode lassen sich bestimmte Parameter lesen und mit dem gelesenen Wert neu beschreiben. Der Aufruf dieser Methode kann z.B. nach einer Wichtungsumschaltung nötig sein.

Der Übergabeparameter **iType** bestimmt, welche Parameter synchronisiert werden sollen. Wenn der Wert K_WEIGHTING_ALL verwendet wird werden alle Parameter synchronisiert, ansonsten nur die Parameter bei denen **iType** mit **iGetParamType()** identisch ist.

Signatur

```
METHOD dwReadWriteParamType : DWORD
VAR_INPUT
    iType : INT := K_WEIGHTING_ALL;
END_VAR
```

Zyklischer Schreibschutz

Die Methode setzt die Parametereigenschaft zyklische Sollwerte zu bekommt. Der Parameter wird über die Parameternummer identifiziert.

Nähere Informationen unter [Kap. "Administration" auf Seite 462](#).

Der Eingang **eiEidn** identifiziert den Parameter.

Signatur

```
METHOD dwSetCyclicConfigured : DWORD
VAR_INPUT
    eiEidn : DWORD;
    boValue : BOOL;
END_VAR
```

10.2.2 PARAM_SercosParameter**Allgemeines**

Diese Klasse ist die Basisimplementation eines Parameters. Nachfolgend wird ihre Verwendung erläutert.

Interfaces**Verwendung**

Damit ein Parameter die Möglichkeit erhält, spezielle Überprüfungen und Verarbeitungen vorzunehmen, besitzt er drei Interfaces:

Dem Parameter kann somit für jedes Interface ein Funktionsbaustein, der dieses Interface implementiert, zugeordnet werden, der die gewünschte Methode durchführt.

Bei einem Kommandoparameter wird nur auf das Kommando-Interface zugegriffen.

Die anderen beiden Interface werden bei normaler Parameterverwendung für den Lese- und Schreibaufruf verwendet.

Damit die Methode die Informationen über den Parameter besitzt wird, jeweils der Pointer des Parameters mit übergeben.

Lese-Interface**Kurzbeschreibung**

Das Interface **I_PARAM_ReadAccess** besitzt die Methode **dwReadData**.

Signatur

```

METHOD dwReadData : DWORD
VAR INPUT
  // Pointer to parameter to modify, read, ...
  poParam: POINTER TO PARAM_SercosParameter;
  // Pointer to data to write
  upData: PARAM_upData;
END_VAR

```

Schreib-Interface

Kurzbeschreibung Das Interface **I_PARAM_WriteAccess** besitzt die Methode **dwWriteData**.

Signatur

```

METHOD dwWriteData : DWORD
VAR INPUT
  // Pointer to parameter to write
  poParam: POINTER TO PARAM_SercosParameter;
  // Pointer to data
  upData: PARAM_upData;
  // Accessmode
  eAccessMode: PARAM_Access;
END_VAR

```

Kommando-Interface

Kurzbeschreibung Das Interface **I_PARAM_CmdAccess** besitzt die Methoden **dwCmdInit**, **dwCmdCyclic** und **dwCmdAbort**.

In der Methode **dwCmdInit** wird die Initialisierung des Kommandos durchgeführt. (Initialisierung Prozessdatenblock, Dateien öffnen, ...)

Signatur dwCmdInit

```

METHOD dwCmdInit : SERCOS_CMD_RETURN
VAR INPUT
  // Pointer to parameter
  poParam: POINTER TO PARAM_SercosParameter;
  // Pointer to process data block
  puPDB: POINTER TO PARAM_upDB;
END_VAR

```

Die zyklische Abarbeitung des Kommandos erfolgt in der Methode **dwCmdCyclic**.

Signatur dwCmdCyclic

```

METHOD dwCmdCyclic : SERCOS_CMD_RETURN
VAR INPUT
  // Pointer to parameter
  poParam: POINTER TO PARAM_SercosParameter;
  // Pointer to process data block
  puPDB: POINTER TO PARAM_upDB;
END_VAR

```

Beim Zurücksetzen des Kommandos wird die Methode **dwCmdAbort** aufgerufen. Diese dient den Aufräumarbeiten. (Dateien schließen, ...)

Signatur dwCmdAbort

```

METHOD dwCmdAbort : SERCOS_CMD_RETURN
VAR INPUT
  // Pointer to parameter
  poParam: POINTER TO PARAM_SercosParameter;
  // Pointer to process data block
  puPDB: POINTER TO PARAM_upDB;
END_VAR

```

ML_ParamBase.library

Attribute

Überblick Der Funktionsbaustein PARAM_SercosParameter besitzt folgende öffentliche Attribute.

Klassenvariablen PARAM_SercosParameter

Variablenname	Datentyp	Beschreibung
c_dwDataStatus	DWORD	Zeigt den Fehlercode des Parameters. Kein Fehler = 0
c_dwEidn	DWORD	EIDN des Parameters
c_paName	POINTER TO PARAM_Language_Array	Parametername
c_ppdwAttribute	POINTER TO POINTER TO DWORD	Attribut des Parameters. Analog sercos Attribut
c_ppaUnit	POINTER TO POINTER TO PARAM_Language_Array	Einheit des Parameters
c_puMin	POINTER TO PARAM_uData	Minimalwert des Parameters
c_puMax	POINTER TO PARAM_uData	Maximalwert des Parameters
c_upActData	PARAM_uData	Parameterwert
c_uDefault	PARAM_uData	Defaultwert
itfWrite	I_PARAM_WriteAccess	Interface für die Schreibmethode
itfRead	I_PARAM_ReadAccess	Interface für die Lesemethode
itfCmd	I_PARAM_CmdAccess	Interface für die Kommandoausführung

c_paName Die Variable verweist auf ein globales Feld mit den Zeigern auf die übersetzten Namen ([Sprache, Seite 468](#)). Damit lässt sich bei mehreren Reglerachsen Speicherplatz sparen.

c_ppdwAttribute Die Variable für das Attribut zeigt auf einen Pointer, der auf das Attribut zeigt. Damit ist es möglich richtungsabhängige Attribute zu erstellen. Der Pointer muss in den Funktionsbaustein liegen, welcher den Parameter instanziiert. Das Attribut selbst kann global abgelegt werden.

c_ppaUnit Die Einheit wird, wie der Name, sprachabhängig angezeigt. Dazu wird die selbe Datenstruktur genommen. Allerdings wird, im Gegensatz zum Namen, über einen Pointer darauf verwiesen. Damit lässt sich auch hier eine Richtungsabhängigkeit erzeugen.

c_puMin, c_puMax Da bei Listen nur die einzelnen Listenelemente überprüft werden, ist der Extremwert auch nur eine Einzelvariable. Zur Kennzeichnung, dass ein Parameter keine Extremwerte besitzt, kann hier der Wert IL_NULL zugewiesen werden.

c_upActData Das Datum des Parameters wird innerhalb des Funktionsbausteins gespeichert in dem auch der Parameter instanziiert wurde. Der Pointer auf dieses wird in dieser Variable gespeichert.

c_uDefault Der Defaultwert wird als Einzelwert abgespeichert.



Der Defaultwert ist nicht für Listenparameter definiert.

Zugriffsmethoden

Datenstatus lesen

Kurzbeschreibung

Der Datenstatus gibt die Gültigkeit des Parameters an. Dieser ist mit einem Wert von Null gültig.

Signatur

```
METHOD dwGetDataStatus : SERCOS_ERROR
VAR_INPUT
    data : POINTER TO PARAM_uData; // Pointer to data to write (dwData)
END_VAR
```

IDN lesen

Kurzbeschreibung

Über die folgende Methode kann die IDN eines Parameters gelesen werden.

Signatur

```
METHOD dwGetIDN : SERCOS_ERROR
VAR_INPUT
    data : PARAM_uData; //Object contains idn (PARAM_uData.eiData)
END_VAR
```

Name lesen

Kurzbeschreibung

Das Lesen des Parameternamens geschieht über diese Methode.

Signatur

```
METHOD dwGetName : SERCOS_ERROR
VAR_INPUT
    // Pointer to Data to write
    pstrData : POINTER TO PARAM_rStringData;
    // Size of data incl. Header
    puiLength : POINTER TO UINT;
    // Read Name in this language
    eLanguage : PARAM_Language := PARAM_Language.PARAM_LANG_DE;
END_VAR
```

Attribut lesen

Kurzbeschreibung

Das sercos Attribut erhält man mit Hilfe dieser Methode.

Signatur

```
METHOD dwGetAttribute : SERCOS_ERROR
VAR_INPUT
    // pointer to data to write (PARAM_uData.dwData)
    puData : POINTER TO PARAM_uData;
END_VAR
```

Minimalwert lesen

Kurzbeschreibung

Den Minimalwert erhält man mit Hilfe dieser Methode.

Signatur

```
METHOD dwGetMinValue : SERCOS_ERROR
VAR_INPUT
    // pointer to data to write (PARAM_uData.<Datatype>)
    puData : POINTER TO PARAM_uData;
    // pointer to data to set written size
    puiLength : POINTER TO UINT;
END_VAR
```

Maximalwert lesen

Kurzbeschreibung

Den Minimalwert erhält man mit Hilfe dieser Methode.

Signatur

```
METHOD dwGetMaxValue : SERCOS_ERROR
VAR_INPUT
```

ML_ParamBase.library

```
// pointer to data to write (PARAM_uData.<Datatype>)
puData : POINTER TO PARAM_uData;
// pointer to data to set written size
puiLength : POINTER TO UINT;
END_VAR
```

Datum lesen**Kurzbeschreibung**

Die Schnittstelle nach außen, um das Datum lesen zu können, ist wie folgt definiert.

Beim Lesen wird überprüft, ob ein Leseinterface angemeldet ist. Wenn dies der Fall ist, wird in die Union **upDate** der Wert geschrieben. Ansonsten wird der Inhalt vom **c_upActData** kopiert.

Der Accesstype wird ignoriert.

Signatur

```
METHOD dwGetOperationData : SERCOS_ERROR
VAR_INPUT
  // Pointer to data to write (<Datatype>)
  upData: PARAM_upData;
  // written size
  puiLength: POINTER TO UINT := IL_NULL;
  // access type
  eAccess: PARAM_Access := PARAMETER_ACCESS;
END_VAR
```

Datum schreiben**Kurzbeschreibung**

Für das Schreiben eines Datums steht die folgende Methode zur Verfügung.

Innerhalb dieser Methode wird überprüft, ob es sich um ein Parameter oder ein Kommando handelt.

Sollte es sich um ein Kommando handeln, wird überprüft, ob das notwendige Interface angemeldet ist. Ist dies der Fall wird die Methode aufgerufen. Ansonsten wird der Fehlercode K_STS_DATA_NO_ACCESS zurückgegeben.

Im Fall eines Parameters wird, wenn vorhanden, das Schreib-Interface aufgerufen. Alternativ wird **c_upActData** kopiert. Obwohl der Parameter über Extremwerte verfügen kann, ist es nicht möglich anzugeben, dass ein Extremwertcheck durchgeführt werden soll.

Der Accesstype wird ignoriert.

Signatur

```
METHOD dwSetOperationData : SERCOS_ERROR
VAR_INPUT
  upData: PARAM_upData; // Pointer with data to write
  puiLength: POINTER TO UINT; // size to write
  eAccess: PARAM_Access := PARAMETER_ACCESS; // access type
END_VAR
```

Administration**Defaultwerte laden****Kurzbeschreibung**

Für das Laden der Defaultwerte verfügt der Parameter über die unten genannte Methode. Die Methode arbeitet nur wenn sie mit der Option K_LOADDEFAULT_NO_OPTIONS aufgerufen wird.

Signatur

```
METHOD dwLoadDefaultData : DWORD // Options to load default data
VAR_INPUT
  dwOptions: DWORD := K_LOADDEFAULT_NO_OPTIONS;
END_VAR
```

Parametertyp

Kurzbeschreibung

Verfügbar ab ML_ParamBase 12.4.0.0

Diese Methode liefert den Wert K_WEIGHTING_NONE zurück.

Signatur

```
METHOD iGetParamType : INT
VAR_INPUT
END_VAR
```

Zyklischer Schreibschutz

Für Parameter, welche eine zyklische Sollwertvorgabe liefern, kann ein Schreibschutz über den Parameterzugriff erfolgen.

Diese Information kann mit Hilfe der folgenden Methode gesetzt werden. Wenn **boValue** TRUE ist, wird der Parameter als zyklisch konfiguriert markiert. Beim Schreiben des Parameters in Phase 4 erhält man den Fehlercode K_STS_DATA_CONFCYCLIC ("**SERCOS_ERROR**" auf Seite 470).

Signatur

```
METHOD dwSetCyclicConfigured : DWORD
VAR_INPUT
    boValue : BOOL := FALSE;
END_VAR
```

10.2.3 PARAM_ExtSercosParameter

Allgemeines

Der Funktionsbaustein PARAM_ExtSercosParameter erbt die Eigenschaften von [PARAM_SercosParameter](#), Seite 458. Deshalb werden hier nur die Unterschiede und Erweiterungen dargestellt.

Attribute

Überblick

Der Funktionsbaustein PARAM_ExtSercosParameter besitzt zusätzlich zum Funktionsbaustein PARAM_SercosParameter folgende Eingänge.

Klassenvariablen PARAM_ExtSercosParameter

Variablenname	Datentyp	Beschreibung
c_combination	POINTER TO PARAM_uData	Gültige Bitkombinationen
c_dwControl	DWORD	Erweitertes Steuerwort des Parameters
c_upParamData	PARAM_upData	Parameterwert (durch Parametersystem geschrieben)
c_upPropData	PARAM_upData	Parameterwert (durch Property geschrieben)
c_iParamType	INT	Klassifizierung des Parameters
c_plrScalingFactor	POINTER TO LREAL	Wichtungsfaktor

c_combination

In dieser Variablen kann ein Pointer auf ein Feld von gültigen Kombinationen hinterlegt werden. Feldelement 0 enthält die Abfragemaske. Beim letzten Element des Feldes werden alle Bits auf 1 gesetzt.

Eine Überprüfung der Gültigkeit erfolgt nur, wenn im erweiterten Steuerwort das Flag **PARAM_ESP_CHECKCOMB** gesetzt ist.

ML_ParamBase.library

c_dwControl Mit Hilfe dieses Steuerwortes können noch unterschiedliche Optionen gesetzt werden.

Steuerflags

Flag	Bedeutung
PARAM_ESP_CHECK-EXTREM	Extremwerte sollen überprüft werden
PARAM_ESP_CHECK-COMB	Kombinationsprüfung soll vorgenommen werden
PA-RAM_ESP_VAR_MIN	Parameter hat einen Minimalwert, dessen Wert den Datum eines andere Parameters entspricht.
PA-RAM_ESP_VAR_MAX	Parameter hat einen Maximalwert, dessen Wert den Datum eines andere Parameters entspricht.
PARAM_ESP_MEMO-RY_CF	Parameter soll auf der CF-Karte gespeichert werden.
PARAM_ESP_WRITE-PROTECTION_P2	Parameter in P2 beschreibbar
PARAM_ESP_WRITE-PROTECTION_P4	Parameter in P4 beschreibbar

c_upParamData, c_upPropData Wie beim Attribut **c_upActData** wird hier nur der Pointer gespeichert. Beiden Pointern kann (wie z.B. bei einem Istwert) der Wert IL_NULL zugeordnet werden.

c_iParamType Mit Hilfe dieser Variablen lässt sich ein Parameter klassifizieren, d.h. ob er wichtungsunabhängig ist, die Wichtung bestimmt oder wie er gewichtet ist.

Übersicht Parametertypen

Konstante	Beschreibung
K_WEIGHTING_NONE	Wichtungsunabhängiger Parameter
K_WEIGHTING_PARAM	Wichtungsbestimmender Parameter

Es können benutzerdefinierte Wichtungen erstellt werden. Diese müssen einen Wert größer 0 besitzen.

c_plrWeightingFactor Ist ein Wichtungsfaktor gesetzt wird dieser beim Lesen/Schreiben des Aktivdatums (c_upActdata) berücksichtigt.

Zugriffsmethoden**Datum lesen****Kurzbeschreibung**

Beim Lesen wird überprüft, ob ein Leseinterface angemeldet ist. Wenn dies der Fall ist, wird in die Union **upData** der Wert geschrieben. Ansonsten wird in Abhängigkeit des Zugriffspfades das entsprechende Datum gelesen. Wenn weder das Propertydatum noch das Parameterdatum existieren, wird **c_upActData** herangezogen.

Signatur

```

METHOD dwGetOperationData : SERCOS_ERROR
VAR_INPUT
    upData:    PARAM upData;           // Pointer to data to write
    puiLength: POINTER TO UINT := IL_NULL; // written size
    eAccess:    PARAM Access := PARAMETER_ACCESS; // access type
END_VAR

```

Datum schreiben

Kurzbeschreibung

Innerhalb dieser Methode wird überprüft, ob es sich um einen Parameter oder ein Kommando handelt.

Sollte es sich um ein Kommando handeln wird überprüft, ob das notwendige Interface angemeldet ist. Ist dies der Fall wird die Methode aufgerufen. Ansonsten wird der Fehlercode K_STS_DATA_NO_ACCESS zurückgegeben.

Im Fall eines Parameters wird, wenn vorhanden, das Schreib-Interface aufgerufen. Ansonsten wird in Abhängigkeit des Zugriffspfad des entsprechenden Datum geschrieben. Wenn weder das Propertydatum noch das Parameterdatum existiert wird **c_upActData** herangezogen.

Signatur

```
METHOD dwSetOperationData : SERCOS_ERROR
VAR_INPUT
  upData:    PARAM_upData;      // Pointer with data to write
  puiLength: POINTER TO UINT;   // size to write
  eAccess:   PARAM_Access := PARAMETER_ACCESS; // access type
END_VAR
```

Administration

Defaultwerte laden

Kurzbeschreibung

Für das Laden der Defaultwerte verfügt der Parameter über die unten genannte Methode. Die Methode kann mit den Optionen K_LOADDEFAULT_NO_OPTIONS und K_LOADDEFAULT_WEIGHTING_ONLY gerufen werden

Signatur

```
METHOD dwLoadDefaultData : DWORD // Options to load default data
VAR_INPUT
  dwOptions: DWORD := K_LOADDEFAULT_NO_OPTIONS;
END_VAR
```

Parametertyp

Kurzbeschreibung

Verfügbar ab ML_ParamBase 12.4.0.0

Diese Methode liefert den Wert der Variable **c_iParamType**, [Seite 463](#) zurück.

Signatur

```
METHOD iGetParamType : INT
VAR_INPUT
END_VAR
```

10.2.4 PARAM_CmdExecuter

Kurzbeschreibung

Dieser Funktionsbaustein realisiert die Abarbeitung einer Folge von Kommandos. Er beinhaltet dazu eine Zustandsmaschine.

Die Kommandos werden nacheinander in Abhängigkeit von ihrem Zustand abgearbeitet, d.h. es wird die entsprechenden Methode des **I_PARAM_CmdAccess** aufgerufen.

Anhand des Rückgabewertes der zuletzt gerufenen Methode und der Benutzereingaben wird die Zustandsmaschine in der Kommandoabarbeitung gesteuert ([Abb. 10-2 "Zustandsmaschine des Kommandohandlers" auf Seite 467](#)).

Befindet sich ein Kommando im Zustand K_STATE_NOT_ACTIVE wird es aus dem Speicher gelöscht.

ML_ParamBase.library

Methoden

Kommando hinzufügen

Zum Hinzufügen eines Kommandos wird die Methode **dwAddCmd** verwendet.

Innerhalb dieser Methode wird überprüft, ob ein weiteres Kommando in die Verwaltungsstruktur aufgenommen werden kann und dieses noch nicht eingetragen wurde. Wenn beides der Fall ist wird die Init-Methode des Kommandos aufgerufen.

Signatur

```
METHOD dwAddCmd : SERCOS_ERROR
VAR_INPUT
  // Pointer to Parameter
  pParam : POINTER TO PARAM_SercosParameter;
  // Interface to work parameter
  itfCmd : I_PARAM_CmdAccess := IL_NULL;
  // Control Cmd: decide next state
  usiCC : USINT := 2#11;
END_VAR
```

Rückgabewert	Beschreibung
K_STS_DATA_INVALID	Funktionsparameter ungültig
K_STS_DATA_CMDACTIVE	Kommando bereits aktiv
K_STS_DATA_CMDNOTEXEETIME	Array ist voll. Kommando kann z.Z. nicht ausgeführt werden.
K_STS_DATA_CMDNOTEXEETIME (Parameterstatus ist K_STATE_ERROR)	Init-Methode nicht erfolgreich ausgeführt
K_STS_OK	Methode erfolgreich durchgeführt

Tab. 10-1: Rückgabewerte

Kommandospeicher löschen

Diese Methode dient zum Löschen des internen Speichers. Diese wird vom Funktionsbaustein beim Initialisieren automatisch vorgenommen.

Die Methode liefert K_STS_OK zurück.

Signatur

```
METHOD dwClearStorage : SERCOS_ERROR
VAR_INPUT
END_VAR
```

Kommando steuern

Mit dieser Methode kann dem Kommando das CC (Control Command) gesetzt werden.

Signatur

```
METHOD dwSetControlCmd : SERCOS_ERROR
VAR_INPUT
  // Pointer to Parameter
  pParam : POINTER TO PARAM_SercosParameter;
  // CC Controlvalue
  usiCC : USINT;
END_VAR
```



Beim Aufruf dieser Methode wird der entsprechende Parameterwert nicht gesetzt.

10.3 Kommandohandling

In dem Parametersystem wurde ein Kommandohandling nach sercos Spezifikation erstellt. Dieses erlaubt die parallele Abarbeitung von mehreren Kommandos in einer gemeinsamen Task.

Dafür ist es notwendig das Programm **HydrControlCmdProg**, welches in dieser Bibliothek bereitgestellt wird, in einer freilaufenden Task zu konfigurieren. Das Programm ruft zyklisch den Funktionsbaustein **PARAM_CmdExecuter** auf. Dieser ist in [PARAM_CmdExecuter auf Seite 465](#) beschrieben.

Das Konzept sieht vor, dass jeder Funktionsbaustein, der für die Kommandoverarbeitung verwendet wird, das Interface **I_PARAM_CmdAccess** implementieren muss. Die einzelnen Methoden dieses Interfaces steuern den Ablauf der sercos Kommando-Zustandsmaschine durch die unterschiedlichen Rückgabewerte. Zur Steuerung der Zustandsmaschine kann der Benutzer den dazugehörigen Parameter schreiben. Die entsprechenden Eingaben sind durch **CC** dargestellt. Die Darstellung des Kommandos im Parameterstatus wird durch das folgende Bild dargestellt.

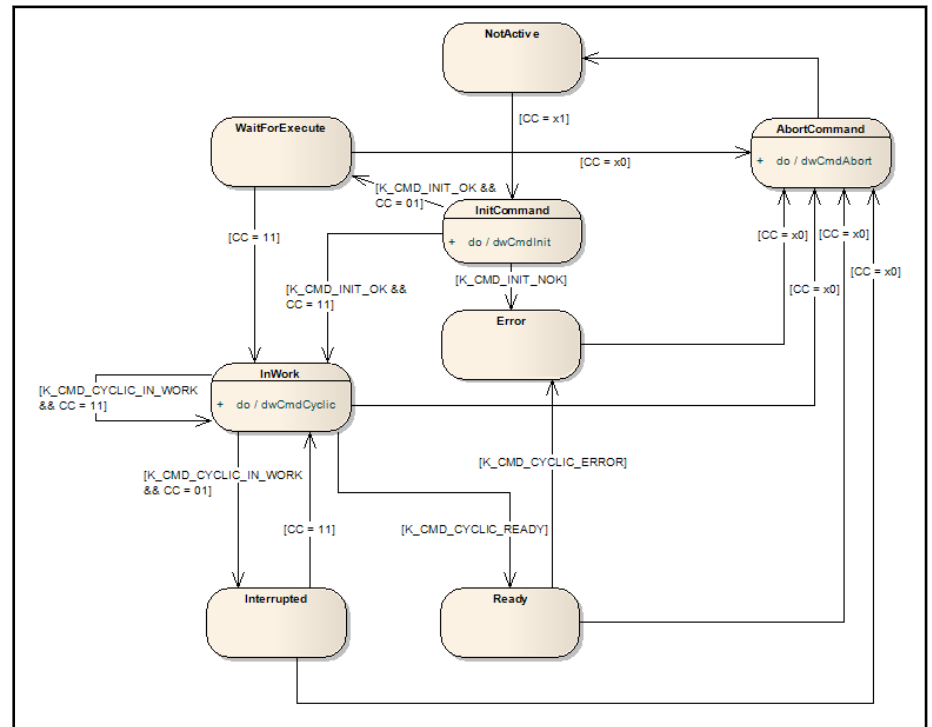


Abb. 10-2: Zustandsmaschine des Kommandohandlers

Da der Kommandohandler mehrere Kommandos parallel abarbeiten soll, ist das Kommando verantwortlich, dass es Zeitschlitze erzeugt. Dieses Verhalten wird über die Rückgabewerte der Methode **dwCmdCyclic** erreicht.

Ein Kommando wird automatisch der Abarbeitung hinzugefügt, wenn der dazugehörige Parameter als Kommando gekennzeichnet und ein Kommandointerface angegeben ist.

ML_ParamBase.library

10.4 Datentypen

10.4.1 Sprache

PARAM_Language *Typdeklaration*

```

TYPE PARAM_Language :
(
    PARAM_LANG_DE := 0, (* German, 0 *)
    PARAM_LANG_EN := 1, (* English, 1 *)
    PARAM_LANG_FR := 2, (* French, 2 *)
    PARAM_LANG_ES := 3, (* Spanish, 3 *)
    PARAM_LANG_IT := 4  (* Italian, 4 *)
) WORD;
END_TYPE

```

In dieser Enumeration sind die verfügbaren Sprachen aufgelistet.

Die Nummern entsprechen der Definition des Parameters S-0-0265.

PARAM_Language_Array *Typdeklaration*

```

TYPE PARAM_Language_Array :
    ARRAY[PARAM_LANG_DE..PARAM_LANG_IT] OF POINTER TO STRING;
END_TYPE

```

Durch diese Felddefinition können der Name und die Einheit der Parameter sprachabhängig gespeichert werden. In den genannten Fällen greift der Parameter auf das Element zu, welches der eingestellten Sprache zugeordnet ist.

10.4.2 Datenzugriff

PARAM_uData

Diese Union enthält alle benötigten Datentypen und deren Pointer. Auf die Darstellung der Definition wird wegen der Übersichtlichkeit verzichtet.

PARAM_rStringData

Diese Typdeklaration kann für Zugriffe auf ein Stringelement (Name, Einheit) oder einen Stringparameter verwendet werden.

Entsprechend der sercos Norm beinhaltet das Datum einen Header mit der aktuellen und maximalen Länge. Beide Werte sind in dieser Verwendung identisch.

Typdeklaration

```

TYPE PARAM_rStringData :
    STRUCT
        uiActLen: UINT;    (* Actual Length *)
        uiMaxLen: UINT;    (* Maximal Length *)
        strData: STRING;  (* Data as String *)
    END_STRUCT
END_TYPE

```

PARAM_rListData

Für den Zugriff auf Listenparameter kann der notwendige Speicher mit dieser Variable reserviert werden. Analog zu Typ **PARAM_rStringData** enthält die Struktur den sercos spezifischen Header.

Typdeklaration

```

TYPE PARAM_rListData :
    STRUCT
        uiActLen: UINT;    (* Actual Length *)
        uiMaxLen: UINT;    (* Maximal Length *)
        rData:    PARAM_uArray; (* Data *)
    END_STRUCT
END_TYPE

```



Mit der Deklaration einer Variablen dieses Typs werden etwa 65532 Bytes Speicher reserviert.

PARAM_upData Für Datenzugriffe, bei denen die Datengröße nicht bekannt oder variabel ist, kann als Übergabewert eine Variable dieses Typs verwendet werden.

Typdeklaration

```
TYPE PARAM_upData :
  UNION
    puData:      POINTER TO PARAM_uData;      // Singleparameter
    pStringData: POINTER TO PARAM_rStringData; // Stringparameter
    pListData:   POINTER TO PARAM_rListData;   // Listparameter
  END UNION
END_TYPE
```

Der benötigte Datentyp kann nun dem entsprechenden Pointer zugewiesen werden.

PARAM_uArray Diese Union definiert ein Feld mit unterschiedlicher Datenbreite so, dass die Gesamtgröße des Feldes jeweils 65532 (64kB abzüglich der Größe des Headers) beträgt.

Die Datentypen **PARAM_u1Byte** bis **PARAM_u8Byte** sind Unions mit den Datentypen entsprechender Größe.

Typdeklaration

```
TYPE PARAM_uArray :
  UNION
    ar8Byte: ARRAY[0.. 8191] OF PARAM_u8Byte; // Num. of Elem., size 8 Byte
    ar4Byte: ARRAY[0..16382] OF PARAM_u4Byte; // Num. of Elem., size 4 Byte
    ar2Byte: ARRAY[0..32765] OF PARAM_u2Byte; // Num. of Elem., size 2 Byte
    ar1Byte: ARRAY[0..65531] OF PARAM_u1Byte; // Num. of Elem., size 1 Byte
    str : STRING(65532);
  END UNION
END_TYPE
```



Mit der Deklaration einer Variablen dieses Typs werden etwa 65532 Bytes Speicher reserviert.

10.5 Hilfsfunktionen

dwGetEidnS *Signatur*

```
FUNCTION dwGetEidnS : DWORD
VAR_INPUT
  uIDN : UINT;
END_VAR
```

Mit Hilfe dieser Funktion kann die EIDN eines S-Parameters ohne SE/SI erzeugt werden.

dwGetEidnS_SESI *Signatur*

```
FUNCTION dwGetEidnS_SESI : DWORD
VAR_INPUT
  uIDN : UINT;
  usiSI : USINT;
  usiSE : USINT;
END_VAR
```

Mit Hilfe dieser Funktion kann die EIDN eines S-Parameters mit SE/SI erzeugt werden.

dwGetEidnP *Signatur*

```
FUNCTION dwGetEidnP : DWORD
VAR_INPUT
  uIDN : UINT;
END_VAR
```

Mit Hilfe dieser Funktion kann die EIDN eines P-Parameters ohne SE/SI erzeugt werden.

ML_ParamBase.library

dwGetEidnP_SESI

Signatur

```

FUNCTION dwGetEidnP_SESI : DWORD
VAR_INPUT
    uIDN : UINT;
    usiSI : USINT;
    usiSE : USINT;
END_VAR

```

Mit Hilfe dieser Funktion kann die EIDN eines P-Parameters mit SE/SI erzeugt werden.

10.6 Konstanten

usiSIZE_OF_SERCOSHEADER

Konstante für die Größe des sercos spezifischen Headers ([Datenzugriff, Seite 468](#)).

STRING_NO_UNIT

Konstante zur Darstellung, dass ein Parameter keine Einheit besitzt.

SERCOS_ERROR

Fehlercode	Wert	Beschreibung
K_STS_OK	16#0000	Operation erfolgreich
K_STS_IDN_NONE	16#1001	IDN ungültig
K_STS_IDN_INVALIDACCESS	16#1009	Ungültiger Zugriff auf Element 1
K_STS_NAME_NONE	16#2001	Name existiert nicht
K_STS_NAME_TOOSHORT	16#2002	Name zu kurz übertragen
K_STS_NAME_TOOLONG	16#2003	Name zu lang übertragen
K_STS_NAME_READONLY	16#2004	Name ist read only
K_STS_NAME_WRITEPROTECT	16#2005	Name ist aktuell schreibgeschützt
K_STS_ATTR_NONE	16#3001	Attribut existiert nicht
K_STS_ATTR_TOOSHORT	16#3002	Attribut zu kurz übertragen
K_STS_ATTR_TOOLONG	16#3003	Attribut zu lang übertragen
K_STS_ATTR_READONLY	16#3004	Attribut ist read only
K_STS_ATTR_WRITEPROTECT	16#3005	Attribut ist aktuell schreibgeschützt
K_STS_UNIT_NONE	16#4001	Einheit existiert nicht
K_STS_UNIT_TOOSHORT	16#4002	Einheit zu kurz übertragen
K_STS_UNIT_TOOLONG	16#4003	Einheit zu lang übertragen
K_STS_UNIT_READONLY	16#4004	Einheit ist read only
K_STS_UNIT_WRITEPROTECT	16#4005	Einheit ist aktuell schreibgeschützt
K_STS_MIN_NONE	16#5001	Minimalwert existiert nicht
K_STS_MIN_TOOSHORT	16#5002	Minimalwert zu kurz übertragen
K_STS_MIN_TOOLONG	16#5003	Minimalwert zu lang übertragen
K_STS_MIN_READONLY	16#5004	Minimalwert ist read only
K_STS_MIN_WRITEPROTECT	16#5005	Minimalwert ist aktuell schreibgeschützt

ML_ParamBase.library

K_STS_MAX_NONE	16#6001	Maximalwert existiert nicht
K_STS_MAX_TOOSHORT	16#6002	Maximalwert zu kurz übertragen
K_STS_MAX_TOOLONG	16#6003	Maximalwert zu lang übertragen
K_STS_MAX_READONLY	16#6004	Maximalwert ist read only
K_STS_MAX_WRITEPROTECT	16#6005	Maximalwert ist aktuell schreibgeschützt
K_STS_DATA_NONE	16#7001	Datum existiert nicht
K_STS_DATA_TOOSHORT	16#7002	Datum zu kurz übertragen
K_STS_DATA_TOOLONG	16#7003	Datum zu lang übertragen
K_STS_DATA_READONLY	16#7004	Datum ist read only
K_STS_DATA_WRITEPROTECT	16#7005	Datum ist aktuell schreibgeschützt
K_STS_DATA_SMALLER	16#7006	Datum ist kleiner als Minimalwert
K_STS_DATA_GREATER	16#7007	Datum ist größer als Maximalwert
K_STS_DATA_INVALID	16#7008	Datum ist ungültig (z.B. Bitkombination)
K_STS_DATA_PASSWORD	16#7009	Datum ist durch ein Passwort geschützt
K_STS_DATA_CONFCYCLIC	16#700A	Parameter ist zyklisch konfiguriert
K_STS_DATA_INVALIDINDEX	16#700B	Index ist ungültig
K_STS_DATA_WRITEPROTECTOS	16#700C	Parameter ist schreibgeschützt
K_STS_DATA_ISNAN	16#700D	Datum ist Not a Number (NaN)
K_STS_DATA_CMDACTIVE	16#7010	Kommando bereits aktiv
K_STS_DATA_CMDNOTINTERR	16#7011	Kommando nicht unterbrechbar
K_STS_DATA_CMDNOTEXECUTE	16#7012	Kommando zur Zeit nicht ausführbar
K_STS_DATA_CMDNOTEXECPARAM	16#7013	Kommando wegen falschen Parameter nicht ausführbar
K_STS_DATA_MAXRANGEEXCEED	16#710C	Datum überschreitet maximalen Bereich
K_STS_DATA_NO_ACCESS	16#8001	Zur Zeit kein Zugriff möglich

Tab. 10-2: SERCOS_ERROR

SERCOS_CMD_STATES

Kommandostatus	Wert	Beschreibung
K_STATE_NOT_ACTICE	2#0000	Kommando nicht aktiv
K_STATE_WAIT_FOR_EXEC	2#0001	Kommando ist gesetzt, wartet auf Freigabe

ML_ParamBase.library

K_STATE_READY	2#0011	Kommando ist fertig ausgeführt, wartet auf Rücksetzen
K_STATE_INTERRUPTED	2#0101	Kommando ist unterbrochen, wartet auf Freigabe oder Abbruch
K_STATE_IN_WORK	2#0111	Kommando ist aktiv
K_STATE_ERROR	2#1111	Kommando mit Fehler abgebrochen

Tab. 10-3: SERCOS_CMD_STATES

SERCOS_CMD_RETURN

Rückgabewert	Bedeutung
K_CMD_INIT_OK	Kommando-Init-Funktion ohne Fehler beendet
K_CMD_INIT_NOK	Kommando-Init-Funktion mit Fehler beendet
K_CMD_CYCLIC_IN_WORK	Zyklischer Kommandoaufruf unterbrochen (Zeitschlitz)
K_CMD_CYCLIC_READY	Zyklischer Kommandoaufruf ohne Fehler beendet
K_CMD_CYCLIC_ERROR	Zyklischer Kommandoaufruf mit Fehler beendet
K_CMD_ABORT_OK	Kommando-Abort-Funktion ohne Fehler beendet
K_CMD_ABORT_NOK	Kommando-Abort-Funktion mit Fehler beendet

Tab. 10-4: SERCOS_CMD_RETURN

11 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

Achsdaten über Funktionsvariablenzugriff.....	94
Achsdaten über Funktionsvariablenzugriff synchron im C2C-Verbund	96
Achstypen und durch sie unterstützte Funktionsbausteine.....	344
Active.....	22
Allgemeines zur geregelten Bewegung.....	265
Allgemeines zur Nutzung des zyklischen Datenkanals.....	252
ANSI Z535.6-2006.....	14
Anwenderdiagnosen.....	123
APP_INFO.....	162
AXIS_GROUP_REF.....	39
AXIS_REF.....	37, 361
Anwendung Feldbusantrieb.....	429
AxisData[] - Achsdaten im Shared Memory.....	91
AxisData[], Anwendung Feldbusantrieb.....	430

B

Basisklasse	
ML_ControllerBase.....	433
Bibliothek	
ML_Base.library.....	79
ML_ControllerBase.....	433
ML_IPOImpactControl.library.....	443
ML_ParamBase.library.....	453
ML_PLCopen.library.....	169
RIL_CommonTypes.library.....	35
RMB_PLCopenFieldBus.library.....	359

C

CommandAborted.....	22
CONTROLS.....	37

D

Daten eines Reglers.....	434
Datentypen zum Aufsynchronisieren in den lagegeregelten Synchron-Betriebsarten.....	66
Datentypen zur Fehlerdiagnose bei IndraMotion-Bausteinen.....	69
Datentypen zur Referenzierung von Objekten bei IndraMotion-Bausteinen.....	36
Datentypen zur Unterstützung von Kurvenscheiben, Motion- und FlexProfilen.....	41
Datentypen zur Unterstützung von Mess- taster-Funktionen.....	73
Datentypen zur Unterstützung von Motion- und FlexProfile-Funktionen.....	82
Datentypen zur Wiedergabe des Datum und absoluter Zeitangaben bei IndraMotion- Bausteinen.....	71
Diagnosen von MB_ReadxxxParameter / MB_WritexxxParameter / Direktzugriff.....	355

Diagnosen zum Kinematikzustand.....	356
Done.....	22
Doppelachsmodule	
Profibus.....	386
DV_Axis[].....	94
DV_Control[].....	101
DV_Osci[].....	115
DV_PlS[].....	107
DV_Probe[].....	112
dwGetEidnP.....	469
dwGetEidnP_SESI.....	470
dwGetEidnS.....	469
dwGetEidnS_SESI.....	469

E

E_LOG_CLASS.....	128
E_LOG_FACILITY.....	128
E_LOG_SEVERITY.....	128
E_SCALING_TYPE.....	162
E/A-Adresszuweisung der Feldbusantriebe.....	430
Enable.....	22
ERROR_CODE.....	69
ERROR_STRUCT.....	70
ERROR_TABLE.....	71
Error_Tables.....	346
MLC_TABLE.....	348
SERCOS_TABLE.....	346
Execute.....	22
ExecuteLock.....	22
EXT_SYS_TIME_DATE.....	72

F

Fehlertabellen	
MLC_TABLE.....	348
SERCOS_TABLE.....	346
Feldbus-Prozessdaten-Eingangsbeschreibung.....	416, 420
Feldbusantrieb	
Antrieb im Urzustand.....	424
Antrieb referenzieren.....	426
AXIS_REF.....	429
AxisData[].....	430
azyklischen Parameterkommunikation.....	430
Betriebsarteneinstellungen.....	426
Bibliotheken einbinden.....	426
Busteilnehmeradresse.....	426
E/A-Adressierung	430
im Projekt anlegen.....	422
Kommunikationseinstellungen IndraDrive....	426
kommunikationsnotwendige Einstellungen..	424
Länge des Parameterkanals.....	426
Prozessdaten und Profiltyp.....	426
Signalstatuswort.....	426
Signalsteuerwort.....	426
SPS-Programm, notwendige Deklaration....	429

Index

zyklischer Datenaustausch.....	430	IActualValue.....	433
Flexprofile		getIActualValue.....	433
Abarbeitungsarten.....	236	IAdditionalSetValue	
Aufsynchronisieren.....	233	getIAdditionalSetValue.....	434
Auswahl Koordinatensystem.....	233	IDigitalIO	
Ereignissteuerung.....	239	getIDigitalIO.....	434
Umschalten.....	233	IL_NULL.....	36
Funktionsbaustein-Präfixe.....	359	IL_SercosAttribute.....	137
Funktionsbausteine zum Parameterzugriff		IL_SercosElementsToParaNo.....	140
Profibus.....	383	IL_SercosStringToParaNo.....	141
sercos.....	278	IL_SetDateTime.....	151
Funktionsbausteine zur Achsgruppensteuerung.....	250	IModuleControl.....	433
Funktionsbausteine zur Einzelachssteuerung..	176	getIModuleControl.....	434
Funktionsbausteine zur geregelten Bewegung.	265	IndraDrive	
Funktionsbausteine zur Multiachssteuerung.....	199	Busteilnehmeradresse.....	426
Funktionsbausteine zur Nutzung der Mess-		Einstellungen Prozessdaten und Profiltypen.	426
taster-Funktion.....	325	Länge des Parameterkanals.....	426
Funktionsbausteine zur Nutzung des zyklischen Datenkanals.....	252	Signalstatuswort.....	426
Funktionsvariablenzugriff		Signalsteuerwort	426
DV_Axis[].....	94	IndraWorks DS	
DV_Control[].....	101	Einstellungen Feldbusantriebe.....	424
DV_Oscil[].....	115	Indraworks Engineering Desktop	
DV_Plis[].....	107	Einstellungen Feldbusantriebe.....	424
DV_Probe[].....	112	INFO_APPLICATIONS.....	163
Funktionsvariablenzugriff im C2C-Verbund		InGear.....	22
GV_LinkAxis[].....	96	InOperation.....	22
		InSync.....	22
G		IntelliSense	
Gefahrenhinweise.....	13	Achsen.....	91, 94
GV_LinkAxis[].....	96	Messstasterdaten.....	110, 112
GVL_A_GlobalVarIdx.....	131	Nockenschaltwerkdaten.....	105, 107
GVL_A_Param_ID.....	134	Oszilloskopdaten.....	115
GVL_Base.....	129	Steuerungsdaten.....	100, 101
GVL_BaseConstant.....	130	IntelliSense einer Funktionsvariablen synchron im C2C-Verbund.....	97
GVL_C_Param_ID.....	134	Interface IActualValue.....	436
GVL_FieldBusDrives.....	362, 429	runActualValue.....	436
GVL_M_Param_ID.....	135	Interface IModuleControl.....	436
GVL_N_Param_ID.....	135	InitModule.....	436
GVL_O_Param_ID.....	136	SwitchToOperationMode.....	437
GVL_P_Param_ID.....	136, 360	SwitchToParamMode.....	437
GVL_S_Param_ID.....	137, 359	Interface IOperationModeHandler.....	437
		bCyclic.....	437
H		bInitOperationCard.....	437
Helpdesk.....	473	Interface IOutputData.....	437
HNC100-3X.....	254	runOutputData.....	437
Hotline.....	473	InTorque.....	22
		InVelocity.....	22
I		IOperationModeHandler.....	433
I_PARAM_CmdAccess		getIOperationModeHandler.....	434
dwCmdAbort.....	459	IOutputData.....	433
dwCmdCyclic.....	459	getIOutputData.....	434
dwCmdInit.....	459	IRetainHandler	
I_PARAM_ReadAccess		getIRetainHandler.....	434
dwReadData.....	458		
dwWriteData.....	459	K	
		KIN_OBJECTS.....	40

Kommandohandling..... 467

L

Länge des Parameterkanals, IndraDrive..... 426

Logbuch-Zugriffe..... 123

M

MB_ChangeCamData..... 225

MB_ChangeCamData, Besonderheiten bei

SERCOS-basierten Achsen..... 226

MB_ChangeFlexEventSet..... 242

MB_ChangeFlexProfileSet..... 238

MB_ChangeProfileSet..... 217

MB_ChangeProfileSet, Besonderheiten bei

SERCOS-basierten Achsen..... 220

MB_ChangeProfileStep..... 222

MB_ChangeProfileStep, Besonderheiten bei

SERCOS-basierten Achsen..... 224

MB_CHANGESTEP_MODE..... 82

MB_Command..... 281

MB_Command, Besonderheiten bei

SERCOS-basierten Achsen..... 284

MB_DIFF_EVAL..... 73

MB_EVENT_ACTION..... 83

MB_EVENT_TRIGGER..... 84

MB_EXECUTION_MODE..... 84

MB_EXPECT_WINDOW_MODE..... 74

MB_FLEX_EVENT..... 85

MB_GearInPos 247

MB_GearInPos, Besonderheiten bei Sercos

basierten Achsen..... 248

MB_Home..... 197, 371

MB_Home, Besonderheiten bei Sercos ba-

sierten Achsen..... 198

MB_Home, Besonderheiten der Feldbusan-

triebe..... 372

MB_IDN..... 81, 361

MB_INTERPOLATION_MODE..... 75

MB_MASTER_TYPE..... 85

MB_MotionProfile..... 211

MB_MotionProfile, Besonderheiten bei

Sercos basierten Achsen..... 214

MB_Phasing..... 202

MB_PhasingSlave..... 204

MB_POSITION_EVAL..... 75

MB_PROBE_CONFIG..... 77

MB_PROBE_CONFIG, Besonderheiten bei

SERCOS-basierten Achsen..... 77

MB_PROBE_DATA..... 76

MB_PROBE_INPUT..... 76

MB_PROBE_MODULE..... 75

MB_PROFILE_START_POINT..... 86

MB_PROFILE_STEP..... 87

MB_ReadLintParameter..... 285

MB_ReadLintParameter, Besonderheiten

bei SERCOS-basierten Achsen..... 287

MB_ReadListParameter..... 287, 387

MB_ReadListParameter, Besonderheiten bei
SERCOS-basierten Achsen..... 289

MB_ReadListParameter, Besonderheiten

der Feldbusantriebe..... 390

MB_ReadListSegParameterPtr..... 390

MB_ReadListSegParameterPtr, Besonder-

heiten der Feldbusantriebe..... 393

MB_ReadLrealListParameter..... 290

MB_ReadLrealListParameter, Besonderhei-

ten bei SERCOS-basierten Achsen..... 292

MB_ReadLrealParameter..... 292

MB_ReadLrealParameter, Besonderheiten

bei SERCOS-basierten Achsen..... 293

MB_ReadMaxRealValue..... 294, 393

MB_ReadMaxRealValue, Besonderheiten

bei SERCOS-basierten Achsen..... 295

MB_ReadMaxRealValue, Besonderheiten

der Feldbusantriebe..... 394

MB_ReadMaxValue..... 295, 395

MB_ReadMaxValue, Besonderheiten bei

SERCOS-basierten Achsen..... 297

MB_ReadMaxValue, Besonderheiten der

Feldbusantriebe..... 396

MB_ReadMinRealValue..... 297, 396

MB_ReadMinRealValue, Besonderheiten bei

SERCOS-basierten Achsen..... 298

MB_ReadMinRealValue, Besonderheiten

der Feldbusantriebe..... 397

MB_ReadMinValue..... 299, 398

MB_ReadMinValue, Besonderheiten bei

SERCOS-basierten Achsen..... 300

MB_ReadMinValue, Besonderheiten der

Feldbusantriebe..... 399

MB_ReadName..... 300

MB_ReadName, Besonderheiten bei

SERCOS-basierten Achsen..... 301

MB_ReadParameter..... 302, 399

MB_ReadParameter, Besonderheiten bei

SERCOS-basierten Achsen..... 303

MB_ReadParameter, Besonderheiten der

Feldbusantriebe..... 401

MB_ReadRealParameter..... 303, 401

MB_ReadRealParameter, Besonderheiten

bei SERCOS-basierten Achsen..... 304

MB_ReadRealParameter, Besonderheiten

der Feldbusantriebe..... 402

MB_ReadSercosAttribute..... 305, 403

MB_ReadSercosAttribute, Besonderheiten

bei SERCOS-basierten Achsen..... 306

MB_ReadSercosAttribute, Besonderheiten

der Feldbusantriebe..... 404

MB_ReadSercosDataStatus..... 306

MB_ReadSercosDataStatus, Besonderhei-

ten bei SERCOS-basierten Achsen..... 308

MB_ReadStringParameter..... 308, 404

MB_ReadStringParameter, Besonderheiten

bei SERCOS-basierten Achsen..... 309

Index

MB_ReadStringParameter, Besonderheiten der Feldbusantriebe.....	405	MC_MoveAbsolute, Besonderheiten der Feldbusantriebe.....	374
MB_ReadUnit.....	310	MC_MoveAdditive.....	183, 375
MB_ReadUnit, Besonderheiten bei SERCOS-basierten Achsen.....	311	MC_MoveAdditive, Besonderheiten bei SERCOS-basierten Achsen.....	186
MB_SERCOS_DATA_STATUS.....	88	MC_MoveAdditive, Besonderheiten der Feldbusantriebe.....	376
MB_SetPositionControlMode.....	196	MC_MoveRelative.....	186, 377
MB_STEP_TYPE.....	88	MC_MoveRelative, Besonderheiten bei SERCOS-basierten Achsen.....	189
MB_Stop.....	369	MC_MoveRelative, Besonderheiten der Feldbusantriebe.....	379
MB_Stop, Besonderheiten der Feldbusantriebe.....	371	MC_MoveVelocity.....	190, 380
MB_SYNC_TYPE.....	88	MC_MoveVelocity, Besonderheiten bei SERCOS-basierten Achsen.....	193
MB_WriteLintParameter.....	311	MC_MoveVelocity, Besonderheiten der Feldbusantriebe.....	383
MB_WriteLintParameter, Besonderheiten bei SERCOS-basierten Achsen.....	312	MC_Power.....	176, 365
MB_WriteListParameter.....	313, 406	MC_Power, Besonderheiten bei Sercos basierten Achsen.....	177
MB_WriteListParameter, Besonderheiten bei SERCOS-basierten Achsen.....	315	MC_Power, Besonderheiten der Feldbusantriebe.....	366
MB_WriteListParameter, Besonderheiten der Feldbusantriebe.....	408	MC_ReadActualPosition.....	321, 415
MB_WriteListSegParameterPtr.....	408	MC_ReadActualPosition, Besonderheiten bei SERCOS-basierten Achsen.....	322
MB_WriteListSegParameterPtr, Besonderheiten der Feldbusantriebe.....	411	MC_ReadActualPosition, Besonderheiten der Feldbusantriebe.....	416
MB_WriteLrealParameter.....	315	MC_ReadAxisError.....	323, 417
MB_WriteLrealParameter, Besonderheiten bei SERCOS-basierten Achsen.....	317	MC_ReadAxisError, Besonderheiten bei SERCOS-basierten Achsen.....	324
MB_WriteParameter.....	317, 411	MC_ReadAxisError, Besonderheiten der Feldbusantriebe.....	418
MB_WriteParameter, Besonderheiten bei SERCOS-basierten Achse.....	318	MC_ReadStatus.....	324, 418
MB_WriteParameter, Besonderheiten der Feldbusantriebe.....	412	MC_ReadStatus, Besonderheiten bei sercos basierten Achsen.....	325
MB_WriteRealParameter.....	318, 412	MC_ReadStatus, Besonderheiten der Feldbusantriebe.....	420
MB_WriteRealParameter, Besonderheiten bei SERCOS-basierten Achsen.....	319	MC_RemoveAxisFromGroup.....	251
MB_WriteRealParameter, Besonderheiten der Feldbusantriebe.....	413	MC_Reset.....	177, 366
MB_WriteStringParameter.....	320, 414	MC_Reset, Besonderheiten bei SERCOS-basierten Achsen.....	178
MB_WriteStringParameter, Besonderheiten bei SERCOS-basierten Achsen.....	321	MC_Reset, Besonderheiten der Feldbusantriebe.....	367
MB_WriteStringParameter, Besonderheiten der Feldbusantriebe.....	415	MC_START_MODE.....	66
MC_AddAxisToGroup.....	250	MC_STEP_MODE.....	42
MC_CAM_ID.....	42	MC_Stop.....	179, 368
MC_CamIn.....	206	MC_Stop, Besonderheiten bei SERCOS-basierten Achsen.....	180
MC_CamIn, Besonderheiten bei Sercos basierten Achsen.....	208	MC_Stop, Besonderheiten der Feldbusantriebe.....	369
MC_CamOut.....	210	MC_SYNC_DIRECTION.....	68
MC_CamOut, Besonderheiten bei SERCOS-basierten Achsen.....	211	MC_TorqueControl.....	193
MC_GearIn.....	243	MC_TorqueControl, Besonderheiten bei Sercos basierten Achsen.....	195
MC_GearIn, Besonderheiten bei Sercos basierten Achsen.....	244	Mehrachsmodule	
MC_GearOut.....	246	Profibus.....	386
MC_GearOut, Besonderheiten bei SERCOS-basierten Achsen.....	247	Messtaster-Daten im Shared Memory.....	110
MC_MoveAbsolute.....	180, 372		
MC_MoveAbsolute, Besonderheiten bei SERCOS-basierten Achsen.....	183		

Messtaster-Daten über Funktionsvariablen- zugriff.....	112	ML_GetMaxTemperature.....	159
ML_AbortTrigger.....	327	ML_GetNewestDiagnosisIndex.....	160
ML_AXISDATA_SM.....	117	ML_GetOhcFan.....	160
ML_Base, Datentypen an der Schnittstelle von IndraMotion-Bausteinen.....	79	ML_GetScalingType.....	160
ML_Base.library.....	79	ML_GetSubnetMask.....	146
ML_CONTROLDATA_SM.....	119	ML_GetSystemTimeUTC.....	152
ML_ControllerBase.....	433	ML_GetTargetInfo.....	161
ML_ControlOff.....	267	ML_GetTemperature.....	161
ML_ControlOn.....	266	ML_GetWarningCpuLoad.....	162
ML_ControlWriteActualData.....	268	ML_InitTouchProbe.....	328
ML_Convert.....	435	ML_IPOImpactControl.....	444
convFromAxis_Acceleration.....	436	mGetAcceleration.....	447
convFromAxis_Analog.....	436	mGetAxisControlledBy.....	449
convFromAxis_Force.....	436	mGetBreakBehaviour.....	447
convFromAxis_Position.....	436	mGetDirection.....	446
convFromAxis_Velocity.....	436	mGetNewState.....	445
convToAxis_Acceleration.....	436	mGetPosition.....	448
convToAxis_Analog.....	436	mGetValidValues.....	449
convToAxis_Force.....	436	mGetVelocity.....	449
convToAxis_Position.....	436	mSetAcceleration.....	446
convToAxis_Velocity.....	436	mSetBreakBehaviour.....	447
ML_Convert_DoublePrecision.....	435	mSetDirection.....	446
convFromAxis_Acceleration.....	436	mSetNewState.....	445
convFromAxis_Analog.....	436	mSetPosition.....	448
convFromAxis_Force.....	436	mSetValidValues.....	448
convFromAxis_Position.....	436	mSetVelocity.....	449
convFromAxis_Velocity.....	436	ML_IPOImpactControl-Methoden.....	445
convToAxis_Acceleration.....	436	ML_IPOImpactControl.library.....	443
convToAxis_Analog.....	436	ML_LINK.....	121
convToAxis_Force.....	436	ML_LOGBOOK.....	129
convToAxis_Position.....	436	ML_MessageHandler.....	442
convToAxis_Velocity.....	436	run_MSGH_ClearErrors.....	442
ML_DAYSOFWEEK.....	155	run_MSGH_ClearWarning.....	442
ML_DIAGNOSIS_LOG.....	163	run_MSGH_SetMessagecode.....	442
ML_FlexProfile.....	227	ML_OORV.....	437
ML_FlexProfile, Besonderheiten bei sercos basierten Achsen.....	230	ML_OORV_CONTROLLERDATA.....	434
ML_Generator.....	441	ML_OpenCyclicAnalogChannel.....	254
GenerateCopyActualAcc.....	441	ML_OpenCyclicAnalogChannelCtrl.....	269
GenerateCopyActualAna.....	441	ML_OpenCyclicPositionChannel.....	256
GenerateCopyActualPos.....	441	ML_OpenCyclicPositionChannelCtrl.....	271
GenerateCopyActualTrq.....	441	ML_OpenCyclicTorqueChannel.....	261
GenerateCopyActualVel.....	441	ML_OpenCyclicTorqueChannelCtrl.....	274
GenerateCopyCyclicAcceleration.....	442	ML_OpenCyclicVelocityChannel.....	263
GenerateCopyCyclicAnalog.....	442	ML_OpenCyclicVelocityChannelCtrl.....	276
GenerateCopyCyclicPosition.....	441	ML_ParamBase.....	
GenerateCopyCyclicTorque.....	442	Datentypen.....	468
GenerateCopyCyclicVelocity.....	442	Datenzugriff.....	468
ML_GetCpuLoad.....	156	Hilfsfunktionen.....	469
ML_GetDiagnosisLog.....	156	Konstanten.....	470
ML_GetFirmwareString.....	157	PARAM_Parametersystem.....	454
ML_GetGateway.....	143	SERCOS_ERROR.....	470
ML_GetHardwareInfoCtrl.....	157	Sprache.....	468
ML_GetInfoApplication.....	158	ML_ParamBase.library.....	453
ML_GetIpAddress.....	144	ML_PLCopen.library.....	169
ML_GetMacAddress.....	145	ML_PLSDATA_SM.....	120
		ML_PROBEDATA_SM.....	120
		ML_ReadDiagLog.....	123
		ML_SERCOS.....	121

Index

ML_SetAxisDiagEntry.....	125
ML_SetDiagEntry.....	126
ML_SetGateway.....	148
ML_SetIpAddress.....	149
ML_SetSubnetMask.....	150
ML_SetSystemTimeUTC.....	154
ML_SetWarningCpuLoad.....	162
ML_SYNC.....	121
ML_TouchProbe.....	334
ML_TouchProbeContinuous.....	337
ML_WriteCyclicAnalog.....	255
ML_WriteCyclicAnalogCtrl.....	270
ML_WriteCyclicPosition.....	258
ML_WriteCyclicPositionCtrl.....	272
ML_WriteCyclicTorque.....	262
ML_WriteCyclicTorqueCtrl.....	275
ML_WriteCyclicVelocity.....	264
ML_WriteCyclicVelocityCtrl.....	277
ML_WriteExpectWindow.....	339
MLC_TABLE.....	348
MotionControl-Wirkungskette.....	199
MP_Drive.....	420
MP_DriveParameter.....	420
MP_FB_IndraDrv_AcyclicCommunication.....	420

N

Name space.....	428
Namensraum.....	428
Nockenschaltwerkdaten über Funktionsvariablenzugriff.....	107

O

OBJECTS.....	38
Oszilloskopdaten über Funktionsvariablenzugriff.....	115

P

PARAM_CmdExecuter.....	465
dwAddCmd.....	466
dwClearStorage.....	466
dwSetControlCmd.....	466
PARAM_ExtSercosParameter.....	463
Administration.....	465
Attribute.....	463
Datum lesen.....	464
Datum schreiben.....	465
Defaultwerte laden.....	465
dwGetOperationData.....	464
dwLoadDefaultData.....	465
dwSetOperationData.....	465
iGetParamType.....	465
Parametertyp.....	465
Zugriffsmethoden.....	464
PARAM_Language.....	468
PARAM_Language_Array.....	468
PARAM_Parametersystem.....	454
addParameter.....	454

Anmelden von Parametern.....	454
Defaultwerte laden.....	456
dwCreateListAllCommands.....	457
dwCreateListAllParam.....	457
dwCreateListSaveOnCF.....	457
dwLoadDefaultParam.....	456
dwLoadFromCF.....	456
dwReadWriteParamType.....	458
dwSaveToCF.....	456
dwSetCyclicConfigured.....	458
Funktionalitäten.....	454
Init.....	454
Initialisierung.....	454
Liste aller auf der CF-Karte gesicherten Parameter.....	457
Liste aller Kommandos.....	457
Liste aller Parameter.....	457
LoadDefaultParam.....	456
Parameter lesen.....	455
Parameter schreiben.....	455
Parameter sichern.....	455
Parameter wiederherstellen.....	456
Parameterwerte synchronisieren.....	458
ParamRead.....	455
ParamWrite.....	455
Sprache umschalten.....	454
Zyklischer Schreibschutz.....	458
PARAM_SercosParameter.....	458
Administration.....	462
Attribut lesen.....	461
Attribute.....	460
Datenstatus lesen.....	461
Datum lesen.....	462
Datum schreiben.....	462
Defaultwerte laden.....	462
IDN lesen.....	461
Interfaces.....	458
Kommando-Interface.....	459
Lese-Interface.....	458
Maximalwert lesen.....	461
Minimalwert lesen.....	461
Name lesen.....	461
Parametertyp.....	463
Schreib-Interface.....	459
Zugriffsmethoden.....	461
Zyklischer Schreibschutz.....	463
Parameterzugriff aus SPS-Programm.....	132, 278
PlsData[] - Nockenschaltwerkdaten im Shared Memory.....	105
PROBECONFIG.....	172
ProbeSelect.....	172

R

REFERENCE_TO

Achsdaten.....	91, 94
Messtasterdaten.....	110, 112
Nockenschaltwerkdaten.....	105, 107

Oszilloskopdaten.....	115
Steuerungsdaten.....	100, 101
REFERENCE_TO auf eine Funktionsvariab-	
le synchron im C2C-Verbund.....	97
Reglerverwaltung.....	437
Anbinden des Reglers.....	437
Betriebsartenhandling.....	439
Vordefinierte Generatoren der Betriebs-	
arten.....	441
Zustandsmaschine Betriebszustand.....	438
Zyklischer Ablauf der Reglerverwaltung.....	437
Zyklischer Ablauf des Reglers.....	438
RIL_CommonTypes, Datentypen an der	
Schnittstelle von IndraMotion-Bausteinen.....	35
RIL_CommonTypes.library.....	35
RMB_PLCOpenFieldBus.library.....	359
Anwendung.....	427

S

sercos III, C2C-Verbund	
MotionControl-Wirkungskette.....	201
SERCOS_TABLE.....	346
Service-Hotline.....	473
Shared Memory	
AxisData[].....	91
ControlData[].....	100
PlsData[].....	105
ProbeData[].....	110
Shutdown.....	22
Sicherheitshinweise.....	13
Signalgrafik.....	14
Signalwörter.....	14
SM_EXT_TYPES.....	122
SM_TYPES.....	122
SPS steuert Antriebe über Feldbusverbindung.	421
SPS-Zugriff auf Parameterwerte	
mit Funktionsvariablenzugriff.....	95
mit Funktionsvariablenzugriff synchron im	
C2C-Verbund.....	97
über Shared Memory.....	92
Standardschnittstellen an Funktionsbausteinen.	21
Status bei Funktionsvariablenzugriff 96, 103,	
108, 114, 117	
Status bei Funktionsvariablenzugriff syn-	
chron im C2C-Verbund.....	98
Steuerungsdaten im Shared Memory.....	100
Steuerungsdaten über Funktionsvariablen-	
zugriff.....	101
Support.....	473
Symbole.....	15
SYS_TIME_DATE.....	72
SYS_TIME64.....	72

T

TARGET_INFO.....	163
TOUCHPROBE_REF.....	40
TP_OBJECTS.....	41

W

Warnhinweise.....	13

Z

Zeitbedarf bei der Bearbeitung von Motion-	
Funktionsbausteinen in der IndraMotion MLC..	342
Zeitbedarf für den Datentransport mit Read-	
und WriteParameter-Funktionsbausteinen.....	280
Zu dieser Dokumentation.....	13
Gültigkeit der Dokumentation.....	13
Zugriff auf Achsdaten über Shared Memory	
und Funktionsvariablenzugriff.....	89
Zugriff auf Daten einer Achse	
Profibus.....	385
Zugriff auf Daten einer Achse, Steuerung	
oder eines anderen Objektes.....	132
Zugriff auf Daten einer Achse, Steuerung,	
Kinematik (Achsgruppe) oder eines anderen	
Objektes.....	278
Zugriff auf Daten eines Feldbusantriebs.....	359
Zugriff auf Messtasterdaten über Shared	
Memory und Funktionsvariablenzugriff.....	109
Zugriff auf Nockenschaltwerkdaten über	
Shared Memory und Funktionsvariablenzugriff	
.....	104
Zugriff auf Oszilloskopdaten über Funktions-	
variablenzugriff.....	115
Zugriff auf Steuerungsdaten über Shared	
Memory und Funktionsvariablenzugriff.....	98
Zustandsdiagramm.....	363
Zustandsdiagramm einer realen Achse.....	173

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



R911336359