

MTX 15VRS

SPS-Interface

Projektierungsbeschreibung
R911400171

Ausgabe 02



Titel	MTX 15VRS SPS-Interface
Art der Dokumentation	Projektierungsbeschreibung
Dokumentations-Type	DOK-MTX***-PLC*INT*V15-PR02-DE-P
Interner Ablagevermerk	RS-dd6592dad401e6c90a347e864072ef42-2-de-DE-6
Änderungsverlauf	Ausgabe 02, 2020-11 Siehe Kap. 1.1 "Änderungsverlauf" auf Seite 1
Schutzvermerk	© Bosch Rexroth AG 2020 Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.
Verbindlichkeit	Die angegebenen Daten dienen allein der Produktbeschreibung und sind nicht als zugesicherte Eigenschaften im Rechtssinne zu verstehen. Änderungen im Inhalt der Dokumentation und Liefermöglichkeiten der Produkte sind vorbehalten.
Redaktion	Entwicklung Automationssysteme - Lösungsintegration HMI, NC-Steuerungen und Support, ToMa (MiSc)

Inhaltsverzeichnis

	Seite
1 Über diese Dokumentation.....	1
1.1 Änderungsverlauf.....	1
1.2 Zweck.....	1
1.3 Übersicht über Zielgruppen und Produktphasen.....	1
1.4 Erforderliche und ergänzende Dokumentationen MTX.....	2
1.4.1 Auswählen/Zusammenstellen.....	2
1.4.2 Konfigurieren.....	2
1.4.3 In Betrieb nehmen.....	3
1.4.4 Bedienen.....	4
1.4.5 OEM-Entwicklung.....	5
1.4.6 AddOns.....	5
1.5 Darstellung von Informationen.....	5
1.5.1 Gebrauch der Sicherheitshinweise.....	5
1.5.2 Bezeichnungen und Abkürzungen.....	7
1.5.3 Kundenfeedback.....	7
2 Wichtige Gebrauchshinweise.....	9
2.1 Bestimmungsgemäßer Gebrauch.....	9
2.1.1 Einführung.....	9
2.1.2 Einsatz- und Anwendungsbereiche.....	10
2.2 Nicht-bestimmungsgemäßer Gebrauch.....	10
3 Übersicht.....	11
4 Konfigurationen.....	15
4.1 Konfiguration des SPS-NC-Bit-Interface.....	15
4.1.1 Allgemeines.....	15
4.1.2 Das Globale Interface (General Interface).....	15
4.1.3 Das Kanal-Interface (Channel Interface).....	15
4.1.4 Das Achs-Interface (Axes Interface).....	16
4.1.5 Das Spindel-Interface (Spindle Interface).....	17
5 Übersicht Interface-Signale.....	19
5.1 Globale Ausgangssignale (SPS --> NC).....	19
5.2 Globale Eingangssignale (NC --> SPS).....	19
5.3 Kanalbezogene Ausgangssignale (SPS --> NC).....	20
5.4 Kanalbezogene Eingangssignale (NC --> SPS).....	22
5.5 Achsbezogene Ausgangssignale (SPS --> NC).....	24
5.6 Achsbezogene Eingangssignale (NC --> SPS).....	25
5.7 Spindelbezogene Ausgangssignale (SPS --> NC).....	27
5.8 Spindelbezogene Eingangssignale (NC --> SPS).....	29

	Seite
6 Globale Interface-Signale.....	31
6.1 Übersicht globale Interface-Signale.....	31
6.1.1 Allgemeines.....	31
6.1.2 Übersicht Ausgangssignale (SPS --> NC).....	31
6.1.3 Übersicht Eingangssignale (NC --> SPS).....	31
6.2 Signalbeschreibung.....	32
6.2.1 Ausgangssignale (SPS --> NC)	32
6.2.2 Eingangssignale (NC --> SPS)	35
7 Kanalbezogene Interface-Signale.....	37
7.1 Übersicht kanalbezogene Interface-Signale.....	37
7.1.1 Allgemeines.....	37
7.1.2 Übersicht kanalbezogene Ausgangssignale (SPS --> NC)	37
7.1.3 Übersicht Eingangssignale (NC --> SPS).....	39
7.2 Signalbeschreibung.....	41
7.2.1 Kanalbezogene Ausgangssignale (SPS --> NC)	41
7.2.2 Eingangssignale (NC --> SPS).....	53
8 Achsbezogene Interface-Signale.....	67
8.1 Übersicht achsbezogene Interface-Signale.....	67
8.1.1 Allgemeines.....	67
8.1.2 Übersicht achsbezogene Ausgangssignale (SPS --> NC).....	67
8.1.3 Übersicht achsbezogene Eingangssignale (NC --> SPS).....	69
8.2 Signalbeschreibung.....	71
8.2.1 Achsbezogene Ausgangssignale (SPS --> NC).....	71
8.2.2 Eingangssignale (NC --> SPS).....	83
9 Spindelbezogene Interface-Signale.....	95
9.1 Übersicht spindelbezogene Interface-Signale.....	95
9.1.1 Allgemeines.....	95
9.1.2 Übersicht Ausgangssignale (SPS --> NC)	95
9.1.3 Übersicht Eingangssignale (NC --> SPS)	97
9.2 Signalbeschreibung.....	98
9.2.1 Ausgangssignale (SPS --> NC).....	98
9.2.2 Eingangssignal (NC --> SPS).....	108
10 Hilfsfunktionen.....	121
10.1 Allgemeines.....	121
10.2 Bitcodierte Hilfsfunktionen	122
10.3 Nicht bitcodierte kanalunabhängige Hilfsfunktionen	123
10.4 Nicht bitcodierte kanalabhängige Hilfsfunktionen.....	124
11 Anbindung digitale I/Os.....	127

	Seite
12 Programmbausteine	129
12.1 Genereller Aufbau von Programmbausteinen	129
12.2 Achsenistwerte lesen (MT_ScsPos)	129
12.2.1 Funktion	129
12.2.2 Projektierungsbeispiel	130
12.2.3 Statusmeldungen	131
12.3 Programmanwahl/-abwahl (MT_ProgSel)	131
12.3.1 Funktion	131
12.3.2 Projektierungsbeispiel	134
12.3.3 Statusmeldungen	134
12.4 NC-Satzvorgabe (MT_NcBlk)	135
12.4.1 Funktion	135
12.4.2 Projektierungsbeispiel	136
12.4.3 Statusmeldungen	136
12.5 Erweiterte NC-Satzvorgabe (MT_NcBlkExt)	137
12.5.1 Funktion	137
12.5.2 Projektierungsbeispiel	138
12.5.3 Statusmeldungen	139
12.6 Kanal-Neustart (MT_ChanRestart)	139
12.6.1 Funktion	139
12.6.2 Projektierungsbeispiel	140
12.6.3 Statusmeldungen	141
12.7 Temperaturkompensation (MT_TempComp)	141
12.7.1 Funktion	141
12.7.2 Projektierungsbeispiel	142
12.7.3 Statusmeldungen	142
12.8 Maschinenparameter lesen (MT_CfgData)	143
12.8.1 Funktion	143
12.8.2 Projektierungsbeispiel	144
12.8.3 Statusmeldungen	145
12.9 Handraddaten (MT_Handw)	145
12.9.1 Funktion	145
12.9.2 Projektierungsbeispiel	146
12.9.3 Statusmeldungen	146
12.10 Sercos Parameter (MT_ScsData)	147
12.10.1 Funktion	147
12.10.2 Projektierungsbeispiel	148
12.10.3 Statusmeldungen	151
12.11 Werkzeuglisten bearbeiten (MT_P_DbRecList)	152
12.11.1 Funktion	152
12.11.2 Projektierungsbeispiel	155
12.11.3 Statusmeldungen	156
12.12 Datensätze bearbeiten (MT_P_DbData)	157
12.12.1 Funktion	157
12.12.2 Projektierungsbeispiel	158
12.12.3 Statusmeldungen	160

Inhaltsverzeichnis

	Seite
12.13 Datensatzlisten bearbeiten (MT_DbRecList).....	161
12.13.1 Funktion.....	161
12.13.2 Projektierungsbeispiel.....	164
12.13.3 Statusmeldungen.....	166
12.14 Datensätze bearbeiten (MT_DbData).....	167
12.14.1 Funktion.....	167
12.14.2 Projektierungsbeispiel.....	172
12.14.3 Statusmeldungen.....	173
12.15 Datenbanktabelle sichern (MT_DbSave).....	174
12.15.1 Funktion.....	174
12.15.2 Projektierungsbeispiel.....	176
12.15.3 Statusmeldungen.....	177
12.16 Datenbanktabelle laden (MT_DbLoad).....	178
12.16.1 Funktion.....	178
12.16.2 Projektierungsbeispiel.....	179
12.16.3 Statusmeldungen.....	180
12.17 Werkzeugkorrektur (MT_TCorr).....	181
12.17.1 Funktion.....	181
12.17.2 Projektierungsbeispiel.....	183
12.17.3 Statusmeldungen.....	185
12.18 Lesen/Schreiben von permanenten CPL-Variablen (MT_CplData).....	185
12.18.1 Funktion.....	185
12.18.2 Projektierungsbeispiel.....	187
12.18.3 Statusmeldungen.....	188
12.19 Lesen/Schreiben von permanenten CPL-Variablen mit Typkonvertierung (MT_CplDataConv).....	188
12.19.1 Funktion.....	188
12.19.2 Projektierungsbeispiel.....	190
12.19.3 Statusmeldungen.....	191
12.20 Systemdaten lesen (MT_SD_RD).....	192
12.20.1 Funktion.....	192
12.20.2 Projektierungsbeispiel.....	193
12.20.3 Statusmeldungen.....	194
12.21 Systemdaten schreiben (MT_SD_WR).....	195
12.21.1 Funktion.....	195
12.21.2 Projektierungsbeispiel.....	196
12.21.3 Statusmeldungen.....	197
12.22 Diagnosedaten lesen (MT_DiagCode).....	198
12.22.1 Allgemeines.....	198
12.22.2 Bausteinparameter.....	199
12.22.3 Projektierungsbeispiel.....	202
12.22.4 Statusmeldungen.....	202
12.23 Diagnosetexte lesen (MT_DiagText).....	203
12.23.1 Allgemeines.....	203
12.23.2 Bausteinparameter.....	204
12.23.3 Projektierungsbeispiel.....	206
12.23.4 Statusmeldungen.....	206

	Seite
12.24 Lesen der Motion Control Data-Services (MT_MCoDS).....	206
12.24.1 Allgemeines.....	206
12.24.2 Bausteinparameter.....	207
12.24.3 Projektierungsbeispiel.....	207
12.24.4 Statusmeldungen.....	208
12.25 Automatischer Sercos III I/O-Tausch (MT_S3AutoAdapt).....	208
12.25.1 Allgemeines.....	208
12.25.2 Bausteinparameter.....	208
12.25.3 Projektierungsbeispiel.....	208
12.25.4 Statusmeldungen.....	209
12.26 Sercos Ringtopologie auslesen (MT_RingTop).....	209
12.26.1 Allgemeines.....	209
12.26.2 Bausteinparameter.....	209
12.26.3 Projektierungsbeispiel.....	210
12.26.4 Statusmeldungen.....	210
12.27 Log-Eintrag im Aktionsrekorder (MT_LogData).....	211
12.27.1 Allgemeines.....	211
12.27.2 Bausteinparameter.....	211
12.27.3 Projektierungsbeispiel.....	213
12.27.4 Statusmeldungen.....	213
12.28 Systemdaten segmentiert lesen (MT_SD_Rd_Seg).....	214
12.28.1 Allgemeines.....	214
12.28.2 Bausteinparameter.....	214
12.28.3 Projektierungsbeispiel.....	215
12.28.4 Statusmeldungen.....	215
12.29 Systemdaten segmentiert schreiben (MT_SD_Wr_Seg).....	216
12.29.1 Allgemeines.....	216
12.29.2 Bausteinparameter.....	217
12.29.3 Projektierungsbeispiel.....	217
12.29.4 Statusmeldungen.....	218
12.30 Fast Action Eingangsdaten lesen (MT_FaDataRead).....	219
12.30.1 Allgemeines.....	219
12.30.2 Bausteinparameter.....	219
12.30.3 Projektierungsbeispiel.....	220
12.30.4 Statusmeldungen.....	221
12.31 Fast Action Ausgangsdaten schreiben(MT_FaDataWrite).....	222
12.31.1 Allgemeines.....	222
12.31.2 Bausteinparameter.....	222
12.31.3 Statusmeldungen.....	222
12.31.4 Projektierungsbeispiel.....	223
13 Konfiguration von SPS-spezifischen Daten in IndraWorks.....	225
13.1 Konfiguration der Maschinenzustandsanzeige (MZA).....	225
13.1.1 Allgemeines.....	225
13.1.2 Parametrierung der Maschinenzustandsanzeige (MZA).....	225
13.1.3 Konfiguration des MZA-Bit-Interface.....	225

	Seite
13.1.4	Aufbau der MZA-Datei..... 226
13.1.5	Aktivierung der Meldungen..... 227
13.1.6	Ergänzende Hinweise..... 228
14	MT_BaseAdditional.library..... 229
14.1	Einführung und Übersicht..... 229
14.2	Standardschnittstellen an Funktionsbausteinen..... 231
14.2.1	Motivation..... 231
14.2.2	Bausteintypen..... 231
14.2.3	Bausteintypen und ihre Schnittstellen..... 232
14.2.4	Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung..... 234
14.2.5	Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung..... 237
14.2.6	Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung..... 240
14.2.7	Ein- und Ausgänge flankengesteuerter Funktionsbausteine, permanente Bearbeitung..... 243
14.3	NC-Kanal..... 245
14.3.1	Übersicht..... 245
14.3.2	MT_ChNumberValidation, Gültigkeit der Kanalnummer prüfen..... 246
14.3.3	MT_ChGetState, Aktuellen Kanalstatus ermitteln..... 248
14.3.4	MT_ChGetActiveProgram, Aktives NC-Programm ermitteln..... 250
14.3.5	MT_ChNcProgProgress, Aktuellen Programmfortschritt ermitteln..... 252
14.3.6	MT_ChSetReentryBreakPoint, Zum Wiederanfahren den Unterbrechungspunkt speichern..... 258
14.3.7	MT_ChSetFeedOverride, Kanalvorschub ändern..... 261
14.3.8	MT_ChGetFeedOverride, Aktiven Vorschub-Override ermitteln..... 262
14.3.9	MT_ChGetOPMode, Aktive Betriebsart ermitteln..... 263
14.3.10	MT_ChSetMainOPMode, Ändern der NC-Hauptbetriebsarten..... 266
14.3.11	MT_ChSetOPMode, Ändern der NC-Betriebsarten..... 270
14.3.12	MT_ChOPMode, Ändern der NC-Hauptbetriebsarten..... 275
14.4	Achsen..... 280
14.4.1	Übersicht..... 280
14.4.2	MT_DrvGetSafeState, Aktiven Sicherheitstechnik-Betriebszustand ermitteln..... 281
14.4.3	MT_AxGetChIndex, Kanalzuordnung der Achse ermitteln..... 285
14.4.4	MT_AxHandwheelAcs, Handrad ändern..... 286
14.4.5	MT_AxProbeMonitor, Messtasterachse überwachen..... 288
14.4.6	MT_AxGetJogMode, Aktiven JogMode ermitteln..... 291
14.4.7	MT_AxSetJogMode, Ändern des JodMode..... 292
14.4.8	MT_AxJogMode, Ändern und Ermitteln der aktiven Tippdistanz..... 296
14.4.9	MT_AxAsyncGetOPMode, Betriebsart einer asynchronen Achse ermitteln..... 301
14.4.10	MT_AxAsyncSetOPMode, Betriebsart einer asynchronen Achse ändern..... 302
14.4.11	MT_AxAsyncOPMode, Betriebsart einer asynchronen Achse 305
14.4.12	MT_AxAsyncSetOverride, Vorschub für asynchrone Achse ändern..... 308
14.4.13	MT_AxGetPowerState, Antriebsstatus ermitteln..... 309
14.4.14	MT_AxGetPowerStateExt, Antriebsstatus incl. Fehler- und Warnungsdetails ermitteln..... 311
14.4.15	MT_AxSetPowerOn, Setzt die Reglerfreigabe für eine Achse..... 312
14.4.16	MT_AxSetPowerOff, Setzt die Reglerfreigabe für eine Achse..... 314
14.4.17	MT_AxPower, Reglerfreigabe und Status einer Achse..... 316
14.5	Spindeln..... 319

	Seite
14.5.1	Übersicht..... 319
14.5.2	MT_SpSetOverride, Vorschub für eine Spindel ändern..... 320
14.5.3	MT_SpGetJogSpeed, Aktive Tippgeschwindigkeit ermitteln..... 321
14.5.4	MT_SpSetJogSpeed, Ändern der Spindel-Tippgeschwindigkeit..... 322
14.5.5	MT_SpJogSpeed, Ändern und Ermitteln der aktiven Tippgeschwindigkeit..... 325
14.5.6	MT_SpGetPowerState, Antriebsstatus ermitteln..... 328
14.5.7	MT_SpGetPowerStateExt, Antriebsstatus incl. Fehler- und Warnungsdetails ermitteln..... 330
14.5.8	MT_SpSetPowerOn, Setzt die Reglerfreigabe für eine Spindel..... 332
14.5.9	MT_SpSetPowerOff, Setzt die Reglerfreigabe für eine Spindel..... 334
14.5.10	MT_SpPower, Reglerfreigabe und Status einer Spindel..... 336
14.6	Werkzeugmanagement..... 339
14.6.1	Übersicht..... 339
14.6.2	MT_TmToolLife, Werkzeug Standzeitählung 339
14.6.3	MT_TmChGetActiveEdge, Aktive Schneide des aktiven Werkzeug ermitteln..... 345
14.7	Sonstige..... 346
14.7.1	MT_DistanceAccumulator, Berechnet die Wegstrecke aus den angegebenen Positionsdaten..... 346
14.7.2	MT_TimeAccumulator, Berechnet die verstrichene Zeit 349
14.7.3	MT_GetRandomNumber, Generiert eine ganzzahlige Zufallszahl..... 351
14.8	Data Types - Datentypen..... 352
14.8.1	MT_CHANNEL_OPERATION_MODE, Struktur mit allen Kanalbetriebsarten..... 352
14.8.2	MT_CHANNEL_STATE, Struktur enthält alle Kanalzustände..... 353
14.8.3	MT_SD_SYS_TOOL, Struktur von Systemdatum SD.SysTool..... 353
14.8.4	MT_AXIS_ASYNC_OPERATION_MODE, Struktur mit allen Achsbetriebsarten..... 354
14.8.5	MT_AXIS_JOG_MODE, Struktur mit Tippdistanzen einer Achse..... 355
14.8.6	MT_DRIVE_ERROR_CLASS_1, Struktur Fehlerdetails aus S-0-0011 einer Achse..... 355
14.8.7	MT_DRIVE_ERROR_CLASS_2, Struktur Warnungsdetails aus S-0-0012 einer Achse..... 356
14.8.8	MT_DRIVE_POWER_STATE, Antriebsstatus..... 356
14.8.9	MT_DRIVE_SAFE_OP_MODE, Struktur enthält Sicherheitsbetriebszustand eines Antriebs 357
14.8.10	MT_SPINDLE_JOG_SPEED, Struktur mit Tippgeschwindigkeiten einer Spindel..... 357
15	MT_PLCOpen.library..... 359
15.1	Einführung und Übersicht..... 359
15.2	Globale_Variablen..... 360
15.2.1	Data Types - Datentypen..... 360
15.3	POUs..... 361
15.4	MB_Command..... 361
15.5	MB_ReadListParameter..... 366
15.6	MB_ReadMaxRealValue..... 368
15.7	MB_ReadMaxValue..... 370
15.8	MB_ReadMinRealValue..... 372
15.9	MB_ReadMinValue..... 374
15.10	MB_ReadName..... 376
15.11	MB_ReadParameter..... 377
15.12	MB_ReadRealParameter..... 379
15.13	MB_ReadSercosAttribute..... 381
15.14	MB_ReadSercosDataStatus..... 383

	Seite
15.15 MB_ReadStringParameter.....	385
15.16 MB_ReadUnit.....	387
15.17 MB_WriteParameter.....	388
15.18 MB_WriteRealParameter.....	390
15.19 MB_WriteStringParameter.....	392
16 Service und Support.....	395
Index.....	397

1 Über diese Dokumentation

1.1 Änderungsverlauf

Ausgaben dieser Dokumentation

Ausgabe	Stand	Bemerkung
01	2020-07	Erstausgabe
02	2020-11	Änderungen in Kap. "Konfigurationen" Änderungen in Kap. "Konfiguration von SPS-spezifischen Daten in IndraWorks" Änderungen in Kap. "MT_BaseAdditional.library"

Tab. 1-1: Änderungsverlauf

1.2 Zweck

Diese Dokumentation beschreibt die Interface-Signale und die Programmbausteine für die integrierte SPS.

1.3 Übersicht über Zielgruppen und Produktphasen

In der folgenden Grafik beziehen sich die umrandeten Aktivitäten, Produktphasen und Zielgruppen auf die vorliegende Dokumentation.

Beispiel: In der Produktphase "Engineering" kann die Zielgruppe "Programmierer" mit Hilfe dieser Dokumentation die Aktivität "programmieren" ausführen.

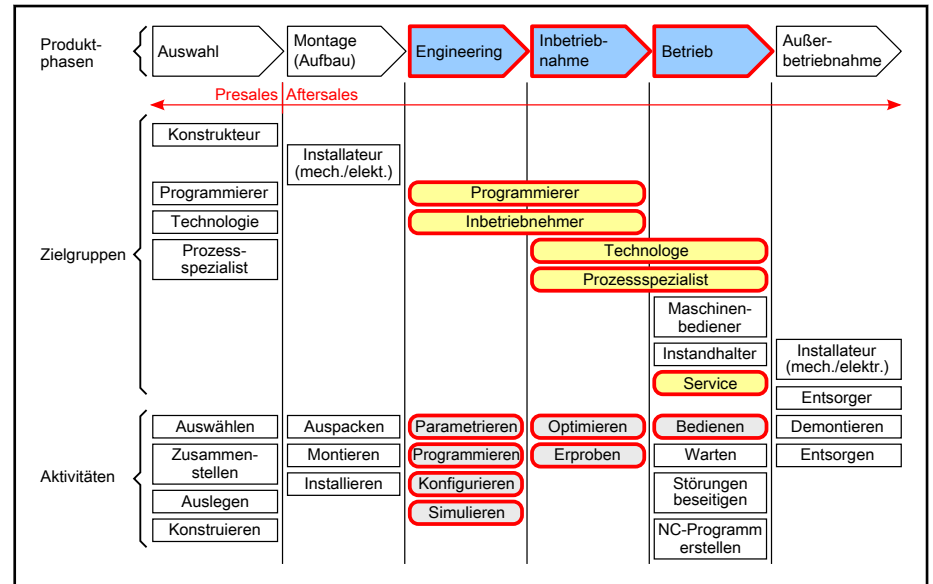


Abb. 1-1: Zuordnung der vorliegenden Dokumentation zu den Zielgruppen, Produktphasen und den Aktivitäten der Zielgruppe

1.4 Erforderliche und ergänzende Dokumentationen MTX

1.4.1 Auswählen/Zusammenstellen

Dokumentationstitel mit Typenschlüssel und Materialnummer

MTX 15VRS Systembeschreibung DOK-MTX***-SYS*DES*V15-PRRS-DE-P, R911397741 Diese Dokumentation beschreibt das Steuerungssystem MTX. Beschrieben werden die Ausführungen, Technischen Daten, Schnittstellen und die Konfiguration der Steuerungsbaugruppen.
MTX 14VRS SafeLogic Systemübersicht DOK-MTX***-SL**SYS*V14-PRRS-DE-P, R911341697 Diese Dokumentation beschreibt den Einsatz der Sicherheitssteuerung SafeLogic im System MTX.
IndraControl XM42 Steuerungen DOK-CONTRL-XM4X*CTRL**-ITRS-DE-P, R911345565 Diese Dokumentation dient der Beschreibung der Steuerungen IndraControl XM42.

RS entsprechender Ausgabestand
Tab. 1-2: MTX-Dokumentationsübersicht - Auswählen/Zusammenstellen

1.4.2 Konfigurieren

Dokumentationstitel mit Typenschlüssel und Materialnummer

MTX 15VRS Maschinenparameter DOK-MTX***-MA*PAR**V15-RERS-DE-P, R911400175 Diese Dokumentation beschreibt den Aufbau und die Anpassung der verfügbaren Parameter der MTX. Ebenfalls beschrieben werden Funktion und Bedienung des NC-Konfigurators.
MTX 15VRS SPS-Interface DOK-MTX***-PLC*INT*V15-PRRS-DE-P, R911400171 Diese Dokumentation beschreibt die Interface-Signale und die Programmbausteine für die integrierte SPS.
MTX 15VRS Funktionsbeschreibung - Grundlagen DOK-MTX***-NC*F*BA*V15-RERS-DE-P, R911401400 Diese Dokumentation beschreibt die Basisfunktionen der MTX. Erläutert werden grundlegende Inbetriebnahmeschritte und die Funktionen der Steuerung als Beschreibung und Handlungsanweisung.
MTX 15VRS Funktionsbeschreibung - Erweiterung DOK-MTX***-NC*F*EX*V15-RERS-DE-P, R911393315 Diese Dokumentation beschreibt die erweiterten Funktionen der MTX. Erläutert werden grundlegende Inbetriebnahmeschritte und die Funktionen der Steuerung als Beschreibung und Handlungsanweisung.
MTX 15VRS Funktionsbeschreibung - Sonderfunktionen DOK-MTX***-NC*F*SP*V15-RERS-DE-P, R911393308 Diese Dokumentation beschreibt die Sonderfunktionen der MTX. Erläutert werden grundlegende Inbetriebnahmeschritte und die Funktionen der Steuerung als Beschreibung und Handlungsanweisung.

MTX Freiformflächenbearbeitung

DOK-MTX***-FREEFORM***-APRS-DE-P, R911341434

Diese Dokumentation beschreibt die Vorgehensweise bei der Freiformflächenbearbeitung mit der MTX-Steuerung. CNC-Programme, die von einem CAD/CAM-System generiert werden, dienen als Grundlage für den gesamten Bearbeitungsprozess. Es wird ein Überblick über folgende Themen gegeben: Beschreibung der MTX- bzw. CNC-Funktionen zur Freiformflächenbearbeitung, Vorgehensweise bei der NC-Parametrierung, Rahmenbedingungen bei der Erzeugung von CNC-Programmen mit CAM-Software.

MTX Konvertierung von MTX-Projekten

DOK-MTX***-PROCONV****-PRRS-DE-P, R911342483

Diese Dokumentation unterstützt bei der Konvertierung von MTX 1.x Projekten nach MTX 2G. Es wird die Vorgehensweise beschrieben und auf mögliche Schwierigkeiten bei der Konvertierung hingewiesen.

RS

entsprechender Ausgabestand

Tab. 1-3:

MTX-Dokumentationsübersicht - Konfigurieren

1.4.3 In Betrieb nehmen

Dokumentationstitel mit Typenschlüssel und Materialnummer

MTX 15VRS Inbetriebnahme

DOK-MTX***-STARTUP*V15-CORS-DE-P, R911393280

Diese Dokumentation beschreibt die Inbetriebnahme der Steuerung MTX. Neben einer Gesamtübersicht werden die Inbetriebnahme und Konfiguration der Achsen und Bedienoberfläche und der SPS-Daten erläutert.

IndraWorks 15VRS Basis-Bibliotheken IndraLogic 2G

DOK-IL*2G*-BASLIB**V15-LIRS-DE-P, R911398632

Das vorliegende Handbuch beschreibt die systemübergreifenden SPS-Bibliotheken.

IndraWorks 15VRS Feldbusse

DOK-IWORKS-FB*****V15-APRS-DE-P, R911393282

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

IndraWorks 14VRS WinStudio 7.4

DOK-IWORKS-WINSTUD*V14-APRS-DE-P, R911341584

Dieses "Benutzerhandbuch und Technische Referenz" soll Sie dabei unterstützen, mit Ihrer Software "Rexroth WinStudio"™ optimale Ergebnisse zu erzielen. Das Dokument bietet technische Informationen und Schritt-für-Schritt-Anleitungen zum Anlegen von webfähigen HMI/SCADA-Programmen.

IndraWorks 15VRS Software-Installation

DOK-IWORKS-SOFTINS*V15-CORS-DE-P, R911393449

Die Dokumentation beschreibt den Installationsprozess von IndraWorks.

IndraWorks 15VRS Engineering

DOK-IWORKS-ENGINEE*V15-APRS-DE-P, R911393302

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.

IndraWorks 15VRS SPS-Programmiersystem IndraLogic 2G

DOK-IWORKS-IL2GPRO*V15-APRS-DE-P, R911396134

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

IndraWorks 15VRS HMI

DOK-MLC***-HMI*****V15-APRS-DE-P, R911399269

Diese Dokumentation beschreibt die Funktionen, die Konfiguration und die Bedienung der Bedienoberflächen IndraWorks HMI Engineering und IndraWorks HMI Operation.

MTX 15VRS Technologie Strahlschneiden

DOK-MTX***-TECHCUT*V15-CORS-DE-P, R911399148

Diese Dokumentation beschreibt die Technologie Strahlschneiden, über die Maschinen mittels eines NC-geführten Bearbeitungskopfs Materialien schneiden können.

RS

entsprechender Ausgabestand

Tab. 1-4:

MTX-Dokumentationsübersicht - In Betrieb nehmen

1.4.4 Bedienen

Dokumentationstitel mit Typenschlüssel und Materialnummer**MTX 12VRS Satzvorlauf**

DOK-MTX***-BLK*RUN*V12-APRS-DE-P, R911334378

Die Dokumentation soll den Maschinenhersteller in die Lage versetzen, die Funktion "Satzvorlauf" an seiner Maschine für den Endanwender einzurichten.

MTX 15VRS Programmierhandbuch

DOK-MTX***-NC**PRO*V15-RERS-DE-P, R911393317

Das vorliegende Handbuch beschreibt die Standard-Programmierung der Steuerung MTX. Beschrieben werden neben den Grundlagen der NC-Programmierung sowohl die Anwendung der NC-Funktionen nach DIN 66025, als auch der NC-Funktionen mit Hochsprachensyntax und CPL-Funktionen.

MTX 15VRS Standard-NC-Bedienung

DOK-MTX***-NC*OP***V15-APRS-DE-P, R911393312

Diese Dokumentation beschreibt die Bedienung der Standard-Bedienoberfläche der NC-Steuerung MTX. Beschrieben werden die Bedienung der Oberfläche, die NC-Programmerstellung und die Werkzeugverwaltung.

MTX 15VRS Multitouch

DOK-MTX***-MULTI***V15-APRS-DE-P, R911393310

In dieser Dokumentation wird die Bedienung der Multitouch-Bedienoberfläche der NC-Steuerung MTX beschrieben.

MTX 15VRS Standard-NC-Zyklen

DOK-MTX***-NC*CYC**V15-PRRS-DE-P, R911394939

Diese Dokumentation beschreibt für die Steuerung MTX die Anwendung der Standard-Zyklen der verschiedenen Technologien.

MTX 15VRS NC-Simulation

DOK-MTX***-NC*SIM**V15-APRS-DE-P, R911393272

Diese Dokumentation beschreibt für die Steuerung MTX die NC-Simulation.

MTX 15VRS Messfunktionen

DOK-MTX***-MES*FUN*V15-APRS-DE-P, R91194937

Diese Dokumentation beschreibt die Messzyklen der Steuerung MTX.

RS

entsprechender Ausgabestand

Tab. 1-5:

MTX-Dokumentationsübersicht - Bedienen

1.4.5 OEM-Entwicklung

Dokumentationstitel mit Typenschlüssel und Materialnummer

MTX 13VRS Automation Interface DOK-MTX***-AUT*INT*V13-APRS-DE-P, R911337273 Diese Dokumentation beschreibt den scriptbasierten Zugriff auf IndraWorks-Projektdateien über die Schnittstelle des Automation Interfaces. Beschrieben werden die verschiedenen Objekte, einschließlich Code-Beispielen. Eine Beschreibung des Automation Builders schließt sich an.
MTX 15VRS OPC-Kommunikation DOK-MTX***-OPC*COM*V15-PRRS-DE-P, R911399271 Diese Dokumentation beschreibt die Syntax und den Aufbau der Items für die Kommunikation mit den Bosch Rexroth-Geräten.
IndraWorks OPC UA Kommunikation DOK-IWORKS-OPC*UA*****-APRS-DE-P, R911379308 Diese Dokumentation beschreibt die OPC UA Kommunikation der Steuerungssysteme MLC und MTX.

RS entsprechender Ausgabestand
Tab. 1-6: *MTX-Dokumentationsübersicht - OEM-Entwicklung*

1.4.6 AddOns

Dokumentationstitel mit Typenschlüssel und Materialnummer

MTX 11VRS Aktionsrekorder DOK-MTX***-ACR*****V11-APxx-DE-P, R911329942 Diese Dokumentation beschreibt den MTX Aktionsrekorder. Beschrieben werden sowohl die Installation und die Inbetriebnahme, einschließlich Schnittstellensignale, als auch die Anwendung und die Bedienung.
MTX 15VRS Efficiency Workbench MTX cta, MTX ega DOK-MTX***-EWB*****V15-APRS-DE-P, R911400177 Diese Dokumentation beschreibt die Funktionsweise und den Einsatzbereich der Analysetools MTX cta und MTX ega.
MTX Remote Condition Monitoring DOK-MTX***-RCM*****V01-APRS-DE-P, R911334382 Diese Dokumentation beschreibt die Bedienung des Remote Condition Monitoring Systems.
MTX visIREC Anwenderdokumentation DOK-MTX***-VISIREC*V01-APRS-DE-P, R911344241 Diese Dokumentation beschreibt das Analysewerkzeug visIREC, das zur Optimierung des Freiformflächenbearbeitungsprozesses eingesetzt werden kann. 2D- bzw. 4D-Anzeige der bahnbezogenen Daten. 2D- bzw. 4D-Anzeige der koordinatenbezogenen Daten. Analyse von kritischen Bereichen (Bahn- und Orientierungsabweichung). Vergleich zwischen den programmierten und den interpolierten NC-Sätzen.

xx / RS entsprechender Ausgabestand
Tab. 1-7: *MTX-Dokumentationsübersicht - AddOns*

1.5 Darstellung von Informationen

1.5.1 Gebrauch der Sicherheitshinweise

Aufbau der Sicherheitshinweise

Die Sicherheitshinweise sind wie folgt aufgebaut:

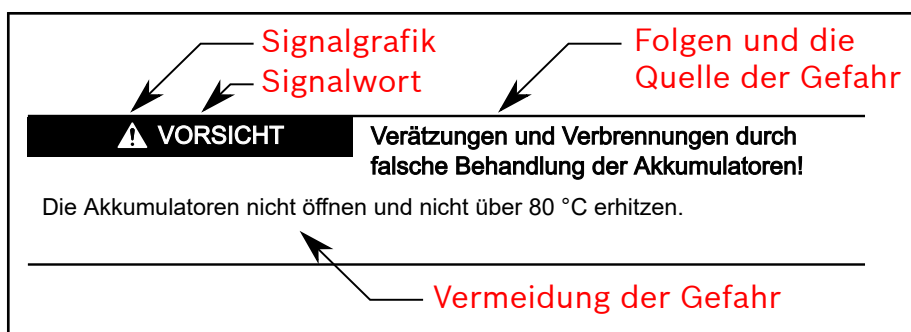


Abb. 1-2: Aufbau der Sicherheitshinweise

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.

Verwendete Symbole

Fingerzeige werden wie folgt dargestellt:



Dies ist ein Hinweis.

Tipps werden wie folgt dargestellt:



Dies ist ein Tipp.

Erläuterung der Signalgrafik auf dem Gerät



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

1.5.2 Bezeichnungen und Abkürzungen

Begriff	Erklärung
IWE	IndraWorks Engineering
IWO	IndraWorks Operation
NC	Numerical Control
OEM	Original Equipment Manufacturer
Profibus	Kommunikationsverbindung
Sercos	Sercos (Serial Realtime Communication System) interface ist eine weltweit genormte digitale Schnittstelle zur Kommunikation zwischen Steuerungen und Antrieben

Tab. 1-8: Verwendete Bezeichnungen und Abkürzungen

1.5.3 Kundenfeedback

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

2 Wichtige Gebrauchshinweise

2.1 Bestimmungsgemäßer Gebrauch

2.1.1 Einführung

Die Produkte von Bosch Rexroth werden nach dem jeweiligen Stand der Technik entwickelt und gefertigt. Vor ihrer Auslieferung werden sie auf ihren betriebssicheren Zustand hin überprüft.

Die Produkte dürfen nur bestimmungsgemäß eingesetzt werden. Wenn sie nicht bestimmungsgemäß eingesetzt werden, dann können Situationen entstehen, die Sach- und Personenschaden nach sich ziehen.



Für Schäden bei nicht-bestimmungsgemäßigem Gebrauch der Produkte leistet Bosch Rexroth als Hersteller keinerlei Gewährleistung, Haftung oder Schadensersatz; die Risiken bei nicht-bestimmungsgemäßigem Gebrauch der Produkte liegen allein beim Anwender.

Bevor Sie die Produkte der Firma Bosch Rexroth einsetzen, müssen die folgenden Voraussetzungen erfüllt sein, um einen bestimmungsgemäßen Gebrauch der Produkte zu gewährleisten:

- Jeder, der in irgendeiner Weise mit einem unserer Produkte umgeht, muss die entsprechenden Sicherheitsvorschriften und den bestimmungsgemäßen Gebrauch lesen und verstehen.
- Sofern es sich bei den Produkten um Hardware handelt, müssen sie in ihrem Originalzustand belassen werden; d. h. es dürfen keine baulichen Veränderungen an ihnen vorgenommen werden. Softwareprodukte dürfen nicht dekompiert werden und ihre Quellcodes dürfen nicht verändert werden.
- Beschädigte oder fehlerhafte Produkte dürfen nicht eingebaut oder in Betrieb genommen werden.
- Es muss gewährleistet sein, dass die Produkte entsprechend den in der Dokumentation genannten Vorschriften installiert sind.



Die der Steuerung vorliegenden oder vom Benutzer eingegebenen bzw. eingelesenen Daten sind vor der Übernahme/Aktivierung auf ihre Richtigkeit zu prüfen, um ungewollte Bewegungen der Achsen auszuschließen. Im Einzelnen kann es sich um folgende ungültige oder veraltete Daten handeln:

- Teileprogramme
- NPV-Tabellen
- Korrekturtabellen
- Werkzeugtabellen
- Permanente CPL-Variablen
- Remanente SPS-Daten
- Permanente Systemdaten

2.1.2 Einsatz- und Anwendungsbereiche

Die MTX-Steuerung dient zum

- Programmieren von Kontur und Bearbeitungstechnologie (Bahnvorschub, Spindeldrehzahl, Werkzeugwechsel) eines Werkstückes.
- Führen eines Bearbeitungswerkzeuges entlang einer programmierten Bahn.

Vorschubantriebe, Spindeln und Hilfsachsen einer Werkzeugmaschine werden per Sercos Schnittstelle angesteuert.



Zusätzlich werden E/A-Komponenten für die integrierte SPS benötigt, die – in Kombination mit der eigentlichen CNC – den Bearbeitungsprozess ganzheitlich steuert und auch in sicherheitstechnischer Hinsicht überwacht.

Der Betrieb darf nur in den ausdrücklich angegebenen Konfigurationen und Kombinationen der Hardware-Komponenten und mit der in den jeweiligen Dokumentationen und den Funktionsbeschreibungen angegebenen und spezifizierten Soft- und Firmware erfolgen.

Das CNC-System MTX bietet die perfekte CNC-Systemlösung für Zerspanen und Umformen für die folgenden Technologien:

- Drehen
- Fräsen
- Bohren
- Schleifen
- Biegen
- Nibbeln
- Stanzen
- Formschneiden
- Handhaben

2.2 Nicht-bestimmungsgemäßer Gebrauch

Die Verwendung des CNC-Systems MTX außerhalb der vorgenannten Anwendungsgebiete oder unter anderen als den in der Dokumentation beschriebenen Betriebsbedingungen und angegebenen technischen Daten gilt als "nicht-bestimmungsgemäß".

Das CNC-System MTX darf nicht eingesetzt werden, wenn sie ...

- Betriebsbedingungen ausgesetzt wird, die die vorgeschriebenen Umgebungsbedingungen nicht erfüllen. Untersagt sind z. B. der Betrieb unter Wasser, unter extremen Temperaturschwankungen oder extremen Maximaltemperaturen.
- Außerdem darf das CNC-System MTX nicht bei Anwendungen eingesetzt werden, die von Bosch Rexroth nicht ausdrücklich freigegeben sind. Beachten Sie hierzu bitte unbedingt die Aussagen in den allgemeinen Sicherheitshinweisen!
- Das CNC-System MTX darf nicht an Anlagen oder Maschinen eingesetzt werden, die über eine ungesicherte Netzwerkverbindung mit dem Internet verbunden ist. Andernfalls besteht die Gefahr von Störungen bzw. einem Steuerungsausfall aufgrund unautorisierter Zugriffe.

3 Übersicht

In diesem Handbuch werden die Software-Schnittstellen und deren Funktionsweise aus Sicht der im CNC-System MTX integrierten SPS-Steuerung dargestellt.



Eine ausführliche Beschreibung der integrierten SPS finden Sie in weiteren Handbüchern.

Das SPS-Ablaufprogramm kann grundsätzlich mit allen Funktionsbereichen bzw. Teilsystemen des Gesamtsystems kommunizieren. Das Handbuch beschreibt die Funktionen, die vom SPS-Ablaufprogramm über die SPS-Operanden direkt angesprochen werden können.

Angaben zur SPS-Schnittstelle

Typ	Größe	Adressbereich
%IB	8 kB	%IB0 ... %IB8192
%QB	8 kb	%QB0 ... %QB8192
%MB	32 kB	%MB0 ... %MB32768

Einzelsignale

Einzelsignale, die dem SPS-Ablaufprogramm einen bestimmten Zustand signalisieren (z. B. Achse In Position, Hilfsfunktionsausgabe) oder eine Funktion in der NC aktivieren (z. B. NC-Start, Stopp, Vorschubfreigabe), werden pro SPS-Zyklus aufgefrischt bzw. beim Auftreten eines Ereignisses in die entsprechende Richtung übertragen.

Sie sind unterteilt in

- [Globale Interface-Signale, Seite 31](#)
- [Kanalbezogene Interface-Signale, Seite 37](#)
- [Achsbezogene Interface-Signale, Seite 67](#)
- [Spindelbezogene Interface-Signale, Seite 95](#)
- Decodierte [Hilfsfunktionen, Seite 121](#)

Diese Interfaces zwischen SPS und NC werden innerhalb der IndraLogic über den Konfigurator der IndraLogic-Programmierschnittfläche im Eingangs- bzw. Ausgangsbereich konfiguriert.

Hierbei kann jedes Interface einzeln konfiguriert werden:

- Die Knoten-Nummer bestimmt hierbei die physikalische Interface-Nummer (Kanal-Nr ab 0, Achs- und Spindelnummer ab 1).
- Jedes Interface kann mit einem symbolischen Bezeichner versehen werden, unter dem das Interface im Ablaufprogramm angesprochen werden kann.
- Jedem Interface kann eine eigne I/O-Adresse zugeordnet werden, so dass durch das Einfügen/Löschen von Interfaces die physikalische Adresse des Interface unverändert bleiben kann.

Sind innerhalb des Systems mehr Interfaces konfiguriert als im I/O-Konfigurator, werden die nicht konfigurierten Interfaces beim Datenaustausch nicht berücksichtigt.

Programmbausteine

Zusätzlich zur Übertragung von Einzelsignalen sind weitere Funktionen implementiert, die seltener oder nur bei speziellen Applikationen benötigt werden. Diese Funktionen können über parametrierbare Programmbausteine aufgerufen und aktiviert werden.

Jeder Aufruf eines Programmbausteins wird quittiert und entsprechend der aufgerufenen Funktion wird das angeforderte Datum auf den vom SPS-Ablaufprogramm parametrisierten Operanden abgelegt.



Verfügbare Programmbausteine siehe [Kap. 12 "Programmbausteine"](#) auf Seite 129.

Beispiel:

Beispiel zur Kommunikationsstruktur

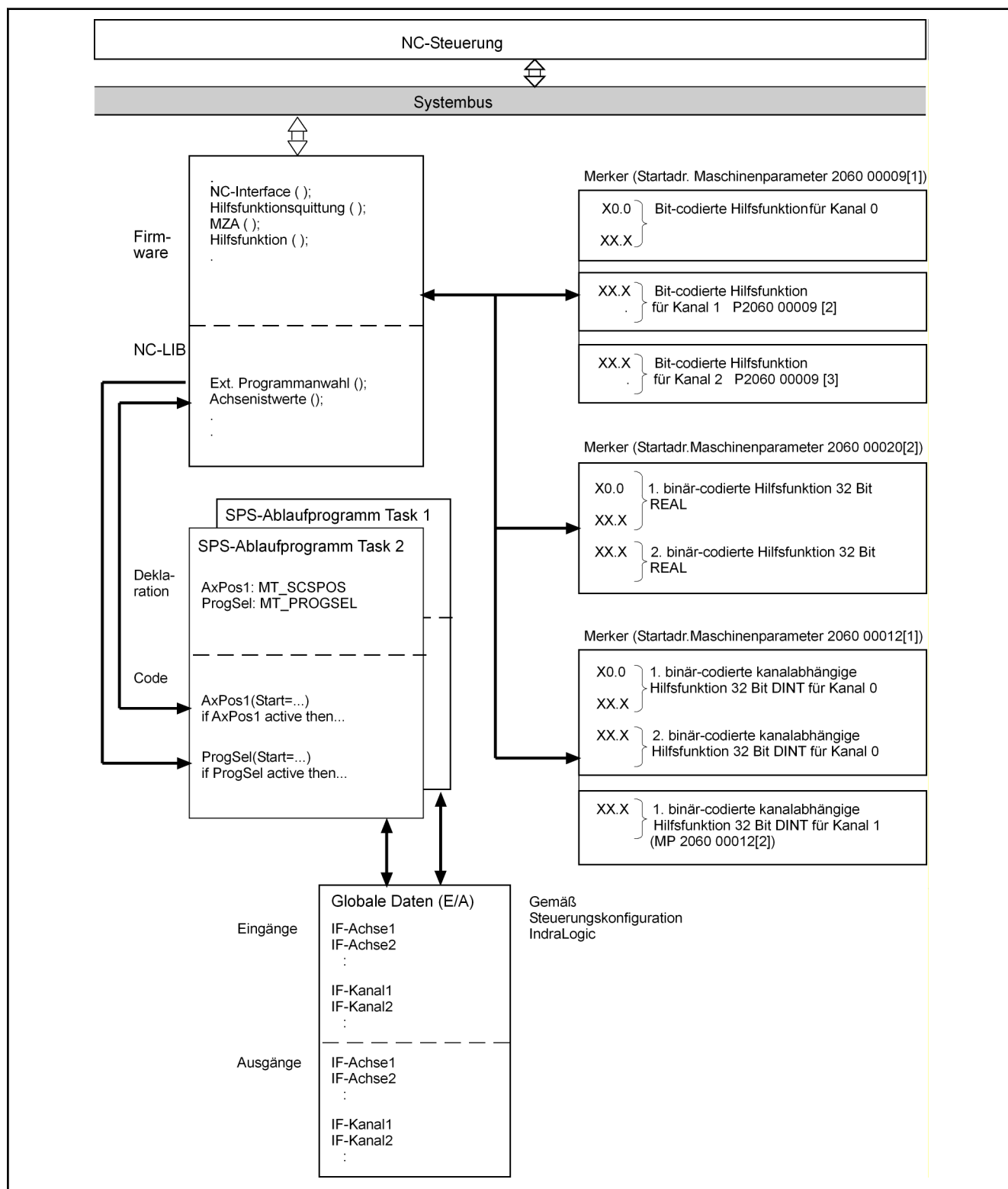


Abb. 3-1: Beispiel zur Kommunikationsstruktur

Ablage der SPS-spezifischen Dateien innerhalb der Steuerung

Alle Dateien, die das IndraLogic-Laufzeitsystem anlegt, werden im RAM-Filesystem unter dem Pfad **/plc/** gespeichert. Dies sind das Boot-Projekt mit Informationen über die ausgelagerten Symbole (Datei mit dem Name <Name der Applikation>.app) sowie einige intern verwendete Dateien. Wird das RAM-Filesystem neu generiert, wie es z. B. beim Firmware-Tausch nötig ist, gehen diese Daten verloren und müssen daher gegebenenfalls vorher auf **/usrfep/plc** oder in einem NC-Archiv (tar-file) gesichert werden. Wurden sie in einem tar-file gesichert, werden die Dateien beim Einlesen des tar-files wieder in **/plc** gespeichert und beim nächsten Hochlauf der Steuerung automatisch geladen.

Wurden die Dateien in **/usrfep/plc** gesichert, können sie nach dem Erstellen des file-Systems entweder durch "händiges" Umkopieren nach **/plc** oder durch einen Steuerungshochlauf mit dem Startup-Mode 2 wieder hergestellt werden.

4 Konfigurationen

4.1 Konfiguration des SPS-NC-Bit-Interface

4.1.1 Allgemeines

Nach dem Erzeugen eines neuen IndraWorks-Projekts und dem Anlegen der Visualisierung (z.B. PR/VR4x) sowie der Steuerung XM4x, findet man die Konfiguration des SPS-NC-Bit-Interfaces als Unterknoten dieser Steuerung. Der zugehörige Knotenname lautet "MTX_NC_PLC_Interface".

Das Bit-Interface besteht aus dem Globalen Interface (General Interface), dem Kanal-Interface (Channel Interface), dem Achs-Interface (Axes Interface) und dem Spindel-Interface (Spindles Interface). Ebenso kann dem Bit-Interface die MZA (Maschinenzustandsanzeige) hinzugefügt werden.

Die Konfiguration der MZA ist in [dem Kapitel 13.1 Konfiguration der Maschinenzustandsanzeige \(MZA\), Seite 225](#) beschrieben.

Standardmäßig ist beim Neuanlegen lediglich das Interface für Globale Daten vorhanden und vorgelegt.

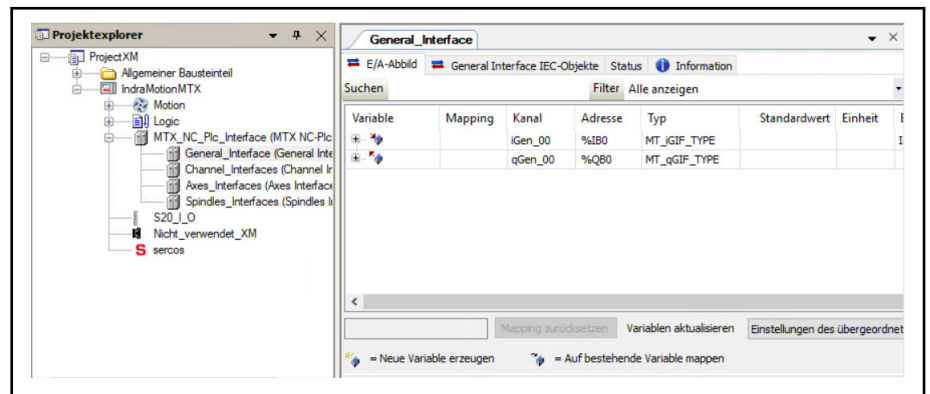


Abb. 4-1: Das vorgelegte Bit-Interface mit dem Eigenschaftendialog für das globale Interface

Die Signale des Bit-Interfaces sind immer aus Sicht der SPS zu betrachten. So sind z. B. die Signale von qGen Signale des Globalen Interfaces von der SPS zur NC.

4.1.2 Das Globale Interface (General Interface)

Das Globale Interface ist immer vorhanden und nicht löscherbar. Der symbolische Name für Ausgangssignale lautet "qGen" und der symbolische Name für Eingangssignale lautet "iGen". Durch einen Doppelklick (oder im Kontextmenü über "Öffnen") auf den Knoten "General Interface" kommt man zum Eigenschaftendialog. In diesem Eigenschaftendialog kann man die vorgelegten Namen und Adressen für die Ein- und Ausgänge verändern sowie Kommentare vergeben.

4.1.3 Das Kanal-Interface (Channel Interface)

Das CNC-System MTX unterstützt einen Schaltfunktionen-Kanal (Kanal 0) und bis zu zwölf Bearbeitungskanäle. Standardmäßig sind noch keine Kanäle definiert. Das Anlegen von neuen Kanälen geschieht über das Kontextmenü auf dem Knoten "Channel Interfaces".

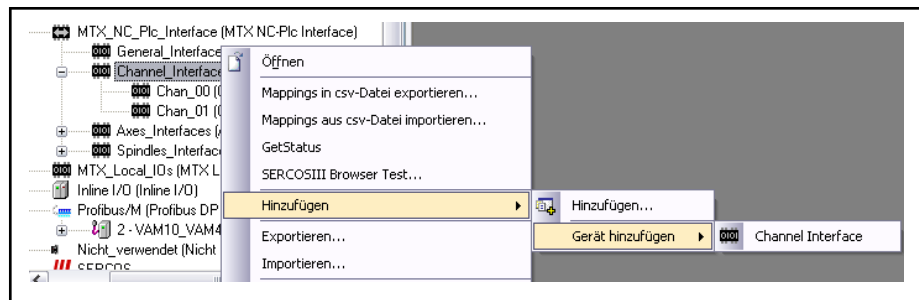


Abb. 4-2: Beispiel für das Anlegen des Kanals 1

Neue Kanäle werden beginnend mit Kanal 0 (Schaltfunktionen-Kanal) fortlaufend angelegt. Entscheidend für die Bedeutung eines Kanals ist sein Index; so ist z. B. ein Kanal-Interface mit dem Index 2 das Interface für den Bearbeitungskanal 2.

Der symbolische Name für Ausgangssignale lautet "qChan_<Knotennummer mit führender Null>" und der symbolische Name für Eingangssignale lautet "iChan_<Knotennummer mit führender Null>".

Durch einen Doppelklick (oder im Kontextmenü über "Öffnen") auf den Knoten "Channel Interface <Knotennummer>" kommt man zum Eigenschaftendialog.

Über den Eigenschaftendialog kann man neben symbolischen Namen, Adressen und Kommentaren auch den jeweiligen Index verändern.

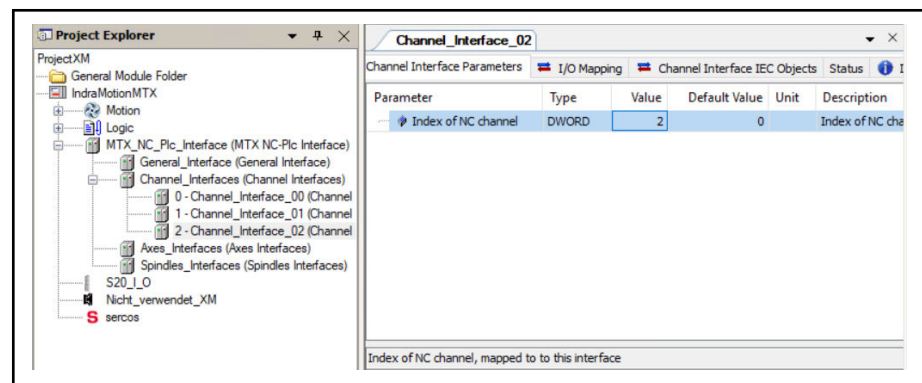


Abb. 4-3: Eigenschaftendialog für das Kanal-Interface



Der Index für die Kanal-Interfaces muss immer eindeutig sein; daher darf jeder Index innerhalb der Kanal-Interfaces nur einmal vorkommen.

4.1.4 Das Achs-Interface (Axes Interface)

Das CNC-System MTX unterstützt bis zu 64 Achsen. Standardmäßig sind noch keine Achsen definiert. Das Anlegen von neuen Achsen geschieht über das Kontextmenü auf dem Knoten "Axes Interfaces".

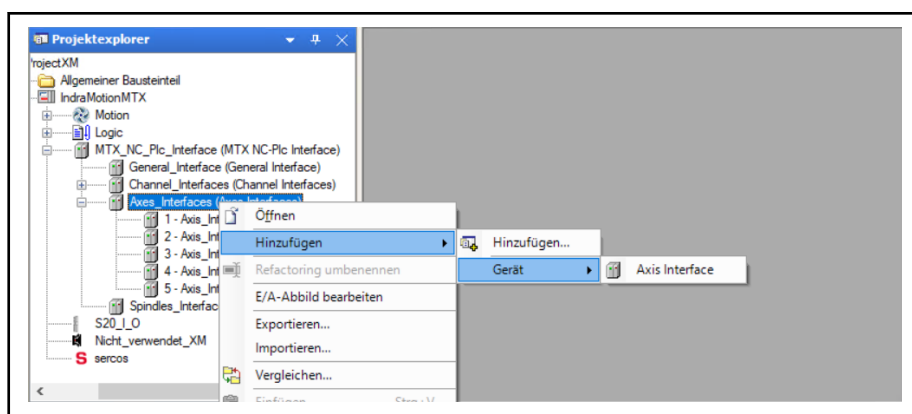


Abb. 4-4: Beispiel für das Anlegen einer Achse

Neue Achsen werden beginnend mit Achse 1 fortlaufend angelegt. Entscheidend für die Bedeutung einer Achse ist sein Index; so ist z. B. ein Achs-Interface mit dem Index 4 das Interface für die Achse 4.

Der symbolische Name für Ausgangssignale lautet "qAxis_<Knotennummer mit führender Null>" und der symbolische Name für Eingangssignale lautet "iAxis_<Knotennummer mit führender Null>".

Durch einen Doppelklick (oder im Kontextmenü über "Öffnen") auf den Knoten "Axis Interface <Knotennummer>" kommt man zum Eigenschaftendialog.

Über den Eigenschaftendialog kann man neben symbolischen Namen, Adressen und Kommentaren auch den jeweiligen Index verändern.

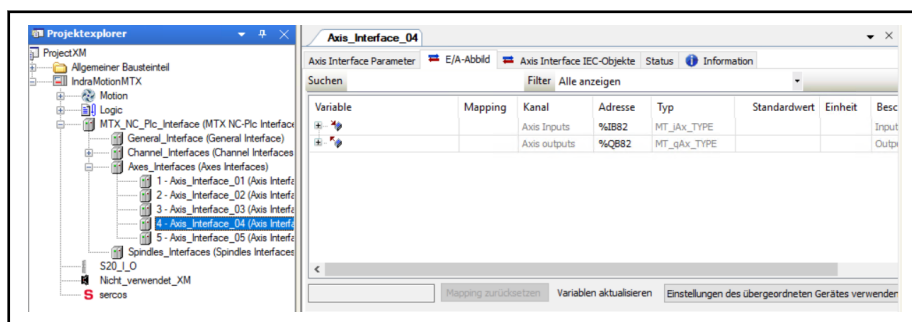


Abb. 4-5: Eigenschaftendialog für das Achs-Interface



Der Index für die Achs-Interfaces muss immer eindeutig sein; daher darf jeder Index innerhalb der Achs-Interfaces nur einmal vorkommen.

4.1.5 Das Spindel-Interface (Spindle Interface)

Das CNC-System MTX unterstützt bis zu 32 Spindeln. Standardmäßig sind noch keine Spindeln definiert. Das Anlegen von neuen Spindeln geschieht über das Kontextmenü auf dem Knoten "Spindles Interfaces".

Konfigurationen

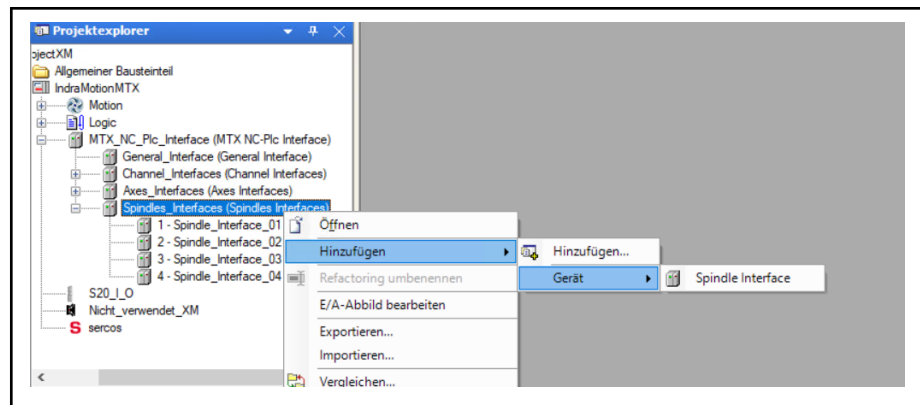


Abb. 4-6: Beispiel für das Anlegen einer Spindel

Neue Spindeln werden beginnend mit Spindel 1 fortlaufend angelegt. Entscheidend für die Bedeutung einer Spindel ist sein Index; so ist z. B. ein Spindel-Interface mit dem Index 6 das Interface für die Spindel 6.

Der symbolische Name für Ausgangssignale lautet "qSpindle_<Knotennummer mit führender Null>" und der symbolische Name für Eingangssignale lautet "iSpindle_<Knotennummer mit führender Null>".

Durch einen Doppelklick (oder im Kontextmenü über "Öffnen") auf den Knoten "Spindle Interface <Knotennummer>" kommt man zum Eigenschaftendialog.

Über den Eigenschaftendialog kann man neben symbolischen Namen, Adressen und Kommentaren auch den jeweiligen Index verändern.

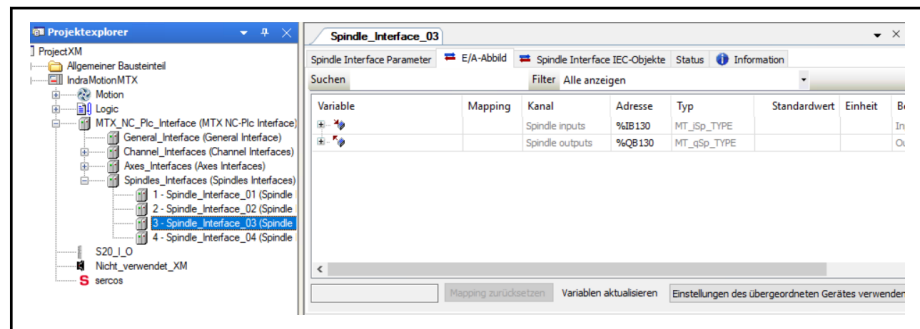


Abb. 4-7: Eigenschaftendialog für das Spindel-Interface



Der Index für die Spindel-Interfaces muss immer eindeutig sein; daher darf jeder Index innerhalb der Spindel-Interfaces nur einmal vorkommen.

5 Übersicht Interface-Signale

5.1 Globale Ausgangssignale (SPS --> NC)

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
0.0	qGen_Reset	Systemgrundstellung	1.0	qGen_StrokeInhibit	Hubsperrung
0.1	-	res.	1.1	qGen_StrokeReserv	Hubreservierung
0.2	qGen_DelErrInfo	Fehler Infos löschen	1.2	qGen_StrokeRel	Hub ein
0.3	qGen_SercosIO_Reset	Sercos-I/O-Geräte-Fehler löschen	1.3	-	res.
0.4	-	res.	1.4	-	res.
0.5	-	res.	1.5	-	res.
0.6	-	res.	1.6	-	res.
0.7	-	res.	1.7	-	res.
2.0	qGen_Crash	Crash-Zustand von SPS an NC (IPO)	3.0	-	res.
2.1	qGen_CrashStoreDataAckn	Quittung Crash-Zustand von NC an SPS	3.1	-	res.
2.2	-	res.	3.2	-	res.
2.3	-	res.	3.3	-	res.
2.4	-	res.	3.4	-	res.
2.5	-	res.	3.5	-	res.
2.6	-	res.	3.6	-	res.
2.7	-	res.	3.7	-	res.

Tab. 5-1: Übersicht Ausgangssignale (SPS --> NC)

5.2 Globale Eingangssignale (NC --> SPS)

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
0.0	iGen_Reset	Systemgrundstellung ausgeführt	1.0	iGen_StrokeIntend	Hub beabsichtigt
0.1	iGen_HardwareState	Hardware Warnung	1.1	iGen_NoStroke	Hub läuft nicht
0.2	-	res.	1.2	-	res.
0.3	iGen_SercosIO_Reset	Sercos-I/O-Geräte-Fehler gelöscht	1.3	-	res.
0.4	-	res.	1.4	-	res.
0.5	-	res.	1.5	-	res.
0.6	-	res.	1.6	-	res.
0.7	-	res.	1.7	-	res.
2.0	iGen_CrashAckn	Quittierung Crash-Zustand von SPS an NC (IPO)	3.0	-	res.

Übersicht Interface-Signale

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
2.1	iGen_CrashStoreData	Crash-Zustand von NC an SPS	3.1	-	res.
2.2	-	res.	3.2	-	res.
2.3	-	res.	3.3	-	res.
2.4	-	res.	3.4	-	res.
2.5	-	res.	3.5	-	res.
2.6	-	res.	3.6	-	res.
2.7	-	res.	3.7	-	res.

Tab. 5-2: Übersicht Eingangssignale (NC --> SPS)

5.3 Kanalbezogene Ausgangssignale (SPS --> NC)

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
0.0	qCh_OpModeSel_00	Anw. Betriebsart Bit 0	1.0	qCh_OpModePlc	Betriebsart von SPS
0.1	qCh_OpModeSel_01	Anw. Betriebsart Bit 1	1.1	qCh_Restart	Automatischer Restart
0.2	qCh_OpModeSel_02	Anw. Betriebsart Bit 2	1.2	qCh_NCStart	NC-Start
0.3	qCh_OpModeSel_03	Anw. Betriebsart Bit 3	1.3	qCh_TransferLock	Einlesesperre
0.4	-	res.	1.4	qCh_FeedHold	Vorschub-Halt
0.5	-	res.	1.5	qCh_FeedStop	Vorschubsperr
0.6	-	res.	1.6	qCh_MachineFunction	Maschinenfunktion
0.7	-	res.	1.7	qCh_ReSelOff	Autom. Wiederanwahl Aus
2.0	qCh_CtrlReset	Grundstellung	3.0	qCh_ASub1	Asynchr.-Unterpr. 1
2.1	qCh_CancSyntaxError	Syntaxfehler löschen	3.1	qCh_ASub2	Asynchr.-Unterpr. 2
2.2	qCh_CancDist	Restweg löschen	3.2	qCh_ASub3	Asynchr.-Unterpr. 3
2.3	qCh_NextBlk	Umschalten auf nächsten Satz	3.3	qCh_ASub4	Asynchr.-Unterpr. 4
2.4	-	res.	3.4	qCh_ASub5	Asynchr.-Unterpr. 5
2.5	-	res.	3.5	qCh_ASub6	Asynchr.-Unterpr. 6
2.6	qCh_RetCont	Wiederanfahren	3.6	qCh_ASub7	Asynchr.-Unterpr. 7
2.7	qCh_Retract	Schneller Rückzug	3.7	qCh_ASub8	Asynchr.-Unterpr. 8
4.0	qCh_JogPlusWcs	WCS Manuell +	5.0	qCh_Blkslash	Satz überlesen
4.1	qCh_JogMinusWcs	WCS Manuell -	5.1	qCh_OptStop	Wahlweise Halt
4.2	-	res.	5.2	qCh_OptJump	Bedingter Sprung
4.3	-	res.	5.3	-	res.
4.4	qCh_HandwSelWcs_00	Handradauswahl Bit0	5.4	qCh_RedRap	Reduzierter Eilgang
4.5	qCh_HandwSelWcs_01	Handradauswahl Bit1	5.5	qCh_Rapid100	Eilgang 100%
4.6	qCh_HandwDirWcs	Handraddrehrichtung	5.6	-	res.

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
4.7	qCh_HandwPosMode	Positionshandrad	5.7	qCh_Override100	Override100%
6.0	qCh_Override_00	Override Bit 0	7.0	qCh_Override_08	Override Bit 8
6.1	qCh_Override_01	Override Bit 1	7.1	qCh_Override_09	Override Bit 9
6.2	qCh_Override_02	Override Bit 2	7.2	qCh_Override_10	Override Bit 10
6.3	qCh_Override_03	Override Bit 3	7.3	qCh_Override_11	Override Bit 11
6.4	qCh_Override_04	Override Bit 4	7.4	qCh_Override_12	Override Bit 12
6.5	qCh_Override_05	Override Bit 5	7.5	qCh_Override_13	Override Bit 13
6.6	qCh_Override_06	Override Bit 6	7.6	qCh_Override_14	Override Bit 14
6.7	qCh_Override_07	Override Bit 7	7.7	qCh_Override_15	Override Bit 15
8.0	qCh_Custom1	Kundeneingang 1	9.0	qCh_OnlCorrWcs	Online-Korrektur Freigabe
8.1	qCh_Custom2	Kundeneingang 2	9.1	qCh_OnlCorrWcsDir	Online-Korrektur Richtung
8.2	qCh_Custom3	Kundeneingang 3	9.2	qCh_Retrace	Rückwärtsbetrieb
8.3	qCh_Custom4	Kundeneingang 4	9.3	qCh_RetraceMovFwd	Bahnbewegung vorwärts
8.4	qCh_Custom5	Kundeneingang 5	9.4	qCh_RetraceMovRev	Bahnbewegung rückwärts
8.5	qCh_Custom6	Kundeneingang 6	9.5	-	res.
8.6	qCh_Custom7	Kundeneingang 7	9.6	qCh_CoordCoupleOff	Kopplung beenden
8.7	qCh_Custom8	Kundeneingang 8	9.7	qCh_TangTRotRel	TTL Freigabe
10.0	qCh_RapOverride00	Eilgang Override Bit 0	11.0	qCh_RapOverride08	Eilgang Override Bit 8
10.1	qCh_RapOverride01	Eilgang Override Bit 1	11.1	qCh_RapOverride09	Eilgang Override Bit 9
10.2	qCh_RapOverride02	Eilgang Override Bit 2	11.2	qCh_RapOverride10	Eilgang Override Bit 10
10.3	qCh_RapOverride03	Eilgang Override Bit 3	11.3	qCh_RapOverride11	Eilgang Override Bit 11
10.4	qCh_RapOverride04	Eilgang Override Bit 4	11.4	qCh_RapOverride12	Eilgang Override Bit 12
10.5	qCh_RapOverride05	Eilgang Override Bit 5	11.5	qCh_RapOverride13	Eilgang Override Bit 13
10.6	qCh_RapOverride06	Eilgang Override Bit 6	11.6	qCh_RapOverride14	Eilgang Override Bit 14
10.7	qCh_RapOverride07	Eilgang Override Bit 7	11.7	qCh_RapOverride15	Eilgang Override Bit 15
12.0	-	res.	13.0	qCH_TestFeed	Testvorschub
12.1	-	res.	13.1	qCH_TestRap	Testeilgang
12.2	-	res.	13.2	qCh_Feedrate2	Zweiter Vorschub
12.3	-	res.	13.3	-	res.
12.4	-	res.	13.4	-	res.
12.5	-	res.	13.5	-	res.
12.6	-	res.	13.6	-	res.
12.7	-	res.	13.7	-	res.

Tab. 5-3: Übersicht Ausgangssignale (SPS → NC)

Übersicht Interface-Signale

5.4 Kanalbezogene Eingangssignale (NC --> SPS)

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
0.0	iCh_OpMode_00	Anw. Betriebsart Bit 0	1.0	iCh_DryRun	Testbetrieb
0.1	iCh_OpMode_01	Anw. Betriebsart Bit 1	1.1	iCh_NCReady	NC-Bereit
0.2	iCh_OpMode_02	Anw. Betriebsart Bit 2	1.2	iCh_ProgRun	Programm läuft
0.3	iCh_OpMode_03	Anw. Betriebsart Bit 3	1.3	iCh_TransferLockAct	Einlesesperre aktiv
0.4		res.	1.4	iCh_FeedHoldAct	Vorschub-Halt aktiv
0.5		res.	1.5	-	res.
0.6		res.	1.6	iCh_ProgStopM0	Programm Halt M0
0.7		res.	1.7	iCh_ProgStopM30	Programm Ende M30
2.0	iCh_Reset	Kanal grundgestellt	3.0	iCh_ASub1	Asynchr.-Unterpr. 1
2.1	-	res.	3.1	iCh_ASub2	Asynchr.-Unterpr. 2
2.2	iCh_RetpaAct	Wiederaufahren zum Anfahrpunkt aktiv	3.2	iCh_ASub3	Asynchr.-Unterpr. 3
2.3	iCh_RetContPath	Wiederaufahren auf Wegstipp-Pfad	3.3	iCh_ASub4	Asynchr.-Unterpr. 4
2.4	iCh_RemoveFinish	Wegfahren beendet	3.4	iCh_ASub5	Asynchr.-Unterpr. 5
2.5	iCh_ReadyReEnter	Bereit zum Wiederauf- fahren	3.5	iCh_ASub6	Asynchr.-Unterpr. 6
2.6	iCh_ReEnterAct	Wiedereintritt aktiv	3.6	iCh_ASub7	Asynchr.-Unterpr. 7
2.7	-	res.	3.7	iCh_ASub8	Asynchr.-Unterpr. 8
4.0	iCh_State_00	Kanalzustand Bit 0	5.0	iCh_BlkSlash	Satz überlesen aktivie- ren
4.1	iCh_State_01	Kanalzustand Bit 1	5.1	iCh_OptStop	Wahlweiser Halt akti- vieren
4.2	iCh_State_02	Kanalzustand Bit 2	5.2	-	res.
4.3	iCh_State_03	Kanalzustand Bit 3	5.3	iCh_SRunAct	Satzvorlauf aktiv
4.4	iCh_State_04	Kanalzustand Bit 4	5.4	iCh_SRunReEnter	Wiedereintritt aktiv
4.5	-	res.	5.5	iCh_SRunRepos	Wiederaufahren aktiv
4.6	-	res.	5.6	iCh_Override0	Override 0%
4.7	-	res.	5.7	iCh_Override100	Override 100%
6.0	iCh_Cpl01	CPL-Kundenausgang 1	7.0	iCh_Cpl09	CPL-Kundenausgang 9
6.1	iCh_Cpl02	CPL-Kundenausgang 2	7.1	iCh_Cpl10	CPL-Kundenausgang 10
6.2	iCh_Cpl03	CPL-Kundenausgang 3	7.2	iCh_Cpl11	CPL-Kundenausgang 11
6.3	iCh_Cpl04	CPL-Kundenausgang 4	7.3	iCh_Cpl12	CPL-Kundenausgang 12
6.4	iCh_Cpl05	CPL-Kundenausgang 5	7.4	iCh_Cpl13	CPL-Kundenausgang 13

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
6.5	iCh_Cpl06	CPL-Kundenausgang 6	7.5	iCh_Cpl14	CPL-Kundenausgang 14
6.6	iCh_Cpl07	CPL-Kundenausgang 7	7.6	iCh_Cpl15	CPL-Kundenausgang 15
6.7	iCh_Cpl08	CPL-Kundenausgang 8	7.7	iCh_Cpl16	CPL-Kundenausgang 16
8.0	iCh_Custom1	Kundenausgang 1	9.0	iCh_G0Act	Eilgang aktiv
8.1	iCh_Custom2	Kundenausgang 2	9.1	iCh_InPosAct	Inpos-Bereich 2 aktiv
8.2	iCh_Custom3	Kundenausgang 3	9.2	iCh_G41G141Act	G41/G141 aktiv
8.3	iCh_Custom4	Kundenausgang 4	9.3	iCh_G42G142Act	G42/G142 aktiv
8.4	iCh_Custom5	Kundenausgang 5	9.4	iCh_RetraceAct	Rückwärtsbetrieb aktiv
8.5	iCh_Custom6	Kundenausgang 6	9.5	-	res.
8.6	iCh_Custom7	Kundenausgang 7	9.6	iCh_CoordCoupleAct	Koord. Kopplung aktiv
8.7	iCh_Custom8	Kundenausgang 8	9.7	iCh_TangTRotCmd	Werkzeugdrehung (TangTool)
10.0	iCh_ActFunc01	G70 aktiv	11.0	iCh_ActFunc09	Werkzeugkorrektur aktiv Bit 0
10.1	iCh_ActFunc02	Vorschub 100% aktiv	11.1	iCh_ActFunc10	Werkzeugkorrektur aktiv Bit 1
10.2	iCh_ActFunc03	G95 aktiv	11.2	iCh_ActFunc11	Werkzeugkorrektur aktiv Bit 2
10.3	iCh_ActFunc04	Achstrafo 2 inaktiv	11.3	iCh_ActFunc12	Werkzeugkorrektur aktiv Bit 3
10.4	iCh_ActFunc05	Programmposition aktiv	11.4	iCh_ActFunc13	Werkzeugkorrektur aktiv Bit 4
10.5	iCh_ActFunc06	Gewindezyklus aktiv	11.5	iCh_ActFunc14	nicht belegt
10.6	iCh_ActFunc07	Gewindebohren aktiv	11.6	iCh_ActFunc15	nicht belegt
10.7	iCh_ActFunc08	Gewindeschneiden aktiv	11.7	iCh_ActFunc16	nicht belegt
12.0	iCh_ActFunc17	nicht belegt	13.0	iCh_TestFeed	Testvorschub aktiv
12.1	iCh_ActFunc18	G96 aktiv	13.1	iCh_TestRap	Testeilgang aktiv
12.2	iCh_ActFunc19	nicht belegt	13.2	iCh_RapOverride0	Eilgang Override 0%
12.3	iCh_ActFunc20	nicht belegt	13.3	iCh_RapOverride100	Eilgang Override 100%
12.4	iCh_ActFunc21	nicht belegt	13.4	iCh_Feedrate2	Zweiter Vorschub
12.5	iCh_ActFunc22	nicht belegt	13.5	-	res.
12.6	iCh_ActFunc23	nicht belegt	13.6	-	res.
12.7	iCh_ActFunc24	nicht belegt	13.7	-	res.

Tab. 5-4: Übersicht Eingangssignale (NC → SPS)

Übersicht Interface-Signale

5.5 Achsbezogene Ausgangssignale (SPS --> NC)

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
0.0	qAx_OpModeSel_00	Achsbetriebsart Bit 0	1.0	qAx_TrVLim_00	Endschalterbereich Bit 0
0.1	qAx_OpModeSel_01	Achsbetriebsart Bit 1	1.1	qAx_TrVLim_01	Endschalterbereich Bit 1
0.2	qAx_JogPlus	Manuell +	1.2	qAx_SwLimOff	Endschalter ausblenden
0.3	qAx_JogMinus	Manuell -	1.3	-	res.
0.4	qAx_JogInch	Schrittmaß in Inch	1.4	-	res.
0.5	qAx_JogDia	Schrittmaß Durchmesser	1.5	-	res.
0.6	qAx_NextNotch	Nächste Rasterposition	1.6	-	res.
0.7	qAx_Reset	Achsgrundstellung	1.7	qAx_FxStopRel	Festanschlag aufheben
2.0	qAx_SafOpModeSwitch	Betriebsartenwahl (BA)	3.0	qAx_HandwSel_00	Handradauswahl Bit 0
2.1	qAx_SafDrvLock	Antriebssperre	3.1	qAx_HandwSel_01	Handradauswahl Bit 1
2.2	qAx_SafEnablCtrl	Zustimmtaste (ZT)	3.2	qAx_HandwDir	Handrad Drehrichtung
2.3	qAx_SafSwitch1	Si-Schalter 1 (S1)	3.3	qAx_HandwPosMode	Positionshandrad
2.4	qAx_SafSwitch2	Si-Schalter 2 (S2)	3.4	-	res.
2.5	qAx_SafCheckInputState	Prüfeingang Si-Status	3.5	-	res.
2.6	qAx_SafTechState	Status Si-Signale	3.6	-	res.
2.7	qAx_SafRedTransTime	Überführzeit abkürzen	3.7	-	res.
4.0	qAx_ManFeed_00	Handvorschub Bit 0	5.0	-	res.
4.1	qAx_ManFeed_01	Handvorschub Bit 1	5.1	-	res.
4.2	qAx_ManFeed_02	Handvorschub Bit 2	5.2	qAx_HomeSwitch	Referenzschalter
4.3	qAx_ManFeed_03	Handvorschub Bit 3	5.3	-	res.
4.4	-	res.	5.4	-	res.
4.5	-	res.	5.5	-	res.
4.6	-	res.	5.6	-	res.
4.7	-	res.	5.7	qAx_Override100	Achs-Override 100 %
6.0	qAx_Override_00	Override Bit 0	7.0	qAx_Override_08	Override Bit 8
6.1	qAx_Override_01	Override Bit 1	7.1	qAx_Override_09	Override Bit 9
6.2	qAx_Override_02	Override Bit 2	7.2	qAx_Override_10	Override Bit 10
6.3	qAx_Override_03	Override Bit 3	7.3	qAx_Override_11	Override Bit 11
6.4	qAx_Override_04	Override Bit 4	7.4	qAx_Override_12	Override Bit 12
6.5	qAx_Override_05	Override Bit 5	7.5	qAx_Override_13	Override Bit 13

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
6.6	qAx_Override_06	Override Bit 6	7.6	qAx_Override_14	Override Bit 14
6.7	qAx_Override_07	Override Bit 7	7.7	qAx_Override_15	Override Bit 15
8.0	qAx_Custom1	Kundeneingang 1	9.0	-	res.
8.1	qAx_Custom2	Kundeneingang 2	9.1	-	res.
8.2	qAx_Custom3	Kundeneingang 3	9.2	-	res.
8.3	qAx_Custom4	Kundeneingang 4	9.3	-	res.
8.4	qAx_Custom5	Kundeneingang 5	9.4	-	res.
8.5	qAx_Custom6	Kundeneingang 6	9.5	-	res.
8.6	qAx_Custom7	Kundeneingang 7	9.6	-	res.
8.7	qAx_Custom8	Kundeneingang 8	9.7	-	res.
10.0	qAx_TrqErrOff	Stillstandsfehler ausblenden	11.0	qAx_Discharge	Achse abgebaut
10.1	qAx_LagErrOff	Kopplungsfehler ausblenden	11.1	qAx_FrzIpoPos	Sollposition halten
10.2	qAx_MasterPos	Gantry auf Masterposition	11.2	-	res.
10.3	qAx_TakeActOffs	Istwert-Versatz übernehmen	11.3	qAx_TrqLim	Momentenreduktion
10.4	-	res.	11.4	-	res.
10.5	-	res.	11.5	-	res.
10.6	-	res.	11.6	qAx_DrvOn	Antrieb Ein
10.7	qAx_RedDelayErr-Class1	Verz. Fehlerreaktion abkürzen	11.7	qAx_DrvLock	Vorschub Sperre

Tab. 5-5: Übersicht Ausgangssignale (SPS → NC)

5.6 Achsbezogene Eingangssignale (NC → SPS)

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
0.0	iAx_RefKnown	Referenzpunkt bekannt	1.0	iAx_DistCtrl	Achse vor Endpunkt
0.1	iAx_RefReached	Referenzpunkt erreicht	1.1	iAx_ProgSpReach	Achsgeschwindigkeit erreicht
0.2	iAx_TrvCmd	Fahrbehl	1.2	-	res.
0.3	iAx_TrvDirNeg	Negative Verfahr-richtung	1.3	-	res.
0.4	iAx_Run	Achse läuft	1.4	-	res.
0.5	iAx_InPos	Achse in Position	1.5	-	res.
0.6	iAx_NotchPos	Achse auf Rasterposition	1.6	iAx_FxStopReached	Festanschlag erreicht
0.7	iAx_Reset	Achse grundgestellt	1.7	iAx_FxStopAct	Festanschlag aktiv

Übersicht Interface-Signale

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
2.0	iAx_SafOpMode_00	Si-Betriebsart Bit 0	3.0	iAx_PosSwitch1	Positionsschaltpunkt 1
2.1	iAx_SafOpMode_01	Si-Betriebsart Bit 1	3.1	iAx_PosSwitch2	Positionsschaltpunkt 2
2.2	iAx_SafOpMode_02	Si-Betriebsart Bit 2	3.2	iAx_PosSwitch3	Positionsschaltpunkt 3
2.3	iAx_SafOpMode_03	Si-Betriebsart Bit3	3.3	iAx_PosSwitch4	Positionsschaltpunkt 4
2.4	iAx_SafStatePos	Status Sichere Lage	3.4	iAx_PosSwitch5	Positionsschaltpunkt 5
2.5	iAx_SafCtrlOutputState	Statusausgang Regler	3.5	iAx_PosSwitch6	Positionsschaltpunkt 6
2.6	-	res.	3.6	iAx_PosSwitch7	Positionsschaltpunkt 7
2.7	-	res.	3.7	iAx_PosSwitch8	Positionsschaltpunkt 8
4.0	iAx_ChIndex_00	Kanalnummer Bit 0	5.0	-	res.
4.1	iAx_ChIndex_01	Kanalnummer Bit 1	5.1	-	res.
4.2	iAx_ChIndex_02	Kanalnummer Bit 2	5.2	-	res.
4.3	iAx_ChIndex_03	Kanalnummer Bit 3	5.3	-	res.
4.4	iAx_ChIndex_04	Kanalnummer Bit 4	5.4	-	res.
4.5	iAx_ChIndex_05	Kanalnummer Bit 5	5.5	-	res.
4.6	-	res.	5.6	iAx_Override0	Achs-Override 0 %
4.7	-	res.	5.7	iAx_Override100	Achs-Override 100 %
6.0	iAx_ScsState00	SCS-Signalstatus 0	7.0	iAx_ScsState08	SCS-Signalstatus 8
6.1	iAx_ScsState01	SCS-Signalstatus 1	7.1	iAx_ScsState09	SCS-Signalstatus 9
6.2	iAx_ScsState02	SCS-Signalstatus 2	7.2	iAx_ScsState10	SCS-Signalstatus 10
6.3	iAx_ScsState03	SCS-Signalstatus 3	7.3	iAx_ScsState11	SCS-Signalstatus 11
6.4	iAx_ScsState04	SCS-Signalstatus 4	7.4	iAx_ScsState12	SCS-Signalstatus 12
6.5	iAx_ScsState05	SCS-Signalstatus 5	7.5	iAx_ScsState13	SCS-Signalstatus 13
6.6	iAx_ScsState06	SCS-Signalstatus 6	7.6	iAx_ScsState14	SCS-Signalstatus 14
6.7	iAx_ScsState07	SCS-Signalstatus 7	7.7	iAx_ScsState15	SCS-Signalstatus 15
8.0	iAx_Custom1	Kundenausgang 1	9.0	iAx_MasterAxIndex_00	Index Master-Achse Bit 0
8.1	iAx_Custom2	Kundenausgang 2	9.1	iAx_MasterAxIndex_01	Index Master-Achse Bit 1
8.2	iAx_Custom3	Kundenausgang 3	9.2	iAx_MasterAxIndex_02	Index Master-Achse Bit 2
8.3	iAx_Custom4	Kundenausgang 4	9.3	iAx_MasterAxIndex_03	Index Master-Achse Bit 3
8.4	iAx_Custom5	Kundenausgang 5	9.4	iAx_MasterAxIndex_04	Index Master-Achse Bit 4
8.5	iAx_Custom6	Kundenausgang 6	9.5	iAx_MasterAxIndex_05	Index Master-Achse Bit 5
8.6	iAx_Custom7	Kundenausgang 7	9.6	iAx_MasterAxIndex_06	Index Master-Achse Bit 6

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
8.7	iAx_Custom8	Kundenausgang 8	9.7	iAx_MasterAxIndex_07	Index Master-Achse Bit 7
10.0	iAx_TrqExceed	Stillstandsfehler	11.0	iAx_DrvErrClass1	Fehler Zustandsklasse-1
10.1	iAx_CoupleLag	Nachlauffehler	11.1	iAx_DrvChangeClass2	Änderung Zustandsklasse-2
10.2	iAx_CmdOffsExst	Gantry-Sollwert-Versatzaktiv	11.2	iAx_DrvChangeClass3	Änderung Zustandsklasse-3
10.3	iAx_CmdOffsExceed	Ausgleichbarer Gantry-Sollwert-Versatz überschritten	11.3	iAx_TrqLim	Moment reduziert
10.4	-	res.	11.4	iAx_DryRun	Testbetrieb
10.5	-	res.	11.5	iAx_DrvPower	Freigabe zur Leistungszuschaltung
10.6	-	res.	11.6	iAx_DrvReady	Antrieb betriebsbereit
10.7	iAx_DelayErrClass1	Verzögerte Fehlerreaktion	11.7	iAx_DrvAct	Antrieb in Betrieb

Tab. 5-6: Übersicht Eingangssignale (NC → SPS)

5.7 Spindelbezogene Ausgangssignale (SPS → NC)

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
0.0	qSp_CAxOn	C-Achse Ein	1.0	qSp_TurnCW	Spindel M3 manuell
0.1	qSp_CAxOff	C-Achse Aus	1.1	qSp_TurnCCW	Spindel M4 manuell
0.2	qSp_JogPlus	Spindel tippen M3	1.2	qSp_Stop	Spindel M5 manuell
0.3	qSp_JogMinus	Spindel tippen M4	1.3	qSp_Orientate	Spindel M19 manuell
0.4	-	res.	1.4	-	res.
0.5	-	res.	1.5	-	res.
0.6	-	res.	1.6	-	res.
0.7	qSp_Reset	Spindelgrundstellung	1.7	-	res.
2.0	qSp_SafOpModeSwitch	Betriebsartenwahl (BA)	3.0	qSp_Gear1Act	Quittung GTS 1
2.1	qSp_SafDrvLock	Antriebssperre	3.1	qSp_Gear2Act	Quittung GTS 2
2.2	qSp_SafEnablCtrl	Zustimmtaste (ZT)	3.2	qSp_Gear3Act	Quittung GTS 3
2.3	qSp_SafSwitch1	Si-Schalter 1 (S1)	3.3	qSp_Gear4Act	Quittung GTS 4
2.4	qSp_SafSwitch2	Si-Schalter 2 (S2)	3.4	-	res.
2.5	qSp_SafCheckInputState	Prüfeingang Si-Status	3.5	-	res.
2.6	qSp_SafTechState	Status Si-Signale	3.6	-	res.
2.7	qSp_SafRedTransTime	Überführzeit abkürzen	3.7	qSp_GearIdleAct	Quittung Leerlauf

Übersicht Interface-Signale

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
4.0	qSp_ManSpeed_00	Drehzahl tippen Bit 0	5.0	qSp_SpeedLimit	Drehzahlbegrenzung
4.1	qSp_ManSpeed_01	Drehzahl tippen Bit 1	5.1	qSp_SValueSD	S-Wert-Vorgabe über SD
4.2	qSp_ManSpeed_02	Drehzahl tippen Bit 2	5.2	qSp_HomeSwitch	Referenzschalter
4.3	-	res.	5.3	-	res.
4.4	-	res.	5.4	-	res.
4.5	-	res.	5.5	-	res.
4.6	-	res.	5.6	-	res.
4.7	-	res.	5.7	qSp_Override100	Spindeloverride 100%
6.0	qSp_Override_00	Override Bit 0	7.0	qSp_Override_08	Override Bit 8
6.1	qSp_Override_01	Override Bit 1	7.1	qSp_Override_09	Override Bit 9
6.2	qSp_Override_02	Override Bit 2	7.2	qSp_Override_10	Override Bit 10
6.3	qSp_Override_03	Override Bit 3	7.3	qSp_Override_11	Override Bit 11
6.4	qSp_Override_04	Override Bit 4	7.4	qSp_Override_12	Override Bit 12
6.5	qSp_Override_05	Override Bit 5	7.5	qSp_Override_13	Override Bit 13
6.6	qSp_Override_06	Override Bit 6	7.6	qSp_Override_14	Override Bit 14
6.7	qSp_Override_07	Override Bit 7	7.7	qSp_Override_15	Override Bit 15
8.0	qSp_Custom1	Kundeneingang 1	9.0	-	res.
8.1	qSp_Custom2	Kundeneingang 2	9.1	-	res.
8.2	qSp_Custom3	Kundeneingang 3	9.2	-	res.
8.3	qSp_Custom4	Kundeneingang 4	9.3	-	res.
8.4	qSp_Custom5	Kundeneingang 5	9.4	-	res.
8.5	qSp_Custom6	Kundeneingang 6	9.5	-	res.
8.6	qSp_Custom7	Kundeneingang 7	9.6	-	res.
8.7	qSp_Custom8	Kundeneingang 8	9.7	-	res.
10.0	-	res.	11.0	-	res.
10.1	-	res.	11.1	-	res.
10.2	-	res.	11.2	-	res.
10.3	-	res.	11.3	-	res.
10.4	-	res.	11.4	-	res.
10.5	-	res.	11.5	-	res.
10.6	-	res.	11.6	qSp_DrvOn	Antrieb Ein
10.7	qSp_RedDelayErr-Class1	Verz. Fehlerreaktion abkürzen	11.7	qSp_DrvLock	Spindel Sperre

Tab. 5-7: Übersicht Ausgangssignale (SPS → NC)

5.8 Spindelbezogene Eingangssignale (NC --> SPS)

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
0.0	iSp_CAxAct	C-Achse ist aktiv	1.0	iSp_ProgSpReach	Drehzahl erreicht
0.1	iSp_CAxSwitch	C-Achs-Umschaltung	1.1	iSp_SpLim	Drehzahl begrenzt
0.2	iSp_TurnCmd	Drehbefehl	1.2	iSp_Stop	Spindel steht
0.3	iSp_TurnDirM4	Drehrichtung M4	1.3	iSp_OrientateFinish	Spindel gerichtet
0.4	-	res.	1.4	iSp_OrientateAct	Spindel richten aktiv
0.5	iSp_InPos	Spindel In Position	1.5	-	res.
0.6	iSp_PosCtrl	Lagereglung aktiv	1.6	-	res.
0.7	iSp_Reset	Spindel grundgestellt	1.7	-	res.
2.0	iSp_SafOpMode_00	Si-Betriebsart Bit 0	3.0	iSp_Gear1Sel	Anwahl GTS 1
2.1	iSp_SafOpMode_01	Si-Betriebsart Bit 1	3.1	iSp_Gear2Sel	Anwahl GTS 2
2.2	iSp_SafOpMode_02	Si-Betriebsart Bit 2	3.2	iSp_Gear3Sel	Anwahl GTS 3
2.3	iSp_SafOpMode_03	Si-Betriebsart Bit3	3.3	iSp_Gear4Sel	Anwahl GTS 4
2.4	iSp_SafStatePos	Status Sichere Lage	3.4	iSp_GearChange	GTS Umschaltung
2.5	iSp_SafCtrlOutputState	Statusausgang Regler	3.5	iSp_IdleSpeed	Trudeldrehzahl erreicht
2.6	-	res.	3.6	-	res.
2.7	-	res.	3.7	iSp_GearIdleSel	Anwahl Getriebe-Leerlauf
4.0	-	res.	5.0	-	res.
4.1	-	res.	5.1	-	res.
4.2	-	res.	5.2	-	res.
4.3	-	res.	5.3	-	res.
4.4	-	res.	5.4	-	res.
4.5	-	res.	5.5	-	res.
4.6	-	res.	5.6	iSp_Override0	Spindeloverride 0 %
4.7	iSp_ToolLimAct	Werkzeugbegrenzungen aktiv	5.7	iSp_Override100	Spindeloverride 100 %
6.0	iSp_ScsState00	SCS-Signalstatus 0	7.0	iSp_ScsState08	SCS-Signalstatus 8
6.1	iSp_ScsState01	SCS-Signalstatus 1	7.1	iSp_ScsState09	SCS-Signalstatus 9
6.2	iSp_ScsState02	SCS-Signalstatus 2	7.2	iSp_ScsState10	SCS-Signalstatus 10
6.3	iSp_ScsState03	SCS-Signalstatus 3	7.3	iSp_ScsState11	SCS-Signalstatus 11
6.4	iSp_ScsState04	SCS-Signalstatus 4	7.4	iSp_ScsState12	SCS-Signalstatus 12
6.5	iSp_ScsState05	SCS-Signalstatus 5	7.5	iSp_ScsState13	SCS-Signalstatus 13
6.6	iSp_ScsState06	SCS-Signalstatus 6	7.6	iSp_ScsState14	SCS-Signalstatus 14
6.7	iSp_ScsState07	SCS-Signalstatus 7	7.7	iSp_ScsState15	SCS-Signalstatus 15
8.0	iSp_Custom1	Kundenausgang 1	9.0	iSp_CoupleIndex_00	Nr. der Kopplung Bit 0

Übersicht Interface-Signale

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
8.1	iSp_Custom2	Kundenausgang 2	9.1	iSp_CoupleIndex_01	Nr. der Kopplung Bit 1
8.2	iSp_Custom3	Kundenausgang 3	9.2	iSp_CoupleIndex_02	Nr. der Kopplung Bit 2
8.3	iSp_Custom4	Kundenausgang 4	9.3	-	res.
8.4	iSp_Custom5	Kundenausgang 5	9.4	-	res.
8.5	iSp_Custom6	Kundenausgang 6	9.5	-	res.
8.6	iSp_Custom7	Kundenausgang 7	9.6	-	res.
8.7	iSp_Custom8	Kundenausgang 8	9.7	-	res.
10.0	iSp_Master	Spindel ist Master	11.0	iSp_DrvErrClass1	Fehler Zustandsklasse-1
10.1	iSp_CoupleErr	Kopplungsfehler	11.1	iSp_DrvChangeClass2	Änderung Zustandsklasse-2
10.2	iSp_Synchr1	Synchronlauf 1	11.2	iSp_DrvChangeClass3	Änderung Zustandsklasse-3
10.3	iSp_Synchr2	Synchronlauf 2	11.3	-	res.
10.4	-	res.	11.4	iSp_DryRun	Testbetrieb
10.5	-	res.	11.5	iSp_DrvPower	Freigabe zur Leistungszuschaltung
10.6	-	res.	11.6	iSp_DrvReady	Antrieb betriebsbereit
10.7	iSp_DelayErrClass1	Verzögerte Fehlerreaktion	11.7	iSp_DrvAct	Antrieb in Betrieb

Tab. 5-8: Übersicht Eingangssignale (NC → SPS)

6 Globale Interface-Signale

6.1 Übersicht globale Interface-Signale

6.1.1 Allgemeines

Die Signale des **globalen Interface** sind übergeordnete Signale, die im Zusammenhang mit Funktionen stehen, die sich auf die gesamte NC beziehen.

6.1.2 Übersicht Ausgangssignale (SPS --> NC)

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
0.0	qGen_Reset	Systemgrundstellung	1.0	qGen_StrokeInhibit	Hubsperr
0.1	-	res.	1.1	qGen_StrokeReserv	Hubreservierung
0.2	qGen_DelErrInfo	Fehler Infos löschen	1.2	qGen_StrokeRel	Hub ein
0.3	qGen_SercosIO_Reset	Sercos-I/O-Geräte-Fehler löschen	1.3	-	res.
0.4	-	res.	1.4	-	res.
0.5	-	res.	1.5	-	res.
0.6	-	res.	1.6	-	res.
0.7	-	res.	1.7	-	res.
2.0	qGen_Crash	Crash-Zustand von SPS an NC (IPO)	3.0	-	res.
2.1	qGen_CrashStoreDataAckn	Quittung Crash-Zustand von NC an SPS	3.1	-	res.
2.2	-	res.	3.2	-	res.
2.3	-	res.	3.3	-	res.
2.4	-	res.	3.4	-	res.
2.5	-	res.	3.5	-	res.
2.6	-	res.	3.6	-	res.
2.7	-	res.	3.7	-	res.

Tab. 6-1: Übersicht Ausgangssignale (SPS --> NC)

6.1.3 Übersicht Eingangssignale (NC --> SPS)

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
0.0	iGen_Reset	Systemgrundstellung ausgeführt	1.0	iGen_StrokeIntend	Hub beabsichtigt
0.1	iGen_HardwareState	Hardware Warnung	1.1	iGen_NoStroke	Hub läuft nicht
0.2	-	res.	1.2	-	res.
0.3	iGen_SercosIO_Reset	Sercos-I/O-Geräte-Fehler gelöscht	1.3	-	res.
0.4	-	res.	1.4	-	res.
0.5	-	res.	1.5	-	res.

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
0.6	-	res.	1.6	-	res.
0.7	-	res.	1.7	-	res.
2.0	iGen_CrashAckn	Quittierung Crash-Zustand von SPS an NC (IPO)	3.0	-	res.
2.1	iGen_CrashStoreData	Crash-Zustand von NC an SPS	3.1	-	res.
2.2	-	res.	3.2	-	res.
2.3	-	res.	3.3	-	res.
2.4	-	res.	3.4	-	res.
2.5	-	res.	3.5	-	res.
2.6	-	res.	3.6	-	res.
2.7	-	res.	3.7	-	res.

Tab. 6-2: Übersicht Eingangssignale (NC --> SPS)

6.2 Signalbeschreibung

6.2.1 Ausgangssignale (SPS --> NC)

Systemgrundstellung qGen_Reset

Dieses Signal wirkt auf alle Kanäle der NC. Die Wirkung auf die einzelnen Kanäle ist identisch mit einer Grundstellung jedes einzelnen Kanals.

Systemgrundstellung wirkt sich aus auf:

- Achsbezogene Signale der NC zur SPS
- *Die kanalbezogenen Signale der NC:*
 - Grundstellung ausgeführt
 - NC-Bereit
 - Programm läuft

Signalpegel 1: In **allen** Kanälen wird Systemgrundstellung ausgelöst.
0: Keine Auswirkung.

Fehler Infos löschen qGen_DelErrInfo

Das Signal "Fehler Infos löschen" erlaubt per SPS das Löschen aller Fehler-, Warnungs- und Meldungseinträge innerhalb des Diagnosespeichers der Diagnoseübersicht (Info-Dialog) der CNC-System MTX Standard-Bedienoberfläche. Das Interface-Signal hat die gleiche Wirkung wie die F-Taste "Löschen", wenn die F-Taste für den Filter auf dem Zustand "Alle" eingestellt ist.

Signalpegel 1: Löschen aller Einträge von Fehlern, Warnungen und Meldungen innerhalb des Diagnosespeichers.
0: Keine Auswirkung.

Sercos-I/O-Geräte-Fehler löschen qGen_SercosIO_Reset

Das Signal "Sercos-I/O-Geräte-Fehler löschen" erlaubt per SPS das Löschen aller C1D-Fehler, die aktuell an den Sercos-I/O-Geräten anstehen.

Signalpegel 1: Löschen der Sercos-I/O-Geräte-Fehler.

0: Keine Auswirkung.

Hubsperrung qGen_StrokeInhibit

Mit diesem Signal sperrt die SPS das Auslösen eines Stanzhubs, d. h. auf der Einsteckkarte High Speed I/O wird der entsprechende schnelle Ausgang gesperrt.

Signalpegel 1: Der schnelle Ausgang HSO ist gesperrt.
0: Keine Auswirkung.

Hubreservierung qGen_StrokeReserv

Mit diesem Signal reserviert sich die SPS einen schnellen Ausgang HSO auf der Einsteckkarte High Speed I/O für eine eigene Hubauslösung (siehe [Kap. "Hub ein qGen_StrokeRel" auf Seite 33](#)).

Signalpegel 1: Schneller Ausgang HSO ist für die SPS reserviert.
0: Keine Auswirkung.

Hub ein qGen_StrokeRel

Mit dem Signal **Hub ein**beauftragt die SPS die NC, einen Stanzhub über den schnellen Ausgang HSO auf der Einsteckkarte High Speed I/O auszulösen.

Signalflanke/-pegel 0 → 1: Stanzhub über den schnellen Ausgang HSO per SPS auslösen.

0: Keine Auswirkung.

Stanz-HS-Logik Das Auslösen einzelner Hübe kann per SPS realisiert werden.

Die SPS kann einen Hub auslösen, indem sie die NC über Interface-Signale beauftragt, dies zu tun.

Zwischen der NC, der Stanzsteuerung und der SPS werden Bit-Signale entsprechend nachfolgender Abbildung verwendet:

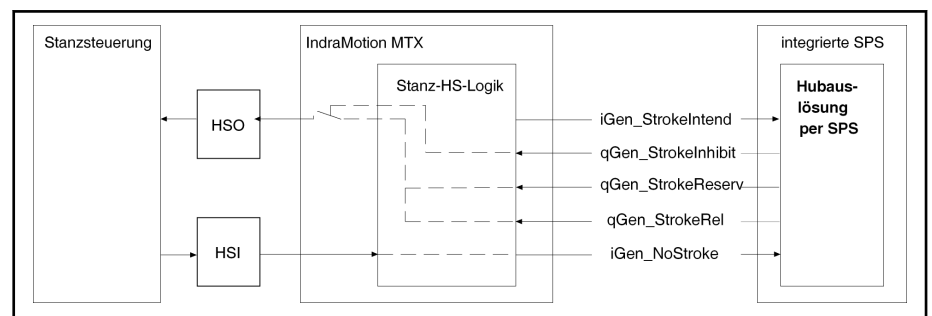


Abb. 6-1: Stanz-HS-Logik

Die Stanz-HS-Logik verarbeitet fünf Interface-Signale:

Interface-Signal	Bedeutung
iGen_StrokeIntend	"Hub beabsichtigt" Die NC teilt der SPS mit, dass sie einen Hub auslösen möchte.
iGen_NoStroke	"Hub läuft nicht" Der schnelle Eingang HSI wird von der Stanz-HS-Logik an die SPS weiter gereicht.
qGen_StrokeInhibit	"Hubsperrung" Die SPS sperrt das Setzen von HSO.

Interface-Signal	Bedeutung
qGen_StrokeReserv	"Hubreservierung" Die SPS reserviert sich den schnellen Ausgang HSO für eine eigene Hubauslösung.
qGen_StrokeRel	"Hub ein" Die SPS beauftragt die NC, einen Hub auszulösen.

Tab. 6-3: Interface-Signale

Ein Stanzhub kann sowohl über die NC als auch über die SPS ausgelöst werden:

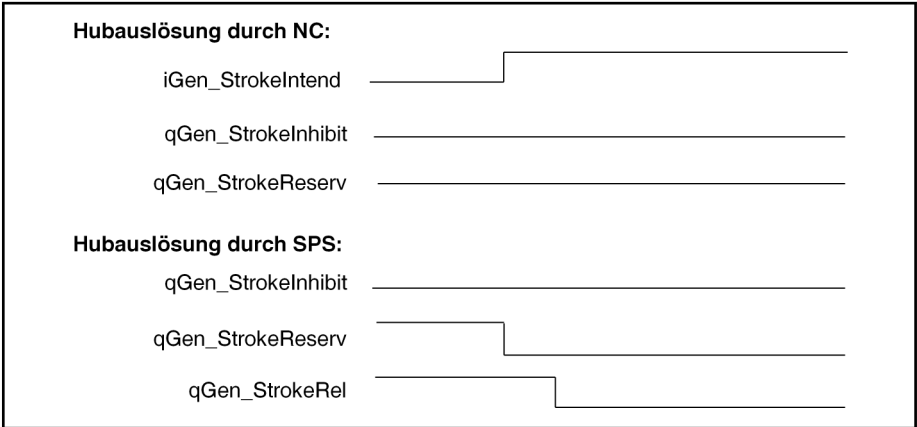


Abb. 6-2: Hubauslösung durch NC

Crash-Zustand von SPS an NC (IPO) `qGen_Crash`

Dieses Signal wird von der SPS gesetzt, wenn sie einen selbst detektierten Crash erkennt. Damit wird beim Aktionsrekorder der Crashtigger "SPS-Fehlermeldung" ausgelöst. Das Signal kann von der SPS nach dem Setzen des Quittungssignals "`iGen_CrashAckn`" wieder zurückgesetzt werden.



Die Verarbeitung eines Crashsignals von der SPS erfolgt im Aktionsrekorder nur bei aktiviertem Eintrag "SPS-Fehlermeldung".

Signalflanke **0 → 1:** Crash durch SPS signalisiert
 1 → 0: Vorgang abgeschlossen

Quittung Crash-Zustand von NC an SPS `qGen_CrashStoreDataAckn`

Quittungssignal der SPS an die NC , dass der Auftrag der NC verarbeitet wurde. Nach dem Erkennen dieses Signals setzt der Aktionsrekorder das Signal "`iGenCrashStoreData`" wieder zurück.



Einen neuen Crash kann der Aktionsrekorder bereits registrieren, sobald der bisherige Crash verarbeitet wurde. Ein neuer Crash wird der SPS jedoch erst dann durch eine 0 -> 1 Flanke angezeigt, wenn diese Flanke den vorherigen Fehler quittiert hat.

Signalpegel **1:** Auftrag der NC fertig bearbeitet.
 0: Vorgang abgeschlossen.

6.2.2 Eingangssignale (NC --> SPS)

Systemgrundstellung ausgeführt

Dieses Signal wirkt im Zusammenhang mit dem Signal Systemgrundstellung oder der Funktion Systemgrundstellung, die über die NC-Bedienoberfläche eingeleitet wird (siehe auch Signaldiagramm Systemgrundstellung auf [Kap. "Systemgrundstellung qGen_Reset" auf Seite 32](#)).

Signalpegel	1:	Steuerungsintern ist die Funktion Grundstellung in allen Kanälen ausgeführt und alle Kanäle befindet sich im Einschaltzustand. Mit diesem Signal muss das Signal Systemgrundstellung zurückgesetzt werden. Der Signalpegel 1 bleibt erhalten, solange sich alle Kanäle im Einschaltzustand befindet und noch kein Programm angewählt ist.
	0:	Mindestens ein Kanal befindet sich nicht mehr im Einschaltzustand, ein NC-Teileprogramm ist angewählt oder bereits aktiv.

Hardware Warnung iGen_HardwareState

*Das Signal **Hardware Warnung** wird gesetzt, wenn eine oder mehrere der folgenden Bedingungen erfüllt ist:*

- Die NC-Baugruppe meldet "Übertemperatur".
- Die NC-Baugruppe meldet "Batterie Vorwarnung".

Das SPS-Programm kann dann die folgenden Systemdaten auswerten und daraus eine adäquate Reaktion ableiten (siehe Funktionsbeschreibung):

- /SysHwStates/BattFail
- /SysHwStates/OverTemp
- /SysStates/BdTemp/CpuTemp

Sercos-I/O-Geräte-Fehler gelöscht

*Das Signal **iGen_SercosIO_Reset** wird gesetzt, wenn die folgenden zwei Bedingungen erfüllt sind:*

- Das SPS-Signal "qGen_SercosIO_Reset" ist gesetzt.
- Die an den Sercos-I/O-Geräten anstehenden C1D-Fehler sind gelöscht.

Hub beabsichtigt iGen_StrokeIntend

Mit dem Signal **Hub beabsichtigt** teilt die NC der SPS mit, dass sie einen Stanzhub auslösen möchte. Erlaubt die SPS das Auslösen (qGen_StrokeInhibit = 0 und qGen_StrokeReserv = 0), wird der schnelle Ausgang HSO auf der Einsteckkarte High Speed I/O von der Stanz-HS-Logik gesetzt ([Kap. 6.2 "Signalbeschreibung" auf Seite 32](#)).

Signalpegel	1:	NC teilt der SPS mit, dass sie beabsichtigt einen Hub auszulösen.
	0:	Keine Auswirkung.

Hub läuft nicht iGen_NoStroke

Das Antwortsignal **Hub läuft nicht** der Stanzsteuerung (schneller Eingang HSI auf der Einsteckkarte High Speed I/O) wird an die SPS weitergereicht.

Signalpegel	1:	Hub läuft nicht = Antwortsignal der Stanzsteuerung an die SPS.

0: Hub läuft.

Quittierung Crash-Zustand von SPS an NC (IPO) iGen_CrashAckn

Die Quittierung dient der SPS als Bestätigung, dass der Aktionsrekorder das Crashsignal erhalten hat. Es wird vom Aktionsrekorder zurückgesetzt, wenn dieser alle notwendigen Daten gesichert hat und der reguläre Betrieb wieder aufgenommen wurde

Signalflanke	0 → 1:	Crash wurde erkannt und wird verarbeitet.
	1 → 0:	Aktionsrekorder hat Crash verarbeitet.

Crash-Zustand von NC an SPS iGen_CrashStoreData

Dieses Signal wird vom Aktionsrekorder gesetzt, wenn im Interpolator der Crashtrigger "Sercos Antriebsfehler (Zustandsklasse 1)" ausgelöst wurde. Damit wird der SPS signalisiert, dass diese gegebenenfalls weitere Operationen ausführen soll.

Nach der Durchführung dieser Operationen signalisiert die SPS mit dem Signal "qGen_CrashStoreDataAckn", dass die Bearbeitung abgeschlossen ist.



Einen neuen Crash kann der Aktionsrekorder bereits registrieren, sobald der bisherige Crash verarbeitet wurde. Ein neuer Crash wird der SPS jedoch erst dann durch eine 0 → 1 Flanke angezeigt, wenn diese Flanke den vorherigen Crash quittiert hat.

Signalflanke	0 → 1:	Ein Crash ist aufgetreten; SPS-seitige Operationen durchführen.
	0:	Vorgang abgeschlossen.

7 Kanalbezogene Interface-Signale

7.1 Übersicht kanalbezogene Interface-Signale

7.1.1 Allgemeines

Das CNC-System MTX verfügt über eine **Kanalstruktur**, die es ermöglicht, mehrere Aufgaben parallel zu bearbeiten.

Die kanalbezogenen Interface-Signale wirken daher hauptsächlich auf die NC-Funktionen, die mit dem Abarbeiten des NC-Teilprogramms im Zusammenhang stehen.

Jeder Kanal hat ein eigenes Interface, d. h., bei n-Kanälen wird das unter [Kap. 7.1.2 "Übersicht kanalbezogene Ausgangssignale \(SPS --> NC\)"](#) auf [Seite 37](#) und [Kap. 7.1.3 "Übersicht Eingangssignale \(NC --> SPS\)"](#) auf [Seite 39](#) dargestellte Interface n-mal auf die entsprechenden Merker abgebildet.

Für die in einem Kanal zusammengefassten Bearbeitungsachsen sowie für die Hilfsachsen ist außerdem das achsbezogene Interface von Bedeutung.

Es müssen immer mindestens 2 Kanäle eingetragen sein:

- Kanal 0 für Schaltfunktionen
- Kanal 1 für einen Bearbeitungskanal

In der folgenden Beschreibung sind die Interface-Signale für einen Kanal dargestellt.

7.1.2 Übersicht kanalbezogene Ausgangssignale (SPS --> NC)

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
0.0	qCh_OpModeSel_00	Anw. Betriebsart Bit 0	1.0	qCh_OpModePlc	Betriebsart von SPS
0.1	qCh_OpModeSel_01	Anw. Betriebsart Bit 1	1.1	qCh_Restart	Automatischer Restart
0.2	qCh_OpModeSel_02	Anw. Betriebsart Bit 2	1.2	qCh_NCStart	NC-Start
0.3	qCh_OpModeSel_03	Anw. Betriebsart Bit 3	1.3	qCh_TransferLock	Einlesesperre
0.4	-	res.	1.4	qCh_FeedHold	Vorschub-Halt
0.5	-	res.	1.5	qCh_FeedStop	Vorschubsperr
0.6	-	res.	1.6	qCh_MachineFunction	Maschinenfunktion
0.7	-	res.	1.7	qCh_ReSelOff	Autom. Wiederanwahl Aus
2.0	qCh_CtrlReset	Grundstellung	3.0	qCh_ASub1	Asynchr.-Unterpr. 1
2.1	qCh_CancSyntaxError	Syntaxfehler löschen	3.1	qCh_ASub2	Asynchr.-Unterpr. 2
2.2	qCh_CancDist	Restweg löschen	3.2	qCh_ASub3	Asynchr.-Unterpr. 3
2.3	qCh_NextBlk	Umschalten auf nächsten Satz	3.3	qCh_ASub4	Asynchr.-Unterpr. 4
2.4	-	res.	3.4	qCh_ASub5	Asynchr.-Unterpr. 5
2.5	-	res.	3.5	qCh_ASub6	Asynchr.-Unterpr. 6
2.6	qCh_RetCont	Wiederanfahren	3.6	qCh_ASub7	Asynchr.-Unterpr. 7
2.7	qCh_Retract	Schneller Rückzug	3.7	qCh_ASub8	Asynchr.-Unterpr. 8
4.0	qCh_JogPlusWcs	WCS Manuell +	5.0	qCh_Blkslash	Satz überlesen

Kanalbezogene Interface-Signale

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
4.1	qCh_JogMinusWcs	WCS Manuell -	5.1	qCh_OptStop	Wahlweise Halt
4.2	-	res.	5.2	qCh_OptJump	Bedingter Sprung
4.3	-	res.	5.3	-	res.
4.4	qCh_HandwSelWcs_00	Handradauswahl Bit0	5.4	qCh_RedRap	Reduzierter Eilgang
4.5	qCh_HandwSelWcs_01	Handradauswahl Bit1	5.5	qCh_Rapid100	Eilgang 100%
4.6	qCh_HandwDirWcs	Handraddrehrichtung	5.6	-	res.
4.7	qCh_HandwPosMode	Positionshandrad	5.7	qCh_Override100	Override100%
6.0	qCh_Override_00	Override Bit 0	7.0	qCh_Override_08	Override Bit 8
6.1	qCh_Override_01	Override Bit 1	7.1	qCh_Override_09	Override Bit 9
6.2	qCh_Override_02	Override Bit 2	7.2	qCh_Override_10	Override Bit 10
6.3	qCh_Override_03	Override Bit 3	7.3	qCh_Override_11	Override Bit 11
6.4	qCh_Override_04	Override Bit 4	7.4	qCh_Override_12	Override Bit 12
6.5	qCh_Override_05	Override Bit 5	7.5	qCh_Override_13	Override Bit 13
6.6	qCh_Override_06	Override Bit 6	7.6	qCh_Override_14	Override Bit 14
6.7	qCh_Override_07	Override Bit 7	7.7	qCh_Override_15	Override Bit 15
8.0	qCh_Custom1	Kundeneingang 1	9.0	qCh_OnlCorrWcs	Online-Korrektur Freigabe
8.1	qCh_Custom2	Kundeneingang 2	9.1	qCh_OnlCorrWcsDir	Online-Korrektur Richtung
8.2	qCh_Custom3	Kundeneingang 3	9.2	qCh_Retrace	Rückwärtsbetrieb
8.3	qCh_Custom4	Kundeneingang 4	9.3	qCh_RetraceMovFwd	Bahnbewegung vorwärts
8.4	qCh_Custom5	Kundeneingang 5	9.4	qCh_RetraceMovRev	Bahnbewegung rückwärts
8.5	qCh_Custom6	Kundeneingang 6	9.5	-	res.
8.6	qCh_Custom7	Kundeneingang 7	9.6	qCh_CoordCoupleOff	Kopplung beenden
8.7	qCh_Custom8	Kundeneingang 8	9.7	qCh_TangTRotRel	TTL Freigabe
10.0	qCh_RapOverride00	Eilgang Override Bit 0	11.0	qCh_RapOverride08	Eilgang Override Bit 8
10.1	qCh_RapOverride01	Eilgang Override Bit 1	11.1	qCh_RapOverride09	Eilgang Override Bit 9
10.2	qCh_RapOverride02	Eilgang Override Bit 2	11.2	qCh_RapOverride10	Eilgang Override Bit 10
10.3	qCh_RapOverride03	Eilgang Override Bit 3	11.3	qCh_RapOverride11	Eilgang Override Bit 11
10.4	qCh_RapOverride04	Eilgang Override Bit 4	11.4	qCh_RapOverride12	Eilgang Override Bit 12
10.5	qCh_RapOverride05	Eilgang Override Bit 5	11.5	qCh_RapOverride13	Eilgang Override Bit 13
10.6	qCh_RapOverride06	Eilgang Override Bit 6	11.6	qCh_RapOverride14	Eilgang Override Bit 14
10.7	qCh_RapOverride07	Eilgang Override Bit 7	11.7	qCh_RapOverride15	Eilgang Override Bit 15
12.0	-	res.	13.0	qCH_TestFeed	Testvorschub
12.1	-	res.	13.1	qCH_TestRap	Testeilgang

Bit	Symbol.Adr.	SPS-Ausgangssignal	Bit	Symbol.Adr.	SPS-Ausgangssignal
12.2	-	res.	13.2	qCh_Feedrate2	Zweiter Vorschub
12.3	-	res.	13.3	-	res.
12.4	-	res.	13.4	-	res.
12.5	-	res.	13.5	-	res.
12.6	-	res.	13.6	-	res.
12.7	-	res.	13.7	-	res.

Tab. 7-1: Übersicht Ausgangssignale (SPS → NC)

7.1.3 Übersicht Eingangssignale (NC → SPS)

Bit	Symbol.Adr.	SPS-Eingangssignal	Bit	Symbol.Adr.	SPS-Eingangssignal
0.0	iCh_OpMode_00	Anw. Betriebsart Bit 0	1.0	iCh_DryRun	Testbetrieb
0.1	iCh_OpMode_01	Anw. Betriebsart Bit 1	1.1	iCh_NCReady	NC-Bereit
0.2	iCh_OpMode_02	Anw. Betriebsart Bit 2	1.2	iCh_ProgRun	Programm läuft
0.3	iCh_OpMode_03	Anw. Betriebsart Bit 3	1.3	iCh_TransferLockAct	Einlesesperre aktiv
0.4		res.	1.4	iCh_FeedHoldAct	Vorschub-Halt aktiv
0.5		res.	1.5	-	res.
0.6		res.	1.6	iCh_ProgStopM0	Programm Halt M0
0.7		res.	1.7	iCh_ProgStopM30	Programm Ende M30
2.0	iCh_Reset	Kanal grundgestellt	3.0	iCh_ASub1	Asynchr.-Unterpr. 1
2.1	-	res.	3.1	iCh_ASub2	Asynchr.-Unterpr. 2
2.2	iCh_RetpaAct	Wiederaufahren zum Anfahrpunkt aktiv	3.2	iCh_ASub3	Asynchr.-Unterpr. 3
2.3	iCh_RetContPath	Wiederaufahren auf Wegtipp-Pfad	3.3	iCh_ASub4	Asynchr.-Unterpr. 4
2.4	iCh_RemoveFinish	Wegfahren beendet	3.4	iCh_ASub5	Asynchr.-Unterpr. 5
2.5	iCh_ReadyReEnter	Bereit zum Wiederaufahren	3.5	iCh_ASub6	Asynchr.-Unterpr. 6
2.6	iCh_ReEnterAct	Wiedereintritt aktiv	3.6	iCh_ASub7	Asynchr.-Unterpr. 7
2.7	-	res.	3.7	iCh_ASub8	Asynchr.-Unterpr. 8
4.0	iCh_State_00	Kanalzustand Bit 0	5.0	iCh_Blklash	Satz überlesen aktivieren
4.1	iCh_State_01	Kanalzustand Bit 1	5.1	iCh_OptStop	Wahlweiser Halt aktivieren
4.2	iCh_State_02	Kanalzustand Bit 2	5.2	-	res.
4.3	iCh_State_03	Kanalzustand Bit 3	5.3	iCh_SRunAct	Satzvorlauf aktiv
4.4	iCh_State_04	Kanalzustand Bit 4	5.4	iCh_SRunReEnter	Wiedereintritt aktiv
4.5	-	res.	5.5	iCh_SRunRepos	Wiederaufahren aktiv
4.6	-	res.	5.6	iCh_Override0	Override 0%
4.7	-	res.	5.7	iCh_Override100	Override 100%
6.0	iCh_Cpl01	CPL-Kundenausgang 1	7.0	iCh_Cpl09	CPL-Kundenausgang 9

Kanalbezogene Interface-Signale

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
6.1	iCh_Cpl02	CPL-Kundenausgang 2	7.1	iCh_Cpl10	CPL-Kundenausgang 10
6.2	iCh_Cpl03	CPL-Kundenausgang 3	7.2	iCh_Cpl11	CPL-Kundenausgang 11
6.3	iCh_Cpl04	CPL-Kundenausgang 4	7.3	iCh_Cpl12	CPL-Kundenausgang 12
6.4	iCh_Cpl05	CPL-Kundenausgang 5	7.4	iCh_Cpl13	CPL-Kundenausgang 13
6.5	iCh_Cpl06	CPL-Kundenausgang 6	7.5	iCh_Cpl14	CPL-Kundenausgang 14
6.6	iCh_Cpl07	CPL-Kundenausgang 7	7.6	iCh_Cpl15	CPL-Kundenausgang 15
6.7	iCh_Cpl08	CPL-Kundenausgang 8	7.7	iCh_Cpl16	CPL-Kundenausgang 16
8.0	iCh_Custom1	Kundenausgang 1	9.0	iCh_G0Act	Eilgang aktiv
8.1	iCh_Custom2	Kundenausgang 2	9.1	iCh_InPosAct	Inpos-Bereich 2 aktiv
8.2	iCh_Custom3	Kundenausgang 3	9.2	iCh_G41G141Act	G41/G141 aktiv
8.3	iCh_Custom4	Kundenausgang 4	9.3	iCh_G42G142Act	G42/G142 aktiv
8.4	iCh_Custom5	Kundenausgang 5	9.4	iCh_RetraceAct	Rückwärtsbetrieb aktiv
8.5	iCh_Custom6	Kundenausgang 6	9.5	-	res.
8.6	iCh_Custom7	Kundenausgang 7	9.6	iCh_CoordCoupleAct	Koord. Kopplung aktiv
8.7	iCh_Custom8	Kundenausgang 8	9.7	iCh_TangTRotCmd	Werkzeugdrehung (TangTool)
10.0	iCh_ActFunc01	G70 aktiv	11.0	iCh_ActFunc09	Werkzeugkorrektur aktiv Bit 0
10.1	iCh_ActFunc02	Vorschub 100% aktiv	11.1	iCh_ActFunc10	Werkzeugkorrektur aktiv Bit 1
10.2	iCh_ActFunc03	G95 aktiv	11.2	iCh_ActFunc11	Werkzeugkorrektur aktiv Bit 2
10.3	iCh_ActFunc04	Achstrafo 2 inaktiv	11.3	iCh_ActFunc12	Werkzeugkorrektur aktiv Bit 3
10.4	iCh_ActFunc05	Programmposition aktiv	11.4	iCh_ActFunc13	Werkzeugkorrektur aktiv Bit 4
10.5	iCh_ActFunc06	Gewindezyklus aktiv	11.5	iCh_ActFunc14	nicht belegt
10.6	iCh_ActFunc07	Gewindebohren aktiv	11.6	iCh_ActFunc15	nicht belegt
10.7	iCh_ActFunc08	Gewindeschneiden aktiv	11.7	iCh_ActFunc16	nicht belegt
12.0	iCh_ActFunc17	nicht belegt	13.0	iCh_TestFeed	Testvorschub aktiv
12.1	iCh_ActFunc18	G96 aktiv	13.1	iCh_TestRap	Testeilgang aktiv
12.2	iCh_ActFunc19	nicht belegt	13.2	iCh_RapOverride0	Eilgang Override 0%
12.3	iCh_ActFunc20	nicht belegt	13.3	iCh_RapOverride100	Eilgang Override 100%
12.4	iCh_ActFunc21	nicht belegt	13.4	iCh_Feedrate2	Zweiter Vorschub

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
12.5	iCh_ActFunc22	nicht belegt	13.5	-	res.
12.6	iCh_ActFunc23	nicht belegt	13.6	-	res.
12.7	iCh_ActFunc24	nicht belegt	13.7	-	res.

Tab. 7-2: Übersicht Eingangssignale (NC → SPS)



Bitte beachten Sie, dass die voreingestellte Belegung der NC-Funktionsspezifischen Bit-Interface-Signale iCh_ActFunc<nn> durch den Maschinenhersteller abweichend konfiguriert sein kann (Konfigurationsparameter: `/PLC/NcFuncBitIf/NcFunc [1] "iChActFunc [1..24 ] //NC-Funktionszuordnung \n (max. 8 Zeichen)" (3020 00001)` bzw. `/PLC/NcFuncBitIf/NcFunc [1] "iChActFunc [1..24 ] //NC-Funktionszuordnung \n (Ergänzung max. 8 Zeichen)" (3020 00002)`).



Das Signal iCh_ActFunc02 war in V01 mit "Vorschub 100%" / "G95 aktiv" doppelt belegt. Diese Doppelbelegung wurde aufgelöst.

Ab V02 gelten in der Voreinstellung folgende Vereinbarungen:

- iCh_ActFunc02 Vorschub 100%
- iCh_ActFunc03 G95 aktiv

G196 wird nicht mehr in iCh_ActFunc18 angezeigt und muss bei Bedarf extra konfiguriert werden.

7.2 Signalbeschreibung

7.2.1 Kanalbezogene Ausgangssignale (SPS → NC)

Anwahl Betriebsart qCh_OpModeSel_00 ... 03

Die Betriebsart kann entweder mit der SPS oder über die NC-Bedienoberfläche vorgegeben werden (siehe qCh_OpModePlc, [Kap. "Betriebsart von SPS qCh_OpModePlc" auf Seite 42](#)).

Es sind folgende Betriebsarten möglich:

Betriebsart	Anwahl Betriebsart			
	Bit 3 (qCh_OpModeSel_03)	Bit 2 (qCh_OpModeSel_02)	Bit 1 (qCh_OpModeSel_01)	Bit 0 (qCh_OpModeSel_00)
Keine Betriebsart	0	0	0	0
Einrichten manuell	0	0	0	1
Einrichten Referenzpunkt anfahren	0	0	1	0
reserviert	0	0	1	1
Handeingabe NC-Satz	0	1	0	0
Folgesatz	0	1	0	1
Programmsatz	0	1	1	0
Einzelschritt	0	1	1	1

Kanalbezogene Interface-Signale

Betriebsart	Anwahl Betriebsart			
	Bit 3 (qCh_Sel_03) OpMode-	Bit 2 (qCh_Sel_02) OpMode-	Bit 1 (qCh_Sel_01) OpMode-	Bit 0 (qCh_Sel_00) OpMode-
Teachen	1	0	0	0
reserviert	1	0	0	1
Einzelsatz	1	0	1	0
Wiederanfahren	1	0	1	1
CPL-Debugger-Programmsatz	1	1	0	0
CPL-Debugger-Folgesatz	1	1	0	1
Einrichten manuell Werkstückkoordinaten	1	1	1	0
reserviert	1	1	1	1

Tab. 7-3: Anwahl Betriebsart

Ein Betriebsartenwechsel ist unter folgenden Bedingungen möglich:

Betriebsart		Betriebsartenwechsel, Kanal aktiv		Betriebsartenwechsel, Kanal nicht aktiv (iCh_Reset = 1, Kanal- grundgestellt)
Folgesatz	Automatikbetriebsart Gruppe 1	ja, innerhalb Gruppe 1	ja, innerhalb Gruppe 1 und 2: Ist der Kanalzustand für Programmstart be- reit, d. h., am Pro- grammanfang oder - ende	ja
Einzelschritt				ja
Einzelsatz				ja
Programmsatz	ja, innerhalb Gruppe 2	ja		
CPL-Debugger-Pro- grammsatz		ja		
CPL-Debugger-Folge- satz		ja		
Handeingabe NC-Satz		ja, löst vor dem Wechsel Grundstellung des Ka- nals aus	ja	
Wiederanfahren			ja	
Einrichten manuell			ja	
Einrichten Referenzpunkt anfahren			ja	
Einrichten manuell Werkstückkoordinaten			ja	
Teachen			ja	

Tab. 7-4: Betriebsartenwechsel

Betriebsart von SPS qCh_OpModePlc

Die Betriebsarten können manuell über die NC-Bedienoberfläche oder über das SPS-Ablaufprogramm angewählt werden.

Signalpegel 1: Die Betriebsart wird von der SPS über qCh_OpModeSel_00...03 vorgegeben.

- Solange der Signalpegel = 1 gesetzt ist, kann über die NC-Bedienoberfläche keine Betriebsart vorgewählt werden.
- 0:** Die Betriebsart wird über die **NC-Bedienoberfläche** vorgegeben.
- Solange der Signalpegel = 0 gesetzt ist, kann die Betriebsart nicht über die SPS vorgegeben werden.

Automatischer Restart qCh_Restart

- Signalpegel** **1:** Nach Programmende wird ein Programm automatisch wieder gestartet.
- 0:** Keine automatische Anwahl.

NC-Start qCh_NCStart

Wirkt in den Betriebsarten **Folgesatz**, **Einzelsatz**, **Einzelschritt**, **Programmsatz** und **Handeingabe NC-Satz**.

In Verbindung mit NC-Start sind die Eingangssignale NC-Bereit, Vorschub-Halt aktiv sowie Programm läuft von Bedeutung.

- Signalflanke** **0 → 1:** Startet das Abarbeiten eines Teileprogramms (Folgesatz) oder eines NC-Satzes. **Vorschub Halt aktiv** wird zurückgesetzt und die Achsbewegung wieder eingeleitet.
- 1 → 0:** Keine Auswirkung.

Einlesesperre qCh_TransferLock

Wirkt in den Betriebsarten **Folgesatz**, **Einzelsatz** und **Handeingabe NC-Satz**.

- Signalpegel** **1:** Das Abarbeiten des nächsten NC-Satzes wird verhindert, der aktive Satz wird jedoch noch ausgeführt. Nach Beenden des aktiven Satzes wird das Signal **Einlesesperre aktiv** ausgegeben.
- Solange Einlesesperre gesetzt ist, kann mit NC-Start das Teileprogramm bzw. der eingegebene NC-Satz nicht ausgeführt werden.
- Die Satzaufbereitung wird durch Einlesesperre nicht angehalten, sondern unabhängig davon solange fortgesetzt, bis der für die Satzaufbereitung konfigurierte Speicherplatz belegt ist.
- 0:** Das mit der Einlesesperre unterbrochene Abarbeiten eines NC-Teileprogramms wird wieder fortgeführt.

Vorschub Halt qCh_FeedHold

Wirkt auf **alle Bearbeitungsachsen** eines Kanals.

Vorschub Halt hat keinen Einfluss auf NC-Sätze, in denen keine Bewegung programmiert ist, d. h., die Ausführung des NC-Programms wird solange fortgeführt, bis eine Verfahrbewegung einer Bearbeitungsachse ansteht.

Vorschub Halt wirkt selbsthaltend. Solange Vorschub Halt wirkt, ist das Eingangssignal **Vorschub Halt aktiv** gesetzt.

Durch das kanalbezogene Interface-Signal **NC-Start** wird die Selbsthaltung aufgehoben.

- Signalpegel** **1:** Die Achsbewegungen werden mit der negativen Bahnbeschleunigung gestoppt, es können vor dem Aufheben der Selbsthaltung keine Verfahrbewegungen gestartet werden.

Bei Gewindeschneiden (G33) hat Vorschub Halt **keine Wirkung**. Die Vorschubgeschwindigkeit ist in diesem Fall von der Spindeldrehzahl und der Gewindesteigung abhängig.

- 0: Ist Vorschub Halt zurückgesetzt ist, kann die Selbsthaltung von Vorschub Halt mit **NC-Start** wieder aufgehoben und eine Bewegung gestartet werden.

Vorschubsperr qCh_FeedStop

Wirkt in der Betriebsart **Folgesatz, Einzelsatz, Einzelschritt** und **Handeingabe NC-Satz** auf alle Bearbeitungsachsen eines Kanals gleichzeitig.

Für die Hilfsachsen hat dieses Signal keine Auswirkung, da die Hilfsachsen keinem speziellen Kanal zugeordnet sind.

Das **Achs-Interface-Signal Vorschubsperr** wirkt jeweils übergeordnet gegenüber diesem kanalbezogenen Signal.

- Signalpegel** 1: Solange das Signal gesetzt ist, verfahren die Achsen nicht. Wird das Signal gesetzt, wenn die Achsen in Bewegung sind, stoppen alle Achsen des Kanals mit der negativen Bahnbeschleunigung.
- Bei Gewindeschneiden (G33) hat Vorschubsperr **keine Wirkung**. Die Vorschubgeschwindigkeit ist von der Spindeldrehzahl und der Gewindesteigung abhängig.
- 0: Ist Vorschubsperr zurückgesetzt, werden die Achsen wieder in Bewegung gesetzt. Sie beschleunigen auf den aktuellen Bahnvorschub.

Maschinenfunktion qCh_MachineFunction

Funktion noch nicht realisiert.

Automatische Wiederanwahl aus qCh_ReSelOff

Dieses Signal beeinflusst die automatische Wiederanwahl eines Teileprogramms.

- Signalpegel** 1: Die automatische Wiederanwahl von Programmen wird unterdrückt.
- 0: Automatische Wiederanwahl erlaubt.

Grundstellung qCh_CtrlReset

Dieses Signal wirkt auf die, in **einem** Kanal interpolierenden Achsen sowie auf das im Kanal ablaufende Teileprogramm. Es hat die gleiche Wirkung wie die Funktion Grundstellung, die über die NC-Bedienoberfläche ausgelöst werden kann.

In Verbindung mit Grundstellung sind die kanalbezogenen Signale **Kanal grundgestellt, NC-Bereit** und **Programm läuft** von Bedeutung. Weiterhin hat Grundstellung Auswirkungen auf die achsbezogenen Signale **Achse in Position, Achse läuft** und **Fahrbehl**.

- Signalpegel** 1: Grundstellung bewirkt u. a. die Abwahl des NC-Teilprogramms und das Rücksetzen von NC-Funktionen auf den im Maschinenparameter **/CHAN/Ch[1]/Ini/ChResetState** "Einschaltzustand nach Grundstellung" (7060 00020) definierten Einschaltzustand. Weiterhin werden die Bearbeitungsachsen bis zum Stillstand gebremst.
- Ist die Funktion Grundstellung steuerungsintern beendet, setzt die NC das Signal **Grundstellung ausgeführt**. Daraufhin setzt die SPS den Signalpegel auf 0 zurück.

Werden die Signalfanken aller Kanäle gleichzeitig von 0 auf 1 gesetzt, wird eine Gesamtgrundstellung durchgeführt. Dabei werden neben den Einzelgrundstellungen auch die Hilfsachsen und Spindeln bis zum Stillstand gebremst und kanalübergreifende Maschinenparameter übernommen.

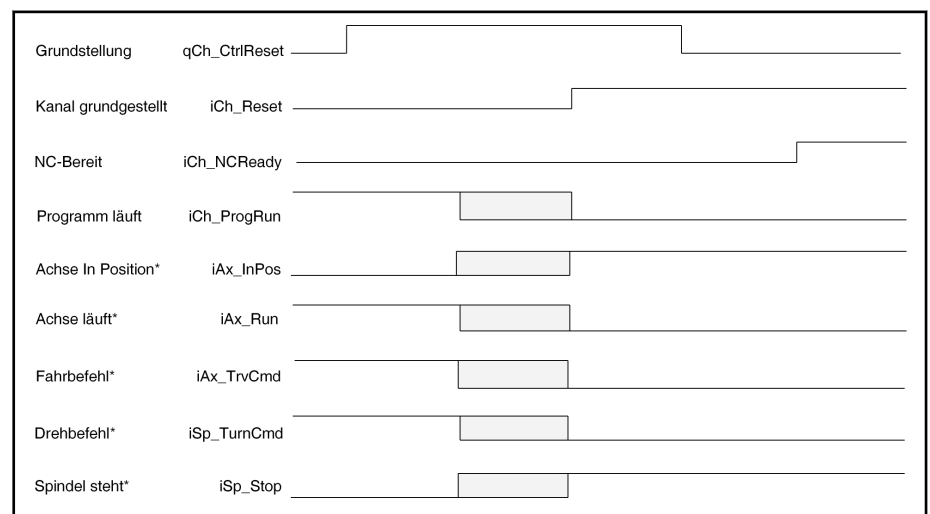
Bei Gewindeschneiden (G33) löst **Grundstellung** einen schnellen Rückzug aus (siehe auch [Kap. "Schneller Rückzug qCh_Retract" auf Seite 47](#)).

0: Keine Auswirkung.

Beispiel:

Grundstellung

Um die Bearbeitung eines Teileprogramms abzubrechen, wird am kanalbezogenen Interface das Signal **Grundstellung** gesetzt.



*) für Hilfsachsen und Spindeln nur bei Gesamtgrundstellung

Abb. 7-1: Grundstellung



Das Signal "NC-Bereit" wird erst gesetzt, nachdem ein NC-Teilprogramm ausgewählt wurde.

Syntaxfehler löschen qCh_CancSyntaxError

Funktion noch nicht realisiert.

Restweg löschen qCh_CancDist

Wirkt in den Betriebsarten **Folgesatz**, **Einzelatz**, **Einzelstschritt**, **Programmsatz** bei aktivem Programm und bei **Handeingabe NC-Satz**.

Zunächst wird das Programm mit Vorschub Halt (qCh_FeedHold) angehalten (dies ist nicht zwingend notwendig). In der Anzeige ist der Restweg zu sehen, den die Achsen bis zum Satzende noch zu verfahren haben.

Nach Auslösen von "Restweg löschen" werden alle vorbereiteten NC-Sätze einschließlich des Restes des aktuellen Satzes verworfen und wieder neu aufbereitet.



Zu beachten ist:

- CPL-Sätze oder CPL-Teile werden nicht berücksichtigt.
Beispiel: Die CPL-Variable POS hatte zur Vorbereitung den Wert 10: dann wird nach "Restweg löschen" das NC-Wort X[POS] als X10 interpretiert, obwohl zu diesem Zeitpunkt POS vielleicht einen ganz anderen Wert hat.
- Eventuell sich ändernde Korrekturwerte werden berücksichtigt.

In der Anzeige wird der Endpunkt auf die aktuelle Position gesetzt und gleichzeitig der angezeigte Restweg gelöscht. Der Kanal geht anschließend in den Zustand NC-Bereit (iCh_NCReady).

Nach NC-Start (iCh_NCStart) wird das Programm **an der Unterbrechungsstelle** fortgesetzt, wobei neue Korrekturwerte berücksichtigt werden.

Signalflanke	0 → 1:	Löst bei aktivem Programm Restweg löschen aus.
	1 → 0:	Keine Auswirkung.



Dieses Signal darf nicht bei aktivem Gewindeschneiden gesetzt werden.

Umschalten auf nächsten Satz qCh_NextBlk

Dieses Signal wirkt bei aktivem Teileprogramm in den Betriebsarten **Folgesatz**, **Einzelsatz**, **Einzelschritt** und **Programmsatz**.

Signalflanke	0 → 1:	Die synchrone Achsbewegung des Kanals wird mit der zulässigen Bahnbeschleunigung gestoppt. Anschließend werden alle vorbereiteten NC-Sätze verworfen. Der Kanal geht in den Zustand NC-Bereit (iCh_NCReady). Nach NC-Start (qCh_NCStart) werden die nachfolgenden Sätze wieder neu aufbereitet und ausgeführt. Das noch nicht abgearbeitete Teilstück des unterbrochenen NC-Satzes wird dabei ignoriert. Um die Funktion gezielt auslösen zu können, muss das Programm zuvor mit Vorschub Halt an definierter Stelle angehalten werden.
	1 → 0:	Keine Auswirkung.



Diese Funktion funktioniert im Allgemeinen nur dann, wenn auf den Abbruchsatz ein Linearsatz folgt (durch die veränderte Startposition ergibt sich beispielsweise mit den programmierten Parametern kein konsistenter Kreis mehr).

Bei aktiver Fräserbahnkorrektur führt die Funktion meistens zu unerwünschten Verfahrbewegungen und damit zur Zerstörung des zu bearbeitenden Teils.

Wiederanfahren qCh_RetCont

Signalpegel/-flanke	1:	Nach NC-Start wird auf der gespeicherten Wegfahrbewegung an die Kontur zurückgefahren.
	0:	Nach NC-Start wird die Bearbeitung direkt fortgesetzt.
	0 → 1:	Die Aufzeichnung der Wegfahrbewegung wird beendet.

Schneller Rückzug qCh_Retract

Löst bei aktivem Gewindeschneiden (G33) die Rückzugsbewegung aus, wenn die Schneidbewegung gestartet ist.

Voraussetzung:

Die Rückzugsbewegung ist über **/THRD/Ch[1]/Retr/DrIndRetrAx**, **/THRD/Ch[1]/Retr/DrIndSeconAx** "Gewindeschneiden: Schneller Rückzug, Systemachsummern" (7050 00645) - **/THRD/Ch[1]/Retr/RetrVectRetrAx**, **/THRD/Ch[1]/Retr/RetrVectSeconAx** "Gewindeschneiden: Schneller Rückzug, Weg und Richtung" (7050 00650) oder mit der Funktion ThreadSet(RD(,)) konfiguriert und über **/THRD/Ch[1]/Retr/EnablRetrThrd** "Gewindeschneiden: Schneller Rückzug, Aktivieren" (7050 00640) oder mit der Funktion ThreadSet(ROn1) freigeschaltet.

Signalpegel	1:	Der Rückzug von der Kontur wird eingeleitet. Der "Nebenschneidachse" wird eine lineare Rückzugsbewegung überlagert, die das Schneidwerkzeug vom bearbeiteten Teil wegführt. <i>Nebenschneidachse:</i> • Nebenachse der Ebene bei Längsgewinde oder • Hauptachse der Ebene bei Plangewinde
	0:	Keine Auswirkung.

Asynchrones Unterprogramm 1 ... 8 qCh_ASub1 ... 8

Jedes dieser Signale löst ein von maximal 8 asynchronen Unterprogrammen aus. Dabei wird die Bearbeitung eines NC-Teilprogramms unterbrochen und das asynchrone Unterprogramm sofort gestartet. Nach Beendigung des asynchronen Unterprogramms kann die Bearbeitung des Teilprogramms fortgesetzt werden.

WCS Manuell + qCh_JogPlusWcs und WCS Manuell - qCh_JogMinusWcs

Diese Signale wirken in der Betriebsart **Einrichten manuell Werkstückkoordinaten**. Eine ausführliche Beschreibung dieser Betriebsart finden Sie im Handbuch "Funktionsbeschreibung".

Signalflanke	0 → 1:	Ist in der Betriebsart "Einrichten manuell Werkstückkoordinaten" eine Koordinate selektiert, wird mit Manuell + oder Manuell - die Bewegung der Koordinate gestartet.
	1 → 0:	Die kontinuierliche Bewegung Manuell + oder Manuell - wird gestoppt.

Handradauswahl Bit 0 qCh_HandwSelWcs_00 und Handradauswahl Bit 1 qCh_HandwSelWcs_01

Diese Signale wirken in der Betriebsart **Einrichten manuell Werkstückkoordinaten**. Eine ausführliche Beschreibung dieser Betriebsart finden sie im Handbuch "Funktionsbeschreibung".

An die NC-Steuerung können zwei Handräder angeschlossen werden. Sobald eines der beiden Signale gesetzt ist, kann die angewählte Koordinate über das ausgewählte Handrad verfahren werden. Die Signale WCS Manuell + und WCS Manuell - sind dann ohne Wirkung.

Signalflanke	0 → 1:	Ist die Betriebsart angewählt und eine Koordinate selektiert, wird mit Handradauswahl Bit0 oder Handradauswahl Bit1 der Handradbetrieb aktiviert.

1 → 0: Der Handradbetrieb wird abgewählt. Eine eventuell noch stattfindende Bewegung wird abgebrochen.

Handraddrehrichtung qCh_HandwDirWcs

Mit diesem Signal ist es möglich, die Bewegungsrichtung der Koordinate unabhängig von der Drehrichtung des Handrads anzugeben.

Signalpegel **1:** Die Bewegungsrichtung der Koordinate wird umgekehrt.
0: Keine Auswirkung.

Positionshandrad qCh_HandwPosMode

Mit diesem Signal ist es möglich, das Verhalten des Handradbetriebs beim Überschreiten der aktuell zulässigen Maximalgeschwindigkeit vorzugeben.



Der Pegel dieses Signals wird beim Setzen von qCh_HandwSelWcs_0x von der NC übernommen.

Signalpegel **1:** Der durch Handradbewegungen vorgegebene Weg wird auch bei einer Überschreitung der Maximalgeschwindigkeit abgearbeitet.
Achtung: Auch nach Beenden der Handradbewegung können noch Achsbewegungen stattfinden.
0: Bei einer Überschreitung der Maximalgeschwindigkeit wird der Fahrweg entsprechend reduziert.

Satz überlesen qCh_BlkSlash

Wirkt in den Betriebsarten **Folgesatz**, **Programmsatz**, **Einzelsatz** und **Einzelschritt**.

Signalpegel **1:** Zum Zeitpunkt der **Satzaufbereitung** werden die im NC-Teilprogramm mit einem Schrägstrich "/" gekennzeichneten NC-Sätze beim Abarbeiten überlesen.
 Die Satzaufbereitung kann gegenüber dem aktiven Satz mehrere NC-Sätze voraus sein. Um sicherzustellen, dass das Überlesen in jedem Fall ausgeführt wird, ist es abhängig von der Anwendung notwendig, die Satzaufbereitung an geeigneter Stelle (vor dem NC-Satz mit dem Schrägstrich /) zu unterbrechen. Die Unterbrechung der Satzaufbereitung wird im Teilprogramm durch den Befehl **WAIT** herbeigeführt.
0: Der mit Schrägstrich "/" gekennzeichnete NC-Satz wird nicht überlesen.

Wahlweiser Halt qCh_OptStop

Wirkt in Verbindung mit der Funktion **M1**, die im NC-Teilprogramm programmiert wird.

Signalpegel **1:** Das Abarbeiten des NC-Teilprogramms wird nach dem NC-Satz angehalten, in dem die Funktion **M1** programmiert ist. Mit **NC-Start** kann es wieder gestartet werden.
0: Die Funktion **M1** ist nicht wirksam, das NC-Teilprogramm wird nicht angehalten.

Bedingter Sprung qCh_OptJump

Wirkt in Verbindung mit der Funktion **GoCond (GOC)**, die im NC-Teilprogramm programmiert wird.

Signalpegel	1:	Im NC-Teileprogramm wird ein Sprung zu dem NC-Satz ausgeführt, der als Sprungziel definiert ist. Dazu muss bereits zum Zeitpunkt der Satzaufbereitung der Signalpegel "Logisch 1" erkannt werden. Die Satzaufbereitung kann mehrere NC-Sätze gegenüber dem aktiven Satz voraus sein. Um sicherzustellen, dass der Sprung in jedem Fall ausgeführt wird, ist es in Abhängigkeit der Anwendung notwendig, die Satzaufbereitung an geeigneter Stelle (vor dem NC-Satz mit der Funktion GoCond) zu unterbrechen. Die Unterbrechung wird im Teileprogramm durch den Befehl WAIT herbeigeführt.
	0:	Der Sprung im Teileprogramm wird nicht ausgeführt.

Reduzierter Eilgang qCh_RedRap

Der G0-Eilgang wird in den Betriebsarten **Handeingabe NC-Satz, Folgesatz, Programmsatz, Einzelschritt** durch Setzen dieses Signals auf einen in Maschinenparameter **/CHAN/Ch[1]/Path/Vel/RedChVel** "Wert für beschränkte Eilgangsgeschwindigkeit" (7030 00110) festgelegten Maximalwert beschränkt.

Wird die Reduzierung erst nach Programmstart geschaltet, wirkt sie erst, nachdem alle in diesem Zeitpunkt vorbereiteten Sätze abgearbeitet sind.

Ist dieses Verhalten nicht erwünscht, muss die Bearbeitung mit Vorschub Halt angehalten, die vorbereiteten Sätze durch Restweg löschen verworfen und die Bearbeitung durch anschließenden NC-Start fortgesetzt werden.

Ein Kanal-Override ist auch bei reduziertem Eilgang wirksam, d. h., 100% Override entsprechen exakt der reduzierten Geschwindigkeit.

Die Funktion wirkt auf alle NC-Funktionen, die mit Eilgang verfahren.

Die Spindeldrehzahl und der programmierte Bahnvorschub bleiben von der Funktion Reduzierter Eilgang unberührt.

Die Funktion hat nur Einfluss auf die, dem Kanal zugeordneten synchronen Achsen.

Signalpegel	1:	Alle Eilgangssätze werden mit reduzierter Eilgangsgeschwindigkeit vorbereitet.
	0:	Reduzierter Eilgang ist nicht aktiv, alle Eilgangssätze werden mit maximal möglicher Geschwindigkeit vorbereitet.

Eilgang 100% qCh_Rapid100

Dieses Signal wirkt nur in Verbindung mit der Funktion **Eilgang Override**.

Wirkt auf alle im Kanal definierten Achsen.

Signalpegel	1:	Unabhängig von der tatsächlichen Stellung des Eilgang Override verfahren Eilgangbewegungen mit der maximal möglichen Geschwindigkeit (sofern die Funktion Eilgang Override im Kanal appliziert wurde).
	0:	Es wirkt der Eilgang Override (sofern die Funktion Eilgang Override im Kanal appliziert wurde).

Override 100% qCh_Override100

Wirkt auf alle im Kanal definierten Achsen.

Signalpegel	1:	Setzt die Override-Funktion außer Kraft, die Achsen fahren mit der programmierten bzw. vorgewählten Geschwindigkeit.

0: Die **Override-Funktion** wirkt.

Override Bit 0 ... Bit 15 qCh_Override_00 ... 15

Die Verfahrensgeschwindigkeiten aller Bearbeitungsachsen eines Kanals können über die **Override-Funktion** beeinflusst werden. Die Override-Funktion wirkt dabei auf die im NC-Teilprogramm programmierten Geschwindigkeiten und auf die vorwählbaren Geschwindigkeiten im Einrichtbetrieb.

Über qCh_Override_00 ... 15 ist eine stufenlose 16 Bit-Vorgabe möglich. Das CNC-System MTX interpretiert den Wert direkt als Override-Wert in 0,01%.



Kanal 0 bestimmt die Werte für die Hilfsachsen, bei denen keine direkte Vorgabe möglich ist.

Kundeneingang 1 ... 8 qCh_Custom1 ... 8

NC-seitig konfigurierbares Interface, das je nach Bedarf vom Anwender (OEM) belegt werden kann.

Anwendungsbeispiele:

- Für OEM-spezifische Firmware
- Melden einer aktiven Online-Korrektur im MCS (Maschinenparameter ONL/McsOnlCorr/Ch[k]/Base/ChIfSigMcsCorr)

Online-Korrektur Freigabe qCh_OnlCorrWcs

Dieses Signal gibt für eine selektierte Koordinate die NC-Funktion "Online-Korrektur in Werkstückkoordinaten" frei.

Bei der Online-Korrektur in Werkstückkoordinaten werden mittels eines Handrades Position und Orientierung im Werkstückkoordinatensystem (WCS) eines Kanals korrigiert. Diese Korrektur ist bei aktivem und inaktivem Teilprogramm möglich. Eine ausführliche Beschreibung finden Sie im Handbuch "Funktionsbeschreibung".



In den Betriebsarten "Einrichten manuell", "Einrichten manuell Werkstückkoordinaten" und "Einrichten Referenzpunkt anfahren" kann die Online-Korrektur nicht freigegeben werden.

Signalpegel	1:	Die angewählte Online-Korrektur verarbeitet die Handraddaten.
	0	Die angewählte Online-Korrektur reagiert nicht auf die Handraddaten.

Online-Korrektur Richtung qCh_OnlCorrWcsDir

Mit diesem Signal ist es möglich, die Korrekturrichtung umzudrehen.

Signalpegel	1:	Positive Drehrichtung des Handrades führt zu negativer Online-Korrektur.
	0:	Positive Drehrichtung des Handrades führt zu positiver Online-Korrektur.

Rückwärtsbetrieb qCh_Retrace

Schaltet den Rückwärtsbetrieb ein.

Der Anwender kann rückwärts oder vorwärts auf aufgezeichneter Kontur fahren.

Signalpegel	1:	Beim Flankenwechsel 0 -> 1 wird das laufende Programm unterbrochen (internes Vorschub Halt), es erfolgt keine wei-

tere Bewegung aus dem Teileprogramm. Bahnbewegungen rückwärts oder vorwärts sind möglich.

- 0: Beim Flankenwechsel 1 -> 0 wird der Rückwärtsbetrieb abgeschaltet. Die Bearbeitung kann mit NC-Start fortgesetzt werden.

Bahnbewegung vorwärts qCh_RetraceMovFwd

Bewegung im Retrace-Speicher vorwärts bis max. zum Anfang der im Retrace-Speicher abgelegten Sätze.

Bedingung:

- Rückwärtsbetrieb eingeschaltet (qCh_Retrace=1)
- Bahnbewegung rückwärts abgeschaltet (qCh_RetraceMovRev=0)
- Aufgezeichnete Sätze im Retrace-Speicher vorhanden

- Signalpegel 1: Bewegung vorwärts
0: Bewegung wird angehalten

Bahnbewegung rückwärts qCh_RetraceMovRev

Bewegung im Retrace-Speicher rückwärts bis max. zum Ende der im Retrace-Speicher abgelegten Sätze.

Bedingung:

- Rückwärtsbetrieb eingeschaltet (qCh_Retrace=1)
- Bahnbewegung vorwärts abgeschaltet (qCh_RetraceMovFwd=0)
- Aufgezeichnete Sätze im Retrace-Speicher vorhanden

- Signalpegel 1: Bewegung rückwärts
0: Bewegung wird angehalten

Kopplung beenden qCh_CoordCoupleOff

Mit diesem Signal wird die Koordinatenkopplung für einen Slave-Kanal beendet.

Da sich der Master-Kanal beim Ankoppeln des Slaves im Stillstand befinden muss, wird empfohlen, den Master während der Ankopplung mit **Einlesesperre** und **Vorschubsperr** anzuhalten. Beide Signale können mit Ausgabe des Signals Koordinatenkopplung aktiv durch den Slave-Kanal wieder zurückgesetzt werden.

Beim Beenden der Koordinatenkopplung sollten sich die Achsen des Master-Kanals im Stillstand befinden.

- Signalflanke 0 --> 1: Die Koordinatenkopplung wird für einen Slave-Kanal beendet. Der Slave-Kanal wechselt dann auf den nächsten Satz, das Signal **Koordinatenkopplung aktiv** wird auf logisch 0 gesetzt.
- 0: Keine Auswirkung.

TTL Freigabe qCh_TangTRotRel

Mit diesem Signal kann die Steuerung einen Zwischensatz ausführen, falls der Konturknickwinkel bei der Werkzeugdrehung zwischen zwei Sätzen zu groß wurde (siehe auch [Kap. "Werkzeugdrehung \(TangTool \(TTL\)\) iCh_TangTRotCmd" auf Seite 62](#)).

- Signalpegel 1: Freigabe für die Steuerung, den Zwischensatz auszuführen.
0: Freigabe für das Ausführen eines Zwischensatzes beendet. Die Steuerung kann weitere Sätze ausführen.

Eilgang Override Bit 0 ... Bit 15 qCh_RapOverride_00 ... 15

Die Verfahrensgeschwindigkeiten aller Bearbeitungsachsen eines Kanals im Eilgang können über die **Eilgang Override-Funktion** beeinflusst werden. Die Eilgang Override-Funktion wirkt dabei auf die im NC-Teilprogramm programmierten Geschwindigkeiten und auf die vorwählbaren Geschwindigkeiten im Einrichtbetrieb.

Über qCh_RapOverride_00 ... 15 ist eine stufenlose 16 Bit-Vorgabe möglich. Das CNC-System MTX interpretiert den Wert direkt als Override-Wert in 0,01%.

Das Signal wird nur dann ausgewertet, wenn der Maschinenparameter / **CHAN/Ch[]/Path/RapTravOvrd/EnableRapTravIOvrd** "Aktiviere Eilgang Override" (7030 00014) auf "Ja" sitzt.

Testvorschub qCh_TestFeed

Mit diesem Signal ignoriert die Steuerung die programmierten Geschwindigkeitsangaben und fährt statt dessen mit dem, in dem Systemdatum /SysCh-TestMode/TestFeedrate abgelegten Geschwindigkeitswert.

Die Funktion Testvorschub kann per NCS-Funktion oder über das Bit-Interface des jeweiligen Kanals aktiviert werden.

Die Umschaltung erfolgt unmittelbar, da die Auswertung der Signale direkt innerhalb der Interpolatorlogik erfolgt und den Sollvorschub beeinflusst.



Es ist jedoch zu beachten, dass eine gewisse Zeit vergeht, bis der Sollvorschub erreicht wird, da die Brems- oder Beschleunigungsvorgänge mit der parametrisierten bzw. programmierten Beschleunigung erfolgen.

Eine Überhöhung des programmierten Vorschubs mittels Testvorschubs ist nur möglich, wenn bereits zum Zeitpunkt der Satz Vorbereitung das Systemdatum /SysChTestMode/Enable gesetzt war.

Auch bei aktivem Testvorschub werden die parametrisierten dynamischen Grenzwerte eingehalten.

Die NC meldet den jeweils aktiven Zustand am NC-Ausgangsinterface zurück.

Die Funktionalität Testvorschub wirkt unter G93 (Zeitprogrammierung), G94 (direkte Vorschubprogrammierung) und G95 (Vorschub/Umdrehung). Das Verhalten ist jedoch unabhängig vom Zustand G93/G94/G95. Der Testvorschub wird bei aktiver Funktionalität jeweils mit dem Vorschuboverride gewichtet. Im Falle von G95 bleibt die Spindeldrehzahl ohne Einfluss.

Signalpegel	1:	Die Steuerung ignoriert die programmierten Geschwindigkeitsangaben und verfährt mit dem im Systemdatum /SysCh-TestMode/TestFeedrate abgelegten Geschwindigkeitswert.
	0:	Testvorschub ist nicht mehr aktiv, die Steuerung verfährt wieder mit der programmierten Geschwindigkeit.

Testeilgang qCh_TestRap

Mit diesem Signal ignoriert die Steuerung die programmierten Geschwindigkeitsangaben und fährt statt dessen mit dem, in dem Systemdatum /SysCh-TestMode/TestRapid abgelegten Geschwindigkeitswert.

Die Funktion Testeilgang kann per NCS-Funktion oder über das Bit-Interface des jeweiligen Kanals aktiviert werden.

Die Umschaltung erfolgt unmittelbar, da die Auswertung der Signale direkt innerhalb der Interpolatorlogik erfolgt und den Sollteilgang beeinflusst.



Jedoch ist zu beachten, dass eine gewisse Zeit vergeht, bis der Sollteilgang erreicht wird, da die Brems- oder Beschleunigungsvorgänge mit der parametrisierten bzw. programmierten Beschleunigung erfolgen.

Die NC meldet den jeweils aktiven Zustand am NC-Ausgangsinterface zurück.

Die Funktionalität Testteilgang wirkt unter G93 (Zeitprogrammierung), G94 (direkte Vorschubprogrammierung) und G95 (Vorschub/Umdrehung). Das Verhalten ist jedoch unabhängig vom Zustand G93/G94/G95. Der Testteilgang wird bei aktiver Funktionalität jeweils mit dem Vorschuboverride gewichtet. Im Falle von G95 bleibt die Spindeldrehzahl ohne Einfluss.

- Signalpegel**
- 1:** Die Steuerung ignoriert die Eilganggeschwindigkeit und verfährt mit dem im Systemdatum /SysChTestMode/TestRapid abgelegten Geschwindigkeitswert.
- 0:** Testteilgang ist nicht mehr aktiv, die Steuerung verfährt wieder mit Eilgang.

Zweiter Vorschub qCh_Feedrate2

Mit diesem Signal wird der Vorschub2-Wert aktiviert, d. h. die Steuerung fährt mit dem per F2 programmierten Geschwindigkeitswert. Die Umschaltung erfolgt unmittelbar, da die Auswertung des Signals direkt innerhalb der Interpolatorlogik erfolgt und den Sollvorschub beeinflusst.

Die NC meldet den jeweils aktiven Zustand am NC-Ausgangsinterface zurück (iCh_Feedrate2).

Die Funktionalität Vorschub2 wirkt unter G94 (direkte Vorschubprogrammierung). Der Vorschub2 wird mit dem Standard-Vorschub-Override und zusätzlich mit einem eigenen Override-Wert, der sich aus dem Systemdatum SD.SysChCmd[...].Feedrate2Ovr ergibt, gewichtet.

- Signalpegel**
- 1:** Die Steuerung verfährt mit der programmierten Geschwindigkeit F2 (Vorschub2)
- 0:** Die Steuerung verfährt mit der programmierten Geschwindigkeit F (Standard- Vorschub)

7.2.2 Eingangssignale (NC --> SPS)

Aktive Betriebsart iCh_OpMode_00 ... 03

Die aktive Betriebsart wird codiert ausgegeben:

Betriebsart	Aktive Betriebsart			
	Bit 3 (iCh_OpMode_03)	Bit 2 (iCh_OpMode_02)	Bit 1 (iCh_OpMode_01)	Bit 0 (iCh_OpMode_00)
Keine Betriebsart	0	0	0	0
Einrichten manuell	0	0	0	1
Einrichten Referenzpunkt anfahren	0	0	1	0
reserviert	0	0	1	1
Handeingabe NC-Satz	0	1	0	0

Betriebsart	Aktive Betriebsart			
	Bit 3 (iCh_ OpMode_03)	Bit 2 (iCh_ OpMode_02)	Bit 1 (iCh_ OpMode_01)	Bit 0 (iCh_ OpMode_00)
Folgesatz	0	1	0	1
Programmsatz	0	1	1	0
Einzelschritt	0	1	1	1
Teachen	1	0	0	0
reserviert	1	0	0	1
Einzelsatz	1	0	1	0
Wiederanfahren	1	0	1	1
CPL-Debugger-Programmsatz	1	1	0	0
CPL-Debugger-Folgesatz	1	1	0	1
Einrichten manuell Werkstück-koordinaten	1	1	1	0
reserviert	1	1	1	1

Tab. 7-5: Aktive Betriebsart

Testbetrieb iCh_DryRun

Dieses Signal ist gesetzt, wenn alle Achsen des Kanals im Testbetrieb sind.

Im Testbetrieb werden Achsen zwar weiterhin intern interpoliert, die berechneten Sollwerte gelangen jedoch lediglich zur Anzeige und werden nicht zum Antrieb übertragen. Die Achse wird auf der Position gehalten, auf der sie zum Zeitpunkt der Aktivierung des Testbetriebs stand.

Testbetrieb kann über Softkey, automatisch bei nicht angeschlossenen Antrieben (keine Sercos Ring vorhanden) oder über den Maschinenparameter / **MAIN/Dr[1]/EnabVirtMode** "Virtueller Antrieb" (1001 00010) eingeschaltet werden.

NC-Bereit iCh_NCReady

Das Signal NC-Bereit ist Voraussetzung für das Setzen des Signals **NC-Start**.

Signalpegel	1:	<i>Wird gesetzt, wenn:</i> • ein Programm angewählt ist oder unter Handeingabe ein Satz aktiviert, aber noch nicht gestartet ist. • in den Betriebsarten Programmsatz, Einzelsatz oder Einzelschrittein Satz abgearbeitet und der neue Satz noch nicht gestartet ist. • in den Betriebsarten Folgesatz, Programmsatz, Einzelsatz oder Einzelschritte in Programm beendet ist. • die NC-Funktionen "Programm Halt" M0 oder M1 ausgeführt wurden. • Restweg löschen (qCh_CancDist) ausgeführt ist.
	0:	<i>Ist zurückgesetzt, wenn:</i> • kein NC-Programm angewählt ist. • NC-Satz eingegeben ist.

- das Abarbeiten mit NC-Start bereits eingeleitet wurde und das Signal Programm läuft gesetzt ist.

Solange NC-Bereit zurückgesetzt ist, darf NC-Start nicht gesetzt werden.

Eine Ausnahme besteht nur in Verbindung mit **Vorschub Halt**. Wurde Vorschub Halt gesetzt, bleibt NC-Bereit = 0, das Bearbeitungsprogramm kann jedoch trotzdem mit NC-Start erneut gestartet werden.

Programm läuft iCh_ProgRun

Wirkt nur in den Betriebsarten **Folgesatz, Programmsatz, Einzelsatz und Einzelschritt**, wenn zuvor ein Programm oder in der Betriebsart **Handeingabe** ein NC-Satz angewählt wurde.

Signalpegel	1:	Das Signal ist gesetzt, wenn in den Betriebsarten Folgesatz, Programmsatz, Einzelsatz und Einzelschritt ein NC-Teilprogramm mit NC-Start gestartet wurde, oder wenn in der Betriebsart Handeingabe mit NC-Start das Abarbeiten eines einzelnen NC-Satzes eingeleitet worden ist. Die Ausgangssignale Vorschubsperr , Einleseperr und Antrieb Aus haben keinen Einfluss auf dieses Signal. Wird Override = 0 vorgegeben, bleibt der Signalpegel auf Logisch 1.
	0:	<i>Ist zurückgesetzt, wenn:</i> • in den Betriebsarten Folgesatz, Programmsatz, Einzelsatz oder Einzelschritt ein Programm angewählt ist. • ein Programm angewählt oder unter Handeingabe ein Satz aktiviert, aber noch nicht gestartet ist. • in den Betriebsarten Programmsatz, Einzelsatz oder Einzelschrittein Satz abgearbeitet und der neue Satz noch nicht gestartet ist. • die NC-Funktionen M0, M1, M2 oder M30 ausgeführt wurden. • Vorschub-Halt (qCh_FeedHold) ansteht.

Einleseperr aktiv iCh_TransferLockAct

Dieses Signal zeigt an, dass der Kanal aufgrund einer anstehenden Einleseperr mit dem Abarbeiten von Sätzen wartet, bis die Einleseperr zurückgesetzt ist.

Signalpegel	1:	Nachdem eine Einleseperr aktiviert wurde und der, zum Zeitpunkt der Aktivierung, aktuelle NC-Satz abgearbeitet wurde, wartet der Kanal auf das Rücksetzen der Einleseperr und setzt das Signal "Einleseperr aktiv". Dieses Signal kann benutzt werden, wenn zum Beispiel für die Übergabe einer externen Korrektur am Satzende Restweg löschen durchgeführt werden soll. Sobald nach dem Aktivieren der Einleseperr dieses Signal gesetzt ist, können über Restweg löschen die vorbereiteten Sätze gelöscht werden, damit die übergebenen Korrekturwerte ab dem nächsten Satz aktiv werden.
	0:	Keine Einleseperr aktiv.



Dieses Signal ist kein Quittungssignal dafür, dass eine Einlesesperre von der NC erkannt wurde, sondern dafür, dass die Einlesesperre aktiv wurde.

Vorschub-Halt aktiv iCh_FeedHoldAct

Dieses Signal ist gesetzt, solange die Selbsthaltung des Signals **Vorschub Halt aktiv** ist.

Signalpegel	1:	Die Selbsthaltung des Signals Vorschub Halt ist aktiv und es können keine Verfahrbewegungen von Bearbeitungsachsen durchgeführt werden, bis Vorschub Halt mit NC-Start aufgehoben wird.
	0:	Vorschub Halt ist nicht aktiv und es können Verfahrbewegungen von Bearbeitungsachsen durchgeführt werden.

Programm Halt M0 iCh_ProgStopM0

Dieses Signal wird ausgegeben, wenn die NC-Funktionen bedingtes und unbedingtes **Programm Halt** ausgeführt werden. Gleichzeitig wird NC-Bereit (iCh_NCRReady) gesetzt.

Wird bis zum Beenden des Satzes kein NC-Start (qCh_NCStart) gesetzt, hält das Programm an und muss erneut mit qCh_NCStart gestartet werden.

Signalpegel	1:	<i>Wird gesetzt, wenn:</i> • ein NC-Satz mit Programm Halt (M0) aktiv wird. • ein NC-Satz mit bedingtem Programm Halt (M1) aktiv wird und gleichzeitig das Signal Wahlweise Halt (qCh_OptStop) gesetzt ist.
	0:	<i>Wird zurückgesetzt, wenn:</i> • ein Programm nach Programm Halt (M0, M1) mit NC-Start wieder gestartet wird. • ein Programm abgewählt wird.

Programm Ende M30 iCh_ProgStopM30

Dieses Signal wird mit Beenden des Hauptprogramms ausgegeben.

Signalpegel	1:	Wird gesetzt, wenn der NC-Satz mit M30 im Hauptprogramm beendet wurde oder wenn das Hauptprogramm vollständig abgearbeitet ist, d. h., M30 beendet ist.
	0:	Mit NC-Start (qCh_NCStart) am Programmanfang oder beim Abwählen des Programms wird das Signal zurückgesetzt.

Kanal Grundstellung iCh_Reset

Dieses Signal wirkt im Zusammenhang mit dem Signal Grundstellung oder der Funktion Grundstellung, die über die NC-Bedienoberfläche eingeleitet wird (siehe auch Signaldiagramm Grundstellung auf [Kap. "Grundstellung qCh_CtrlReset" auf Seite 44](#)).

Signalpegel	1:	Steuerungsintern ist die Funktion Grundstellung ausgeführt und der Kanal befindet sich im Einschaltzustand. Mit diesem Signal muss das Signal Grundstellung zurückgesetzt werden. Der Signalpegel 1 bleibt erhalten, solange sich der Kanal im Einschaltzustand befindet und noch kein Programm angewählt ist.

- 0: Der Kanal befindet sich nicht mehr im Einschaltzustand, ein NC-Teilprogramm ist angewählt oder bereits aktiv.

Wiederanfahren zum Anfahrpunkt aktiv iCh_RetpaAct

- Signalpegel 1: Die Wiederanfahrbewegung zum Anfahrpunkt nach Weg- und Wiederanfahren, ASUP und Satzvorlauf wurde gestartet und der Anfahrpunkt ist noch nicht erreicht.
Als Anfahrpunkt kann der Start-, End- oder Unterbrechungspunkt des unterbrochenen Satzes vorgewählt sein.
- 0: Mit dem Beenden der Wiederanfahrbewegung wird das Signal zurückgesetzt.

Wiederanfahren auf Wegtipp-Pfad iCh_RetContPath

- Signalpegel 1: Der Wiederanfahrweg "Wegtipp-Pfad" wurde vorgewählt. Die Wiederanfahrbewegung im Anfahrmodus "Automatik" und "Einzelsatz" führt rückwärts, entlang des Wegtipp-Pfads, zum Aufsetzpunkt. Der Aufsetzpunkt entspricht dem Endpunkt der ersten Wegtipp-Bewegung. Vom Aufsetzpunkt kann mit NC-Start zum Anfahrpunkt (Start-, End- oder Unterbrechungspunkt) gefahren werden.
Im Anfahrmodus "Manuell" wird die Vorwahl "Wegtipp-Pfad" ignoriert.
- 0: Der Wiederanfahrweg "Direkt" wurde vorgewählt. Die Wiederanfahrbewegung führt mit NC-Start direkt zum Anfahrpunkt.

Wegfahren beendet iCh_RemoveFinish

- Signalpegel 1: Mit der steigenden Flanke von Wiederanfahren (qCh_RetCont) oder mit Betätigen des Softkeys "Wiederanfahren" wurde die Wegfahrbewegung beendet. Nachfolgende Verfahrbewegungen werden nicht mehr aufgezeichnet.
- 0: Wegfahrbewegungen werden bis zu einer maximalen Anzahl von Sätzen (festgelegt in Maschinenparameter /OPF/Return-Path/Ch[1]/NofJogSteps "Minimale Shape-Ordnung zur Bahnrickbegrenzung" (7050 00330)) aufgezeichnet.

Bereit zum Wiederanfahren iCh_ReadyReEnter

- Signalpegel 1: In der Wiederanfahrart "automatisches Anfahren" und gesetztem Signal Wiederanfahren (qCh_RetCont) kann mittels NC-Start auf der gespeicherten Kontur zurückgefahren werden. In allen anderen Fällen kann die Bearbeitung mittels NC-Start fortgesetzt werden.
- 0: Automatisches Anfahren oder Fortsetzung der Bearbeitung ist nicht möglich.

Wiedereintritt aktiv iCh_ReEnterAct

- Signalpegel 1: Automatisches Anfahren auf der gespeicherten Kontur (Wegtipp-Pfad) wurde mit NC-Start gestartet und der Aufsetzpunkt wurde noch nicht erreicht. Der Aufsetzpunkt entspricht dem Endpunkt der ersten Wegtipp-Bewegung.
- 0: Automatisches Anfahren wurde noch nicht gestartet oder ist bereits beendet.

Kanalbezogene Interface-Signale

Asynchrones Unterprogramm 1 ... 8 iCh_ASub1 ... 8

Jedes dieser Signale ist einem, von maximal 8 asynchronen Unterprogrammen zugeordnet.

Signalpegel 1: Asynchrones Unterprogramm Nr. x aktiv
 0: Asynchrones Unterprogramm Nr. x nicht aktiv

Kanalzustand Bit 0 ... 4 iCh_State_00 .. 04

Abhängig von der Betriebsart werden folgende aktive Kanalzustände binär codiert ausgegeben:

Betriebsart	Kanalzustand	Bit 4 (iCh_State_04)	Bit 3 (iCh_State_03)	Bit 2 (iCh_State_02)	Bit 1 (iCh_State_01)	Bit 0 (iCh_State_00)
Inaktiv	Betriebsart ist nicht aktiv und ein Prozess kann angewählt werden.	0	0	0	0	1
Bereit	Betriebsart ist bereit und der Prozess kann gestartet werden.	0	0	0	1	0
Läuft	Betriebsart ist aktiv und bearbeitet ein Programm oder einen NC-Satz.	0	0	0	1	1
Intern/reserviert	reserviert	0	0	1	0	0
Intern/reserviert	reserviert	0	0	1	0	1
Fehler	In der Betriebsart ist ein Fehler aufgetreten, der nur durch "Grundstellung" oder "Programm-Abwahl" behoben werden kann.	0	0	1	1	0
Intern/reserviert	reserviert	0	0	1	1	1
Grundstellung aktiv	Der Kanal führt gerade eine Grundstellung durch. Es dürfen keine Aufträge angenommen werden, bis der Zustand auf "inaktiv" übergeht.	0	1	0	0	0
Programmanwahl läuft	Ein Programm ist angewählt und wird vorbereitet (z. B. gelinkt).	0	1	0	0	1
Restweg löschen Vorbereitung	"Restweg löschen" wurde ausgelöst, Aufräumarbeiten dauern an.	0	1	0	1	0
Restweg löschen läuft	Betriebsart ist aktiv und bearbeitet vorhandene Puffer neu.	0	1	0	1	1
Bereit für Programmstart	Betriebsart ist bereit, der Prozess steht am Programmanfang und kann gestartet werden.	0	1	1	0	0
Bereit für den nächsten Satz	Bei der gepufferten NC-Satzvorgabe sind alle Sätze abgearbeitet. Es wird auf die nächste Vorgabe gewartet.	0	1	1	0	1

Tab. 7-6: Binär codierte aktive Kanalzustände

Satz überlesen aktivieren iCh_BlSlash

Steht im Zusammenhang mit dem im NC-Teileprogramm programmierten Zeichen für **Satz überlesen (/)** und dem Signal **Satz überlesen**.

Signalpegel	1:	Wird gesetzt, wenn auf der NC-Bedienoberfläche "Satz überlesen" betätigt wurde. Das Signal bleibt gesetzt, bis der Softkey Satz überlesen nochmals gedrückt wird. Durch Betätigen des Softkeys ist die NC-interne Funktion Satz überlesen noch nicht aktiviert, dazu muss das Signal Satz überlesen (qCh_BlSlash) gesetzt sein. Auf einfache Art lässt sich das erreichen, wenn Satz überlesen aktivieren auf das Signal Satz überlesen zurückgeführt wird. Alle NC-Sätze des Teileprogramms, die mit einem Schrägstrich "/" gekennzeichnet sind, werden anschließend überlesen.
	0:	Der Softkey Satz überlesen wurde nicht aktiviert.

Wahlweiser Halt aktivieren iCh_OptStop

Steht in Zusammenhang mit der im NC-Teileprogramm programmierbaren Funktion **M1** sowie dem Signal **Wahlweise Halt**.

Signalpegel	1:	Wird gesetzt, wenn auf der NC-Bedienoberfläche der Softkey Wahlweise Halt betätigt wurde. Das Signal bleibt gesetzt, bis der Softkey Wahlweise Halt nochmals gedrückt wird. Durch Betätigen des Softkeys ist die NC-interne Funktion Wahlweiser Halt noch nicht aktiviert, dazu muss zusätzlich das Signal Wahlweise Halt (qCh_OptStop) gesetzt sein. Auf einfache Art lässt sich das erreichen, wenn Wahlweiser Halt aktivieren auf das Signal Wahlweiser Halt zurückgeführt wird. Beim Abarbeiten des NC-Teileprogramms wird der Programmablauf anschließend in dem Satz unterbrochen, indem die Funktion M1 programmiert ist. Der Programmablauf kann durch Setzen von NC-Start fortgeführt werden.
	0:	Der Softkey Wahlweiser Halt wurde nicht aktiviert.

iCh_OptJump

Dieses Signal wird zur Zeit noch nicht unterstützt. Die entsprechende Funktionalität ist innerhalb der MTX-Oberfläche bislang nicht implementiert.

Übersicht Interface-Signale für Satzvorlauf

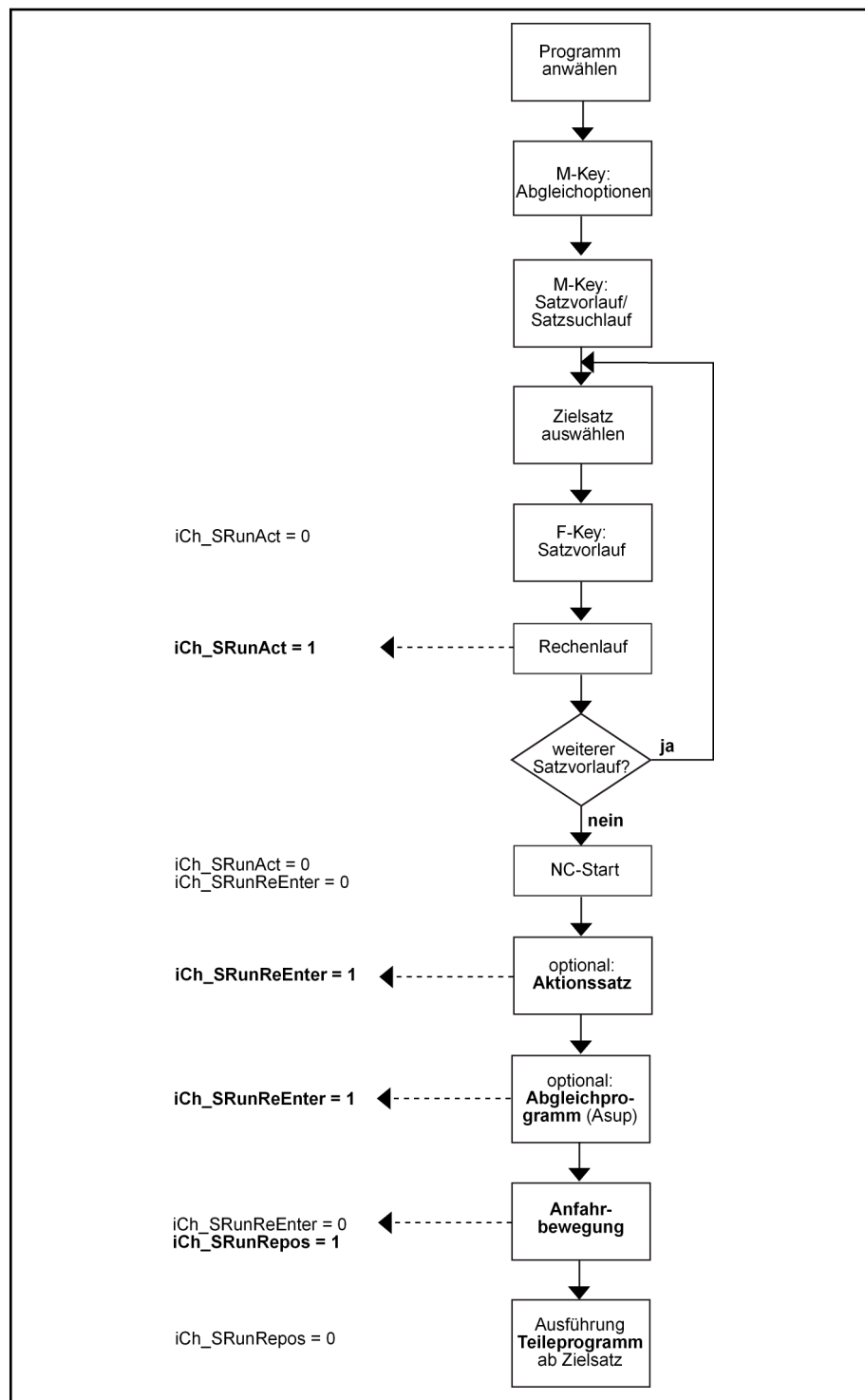


Abb. 7-2: Interface-Signale für Satzvorlauf

Satzvorlauf aktiv iCh_SRunAct

Das Signal wirkt in Verbindung mit der Funktionalität Satzvorlauf.

- Signalpegel**
- 1:** Das Programm wird im "Rechenlauf" bis zum Abbruchsatz (= Zielsatz-1) abgearbeitet.
 - 0:** Kein Rechenlauf aktiv.

Wiedereintritt aktiv iCh_SRunReEnter

Das Signal wirkt in Verbindung mit der Funktionalität Satzvorlauf.

Signalpegel	1:	Der Aktionssatz oder das Abgleichprogramm wird abgearbeitet.
	0:	Es wird kein Aktionssatz oder Abgleichprogramm abgearbeitet.

Wiederanfahren aktiv iCh_SRunRepos

Das Signal wirkt in Verbindung mit der Funktionalität Satzvorlauf.

Signalpegel	1:	Der Anfahrssatz an den Zielsatz ist aktiv.
	0:	Der Anfahrssatz ist nicht aktiv.

Override 0% iCh_Override0

Dieses Signal wirkt in Verbindung mit der Funktion **Override**.

Die Stufen der Vorschubbeeinflussung können mit **/CHAN/Ch[1]/Path/Ovrd/ChOvrdStep[1]**, **/AX/AsynchrAxOvrdIf/AxOvrdStep[1]** "Vorschub-Override für Kanal" (7030 00010) gewichtet werden.

Signalpegel	1:	Das Signal Override 100% (qCh_Override100) ist für den Kanal nicht gesetzt. Ein eventuell programmierter Vorschub 100% mit "OvrDis" (OVD) hat keinen Einfluss auf dieses Signal.
	0:	Override 100% wurde über Interface angewählt.

Override 100% iCh_Override100

Dieses Signal wirkt in Verbindung mit der Funktion **Override**.

Die Stufen der Vorschubbeeinflussung können mit Maschinenparameter **/CHAN/Ch[1]/Path/Ovrd/ChOvrdStep[1]** bzw. **/AX/AsynchrAxOvrdIf/AxOvrdStep[1]** "Vorschub-Override für Kanal" (7030 00010) gewichtet werden.

Signalpegel	1:	Das Signal Override 100% (qCh_Override100) ist für den Kanal gesetzt. Ein eventuell programmierter Vorschub 100% mit "OvrDis"(OVD) hat keinen Einfluss auf dieses Signal.
	0:	Kein Override 100% über Interface angewählt.

CPL-Kundenausgang 1 ... 16 iCh_Cpl01 ... 16

Diese 16 Signale können direkt aus dem CPL-Teilprogramm durch den CPL-Befehl **"BITIF(..)"** beschrieben und gelesen werden (siehe "Programmierhandbuch").

Kundenausgang 1 ... 8 iCh_Custom1 ... 8

NC-seitig konfigurierbares Interface, das je nach Bedarf belegt werden kann.
(Diese Möglichkeit ist z. Zt. noch nicht aktiv)

Eilgang aktiv iCh_G0Act

Dieses Signal wird gesetzt, wenn "Verfahren im Eilgang" modal aktiv ist (G0-Varianten).

Signalpegel	1:	Das Signal wird solange ausgegeben, wie Eilgang-Varianten aktiv sind.
	0:	Kein Eilgang wirksam.

Inpos Bereich 2 aktiv iCh_InPosAct

Dieses Signal zeigt an, dass für alle, dem Kanal zugeordneten Achsen der Genauhalt-Modus im **Positionierfenster 2 (grob)** aktiv ist (z. B. G0(IPS2)). Das Positionierfenster 2 wird achsweise über den Sercos Parameter S-0-0261 festgelegt.

Sind die Genauhalt-Modi ..(IPS1) oder ..(IPS3) aktiv, ist dieses Signal nicht gesetzt und zur Bildung des Achs-Interface-Signals Achse in Position (iAx_InPos) wird für die jeweiligen Kanalachsen das Positionierfenster des Sercos Parameters S-0-0057 zu Grunde gelegt.

Signalpegel	1:	Es wirkt das Positionierfenster 2 (grob) für alle, dem Kanal zugeordneten Achsen.
	0:	Es wirkt das "normale" Positionierfenster für alle, dem Kanal zugeordneten Achsen.

G41/141 aktiv iCh_G41G141Act, G42/142 aktiv iCh_G42G142Act

Diese Signale zeigen an, dass die jeweilige Funktion modal im Kanal aktiv ist. G41/141 und G42/142 schließen sich von der Programmierung gegenseitig aus, so dass immer nur eines von beiden Signalen aktiv sein kann. Ist keines der beiden Signale gesetzt, ist keine Werkzeugradiuskorrektur angewählt, d. h., G40 und G140 sind aktiv.

Signalpegel	1:	G41/141 oder G42/142 ist aktiv.
	0:	G41/141 oder G42/142 ist nicht aktiv.

Rückwärtsbetrieb aktiv iCh_RetraceAct

Rückwärtsbetrieb ist aktiv. Es können Rückwärts- und Vorwärtsbewegungen durchgeführt werden.

Das Signal wird gelöscht, wenn der Rückwärtsbetrieb beendet wurde, d. h. qCh_Retrace=0 **und** die Bearbeitung wurde mit NC-Start fortgesetzt.

Signalpegel	1:	Rückwärtsbetrieb aktiv
	0:	Rückwärtsbetrieb nicht aktiv

Koordinatenkopplung aktiv iCh_CoordCoupleAct

Dieses Signal zeigt an, dass der betreffende Kanal an einen Master-Kanal gekoppelt ist.

Signalpegel	1:	Die Koordinatenkopplung ist aktiv.
	0:	Die Koordinatenkopplung ist nicht aktiv.

Werkzeugdrehung (TangTool (TTL)) iCh_TangTRotCmd

Dieses Signal zeigt an, dass der Winkel zwischen 2 Verfahrsätzen den aktuell mit TangTool (TTL) (Tangentiale Werkzeugführung) programmierten Konturnickwinkel überschritten hat.

Die NC wartet mit der Ausführung des steuerungsintern generierten Zwischensatzes so lange, bis von der SPS die Quittung **TangTool (TTL) Freigabe (qCh_TangTRotRel)** gesetzt wird.

Das Signal iCh_TangTRotCmd bleibt solange anstehen, bis die Ausführung des Zwischensatzes beendet ist.

Signalpegel	1:	Der Winkel zwischen zwei Sätzen übersteigt den programmierten Konturnickwinkel. Die NC wartet auf das Setzen der Freigabe durch die SPS oder führt den Zwischensatz gerade aus.
	0:	Werkzeugdrehung überschreitet nicht den Konturnickwinkel.

Die NC hat aktuell keinen Zwischensatz anstehen, der auf das Setzen der Freigabe wartet.

G70 aktiv iCh_ActFunc01

Dieses Signal wird gesetzt, wenn im aktuellen Kanal G70 angewählt wurde. Die programmierten Werte der linearen synchronen Achsen werden dann in Inch interpretiert.

Signalpegel	1:	G70 ist aktiv (Inch Programmierung).
	0:	G71 ist aktiv (Metrisch Programmierung).

Vorschub 100% aktiv iCh_ActFunc02

Dieses Signal zeigt an, dass im Kanal die Funktion **Override 100%** setzen programmiert wurde und modal ansteht. Das Interface-Signal **Override 100%** hat keinen Einfluss auf das Signal **Vorschub 100% aktiv**. Ist dieses Signal nicht gesetzt, ist die Funktion OvrEna (OVR)(Vorschub 100% aus) im Kanal aktiv, welche wiederum durch das Interface-Signal **Override 100%** überblendet werden kann.

Signalpegel	1:	Die Funktion OvrDis (OVD) ist aktiv.
	0:	Die Funktion OvrDis (OVD) ist nicht aktiv.

G95 aktiv iCh_ActFunc03

Dieses Signal zeigt an, dass im Kanal die Geschwindigkeitsvorgabe im F-Wort als **Vorschub pro Umdrehung** interpretiert wird. G95 wird mit Grundstellung oder durch Programmierung von G95 angewählt.

Signalpegel	1:	Vorschub pro Umdrehung ist wirksam.
	0:	Vorschub pro Umdrehung ist nicht aktiv.

Achstrafo 2 inaktiv iCh_ActFunc04

Dieses Signal zeigt an, ob die Achstransformation 2 ausgeschaltet ist.

Signalpegel	1:	Achstransformation 2 ist inaktiv.
	0:	Achstransformation 2 ist aktiv (z. B. Zylindermantel- oder Transmit-Transformation).

Programmposition aktiv iCh_ActFunc05

Dieses Signal zeigt an, dass im Kanal eine **SetPos-Verschiebung** wirksam ist. Die SetPos-Verschiebung wird mit Grundstellung oder durch Programmierung von SetPos (SPS) ohne Achsadressen jeweils angewählt.

Signalpegel	1:	Eine SetPos-Verschiebung ist wirksam.
	0:	Eine SetPos-Verschiebung ist nicht aktiv.

Gewindezyklus aktiv iCh_ActFunc06

Dieses Signal kann aus einem Teileprogramm mit der Funktion ThreadSet (TST) TCIX gesetzt werden.



Hauptprogrammende (M30) oder Grundstellung löschen das Signal.

Signalpegel	1:	ThreadSet (TST) TC11 ist programmiert.
	0:	ThreadSet (TST) TC10 ist programmiert oder es ist kein Programm aktiv.

Gewindebohren aktiv iCh_ActFunc07

Dieses Signal zeigt an, dass die Funktion Gewindebohren aktiv ist.

- Signalpegel
- 1:

Gewindebohren (G63) ist aktiv
- 0:

Gewindebohren ist nicht aktiv.

Beispiel:

Anhand eines Beispiels sollen die Verläufe der wichtigsten Signale beim Gewindebohren dargestellt werden.

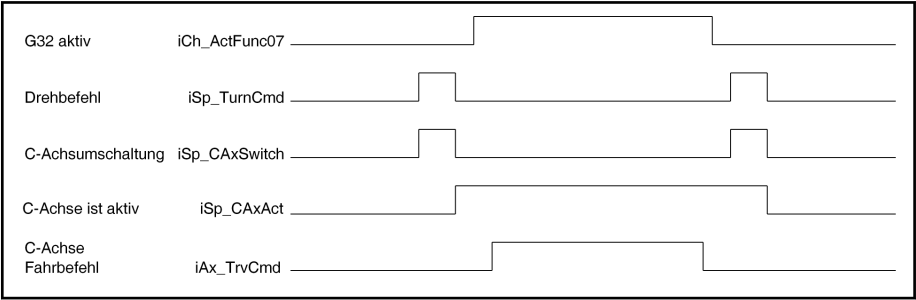


Abb. 7-3: Signale beim Gewindebohren

Gewindeschneiden aktiv iCh_ActFunc08

Dieses Signal wird bei aktivem Gewindeschneiden (G33) gesetzt.

- Signalpegel
- 1:

G33 ist aktiv. Die Vorschubgeschwindigkeit ist von der Spindeldrehzahl und der Gewindesteigung abhängig

Folgendes Signal reagiert **verändert**:

- qCh_CtrlReset (Grundstellung):
Löst, falls konfiguriert und freigeschaltet, einen schnellen Rückzug aus (siehe qCh_Retract, Schneller Rückzug).

Folgende Signale haben **keine Wirkung**:

- qCh_Override100 (Override 100%)
- qCh_Override_00...15 (Override Bit 0...15)
- qCh_FeedStop (Vorschubsperr)
- qCh_FeedHold (Vorschub Halt).

Folgendes Signal darf **nicht gesetzt** werden:

- qCh_CancDist (Restweg löschen)

- 0:

G33 ist nicht aktiv.

Werkzeugkorrektur aktiv Bit 0 ... Bit 4 iCh_ActFunc09 ... 13

Diese Signale zeigen an, welche Werkzeugkorrektur im Kanal aktiv ist.

Werkzeugkorrektur	Ausgangssignal				
	Bit 4 iCh_ActFunc13	Bit 3 iCh_ActFunc12	Bit 2 iCh_ActFunc11	Bit 1 iCh_ActFunc10	Bit 0 iCh_ActFunc09
ED0	0	0	0	0	0
ED1	0	0	0	0	1
ED2	0	0	0	1	0

Werkzeugkorrektur	Ausgangssignal				
	Bit 4 iCh_ActFunc13	Bit 3 iCh_ActFunc12	Bit 2 iCh_ActFunc11	Bit 1 iCh_ActFunc10	Bit 0 iCh_ActFunc09
ED3	0	0	0	1	1
ED4	0	0	1	0	0
ED5	0	0	1	0	1
ED6	0	0	1	1	0
ED7	0	0	1	1	1
ED8	0	1	0	0	0
ED9	0	1	0	0	1
ED10	0	1	0	1	0
ED11	0	1	0	1	1
ED12	0	1	1	0	0
ED13	0	1	1	0	1
ED14	0	1	1	1	0
ED15	0	1	1	1	1
ED16	1	0	0	0	0

Tab. 7-7: Werkzeugkorrektur aktiv

G96 aktiv iCh_ActFunc18

Dieses Signal zeigt an, ob die Spindel im aktuellen Kanal mit konstanter Schnittgeschwindigkeit verfährt (G96).

Signalpegel	1:	Konstante Schnittgeschwindigkeit ist aktiv.
	0:	Direkte Drehzahlprogrammierung für Spindel aktiv.

Eilgang Override 0% iCh_RapOverride0

Dieses Signal wirkt nur in Verbindung mit der Funktion **Eilgang Override**.

Die Stufen der Vorschubbeeinflussung können mit **/CHAN/Ch[]/Path/Ovrd/ChOvrdStep[]** "Override-Wert für Stufe" (7030 00010) gewichtet werden.

Signalpegel	1:	Das Signal Eilgang Override 100% (qCh_Rap100) ist für den Kanal nicht gesetzt. Ein eventuell programmierter Vorschub 100% mit "OvrDis" (OVD) hat keinen Einfluss auf dieses Signal.
	0:	Eilgang Override 100% wurde über Interface angewählt.

Eilgang Override 100% iCh_RapOverride100

Dieses Signal wirkt nur in Verbindung mit der Funktion **Eilgang Override**.

Die Stufen der Vorschubbeeinflussung können mit Maschinenparameter **/CHAN/Ch[]/Path/Ovrd/ChOvrdStep[]** "Override-Wert für Stufe" (7030 00010) gewichtet werden.

Signalpegel	1:	Das Signal Eilgang Override 100% (qCh_Rap100) ist für den Kanal gesetzt. Ein eventuell programmierter Vorschub 100% mit "OvrDis" (OVD) hat keinen Einfluss auf dieses Signal.
	0:	

0: Kein Eilgang Override 100% über Interface angewählt.

Zweiter Vorschub iCh_Feedrate2

Die NC meldet mit diesem Signal, ob der Vorschub2-Wert aktiviert ist. D. h. es zeigt an, ob die Steuerung mit dem Standard-Vorschub F oder mit dem zweiten Vorschub F2 als programmiertem Geschwindigkeitswert fährt.

Signalpegel	1:	Die Steuerung verfährt mit der programmierten Geschwindigkeit F2 (Vorschub2).
	0:	Die Steuerung verfährt mit der programmierten Geschwindigkeit F (Standard- Vorschub).

8 Achsbezogene Interface-Signale

8.1 Übersicht achsbezogene Interface-Signale

8.1.1 Allgemeines

Die achsbezogenen Interface-Signale wirken auf die NC-Funktionen, die mit dem Steuern einer Bearbeitungs- oder Hilfsachse im Zusammenhang stehen.

Die Anzahl der vorhandenen Achs-Interfaces entspricht dem Index der letzten in Maschinenparameter `/MAIN/NofDr`, `/MAIN/Dr[1]/EnablDr`, `/MAIN/Dr[1]/AxFunc`, `/MAIN/Dr[1]/SpFunc`, `/MAIN/Dr[1]/SpFunc/Splnd` "Antriebsfunktions-typ" (1001 00001) eingetragenen Achse oder C-Achse/Spindel (identisch mit dem physikalischen Achs- oder Antriebsindex). Das Achs-Interface einer C-Achse wird nur während des C-Achsbetriebs von der NC ausgewertet oder beschrieben.

Sind vor der letzten eingestellten Achse oder C-Achse/Spindel Achsen ausgeblendet oder "einfache" Spindeln konfiguriert, wird für diese jeweils ein achsbezogenes Interface auf die entsprechenden Merker abgebildet, welches jedoch von der NC nicht bedient wird!

In der folgenden Beschreibung sind die Interface-Signale für eine Achse dargestellt.

8.1.2 Übersicht achsbezogene Ausgangssignale (SPS --> NC)

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
0.0	qAx_OpModeSel_00	Achsbetriebsart Bit 0	1.0	qAx_TrvLim_00	Endschalterbereich Bit 0
0.1	qAx_OpModeSel_01	Achsbetriebsart Bit 1	1.1	qAx_TrvLim_01	Endschalterbereich Bit 1
0.2	qAx_JogPlus	Manuell +	1.2	qAx_SwLimOff	Endschalter ausblenden
0.3	qAx_JogMinus	Manuell -	1.3	-	res.
0.4	qAx_JogInch	Schrittmaß in Inch	1.4	-	res.
0.5	qAx_JogDia	Schrittmaß Durchmesser	1.5	-	res.
0.6	qAx_NextNotch	Nächste Rasterposition	1.6	-	res.
0.7	qAx_Reset	Achsgrundstellung	1.7	qAx_FxStopRel	Festanschlag aufheben
2.0	qAx_SafOpModeSwitch	Betriebsartenwahl (BA)	3.0	qAx_HandwSel_00	Handradauswahl Bit 0
2.1	qAX_SafDrvLock	Antriebssperre	3.1	qAx_HandwSel_01	Handradauswahl Bit1
2.2	qAx_SafEnablCtrl	Zustimmtaste (ZT)	3.2	qAx_HandwDir	Handrad Drehrichtung
2.3	qAx_SafSwitch1	Si-Schalter 1 (S1)	3.3	qAx_HandwPosMode	Positionshandrad
2.4	qAx_SafSwitch2	Si-Schalter 2 (S2)	3.4	-	res.
2.5	qAx_SafCheckInputState	Prüfeingang Si-Status	3.5	-	res.
2.6	qAx_SafTechState	Status Si-Signale	3.6	-	res.

Achsbezogene Interface-Signale

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
2.7	qAx_SafRedTransTime	Überführzeit abkürzen	3.7	-	res.
4.0	qAx_ManFeed_00	Handvorschub Bit 0	5.0	-	res.
4.1	qAx_ManFeed_01	Handvorschub Bit 1	5.1	-	res.
4.2	qAx_ManFeed_02	Handvorschub Bit 2	5.2	qAx_HomeSwitch	Referenzschalter
4.3	qAx_ManFeed_03	Handvorschub Bit 3	5.3	-	res.
4.4	-	res.	5.4	-	res.
4.5	-	res.	5.5	-	res.
4.6	-	res.	5.6	-	res.
4.7	-	res.	5.7	qAx_Override100	Achs-Override 100 %
6.0	qAx_Override_00	Override Bit 0	7.0	qAx_Override_08	Override Bit 8
6.1	qAx_Override_01	Override Bit 1	7.1	qAx_Override_09	Override Bit 9
6.2	qAx_Override_02	Override Bit 2	7.2	qAx_Override_10	Override Bit 10
6.3	qAx_Override_03	Override Bit 3	7.3	qAx_Override_11	Override Bit 11
6.4	qAx_Override_04	Override Bit 4	7.4	qAx_Override_12	Override Bit 12
6.5	qAx_Override_05	Override Bit 5	7.5	qAx_Override_13	Override Bit 13
6.6	qAx_Override_06	Override Bit 6	7.6	qAx_Override_14	Override Bit 14
6.7	qAx_Override_07	Override Bit 7	7.7	qAx_Override_15	Override Bit 15
8.0	qAx_Custom1	Kundeneingang 1	9.0	-	res.
8.1	qAx_Custom2	Kundeneingang 2	9.1	-	res.
8.2	qAx_Custom3	Kundeneingang 3	9.2	-	res.
8.3	qAx_Custom4	Kundeneingang 4	9.3	-	res.
8.4	qAx_Custom5	Kundeneingang 5	9.4	-	res.
8.5	qAx_Custom6	Kundeneingang 6	9.5	-	res.
8.6	qAx_Custom7	Kundeneingang 7	9.6	-	res.
8.7	qAx_Custom8	Kundeneingang 8	9.7	-	res.
10.0	qAx_TrqErrOff	Stillstandsfehler ausblenden	11.0	qAx_Discharge	Achse abgebaut
10.1	qAx_LagErrOff	Kopplungsfehler ausblenden	11.1	qAx_FrzlpoPos	Sollposition halten
10.2	qAx_MasterPos	Gantry auf Masterposition	11.2	-	res.
10.3	qAx_TakeActOffs	Istwert-Versatz übernehmen	11.3	qAx_TrqLim	Momentenreduktion
10.4	-	res.	11.4	-	res.
10.5	-	res.	11.5	-	res.

Bit	Symbol.Adr.	SPS-Ausgangssignal	Bit	Symbol.Adr.	SPS-Ausgangssignal
10.6	-	res.	11.6	qAx_DrvOn	Antrieb Ein
10.7	qAx_RedDelayErr-Class1	Verz. Fehlerreaktion abkürzen	11.7	qAx_DrvLock	Vorschub Sperre

Tab. 8-1: Übersicht Ausgangssignale (SPS → NC)

8.1.3 Übersicht achsbezogene Eingangssignale (NC → SPS)

Bit	Symbol.Adr.	SPS-Eingangssignal	Bit	Symbol.Adr.	SPS-Eingangssignal
0.0	iAx_RefKnown	Referenzpunkt bekannt	1.0	iAx_DistCtrl	Achse vor Endpunkt
0.1	iAx_RefReached	Referenzpunkt erreicht	1.1	iAx_ProgSpReach	Achsgeschwindigkeit erreicht
0.2	iAx_Trvcmd	Fahrbehl	1.2	-	res.
0.3	iAx_TrvcDirNeg	Negative Verfahr-richtung	1.3	-	res.
0.4	iAx_Run	Achse läuft	1.4	-	res.
0.5	iAx_InPos	Achse in Position	1.5	-	res.
0.6	iAx_NotchPos	Achse auf Rasterposition	1.6	iAx_FxStopReached	Festanschlag erreicht
0.7	iAx_Reset	Achse grundgestellt	1.7	iAx_FxStopAct	Festanschlag aktiv
2.0	iAx_SafOpMode_00	Si-Betriebsart Bit 0	3.0	iAx_PosSwitch1	Positionsschalt- punkt 1
2.1	iAx_SafOpMode_01	Si-Betriebsart Bit 1	3.1	iAx_PosSwitch2	Positionsschalt- punkt 2
2.2	iAx_SafOpMode_02	Si-Betriebsart Bit 2	3.2	iAx_PosSwitch3	Positionsschalt- punkt 3
2.3	iAx_SafOpMode_03	Si-Betriebsart Bit 3	3.3	iAx_PosSwitch4	Positionsschalt- punkt 4
2.4	iAx_SafStatePos	Status Sichere Lage	3.4	iAx_PosSwitch5	Positionsschalt- punkt 5
2.5	iAx_SafCtrlOutputState	Statusausgang Regler	3.5	iAx_PosSwitch6	Positionsschalt- punkt 6
2.6	-	res.	3.6	iAx_PosSwitch7	Positionsschalt- punkt 7
2.7	-	res.	3.7	iAx_PosSwitch8	Positionsschalt- punkt 8
4.0	iAx_ChIndex_00	Kanalnummer Bit 0	5.0	-	res.
4.1	iAx_ChIndex_01	Kanalnummer Bit 1	5.1	-	res.
4.2	iAx_ChIndex_02	Kanalnummer Bit 2	5.2	-	res.
4.3	iAx_ChIndex_03	Kanalnummer Bit 3	5.3	-	res.
4.4	iAx_ChIndex_04	Kanalnummer Bit 4	5.4	-	res.
4.5	iAx_ChIndex_05	Kanalnummer Bit 5	5.5	-	res.
4.6	-	res.	5.6	iAx_Override0	Achs-Override 0 %
4.7	-	res.	5.7	iAx_Override100	Achs-Override 100 %
6.0	iAx_ScsState00	SCS-Signalstatus 0	7.0	iAx_ScsState08	SCS-Signalstatus 8
6.1	iAx_ScsState01	SCS-Signalstatus 1	7.1	iAx_ScsState09	SCS-Signalstatus 9

Achsbezogene Interface-Signale

Bit	Symbol. Adr.	SPS-Eingangssignal	Bit	Symbol. Adr.	SPS-Eingangssignal
6.2	iAx_ScsState02	SCS-Signalstatus 2	7.2	iAx_ScsState10	SCS-Signalstatus 10
6.3	iAx_ScsState03	SCS-Signalstatus 3	7.3	iAx_ScsState11	SCS-Signalstatus 11
6.4	iAx_ScsState04	SCS-Signalstatus 4	7.4	iAx_ScsState12	SCS-Signalstatus 12
6.5	iAx_ScsState05	SCS-Signalstatus 5	7.5	iAx_ScsState13	SCS-Signalstatus 13
6.6	iAx_ScsState06	SCS-Signalstatus 6	7.6	iAx_ScsState14	SCS-Signalstatus 14
6.7	iAx_ScsState07	SCS-Signalstatus 7	7.7	iAx_ScsState15	SCS-Signalstatus 15
8.0	iAx_Custom1	Kundenausgang 1	9.0	iAx_MasterAxIndex_00	Index Master-Achse Bit 0
8.1	iAx_Custom2	Kundenausgang 2	9.1	iAx_MasterAxIndex_01	Index Master-Achse Bit 1
8.2	iAx_Custom3	Kundenausgang 3	9.2	iAx_MasterAxIndex_02	Index Master-Achse Bit 2
8.3	iAx_Custom4	Kundenausgang 4	9.3	iAx_MasterAxIndex_03	Index Master-Achse Bit 3
8.4	iAx_Custom5	Kundenausgang 5	9.4	iAx_MasterAxIndex_04	Index Master-Achse Bit 4
8.5	iAx_Custom6	Kundenausgang 6	9.5	iAx_MasterAxIndex_05	Index Master-Achse Bit 5
8.6	iAx_Custom7	Kundenausgang 7	9.6	iAx_MasterAxIndex_06	Index Master-Achse Bit 6
8.7	iAx_Custom8	Kundenausgang 8	9.7	iAx_MasterAxIndex_07	Index Master-Achse Bit 7
10.0	iAx_TrqExceed	Stillstandsfehler	11.0	iAx_DrvErrClass1	Fehler Zustandsklasse-1
10.1	iAx_CoupleLag	Nachlauffehler	11.1	iAx_DrvChangeClass2	Änderung Zustandsklasse-2
10.2	iAx_CmdOffsExst	Gantry-Sollwert-Versatzaktiv	11.2	iAx_DrvChangeClass3	Änderung Zustandsklasse-3
10.3	iAx_CmdOffsExceed	Ausgleichbarer Gantry-Sollwert-Versatz überschritten	11.3	iAx_TrqLim	Moment reduziert
10.4	-	res.	11.4	iAx_DryRun	Testbetrieb
10.5	-	res.	11.5	iAx_DrvPower	Freigabe zur Leistungszuschaltung
10.6	-	res.	11.6	iAx_DrvReady	Antrieb betriebsbereit
10.7	iAx_DelayErrClass1	Verzögerte Fehlerreaktion	11.7	iAx_DrvAct	Antrieb in Betrieb

Tab. 8-2: Übersicht Eingangssignale (NC → SPS)

8.2 Signalbeschreibung

8.2.1 Achsbezogene Ausgangssignale (SPS --> NC)

Allgemeines

Je nach Funktion wertet das CNC-System MTX die Signale als **statische Signale** mit den Signalpegeln Logisch 0 und Logisch 1 aus oder als **flankengesteuerte Signale** mit den Signalfanken 0 → 1 oder 1 → 0.

Achsbetriebsart Bit 0 und 1 qAx_OpModeSel_00 ... 01

Mit diesem Signal wird für Hilfsachsen festgelegt, ob sie sich bei Betätigen von Manuell+/- in den Betriebsarten **Einrichten manuell** oder **Einrichten Referenzpunkt anfahren** bewegen sollen (Voraussetzung: die Achse ist nicht durch Programmierung in einem Kanal belegt, d. h., sie darf sich nicht bewegen).

Die angewählten Betriebsarten der Kanäle spielen keinerlei Rolle, da asynchrone Achsen keinem Kanal zugeordnet sind.

Die Betriebsarten sind folgendermaßen codiert:

Betriebsarten	Betriebsart Anwahl	
	Bit 1 (qAx_OpModeSel_01)	Bit 0 (qAx_OpModeSel_00)
Programmierbetrieb	0	0
Einrichten manuell	0	1
Einrichten Referenzpunkt anfahren	1	0
reserviert	1	1

Tab. 8-3: Codierung der Betriebsarten

Manuell+ qAx_JogPlus, Manuell- qAx_JogMinus

Die Signale Manuell+ und Manuell- wirken für **Kanalachsen** (synchrone Achsen) und **Hilfsachsen** (asynchrone Achsen) in den Betriebsarten **Einrichten Manuell** und **Einrichten Referenzpunkt anfahren**.

Darüber hinaus dienen die Signale zum manuellen Wegtippen von synchronen Achsen beim **Wegfahren von der Kontur**. Das aktive Teileprogramm ist dabei mit Vorschub Halt angehalten

Signalpegel 1: Kontinuierliches Tippen:

Ist die Kanal- bzw. Achsbetriebsart **Einrichten Manuell** gesetzt und sind alle sonstigen Voraussetzungen erfüllt (Vorschub Halt, Vorschub Sperre usw. sind nicht gesetzt), wird die Achse über Manuell+ oder Manuell- in die entsprechende Richtung verfahren.

Der Vorschub wird dabei über "Handvorschub" (qAx_ManFeed_00 ... 03) ausgewählt, die Beschleunigung ist in Maschinenparameter /AX/Dr[1]/Jog/VarDistAxJog "Variabler Inkrementen-Schritt" (1015 00002) parametrierbar.

Am Achsinterface sind dabei die Signale "Fahrbefehl" (iAx_TrvCmd) und "Achse läuft" (iAx_Run) und je nach Fahr- richtung "Negative Fahr- richtung" (iAx_TrvDirNeg) gesetzt. Das Signal "Achse in Position" (iAx_InPos) wird gleichzeitig gelöscht.

Inkrementelles Tippen:

Wie oben, jedoch fährt die Achse das über "Handvorschub" (qAx_ManFeed_00 ... 03) ausgewählte **Schrittmaß** ab. Die Achse verfährt auch dann weiter, wenn das Signal Manuell+ bzw. Manuell- nicht weiter ansteht und der vorgewählte Weg noch nicht fertig abgefahren wurde. Der Weg wird mit der in **Handvorschub Bit 0 ... Bit 3** bei definierte Schritte bzw. variable Schritte eingetragenen Geschwindigkeit abgefahren.

Eine erneute Bewegung kann erst dann ausgelöst werden, wenn Manuell+ bzw. Manuell- zuvor den Signalpegel logisch 0 hatte (Flankentriggerung).

Referenzpunkt anfahren:

Ist die Kanal- bzw. Achsbetriebsart **Einrichten Referenzpunkt anfahren** gesetzt und sind alle sonstigen Voraussetzungen erfüllt, wird für die Achse mit Manuell+ oder Manuell- das antriebsgeführte Referenzieren ausgelöst. Das Verhalten des Antriebs wird dabei im Sercos Referenzfahr-Parameter S-0-0147 festgelegt.

0:**Kontinuierliches Tippen:**

Mit Rücknahme des Signals Manuell+ bzw. Manuell- wird die Verfahrbewegung mit der in den Maschinenparametern eingestellten Beschleunigung gestoppt. Der Fahrbefehl wird zurückgesetzt. Sobald sich die Achse im Inpositions-Fenster befindet wird das Signal "Achse läuft" zurückgesetzt und Achse in Position ausgegeben.

Beispiel:

Eine Achse (synchron oder asynchron) soll in der Betriebsart "Einrichten" in die positive Richtung verfahren werden.

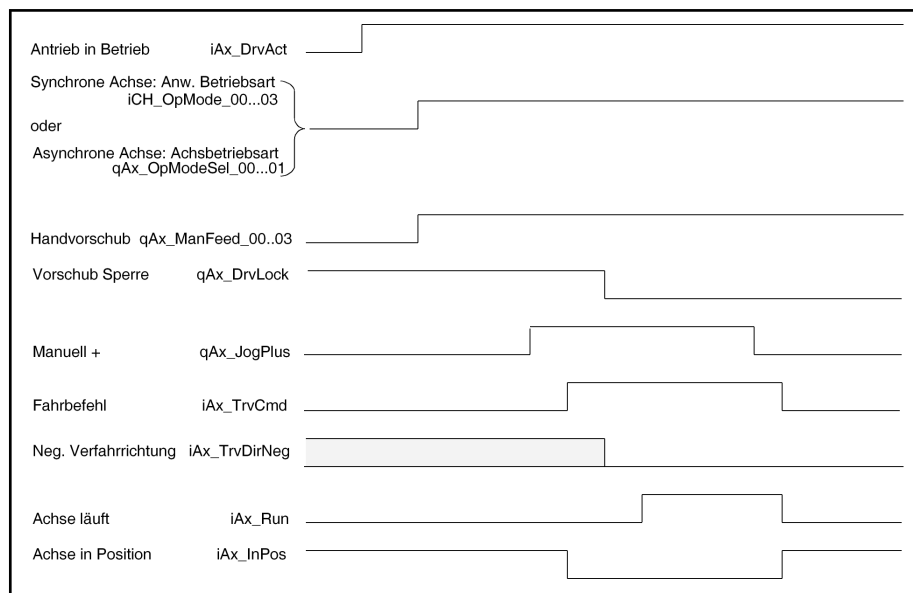


Abb. 8-1: Verfahren in positive Richtung

Die Steuerung befindet sich im eingeschalteten Zustand, der Regelkreis ist bereits geschlossen und der Antrieb erzeugt Drehmoment. **Antrieb in Betrieb** (iAx_DrvAct) ist gesetzt.

Um eine Achse **manuell** in die positive oder negative Richtung zu verfahren, muss bei Bearbeitungsachsen die Betriebsart **Einrichten** des jeweiligen Ka-

nals aktiviert sein. Die Betriebsart **Einrichten** wird im kanalbezogenen Interface codiert an den Signalen iCH_OpMode_00...03 ausgegeben.

Bei Hilfsachsen muss die Betriebsartenwahl qAx_OpModeSel_00...01 entsprechend gesetzt sein. Der **Handvorschub** bzw. das **Schrittmaß** wird über die Signale qAx_ManFeed_00...03 codiert vorgegeben.

Mit Setzen des Signals **Manuell+** wird das Signal **Fahrbefehl** ausgegeben sowie die **Verfahrrichtung**.

Mit Ausgabe des **Fahrbefehls** wird das Signal **Achse In Position** zurückgesetzt.

Solange **Vorschubsperr** ansteht, ist keine Verfahrbewegung möglich. Mit Rücksetzen des Signals Vorschubsperr verfährt die Achse so lange, wie Manuell+ betätigt wird oder bis das vorgegebene **Schrittmaß** verfahren wurde. Während die Achse verfährt, wird **Achse läuft** ausgegeben. Wurde die Vorschubsperr bereits zu einem früheren Zeitpunkt zurückgesetzt, verfährt die Achse unmittelbar nach Vorgabe von Manuell+.

Der Signal **Fahrbefehl** steht an, bis die Verfahrbewegung abgeschlossen ist. Steht die Achse, wird **Achse läuft** zurückgesetzt und das Signal **Achse In Position** ausgegeben, wenn sich die Achse im Inpositions-Fenster befindet. Das Signal **Verfahrrichtung** bleibt solange anstehen, bis der nächste Fahrbefehl mit entgegengesetzter Richtung ausgegeben wird.

Schrittmaß in Inch qAx_JogInch

Durch Setzen dieses Signals ist es möglich, im Einrichtbetrieb "Inch-Inkrement" zu tippen oder mit dem Handrad zu verfahren. Auch das, ebenfalls über Interface vorgegebene Schrittmaß wird in Inch interpretiert.

Das Inch-Inkrement einer Achse ergibt sich aus:

$0.0001 * AXSP\Dr[x]\AxFun\Jog\Incr [Achse] \text{ in Inch.}$

Maschinenparameter AXSP\Dr[x]\AxFun\Jog\Incr entspricht **/AX/Dr[1]/Jog/IncrAxJog** "Achsauflösung" (1015 00001).



Für Rund- und Endlosachsen ist das Interface-Signal "Schrittmaß in Inch" ohne Bedeutung.

Beispiel:

$AXSP\Dr[x]\AxFun\Jog\Incr = 1$

1 Inch-Inkrement = 0.0001 Inch

1 Metrisch-Inkrement = 0.001 mm

Signalpegel	1:	Das vorgegebene Schrittmaß wird bei Tipp- oder Handradbetrieb in der Einheit "Inch" interpretiert.
	0:	Das vorgegebene Schrittmaß wird bei Tipp- oder Handradbetrieb in der Einheit "Metrisch" interpretiert.

Schrittmaß als Durchmesser qAx_JogDia

Durch Setzen dieses Signals ist es möglich, im Einrichtbetrieb "Durchmesser-Inkrement" zu tippen oder per Handrad zu verfahren. Auch das, ebenfalls über Interface vorgegebene Schrittmaß wird als Durchmesserdiffferenz interpretiert.

Die Maßeinheit der Inkremente wird durch das Signal "Schrittmaß in Inch" vorgegeben.

Signalpegel	1:	Das vorgegebene Schrittmaß wird bei Tipp- oder Handradbetrieb als Durchmesserdifférenz interpretiert.
	0:	Das vorgegebene Schrittmaß wird bei Tipp- oder Handradbetrieb als Radiusdifférenz interpretiert.

Nächste Rasterposition qAx_NextNotch

Mit diesem Signal kann die Steuerung eine Hirthachse auf die nächste Rasterposition fahren. Dazu muss sich die NC in der Betriebsart **Einrichten manuell** befinden und die Hirthachse darf gerade nicht durch **Tippen** angesteuert werden (iAx_NotchPos ist gesetzt).

Die zuletzt aktive Verfahrrichtung (siehe iAx_TrVDirNeg) gibt die Richtung vor, in die die Hirthachse fährt.

Das Signal kann verwendet werden, um eine Hirthachse, die zwischen 2 Rasterpositionen zum Stillstand gekommen ist (z. B.: bei NOT AUS), in der anstehenden Verfahrrichtung wieder auf eine Rasterposition zuzufahren.

Achsgrundstellung qAx_Reset

Mit diesem Signal kann die Bewegung einer asynchronen Achse abgebrochen werden. Nachdem die Achse zum Stillstand gekommen ist, wird das Signal **Achse grundgestellt** (iAx_Reset) gesetzt.

Signalfanke	0 → 1:	Das Signal Achse grundgestellt (iAx_Reset) wird gesetzt.
	1 → 0:	Keine Auswirkung.

Endschalterbereich 0 ... 1 qAx_TrVLim_00 ... 01

Selektiert einen von vier möglichen Endschalterbereichen:

Bitkombination 0:	1. Endschalterbereich aktiv
Bitkombination 1:	2. Endschalterbereich aktiv
Bitkombination 2:	3. Endschalterbereich aktiv (noch nicht realisiert)
Bitkombination 3:	4. Endschalterbereich aktiv (noch nicht realisiert)



Auf den selektierten Endschalterbereich wirkt das Signal "Endschalter ausblenden" (qAx_SwLimOff).

Endschalter ausblenden qAx_SwLimOff

Dieses Signal blendet achsweise den positiven und den negativen Endschalter aus. Innerhalb der NC ist dann für diese Achse keine Endlagen-Überwachung aktiv.

Solange der Referenzpunkt der Achse noch nicht bekannt ist (siehe iAx_RefKnown), ist dieses Signal ohne Bedeutung. Die NC-internen Überprüfungen sind in diesem Fall immer ausgeschaltet.

Überwachungen innerhalb des Antriebs bleiben von diesem Signal unberührt (z. B. S-0-0049 Lagegrenzwert positiv; S-0-0050 Lagegrenzwert negativ). Weitere sicherheitsrelevante Funktionen müssen bei ausgeblendeten Endschaltern Maschinen- und SPS-seitig übernommen werden.

Das Signal **Endschalter ausblenden** wirkt in allen Betriebsarten.

Signalpegel	1:	Die Software-Endschalter sind ausgeblendet.
	0:	Die Software-Endschalter sind nicht ausgeblendet. Die NC überwacht die, in den Maschinenparametern definierten Endlagen gemäß dem ausgewählten Endschalterbereich.

Festanschlag aufheben qAx_FxStopRel

Mit Setzen dieses Signals wird für eine asynchrone Achse ein aktiver Festanschlag aufgehoben.

Betriebsartenwahl (BA) qAx_SafOpModeSwitch

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive und gilt nicht für Antriebe mit SMO.

Als Kanal 1 wird hier die Betriebsartenwahl der SPS über Sercos Interface in Echtzeit an den Antrieb übertragen.

Signalpegel	0:	Normalbetriebsart NO
	1:	Sonderbetriebsart SO
		(Sonderbetrieb mit Stillstand/mit Bewegung)

Antriebssperre qAx_SafDrvLock

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive und gilt nicht für Antriebe mit SMO.

Als Kanal 1 wird hier das Signal "Antriebssperre" per SPS über Sercos Interface in Echtzeit an den Antrieb übertragen. Bei Vorgabe der Antriebssperre schaltet sich der Antrieb momentenfrei.

Signalpegel	1:	Antriebssperre ein
	0:	Antriebssperre aus

Zustimmtaste (ZT) qAx_SafEnablCtrl

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive und gilt nicht für Antriebe mit SMO.

Als Kanal 1 wird hier die Umschaltung per SPS in "Sonderbetrieb mit Bewegung" über Sercos Interface in Echtzeit an den Antrieb übertragen.

Signalpegel	1:	Sonderbetrieb mit Bewegung
	0:	Sonderbetrieb mit Stillstand

Si-Schalter 1 (S1) qAx_SafSwitch1, Si-Schalter 2 (S2) qAx_SafSwitch2

Diese Signale sind Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive und gelten nicht für Antriebe mit SMO.

Für den "Sonderbetrieb mit Bewegung" können im Antrieb bis zu 4 verschiedene Parametersätze abgelegt sein. Als Kanal 1 wird hier der per SPS angewählte Parametersatz über Sercos in Echtzeit an den Antrieb übertragen.

Bis MTX14V06 unterstützt die MTX maximal 2 sichere Bewegungen. Ab MTX14V08 können in Verbindung mit der Sicherheitstechnik nach IEC 61508 maximal 4 sichere Bewegungen verwendet werden.

Beachten Sie bitte die Hinweise in der Antriebsdokumentation zu "Zustandsmaschine der integrierten Sicherheitstechnik" und zum Sercos-Parameter "P-0-3212 (SI-Steuerwort, Kanal 1) Bit 3 und Bit 4".



Wenn der SI-Schalter 2 "P-0-3212 (SI-Steuerwort, Kanal 1) Bit 4" zyklisch zum Antrieb übertragen wird, dann erwartet die MTX vier konfigurierte sichere Bewegungen!

Wenn nur maximal 2 sichere Bewegungen verwendet werden, dann sollte nur der SI-Schalter 1 "P-0-3212 (SI-Steuerwort, Kanal 1) Bit 3" zyklisch zum Antrieb übertragen werden.

	Bit 1 (qAx_SafSwitch2)	Bit 0 (qAx_SafSwitch1)
SMM1	0	0
SMM2	0	1
SMM3	1	0
SMM4	1	1

Tab. 8-4: Anwahl der sicheren Bewegungen

Prüfeingang Si-Status qAx_SafCheckInputState

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive und gilt nicht für Antriebe mit SMO.

Als Kanal 1 wird hier ein per SPS gesetztes sicheres Statussignal über Sercos Interface in Echtzeit an den Antrieb übertragen.

Status Si-Signale qAx_SafTechState

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive und gilt nicht für Antriebe mit SMO.

Die Steuersignale der SPS (qAx_Saf...), die über den Kanal 1 zum Antrieb gehen, müssen zur Überprüfung in gewissen Zeitabständen dynamisiert werden. Das heißt, die Signale werden von der SPS für einen Zyklus auf "0" gesetzt. Die SPS setzt gleichzeitig den Si-Signalstatus von gültig auf dynamisiert und teilt dadurch dem Antrieb mit, dass die Steuersignale ungültig sind.

Danach werden die Signale neu gesetzt.

Signalpegel	1:	Die Steuersignale der Sicherheitstechnik sind gültig.
	0:	Die Steuersignale der Sicherheitstechnik werden dynamisiert.

Überführzeit abkürzen qAx_SafRedTransTime

Dieses Signal ist Bestandteil der Sicherheitstechnik der Antriebsbaureihe IndraDrive.

Dieses Signal bietet der SPS die Möglichkeit zum Verkürzen der Überführzeit von Normalbetrieb nach Sonderbetrieb bzw. von Sonderbetrieb nach Sonderbetrieb.

Bei Verwendung der Funktion "Integrierte Sicherheitstechnik nach IEC 61508":

- Beachten Sie bitte die Hinweise in der Antriebsdokumentation zu "Zustandsmaschine der integrierten Sicherheitstechnik" und zum Sercos-Parameter "P-0-3212 (SI-Steuerwort, Kanal 1) Bit 11".

Bei Verwendung der Funktion "Integrierte Sicherheitstechnik Safe Motion":

- Beachten Sie bitte die Hinweise in der Antriebsdokumentation zu "Überföhrfunktionen" und zum Sercos-Parameter "P-0-3261.0.1 (SMO: Steuerwort Zustandsmaschine, funktional) Bit 0".

Signalpegel	1:	Meldet dem Antrieb, dass die Anpassung des Sollwertsystems abgeschlossen ist und der Antrieb aktiviert die Überwachungen der angewählten Sonderbetriebsart.
	0:	Keine Wirkung

Handradauswahl Bit 0 und Bit 1 qAx_HandwSel_00 ... 01

An die NC-Steuerung können zwei Handräder angeschlossen werden. Sobald eines der beiden Signale gesetzt ist, kann die Achse über das ausgewählte Handrad verfahren werden. Die Signale **Manuell+** und **Manuell-** sind wirkungslos.

Wird die Achse über das Handrad verfahren, sind die Signale **Handvorschub Bit 0...3** von Bedeutung. Die anwählbaren Schrittmaße werden hier als Bewertung für die Skalenteilung des Handrades verrechnet.

Handrad	Handradzuweisung	
	Bit 1 (qAx_HandwSel_01)	Bit 0 (qAx_HandwSel_00)
kein Handrad aktiv	0	0
Handrad 1	0	1
Handrad 2	1	0
reserviert	1	1

Tab. 8-5: Handradauswahl Bit 0 und Bit 1

Handrad Drehrichtung qAx_HandwDir

Mit diesem Signal ist es möglich, die Drehrichtung der Achse unabhängig von der Drehrichtung des Handrads anzugeben.

Signalpegel	1:	Die Drehrichtung der Achse ist entgegengesetzt zur Drehrichtung des Handrads festgelegt.
	0:	Drehrichtung von Achse und Handrad stimmen überein.

Positionshandrad qAx_HandwPosMode

Mit diesem Signal ist es möglich, das Verhalten des Handradbetriebs beim Überschreiten der aktuell zulässigen Maximalgeschwindigkeit vorzugeben.



Der Pegel dieses Signals wird beim Setzen von qAx_HandwSel_0x von der NC übernommen.

Signalpegel	1:	Der durch Handradbewegungen vorgegebene Weg wird auch bei einer Überschreitung der Maximalgeschwindigkeit abgearbeitet. Achtung: Auch nach Beenden der Handradbewegung können noch Achsbewegungen stattfinden.
	0:	Bei einer Überschreitung der Maximalgeschwindigkeit wird der Fahrweg entsprechend reduziert.

Handvorschub Bit 0 ... Bit 3 qAx_ManFeed_00 ... 03

Diese Signale wirken in der Betriebsart **Einrichten**. Beim manuellen Verfahren der Achsen über Tipptaster bzw. über Handrad werden die **Geschwindigkeiten** bzw. **Schrittmaße** aktiv, die der entsprechenden Bit-Kombination zugeordnet sind. Die Geschwindigkeiten werden in den Maschinenparametern AXSP\Dr[x]\AxFun\Jog\Vel\... definiert.

Beim Verfahren der Achsen über ein Handrad werden die **Schrittmaße als Bewertungsfaktor** für die Skalenteilung des Handrades ausgewertet.

Die Schrittmaße für **x Inkremente** werden im Maschinenparameter **/AX/Dr[1]/Jog/VarDistAxJog** "Variabler Inkrementen-Schritt" (1015 00002) festgelegt.

Handvorschub/ Schrittmaß	Maschinenparameter	Bit 3 (qAx_Man- Feed_03)	Bit 2 (qAx_Man- Feed_02)	Bit 1 (qAx_Man- Feed_01)	Bit 0 (qAx_Man- Feed_00)
keine Vorgabe	-	0	0	0	0
Eilgang	/AX/Dr[1]/Jog/Vel/RapidAx-Vel "Handvorschub Eilgang" (1005 00006)	0	0	0	1
Schnell	/AX/Dr[1]/Jog/Vel/FastAxVel "Handvorschub schnell" (1005 00005)	0	0	1	0
Mittel	/AX/Dr[1]/Jog/Vel/MediumAxVel "Handvorschub mittel" (1005 00004)	0	0	1	1
Langsam	/AX/Dr[1]/Jog/Vel/SlowAxVel "Handvorschub langsam" (1005 00003)	0	1	0	0
x Inkremente	/AX/Dr[1]/Jog/Vel/VarDistAxVel "Handvorschub für variable Schritte" (1005 00008)	0	1	0	1
1000 Inkremente	/AX/Dr[1]/Jog/Vel/DefStepAxVel "Handvorschub für definierte Schritte" (1005 00007)	0	1	1	0
100 Inkremente		0	1	1	1
10 Inkremente		1	0	0	0
1 Inkrement		1	0	0	1

Tab. 8-6: Handvorschub Bit 0 ... Bit 3 - Schrittmaß

Referenzschalter qAx_HomeSwitch

Überträgt das Signal des Referenzschalters zum Antrieb, wenn dieser nicht direkt am Antrieb verdrahtet ist. Voraussetzung ist, dass dies im Antrieb entsprechend konfiguriert ist.



- Das Signalsteuerwort S-0-0145 muss im Master-Daten-Telegramm (S-0-1050.0.6) enthalten sein
- Bit 0 von S-0-0400 muss im Signal-Steuerwort (S-0-0145) konfiguriert sein

Signalpegel 1: Der Referenzschalter ist betätigt.

0: Der Referenzschalter ist nicht betätigt.

Achs-Override 100% qAx_Override100

Dieses Signal blendet den Achs-Override einer asynchronen Achse aus. Die Achse verfährt dann mit der vorgewählten bzw. programmierten Geschwindigkeit. Das Signal bezieht sich sowohl auf programmierte asynchrone Bewegungen und auf externe Sollvorgabe, als auch auf den Einrichtbetrieb (Tippen).

Das Signal hat keinen Einfluss auf Bewegungen, die vom Antrieb interpoliert werden (z. B. Referenzpunkt anfahren).

Signalpegel 1: Der Achs-Override ist ausgeblendet.
0: Der Achs-Override ist wirksam.

Override Bit 0 ... Bit 15 qAx_Override_00 ... 15

Die Verfahrgeschwindigkeit von asynchronen Achsen kann über die **Override-Funktion** beeinflusst werden. Sie wirkt auf die im **Teileprogramm** programmierten Geschwindigkeiten und auf die vorgewählten Geschwindigkeiten im **Einrichtbetrieb**.

Mit diesem Signal wird jeweils eine Bewertungsstufe des Overrides für eine Hilfsachse aktiviert, sofern das Override weder über Interface (qAx_Override100) noch über Maschinenparameter ausgeblendet wurde.

Die Stufen des Override-Schalters werden in **binär codierter** Form von der NC abgefragt. Jedem Binärcode wird über Maschinenparameter **/CHAN/Ch[1]/Path/Ovrd/ChOvrdStep[1]**, **/AX/AsynchrAxOvrdIf/AxOvrdStep[1]** "Vorschub-Override für Kanal" (7030 00010) von **Kanal 0 (!)** ein Wert zugewiesen, der die jeweilige Geschwindigkeit als Prozentwert darstellt.

Kundeneingang 1 ... 8 qAx_Custom1 ... 8

NC-seitig konfigurierbares Interface, das je nach Bedarf belegt werden kann.
(Diese Möglichkeit ist z. Z. noch nicht aktiv)

Stillstandsfehler ausblenden qAx_TrqErrOff

Dieses Signal unterdrückt bei Überschreitung des Stillstandsmoments einen internen Fehlerzustand. In diesem Fall muss in der SPS eine Überwachung des Stillstandsmoments programmiert werden.



Unabhängig von diesem Signal wird bei Überschreitung des Stillstandsmoments das Signal "Stillstandsfehler" (iAx_TrqExceed) gesetzt!

Signalpegel 1: Bei Überschreitung des Stillstandsmoments erzeugt die NC **keinen** internen Fehlerzustand. Die Überschreitung wird toleriert.
0: Fehlerzustand bei Überschreiten des Stillstandsmoments.

Kopplungsfehler ausblenden qAx_LagErrOff

Dieses Signal unterdrückt für Gantry-Achsen bei Koppelnachlaufüberschreitung einen internen Fehlerzustand. In diesem Fall muss in der SPS eine Überwachung des Koppelnachlaufs programmiert werden.



Unabhängig von diesem Signal wird bei Koppelnachlaufüberschreitung das Signal "Nachlauffehler" (iAx_CoupleLag) gesetzt!

Signalpegel	1:	Bei Koppelnachlaufüberschreitung erzeugt die NC keinen internen Fehlerzustand. Die Überschreitung wird toleriert.
	0:	Koppelnachlaufüberwachung erzeugt Fehler.

Gantry auf Masterposition qAx_MasterPos



VORSICHT

Das Signal darf nur gesetzt werden, wenn die Geber in Betrieb genommen sind.

Sonst kann es zu mechanischen Schäden kommen.

Über dieses Signal wird bei einem Gantry-Slave die Sollposition der Master-Achse ohne Versatz als Sollposition der Slave-Achse übernommen. Das Signal wirkt auf die Slave-Achse.

Voraussetzungen:

- MasterAchse und Slave-Achse haben bekannte Lageistwerte .
- Differenz der Positionen kleiner als der in den Maschinenparametern konfigurierte maximale Schleppabstand.
- Das Signal wirkt auch bei inkrementellen Messsystemen mit abstandskodierten Gebern, wenn das Referenzieren mit "Strecke fahren" eingestellt ist.
- *Das Signal wirkt unter folgenden Bedingungen:*
 - Antrieb in Referenz.
 - Maximaler Schleppabstand nicht überschritten oder Schleppabstandsüberwachung deaktiviert.

Bei zyklisch absoluten Gebern sind die Achspositionen vor dem Referenzieren nicht bekannt. Deshalb wird aus Sicherheitsgründen der Abstand zwischen den Gantry-Achsen, der aus den von den Gebern gelieferten Lage-Istwerten gebildet wird, bis zum Referenzieren eingefroren.

Bei Achsen mit Absolutwertgebern sind dagegen die Lage-Istwerte auch ohne Referenzieren bekannt. Sind die Geber in Betrieb genommen, kann aus den von ihnen gelieferten Werten auf eine Schräglage geschlossen werden.

Signalflanke	0 → 1:	Die Schräglage der Slave-Achse wird interpolierend ausgeglichen, sobald die Antriebe in Betrieb sind und alle Voraussetzungen erfüllt sind. Sind die Voraussetzungen nicht erfüllt, so kommt es zu einer Warnung (s. Funktionshandbuch MTX Erweiterung) und die Flanke wird verworfen. Nach der Korrektur der Voraussetzungen muss die Flanke erneut vorgegeben werden.

Istwert-Versatz übernehmen qAx_TakeActOffs

Übernahme des Istwert-Versatzes dieser Slaves als Sollwert-Versatz.

Signalflanke	0 → 1:	Übernahme des Istwert-Versatzes bei offenem Regler.

Der Sollwert-Versatz eines Gantry-Slaves wird von der Steuerung aus dem aktuellen Istwert-Versatz gebildet und nachgezogen, solange der Regler offen ist. Beim Schließen des Reglers wird der Wert eingefroren.

Das Signal muss für die Slave-Achse gesetzt werden.

Verzögerte Fehlerreaktion abkürzen qAx_RedDelayErrClass1

Dieses Signal steht in Zusammenhang mit iAx_DelayErrClass1 (Verzögerte Fehlerreaktion).

Dieses Signal bietet der SPS die Möglichkeit die Zeit bis zum Stillsetzen der Achse durch die NC zu verkürzen.



Weitere Informationen sind dem Kapitel "Verzögerte NC-Fehlerreaktion bei nichtfatalen Antriebsfehlern" im Handbuch "Funktionsbeschreibung" zu entnehmen.

Signalfanke	0 → 1:	Die verzögerte Fehlerreaktion wird beendet. Die NC stoppt die Achse.

Achse abgebaut qAx_Discharge

Mit Setzen dieses Signals wird die Achse nicht mehr innerhalb der Inpos-Logik überprüft. Die Achse kann trotzdem in der Konfiguration eines Kanals verbleiben, wenn zusätzlich das Signal **Antrieb Ein** nicht gesetzt ist.

Sollposition halten qAx_FrzlpoPos

Dieses Signal verhindert das Nachführen der Antriebs-Istposition bei geöffnetem Lageregelkreis, d. h., "Antrieb in Betrieb" (iAx_DrvAct) ist nicht gesetzt.

Signalpegel	1:	Die Istposition des Antriebs wird bei geöffnetem Lageregelkreis nicht in die NC übernommen.
	0:	Die NC übernimmt die Istposition des Antriebs bei geöffnetem Lageregelkreis.

Momentenreduktion qAx_TrqLi

Mit Hilfe dieses Signals kann der, im Antrieb gültige Momentengrenzwert (**Sercos Parameter S-0-0092**) verändert werden.

Dieses Signal hat insbesondere Bedeutung im Zusammenhang mit klemmbaren Achsen; so kann während der Klemmung die Lageregelung geschlossen bleiben, das Moment des Antriebs jedoch reduziert werden, um die mechanische Beanspruchung zu minimieren. Hierdurch wird ein mechanisches Verrücken der Achse beim Schließen bzw. Öffnen der Klemmung weitgehend vermieden.

Der Wert des veränderten Momentengrenzwertes wird über den Maschinenparameter **RedMaxTorq** "Reduziertes Drehmoment bei aktiver Momentreduktion" (1003 00010) achsweise eingestellt und lässt sich außerdem über die NC-Funktion RedTorque (RDT) programmieren.

Eine Momentenumschaltung ist nur bei Stillstand der Achse zulässig.

Verweigert der Antrieb die Annahme des Momentengrenzwertes (z. B. Parameter ist schreibgeschützt), erscheint keine Fehlermeldung. Das Signal "Moment reduziert" (iAx_TrqLim) wird jedoch nicht gesetzt.

Signalfanke	0 → 1:	Über die Sercos Interface-Schnittstelle wird das reduzierte Drehmoment zum Antrieb übertragen.
	1 → 0:	Der originale Momentengrenzwert aus dem Sercos Parameter S-0-0092 wird zum Antrieb übertragen.

Antrieb Ein qAx_DrvOn

Dieses Signal schaltet im Antrieb das Drehmoment ein, wenn er betriebsbereit ist (iAx_DrvReady gesetzt). Der Regelkreis ist geschlossen.

Das Rücksetzen von **Antrieb Ein** leitet einen Ausschaltvorgang ein. Ist die Achse zu diesem Zeitpunkt noch in Bewegung, wird sie mit der voreingestellten Bremsrampe zum Stillstand gebracht.

Im Stillstand wird der Lageregelkreis geöffnet und das Signal **Antrieb in Betrieb** zurückgesetzt. Der Antrieb ist jetzt drehmomentfrei.

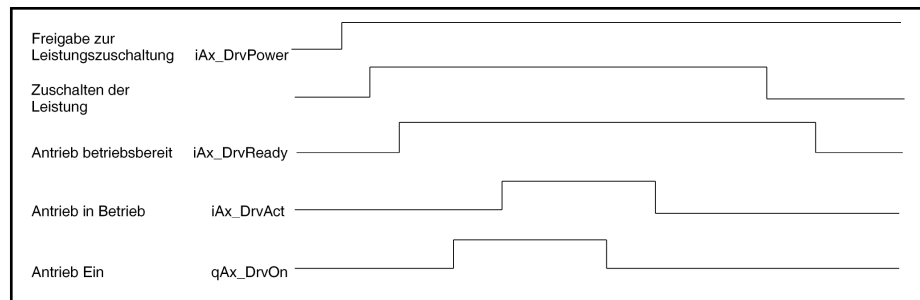


Abb. 8-2: Antrieb Ein

Beispiel:**Funktionsweise des Ein- und Ausschaltvorgangs**

Nach dem Einschalten der Maschine bzw. Anlegen der Spannung durchlaufen NC-Steuerung, Servo- und Spindeltrieb eine Initialisierungsphase.

Während der Initialisierungsphase der einzelnen Gerätekomponenten wird u. a. die digitale Schnittstelle zur Kommunikation zwischen NC-Steuerung und Antrieben aufgebaut.

Diese digitale Schnittstelle entspricht den Festlegungen des **Sercos Interface**.

Die Initialisierung des **Sercos Interface** erfolgt in 4 Phasen systemintern und automatisch. Nachdem die Schnittstelle initialisiert und der zyklische Betrieb aktiv ist, wird das Signal **Freigabe zur Leistungszuschaltung** ausgegeben.

Die Leistung an den betreffenden Antrieben kann zugeschaltet werden. Nachdem die Leistung zugeschaltet wurde, wird das Signal **Antrieb betriebsbereit** ausgegeben.

Mit Setzen von **Antrieb Ein** wird der Lageregelkreis geschlossen. Der Antrieb steht unter Drehmoment und das Signal **Antrieb in Betrieb** wird ausgegeben.

Das Einleiten des Ausschaltvorgangs bzw. Abschalten der Leistung erfolgt, indem das SPS-Ablaufprogramm das Signal **Antrieb Ein** zurücksetzt.

Ist die Achse zu diesem Zeitpunkt noch in Bewegung, wird sie mit der voreingestellten Bremsrampe zum Stillstand gebracht.

Im Stillstand wird der Lageregelkreis geöffnet und das Signal **Antrieb in Betrieb** zurückgesetzt. Der Antrieb ist jetzt drehmomentfrei.

Im nächsten Schritt kann die Leistung abgeschaltet werden.

Das Signal **Freigabe zur Leistungszuschaltung** bleibt anstehen, solange die Gerätekomponenten (NC und Antriebe) weiterhin initialisiert sind.

Vorschub Sperre qAx_DrvLock

Dieses Signal wirkt in jeder Betriebsart, wenn der Regelkreis geschlossen ist und die Achse Drehmoment hat.

Es verhindert, dass eine programmierte oder manuell vorgegebene Achsbewegung ausgeführt wird, hat jedoch keinen Einfluss auf die Eingangssignale Fahrbefehl, Negative Fahrriichtung und Achse In Position.

Signalpegel 1: Die Achse kann nicht verfahren werden oder sie wird mit der parametrisierten Bremsbeschleunigung gestoppt, wenn eine Bewegung bereits aktiv ist.

Ist **Vorschub Sperre** für eine Achse gesetzt, die mit anderen Achsen an einer Interpolation teilnimmt, können die anderen Achsen ebenfalls nicht verfahren werden.

0: Der Vorschub einer Achse ist nicht gesperrt. Wurde über das NC-Teileprogramm oder manuell ein Kommando zum Verfahren der Achse gegeben, kann die Achse unmittelbar verfahren.

Wurde eine fahrende Achse zuvor mit **Vorschub Sperre** zum Stillstand gebracht, beschleunigt sie mit Rücksetzen wieder auf die programmierte Geschwindigkeit.

Beispiel:

Vorschub Sperre

Die Maschine ist eingeschaltet und der Regelkreis ist bereits geschlossen. Über das NC-Teileprogramm ist für eine Achse eine negative Verfahrbewegung programmiert.

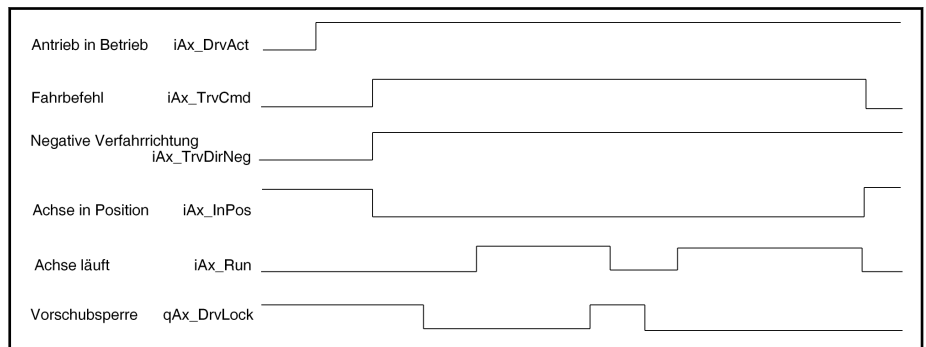


Abb. 8-3: Antrieb in Betrieb

Mit dem Signal **Antrieb in Betrieb** wird dem SPS-Ablaufprogramm mitgeteilt, dass der Regelkreis geschlossen ist.

Fahrbefehl und **Neg. Verfahrrichtung** werden mit Ausführung des NC-Satzes, in dem die Verfahrbewegung programmiert ist, ausgegeben. Die Verfahrrichtung ist nur dann aktuell, wenn ein Fahrbefehl ansteht.

Das Signal **Achse in Position** wird mit Aktivieren des Fahrbefehls zurückgesetzt.

Solange **Vorschub Sperre** gesetzt ist, wird die Verfahrbewegung noch nicht ausgeführt.

Mit Rücksetzen von Vorschub Sperre wird die programmierte Verfahrbewegung ausgeführt und **Achse läuft** gesetzt.

Wird während der Verfahrbewegung **Vorschub Sperre** wieder gesetzt, wird die Verfahrbewegung gestoppt und **Achse läuft** zurückgesetzt, wenn die Achse zum Stillstand gekommen ist.

Mit Rücksetzen der **Vorschub Sperre** beschleunigt die Achse auf die zuvor gültige Geschwindigkeit, **Achse läuft** wird gesetzt und die Verfahrbewegung wird zu Ende geführt.

Nach Erreichen der programmierten Endposition wird der **Fahrbefehl** und **Achse läuft** zurückgesetzt und das Signal **Achse In Position** gesetzt.

8.2.2 Eingangssignale (NC --> SPS)

Referenzpunkt bekannt iAx_RefKnown

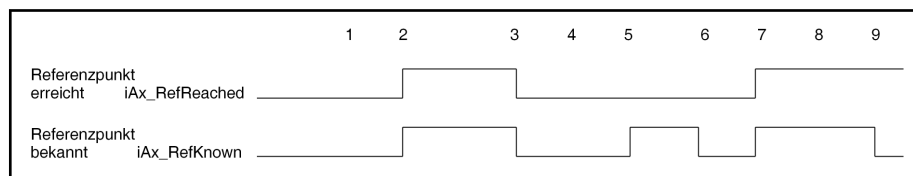
Dieses Signal wird gesetzt, sobald der Referenzpunkt der Achse angefahren wird, bzw. bei einem Absolutwertgeber dieser von der Steuerung erkannt wird. Die Steuerung wertet dazu den Sercos Parameter **Status Lageistwert** S-0-0403 aus. Solange das Signal **Referenzpunkt bekannt** noch nicht gesetzt wurde, überwacht die NC noch keine Endschalter.



Nachdem dieses Signal einmal gesetzt wurde, wird es nur dann wieder zurückgesetzt, wenn ein neuer Sercos Phasenhochlauf ausgelöst wird, oder eine Störung des Antriebs (z. B. Geber defekt) vorliegt.

Soll das Signal zu jeder Zeit den aktuellen Zustand von **S-0-403, Status Lageistwert** anzeigen, dann muss Bit 0 von S-0-403 zyklisch via S-0-144 vom Antrieb in die NC übertragen werden.

- Signalpegel**
- 1:** Der Referenzpunkt wurde angefahren oder die Achse ist mit Absolutwertgebern ausgestattet.
- 0:** Es wurde noch kein Referenzpunkt angefahren.



- 1 Start Referenzpunkt Anfahren
- 2 Referenzpunkt Anfahren beendet
- 3 Start Referenzpunkt Anfahren
- 4 Abbruch von Referenzpunkt Anfahren (Referenzpunkt wurde vom Antrieb schon erkannt)
- 5 Grundstellung
- 6 Start Referenzpunkt Anfahren
- 7 Referenzpunkt Anfahren beendet
- 8 Antrieb verliert durch Fehler den Referenzpunkt
- 9 Grundstellung

Abb. 8-4: Referenzpunkt

Referenzpunkt erreicht iAx_RefReached

Im Gegensatz zum Signal **Referenzpunkt bekannt**, wird dieses Signal nur dann gesetzt, wenn eine Referenzpunktfahrt erfolgreich bis zum Ende ausgeführt wurde. Bei abgebrochener Referenzpunktfahrt werden beide Signale nicht gesetzt.

Während eine anschließende Grundstellung den Status der Lageistwerte im Antrieb prüft (Parameter S-0-0403) und gegebenenfalls das Signal **Referenzpunkt bekannt** setzt (oder löscht), bleibt das Signal **Referenzpunkt erreicht** unverändert.

Zu Beginn einer jeden Referenzpunktfahrt wird das Signal zurückgesetzt.

- Signalpegel**
- 1:** Die letzte Referenzpunktfahrt endete ordnungsgemäß mit Erreichen des Referenzpunktes.
- 0:** Es wurden entweder bisher noch keine Referenzpunkte angefahren, oder es ist noch eine Referenzpunktfahrt aktiv und der Referenzpunkt wurde bisher noch nicht erreicht, oder die letzte Referenzpunktfahrt wurde abgebrochen.

Fahrbehl iAx_TrVCmd

Die Signale **Fahrbehl**, **Neg. Verfahrriichtung**, **Achse läuft** und **Achse in Position** beeinflussen sich wechselseitig.

In den Signaldiagrammen und Erläuterungen zu den Signalen **Manuell+/-** (**Manuell+ qAx_JogPlus**, Seite 71) und (**Vorschub Sperre qAx_DrvLock**, Seite 82) sind die Zusammenhänge beispielhaft dargestellt.

In Verbindung mit dem Signal **Fahrbehl** wird das Signal **Neg. Verfahrriichtung** ausgegeben.

Signalpegel	1:	Die Vorgabe für eine Verfahrbewegung steht an. Sie kann manuell oder über das NC-Teilprogramm erfolgen.
	0:	Es liegt kein Kommando für eine Verfahrbewegung vor.

Negative Verfahrrichtung iAx_TrDirNeg

Dieses Signal wirkt nur in Zusammenhang mit dem Signal **Fahrbehl**. Ist kein Fahrbehl gesetzt, hat dieses Signal keine Bedeutung.

Signalpegel	1:	"Negative Verfahrrichtung" und "Fahrbehl" gesetzt bedeutet, dass die betreffende Achse eine Verfahrbewegung in die negative Richtung ausführen soll oder die letzte Bewegung in negative Richtung ausgeführt hat.
	0:	"Negative Verfahrrichtung" nicht gesetzt und "Fahrbehl" gesetzt bedeutet, dass die betreffende Achse eine Verfahrbewegung in die positive Richtung ausführen soll oder die letzte Bewegung in positiver Richtung ausgeführt hat.

Achse läuft iAx_Run

Die Signale **Fahrbehl**, **Negative Verfahrrichtung**, **Achse läuft** und **Achse in Position** beeinflussen sich wechselseitig.

In den Signaldiagrammen und Erläuterungen zu den Signalen **Manuell+/-** (**Manuell+ qAx_JogPlus**, Seite 71) und (**Vorschub Sperre qAx_DrvLock**, Seite 82) sind die Zusammenhänge beispielhaft dargestellt.

Ist NC-seitig die Verfahrbewegung einer Achse abgeschlossen, steht kein **Fahrbehl** mehr an und es wird entweder das Signal **Achse in Position** oder **Achse läuft** ausgegeben.

Signalpegel	1:	"Achse läuft" bleibt solange gesetzt, wie die Achse eine programmierte oder manuell vorgegebene Bewegung ausführt. Beim antriebsgeführten Referenzpunkt anfahren wird das Signal nicht gesetzt, selbst wenn die Achse noch in Bewegung ist.
	0:	Die Achse führt keine Bewegung aus.

Achse in Position iAx_InPos

Das Signal ist aktiv, wenn der Regelkreis geschlossen ist und die Achse unter Drehmoment steht.

Signalpegel	1:	Achse befindet sich im parametrierten Inpositionsfenster und es steht kein Fahrbehl an. Das Inpositionsfenster wird mit dem Sercos Parameter S-0-0057 festgelegt. Das Fenster "In Position grob" wird mit dem Sercos Parameter S-0-0261, für DiAx/EcoDrive und HNC100 mit dem Maschinenparameter /NCO/InPosWin/Dr[1]/InPosRough "In-Positions-Fenster" (1015 00100) festgelegt.
	0:	Die Achse befindet sich nicht im parametrierten Inpositionsfenster.

Achse auf Rasterposition iAx_NotchPos

Mit diesem Signal wird angezeigt, wenn die Hirthachse eine Rasterposition erreicht hat.

Dieses Signal ist immer 0, solange der Referenzpunkt nicht angefahren wurde. Das Signal wird auf 1 gesetzt, sobald die Achse sich im angewählten Inpositionsfenster befindet.

Signalpegel	1:	Hirthachse befindet sich im angewählten Inpositionsfenster.

- 0: Zeigt an, dass die Hirthachse noch keinen Referenzpunkt angefahren hat.

Achse grundgestellt iAx_Reset

Dieses Signal zeigt an, dass der Grundstellungsauftrag von der asynchronen Achse angenommen wurde, und dass die asynchrone Achse wieder bereit ist, neue Bewegungsvorgaben anzunehmen.

Achse vor Endpunkt iAx_DistCtrl

Dieses Signal zeigt an, dass sich die Achse in einem über Maschinenparameter `/SFU/AxDistCtrl/Dr[1]/EnablAxDistCtrl`, `/SFU/AxDistCtrl/Dr[1]/DistEndPoint` "Distanz vor Endpunkt" (1015 00010) eingestellten Bereich vor ihrem Endpunkt befindet.

- Signalpegel**
- 1: Achssollposition innerhalb der angegebenen Distanz.
- 0: Achssollposition außerhalb der angegebenen Distanz oder nicht konfiguriert oder Referenzpunkt Anfahren aktiv oder Handrad aktiv.

Achsgeschwindigkeit erreicht iAx_ProgSpReach

- Signalpegel**
- 1: Die asynchrone Achse hat die programmierte Drehzahl bzw. Modulo-Geschwindigkeit `<Achse>=S<Wert>` erreicht. Die Stellung des Override-Schalters wird berücksichtigt.
- 0: Die Drehzahl bzw. Modulogeschwindigkeit der asynchronen Achse ist noch nicht erreicht oder die Geschwindigkeitsinterpolation wurde mit `<Achse>=S0` oder einem Positioniervorgang abgebrochen.

Festanschlag erreicht iAx_FxStopReached

Dieses Signal wird ausgegeben, solange die Achse am Festanschlag steht.

Das Signal wird zurückgesetzt, wenn die Achse das Überwachungsfenster Festanschlag (Maschinenparameter `/MPS/Dr[1]/MaxDevPoStop` "Überwachungsfenster Festanschlag" (1003 00032)) verlässt oder der Festanschlag mit "FsReset" (FSR) aufgehoben wird.

- Signalpegel**
- 1: Achse hat den Festanschlag erreicht.
- 0: Festanschlag verlassen oder aufgehoben.

Festanschlag aktiv iAx_FxStopAct

Dieses Signal wird ausgegeben, solange die Verfahrbewegung zum Festanschlag aktiv ist.

Eine Bewegung zum Festanschlag wird durch die Funktionen "FsProbe" (FSP, Messen an Festanschlag) bzw. "FsMove" (FSM, Fahren auf Festanschlag) ausgelöst.

- Signalpegel**
- 1: Achse fährt zum Festanschlag.
- 0: Keine Verfahrbewegung zum Festanschlag aktiv.

Si-Betriebsart Bit 0 ... Bit 3 iAx_SafOpMode_00 ... 03

Diese Signale sind Bestandteil der Sicherheitstechnik der Antriebsbaureihe IndraDrive.

Die NC liest die angewählte Betriebsart des Antriebs (P-0-3215 oder P-0-3262) und gibt die Information über diese Ausgänge an die SPS weiter.



Die sicheren Bewegungen SMM14 bis SMM16 werden im Bitinterface nicht angezeigt.

Sicherheitsfunktion	Si-Betriebsart			
	Bit 3 (iAx_SafOpMode_03)	Bit 2 (iAx_SafOpMode_02)	Bit 1 (iAx_SafOpMode_01)	Bit 0 (iAx_SafOpMode_00)
Normalbetrieb NO	0	0	0	0
Sicherer Stopp 1 (NOT-HALT) (SS1ES)	0	0	0	1
Sicherer Stopp 1 (SS1) / Sicherer Stopp 2 (SS2)	0	0	1	0
Sichere Bewegung (SMM1)	0	0	1	1
Sichere Bewegung (SMM2)	0	1	0	0
Sichere Bewegung (SMM3)	0	1	0	1
Sichere Bewegung (SMM4)	0	1	1	0
Sichere Bewegung (SMM5)	0	1	1	1
...				
Sichere Bewegung (SMM13)	1	1	1	1

Tab. 8-7: Si-Betriebsart

Status sichere Lage iAx_SafStatePos

Dieses Signal ist Bestandteil der Sicherheitstechnik der Antriebsbaureihe IndraDrive.

Als Voraussetzung für die Sicherheitsfunktion "Sichere Absolutlage" meldet der Antrieb über dieses Signal, ob sicher referenziert wurde.

Signalpegel	1:	Der Antrieb ist sicher referenziert.
	0:	Keine sichere Referenzierung.

Statusausgang Regler iAx_SafCtrlOutputState

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive.

Über dieses Signal gibt der Antrieb Rückmeldung über seinen sicheren Zustand.

Signalpegel	1:	Sicherer Zustand des Antriebs.
	0:	Kein sicherer Zustand des Antriebs.

Positionsschaltpunkt 1 ... 8 iAx_PosSwitch1 ... 8

Mit diesem Signal wird die aktuelle Position einer Achse überwacht. Maximal 8 Achsüberwachungspositionen (elektrische Endschalter) lassen sich einer Achse zuordnen.

Mit Maschinenparameter SFU\WayPoint\Point[x] (/SFU/WayPoint/Point[1]/DrIndWayPoint, /SFU/WayPoint/Point[1]/MonTypeWayPoint "Überwachte Achse" (2010 00100), /SFU/WayPoint/Point[1]/PosWayPoint "Position des Punkts" (2010 00110), /SFU/WayPoint/Point[1]/IfSigWayPoint "Interfacesignal des Punkts" (2010 00120), /SFU/WayPoint/Point[1]/MonTypeWayPoint "Zyklische Überwachung bei Rundachsen" (2010 00130)) können insgesamt bis zu 64 Achsüberwachungspositionen für das ganze System definiert werden.

Durch **Kombination** zweier Signale können "Überwachungsbereiche" erzeugt werden.

Weitere Informationen siehe Handbuch "Maschinenparameter". Im Kapitel 1 Zu dieser Dokumentation sind zusätzliche Dokumentationstitel aufgeführt.

- Signalpegel**
- 1:** Die Achs-Istposition ist größer oder gleich der Überwachungsposition.
- 0:** Die Achs-Istposition ist kleiner als die Überwachungsposition.

Kanalnummer Bit 0 ... Bit 5 iAx_ChIndex_00 ... 05

Mit diesem Signal wird die binär codierte Kanalnummer des zugehörigen Kanals ausgegeben.

"0" wird ausgegeben, wenn die Achse keinem Kanal zugeordnet ist (z. B.: asynchrone Achse).

Kanalnummer	Bit 5 (iAx_ dex_05)	ChIn- dex_04)	Bit 4 (iAx_ dex_04)	ChIn- dex_03)	Bit 3 (iAx_ dex_03)	ChIn- dex_02)	Bit 2 (iAx_ dex_02)	ChIn- dex_01)	Bit 1 (iAx_ dex_01)	ChIn- dex_00)	Bit 0 (iAx_ dex_00)
kein Kanal	0		0		0		0		0		0
Kanal 1	0		0		0		0		0		1
...	...		...		...		...		...		...
Kanal 12	0		0		1		1		0		0
...	...		...		...		...		...		...
Kanal 60	1		1		1		1		0		0

Tab. 8-8: Kanalnummer Bit 0 ... 5

An einen anderen Kanal verliehene Achsen werden auf folgende Art ausgegeben:

Zustand	Kanal-Information
Kanal wird mit einer Achse aktiv (bei Programm-anwahl oder Handeingabe)	Nummer dieses Kanals
Kanal wird mit einer Achse inaktiv (bei Program-mabwahl oder Ende der Handeingabe)	Nummer dieses Kanals
Kanal gibt Achse mit RemAxis frei und Achse ist im Ursprungskanal nicht freigegeben	Nummer des Ursprungskanals
Kanal gibt Achse mit RemAxis frei und Achse ist im Ursprungskanal freigegeben	0

Tab. 8-9: Kanalausgabe

Begriffe:

- Verliehene Achse:** Verliehen ist eine Achse, wenn ein Kanal auf eine "Achse aus einem inaktiven Kanal" zugreift, ohne dass dieser die Achse mit "RemAxis" (RAX) freigegeben hat.
- Ursprungskanal:** Nach dem Steuerungshochlauf haben alle synchronen Achsen einen Ursprungskanal.

Achs-Override 0% iAx_Override0

Dieses Signal zeigt für eine asynchrone Achse an, dass ihr Achs-Override auf 0% geschaltet ist und kein Override 100% über Interface (qAx_Override100) angewählt ist.

Signalpegel	1:	Auf die Achse wirken 0%.
	0:	Auf die Achse wirkt ein Wert ungleich 0%.

Achs-Override 100% iAx_Override100

Dieses Signal zeigt für eine asynchrone Achse an, dass ihr Override auf 100% geschaltet ist und/oder Override 100% über Interface (qAx_Override100) angewählt ist.

Signalpegel	1:	Auf die Achse wirkt ein Wert von 100%.
	0:	Auf die Achse wirken keine 100%.

SCS-Signalstatus 0 ...15 iAx_ScsState00 ... 15

Im Signalstatuswort (Sercos Parameter S-0-0144) werden die konfigurierten Bit-Signale aus dem zyklischen Telegramm des Achsantriebs in die NC übertragen.

Die Statussignale lassen sich im Parameter S-0-0026 konfigurieren. Für die Übertragung der Signale im zyklischen Telegramm muss der Parameter S-0-0144 in der Konfigurationsliste des Antriebstelegramm (S-0-0016) aufgenommen werden.

Beispiel:

Konfiguration:

- S-0-0026 = (330, 331, 0, 333, 0, 335, 336, 337, 0, 0, 0, 0, 0, 0, 0):
Konfiguration des Signalstatuswortes
- S-0-0016 = (40, 144):
Konfiguration des Antriebs-Telegramms

Damit werden den einzelnen Signalen folgende Statusmeldungen zugeordnet:

- iAx_ScsState00: S-0-0330 Meldung $n_{ist} = n_{soll}$
- iAx_ScsState01: S-0-0331 Meldung $n_{ist} = 0$
- iAx_ScsState02: -
- iAx_ScsState03: S-0-0333 Meldung $M_d \geq M_{dx}$
- iAx_ScsState04: -
- iAx_ScsState05: S-0-0335 Meldung $n_{soll} > n_{Grenz}$
- iAx_ScsState06: S-0-0336 Meldung In-Position
- iAx_ScsState07: S-0-0337 Meldung $P \geq P_x$
- iAx_ScsState08: -
- iAx_ScsState09: -
- iAx_ScsState10: -
- iAx_ScsState11: -
- iAx_ScsState12: -
- iAx_ScsState13: -
- iAx_ScsState14: -

- iAx_ScsState15: -



Eine ausführliche Beschreibung aller zur Verfügung stehenden Statusmeldungen siehe "Sercos Interface-Spezifikation".

Kundenausgang 1 ... 8 iAx_Custom1 ... 8

NC-seitig konfigurierbares Interface, das je nach Bedarf belegt werden kann.
(Diese Möglichkeit ist z. Zt. noch nicht aktiv)

Index Master-Achse Bit 0 ... Bit 7 iAx_MasterAxindex_00 ... 07

Jede Achse erhält die binär codierte Achsnummer ihrer Master-Achse, wobei gilt:

- Ist die Achse nicht gekoppelt, hat die Nummer der Master-Achse den Wert 0.
- Ist die Achse selbst Master, so ist die Nummer der Master-Achse identisch der Achsnummer.

Stillstandsfehler iAx_TrqExceed

Mit diesem Signal wird ein vorgegebenes Stillstandsmoment für eine Achse überwacht.



Bei Gantry-Achsen bezieht sich das Signal auf die vorzeichenunabhängige Summe der Stillstandsmomente aller Achsen eines Verbands und wird nur für die Master-Achse gesetzt.

Signalpegel	1:	Das vorgegebene Stillstandsmoment für eine Achse ist überschritten.
	0:	Das Stillstandsmoment für eine Achse ist nicht überschritten.

Nachlauffehler iAx_CoupleLag

Mit diesem Signal wird eine Überschreitung des Koppelabstands einer Gantry-Achse gemeldet.

Signalpegel	1:	Der maximale Koppelabstand für eine Achse ist überschritten.
	0:	Der maximale Koppelabstand für eine Achse ist nicht überschritten.

Gantry-Sollwert-Versatz aktiv iAx_CmdOffsExst

Signalpegel	1:	Für diesen Slave existiert ein Sollwert-Versatz.

Ausgleichbarer Gantry-Sollwert-Versatz überschritten iAx_CmdOffsExceed

Signalpegel	1:	Für diesen Slave ist der ausgleichbare Sollwert-Versatz überschritten.

Verzögerte Fehlerreaktion iAx_DelayErrClass1

Dieses Signal steht in Zusammenhang mit iAx_DrvErrClass1 (Fehler Zustandsklasse-1).

Bei aktivierter Funktion "Verzögerte NC-Fehlerreaktion bei nichtfatalen Antriebsfehlern" wird dieses Signal zusammen mit iAx_DrvErrClass1 gesetzt.



Weitere Informationen sind dem Kapitel "Verzögerte NC-Fehlerreaktion bei nichtfatalen Antriebsfehlern" im Handbuch "Funktionsbeschreibung" zu entnehmen.

Signalpegel	1:	Ein nichtfataler Antriebsfehler ist aufgetreten und der Antrieb räumt der SPS und der NC Zeit zum koordinierten Stillsetzen der Bewegung ein.
	0:	Es ist keine verzögerte Fehlerreaktion aktiv oder die verzögerte Fehlerreaktion ist beendet.

Fehler Zustandsklasse-1 iAx_DrvErrClass1

Dieses Signal ist ein direktes Abbild des entsprechenden Signals im Sercos Antriebs-Statuswort der betreffenden Achse (der Antrieb schaltet ab und wird verriegelt).

Die Verriegelung kann nur durch das Kommando "Reset Zustandsklasse-1", das bei **Grundstellung** automatisch von der NC gesetzt wird, wieder aufgehoben werden.

Eine Auswertung des anstehenden Fehlers kann manuell über den Sercos Monitor, oder auch automatisch durch direktes Lesen durch die SPS erfolgen.

Der Parameter S-0-0011 besteht aus 2 Byte und hat folgenden Aufbau:

Bit 0:	reserviert	
Bit 1:	Verstärker-Abschalttemperatur	S-0-0203
Bit 2:	Motor-Abschalttemperatur	S-0-0204
Bit 3:	reserviert	
Bit 4:	Steuerspannungsfehler	
Bit 5:	Feedbackfehler (Geber, Messsystem)	
Bit 6:	Fehler im elektronischen Kommutierungssystem	
Bit 7:	Überstrom	
Bit 8:	Überspannung	
Bit 9:	Unterspannungsfehler	
Bit 10:	Phasenfehler der Leistungsverorgung	
Bit 11:	exzessive Reglerabweichung	S-0-0159
Bit 12:	Kommunikationsfehler	
Bit 13:	Lagegrenzwert überschritten	S-0-0049
Bit 14:	reserviert	
Bit 15:	Herstellerspezifischer Fehler	S-0-0129

Tab. 8-10: Aufbau Parameter S-0-0011

Änderung Zustandsklasse-2 iAx_DrvChangeClass2

Dieses Signal ist ein direktes Abbild des entsprechenden Signals im Sercos Antriebs-Statuswort der betreffenden Achse (Abschaltvorwarnung). Mit einer Änderung der Zustandsklasse 2 wird das Bit gesetzt, nach Lesen des Status über den Parameter S-0-0012 wird es wieder zurückgesetzt. Das Lesen kann manuell über den Sercos Monitor, oder auch direkt von der SPS erfolgen.

Der Parameter S-0-0012 besteht aus 2 Byte und enthält folgenden Warnungen:

Bit 0:	Überlastwarnung	
Bit 1:	Verstärker-Übertemperatur-Warnung	
Bit 2:	Motor-Übertemperatur-Warnung	S-0-0201
Bit 3:	reserviert	
Bit 4:	reserviert	
Bit 5:	reserviert	
Bit 6:	reserviert	
Bit 7:	reserviert	
Bit 8:	reserviert	
Bit 9:	Unterspannung im Zwischenkreis	
Bit 10:	reserviert	
Bit 11:	reserviert	
Bit 12:	reserviert	
Bit 13:	reserviert	
Bit 14:	reserviert	
Bit 15:	Herstellerspezifische Warnung	S-0-0095

Tab. 8-11: Warnungen Parameter S-0-0012

Über die Maske der **Zustandsklasse 2** (S-0-0097) können die Abschaltvorwarnungen bezüglich ihrer Wirkung auf das Änderungsbit ausmaskiert werden.

Änderung Zustandsklasse-3 iAx_DrvChangeClass3

Dieses Signal ist ein direktes Abbild des entsprechenden Signals im Sercos Antriebs-Statuswort der betreffenden Achse (Meldungen der Betriebszustände). Mit einer Änderung der Zustandsklasse 3 wird das Bit gesetzt, nach Lesen des Status über den Parameter S-0-0013 wird es wieder zurückgesetzt. Das Lesen kann manuell über den Sercos Monitor, oder auch direkt von der SPS erfolgen.

Der Parameter S-0-0013 besteht aus 2 Byte und enthält folgenden Meldungen:

Bit 0:	$n_{\text{ist}} = n_{\text{soll}}$	S-0-0330
Bit 1:	$n_{\text{ist}} = 0$	S-0-0331
Bit 2:	$ n_{\text{ist}} < n_x $	S-0-0332
Bit 3:	$ M_d \geq M_{d_x} $	S-0-0333
Bit 4:	$ M_d \geq M_{d_{\text{grenz}}} $	S-0-0334
Bit 5:	$ n_{\text{soll}} > n_{\text{grenz}} $	S-0-0335
Bit 6:	In Position	S-0-0336
Bit 7:	$ P \geq P_x $	S-0-0337
Bit 8:	-	

Bit 9:	$ n_{\text{ist}} \geq \text{min. Spindeldrehzahl}$	S-0-0339
Bit 10:	$ n_{\text{ist}} \geq \text{max. Spindeldrehzahl}$	S-0-0340
Bit 11:	In Position grob	S-0-0341
Bit 12:	Zielposition erreicht	S-0-0342
Bit 13:	Interpolator angehalten	S-0-0343
Bit 14:	-	
Bit 15:	Herstellerspezifische Betriebszustände	S-0-0182

Tab. 8-12: Meldungen Parameter S-0-0013

Über die Maske der Zustandsklasse 3 (S-0-0098) können die Betriebszustände bezüglich ihrer Wirkung auf das Änderungsbit ausmaskiert werden.

Moment reduziert iAx_TrqLim

Dieses Signal gibt an, dass die betreffende Achse mit reduziertem Maximalmoment betrieben wird.

Signalpegel	1:	Maximalmoment der Achse ist reduziert.
	0:	Maximalmoment der Achse ist nicht reduziert.

Testbetrieb iAx_DryRun

Dieses Signal ist gesetzt, wenn die Achse im Testbetrieb ist.

Im Testbetrieb werden Achsen zwar weiterhin intern interpoliert, die berechneten Sollwerte gelangen jedoch lediglich zur Anzeige und werden nicht zum Antrieb übertragen. Die Achse wird auf der Position gehalten, auf der sie zum Zeitpunkt der Aktivierung des Testbetriebs stand.

Testbetrieb kann über Softkey, automatisch bei nicht angeschlossenen Antrieben (kein Sercos Ring vorhanden) oder über den Maschinenparameter / **MAIN/Dr[1]/EnablVirtMode** "Virtueller Antrieb" (1001 00010) eingeschaltet werden.

Freigabe zur Leistungszuschaltung iAx_DrvPower

Dieses Signal steht in engem Zusammenhang mit den Signalen **Antrieb Ein**, **Antrieb betriebsbereit** und **Antrieb in Betrieb** (siehe S-0-0135, Bit 14/15).

Der Funktionszusammenhang ist im Signaldiagramm für das Signal ([Antrieb Ein qAx_DrvOn](#), Seite 81) ausführlich dargestellt.

Signalpegel	1:	Die Initialisierung des Sercos Interface, des Antriebs und der NC-Steuerung sind abgeschlossen und es liegt kein Fehler vor. Dieses Signal ist Voraussetzung für das Zuschalten der Leistung an den Antrieb.
	0:	Die Leistung am Antrieb kann nicht zugeschaltet werden.

Antrieb betriebsbereit iAx_DrvReady

Dieses Signal steht in engem Zusammenhang mit den Signalen **Antrieb Ein**, **Leistung zugeschaltet** und **Antrieb in Betrieb** (siehe S-0-0135, Bit 14/15).

Der Funktionszusammenhang ist im Signaldiagramm für das Signal ([Antrieb Ein qAx_DrvOn](#), Seite 81) ausführlich dargestellt.

Signalpegel	1:	Die Leistung am betreffenden Antrieb ist zugeschaltet. Das Signal bleibt gesetzt, solange die Leistung nicht abgeschaltet wird.
	0:	Am Antrieb ist keine Leistung zugeschaltet.

Antrieb in Betrieb iAx_DrvAct

Dieses Signal steht in engem Zusammenhang mit den Signalen **Antrieb Ein**, **Leistung zugeschaltet** und **Antrieb betriebsbereit** (siehe S-0-0135, Bit 14/15).

Der Funktionszusammenhang ist im Signaldiagramm für das Signal (**Antrieb Ein qAx_DrvOn**, Seite 81) ausführlich dargestellt.

Signalpegel	1:	Der Regelkreis ist geschlossen und der Antrieb steht unter Drehmoment. Sofern keine weiteren Verriegelungen, wie z. B. Vorschub Sperre vorliegen, kann die Achse verfahren werden.
	0:	Der Regelkreis ist geöffnet, der Antrieb steht nicht unter Drehmoment.

9 Spindelbezogene Interface-Signale

9.1 Übersicht spindelbezogene Interface-Signale

9.1.1 Allgemeines

Die spindelbezogenen Interface-Signale wirken auf die NC-Funktionen, die mit der Steuerung einer Bearbeitungsspindel im Zusammenhang stehen.

Für jede projektierte Spindel ist ein Interface vorhanden, d. h., bei n-Spindeln ist auch das spindelbezogene Interface n-mal auf die entsprechenden Merker abgebildet.

Für die Spindeln, die einem Kanal zugeordnet sind, ist außerdem das kanalbezogene Interface von Bedeutung.

In der folgenden Beschreibung sind die Interface-Signale für eine Spindel dargestellt.



Der unterstützte Funktionsumfang einer Spindel ist von der Art der Spindel (Maschinenparameter SP/SpFunc/Sp[1]/Base/SpType "Spindelart" (1040 00001)), anderen Einstellungen in den Maschinenparametern und den Antrieben abhängig.

9.1.2 Übersicht Ausgangssignale (SPS --> NC)

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
0.0	qSp_CAxOn	C-Achse Ein	1.0	qSp_TurnCW	Spindel M3 manuell
0.1	qSp_CAxOff	C-Achse Aus	1.1	qSp_TurnCCW	Spindel M4 manuell
0.2	qSp_JogPlus	Spindel tippen M3	1.2	qSp_Stop	Spindel M5 manuell
0.3	qSp_JogMinus	Spindel tippen M4	1.3	qSp_Orientate	Spindel M19 manuell
0.4	-	res.	1.4	-	res.
0.5	-	res.	1.5	-	res.
0.6	-	res.	1.6	-	res.
0.7	qSp_Reset	Spindelgrundstellung	1.7	-	res.
2.0	qSp_SafOpModeSwitch	Betriebsartenwahl (BA)	3.0	qSp_Gear1Act	Quittung GTS 1
2.1	qSp_SafDrvLock	Antriebssperre	3.1	qSp_Gear2Act	Quittung GTS 2
2.2	qSp_SafEnablCtrl	Zustimmtaste (ZT)	3.2	qSp_Gear3Act	Quittung GTS 3
2.3	qSp_SafSwitch1	Si-Schalter 1 (S1)	3.3	qSp_Gear4Act	Quittung GTS 4
2.4	qSp_SafSwitch2	Si-Schalter 2 (S2)	3.4	-	res.
2.5	qSp_SafCheckInputState	Prüfeingang Si-Status	3.5	-	res.
2.6	qSp_SafTechState	Status Si-Signale	3.6	-	res.
2.7	qSp_SafRedTransTime	Überführzeit abkürzen	3.7	qSp_GearIdleAct	Quittung Leerlauf
4.0	qSp_ManSpeed_00	Drehzahl tippen Bit 0	5.0	qSp_SpeedLimit	Drehzahlbegrenzung

Spindelbezogene Interface-Signale

Bit	Symbol. Adr.	SPS-Ausgangssignal	Bit	Symbol. Adr.	SPS-Ausgangssignal
4.1	qSp_ManSpeed_01	Drehzahl tippen Bit 1	5.1	qSp_SValueSD	S-Wert-Vorgabe über SD
4.2	qSp_ManSpeed_02	Drehzahl tippen Bit 2	5.2	qSp_HomeSwitch	Referenzschalter
4.3	-	res.	5.3	-	res.
4.4	-	res.	5.4	-	res.
4.5	-	res.	5.5	-	res.
4.6	-	res.	5.6	-	res.
4.7	-	res.	5.7	qSp_Override100	Spindeloverride 100%
6.0	qSp_Override_00	Override Bit 0	7.0	qSp_Override_08	Override Bit 8
6.1	qSp_Override_01	Override Bit 1	7.1	qSp_Override_09	Override Bit 9
6.2	qSp_Override_02	Override Bit 2	7.2	qSp_Override_10	Override Bit 10
6.3	qSp_Override_03	Override Bit 3	7.3	qSp_Override_11	Override Bit 11
6.4	qSp_Override_04	Override Bit 4	7.4	qSp_Override_12	Override Bit 12
6.5	qSp_Override_05	Override Bit 5	7.5	qSp_Override_13	Override Bit 13
6.6	qSp_Override_06	Override Bit 6	7.6	qSp_Override_14	Override Bit 14
6.7	qSp_Override_07	Override Bit 7	7.7	qSp_Override_15	Override Bit 15
8.0	qSp_Custom1	Kundeneingang 1	9.0	-	res.
8.1	qSp_Custom2	Kundeneingang 2	9.1	-	res.
8.2	qSp_Custom3	Kundeneingang 3	9.2	-	res.
8.3	qSp_Custom4	Kundeneingang 4	9.3	-	res.
8.4	qSp_Custom5	Kundeneingang 5	9.4	-	res.
8.5	qSp_Custom6	Kundeneingang 6	9.5	-	res.
8.6	qSp_Custom7	Kundeneingang 7	9.6	-	res.
8.7	qSp_Custom8	Kundeneingang 8	9.7	-	res.
10.0	-	res.	11.0	-	res.
10.1	-	res.	11.1	-	res.
10.2	-	res.	11.2	-	res.
10.3	-	res.	11.3	-	res.
10.4	-	res.	11.4	-	res.
10.5	-	res.	11.5	-	res.
10.6	-	res.	11.6	qSp_DrvOn	Antrieb Ein
10.7	qSp_RedDelayErr-Class1	Verz. Fehlerreaktion abkürzen	11.7	qSp_DrvLock	Spindel Sperre

Tab. 9-1: Übersicht Ausgangssignale (SPS → NC)

9.1.3 Übersicht Eingangssignale (NC --> SPS)

Bit	Symbol.Adr.	SPS-Eingangssignal	Bit	Symbol.Adr.	SPS-Eingangssignal
0.0	iSp_CAxAct	C-Achse ist aktiv	1.0	iSp_ProgSpReach	Drehzahl erreicht
0.1	iSp_CAxSwitch	C-Achs-Umschaltung	1.1	iSp_SpLim	Drehzahl begrenzt
0.2	iSp_TurnCmd	Drehbefehl	1.2	iSp_Stop	Spindel steht
0.3	iSp_TurnDirM4	Drehrichtung M4	1.3	iSp_OrientateFinish	Spindel gerichtet
0.4	-	res.	1.4	iSp_OrientateAct	Spindel richten aktiv
0.5	iSp_InPos	Spindel In Position	1.5	-	res.
0.6	iSp_PosCtrl	Lagereglung aktiv	1.6	-	res.
0.7	iSp_Reset	Spindel grundgestellt	1.7	-	res.
2.0	iSp_SafOpMode_00	Si-Betriebsart Bit 0	3.0	iSp_Gear1Sel	Anwahl GTS 1
2.1	iSp_SafOpMode_01	Si-Betriebsart Bit 1	3.1	iSp_Gear2Sel	Anwahl GTS 2
2.2	iSp_SafOpMode_02	Si-Betriebsart Bit 2	3.2	iSp_Gear3Sel	Anwahl GTS 3
2.3	iSp_SafOpMode_03	Si-Betriebsart Bit3	3.3	iSp_Gear4Sel	Anwahl GTS 4
2.4	iSp_SafStatePos	Status Sichere Lage	3.4	iSp_GearChange	GTS Umschaltung
2.5	iSp_SafCtrlOutputState	Statusausgang Regler	3.5	iSp_IdleSpeed	Trudeldrehzahl erreicht
2.6	-	res.	3.6	-	res.
2.7	-	res.	3.7	iSp_GearIdleSel	Anwahl Getriebe-Leerlauf
4.0	-	res.	5.0	-	res.
4.1	-	res.	5.1	-	res.
4.2	-	res.	5.2	-	res.
4.3	-	res.	5.3	-	res.
4.4	-	res.	5.4	-	res.
4.5	-	res.	5.5	-	res.
4.6	-	res.	5.6	iSp_Override0	Spindeloverride 0 %
4.7	iSp_ToolLimAct	Werkzeugbegrenzungen aktiv	5.7	iSp_Override100	Spindeloverride 100 %
6.0	iSp_ScsState00	SCS-Signalstatus 0	7.0	iSp_ScsState08	SCS-Signalstatus 8
6.1	iSp_ScsState01	SCS-Signalstatus 1	7.1	iSp_ScsState09	SCS-Signalstatus 9
6.2	iSp_ScsState02	SCS-Signalstatus 2	7.2	iSp_ScsState10	SCS-Signalstatus 10
6.3	iSp_ScsState03	SCS-Signalstatus 3	7.3	iSp_ScsState11	SCS-Signalstatus 11
6.4	iSp_ScsState04	SCS-Signalstatus 4	7.4	iSp_ScsState12	SCS-Signalstatus 12
6.5	iSp_ScsState05	SCS-Signalstatus 5	7.5	iSp_ScsState13	SCS-Signalstatus 13
6.6	iSp_ScsState06	SCS-Signalstatus 6	7.6	iSp_ScsState14	SCS-Signalstatus 14
6.7	iSp_ScsState07	SCS-Signalstatus 7	7.7	iSp_ScsState15	SCS-Signalstatus 15
8.0	iSp_Custom1	Kundenausgang 1	9.0	iSp_CoupleIndex_00	Nr. der Kopplung Bit 0

Spindelbezogene Interface-Signale

Bit	Symbol.Adr.	SPS-Eingangssignal	Bit	Symbol.Adr.	SPS-Eingangssignal
8.1	iSp_Custom2	Kundenausgang 2	9.1	iSp_CoupleIndex_01	Nr. der Kopplung Bit 1
8.2	iSp_Custom3	Kundenausgang 3	9.2	iSp_CoupleIndex_02	Nr. der Kopplung Bit 2
8.3	iSp_Custom4	Kundenausgang 4	9.3	-	res.
8.4	iSp_Custom5	Kundenausgang 5	9.4	-	res.
8.5	iSp_Custom6	Kundenausgang 6	9.5	-	res.
8.6	iSp_Custom7	Kundenausgang 7	9.6	-	res.
8.7	iSp_Custom8	Kundenausgang 8	9.7	-	res.
10.0	iSp_Master	Spindel ist Master	11.0	iSp_DrvErrClass1	Fehler Zustandsklasse-1
10.1	iSp_CoupleErr	Kopplungsfehler	11.1	iSp_DrvChangeClass2	Änderung Zustandsklasse-2
10.2	iSp_Synchr1	Synchronlauf 1	11.2	iSp_DrvChangeClass3	Änderung Zustandsklasse-3
10.3	iSp_Synchr2	Synchronlauf 2	11.3	-	res.
10.4	-	res.	11.4	iSp_DryRun	Testbetrieb
10.5	-	res.	11.5	iSp_DrvPower	Freigabe zur Leistungszuschaltung
10.6	-	res.	11.6	iSp_DrvReady	Antrieb betriebsbereit
10.7	iSp_DelayErrClass1	Verzögerte Fehlerreaktion	11.7	iSp_DrvAct	Antrieb in Betrieb

Tab. 9-2: Übersicht Eingangssignale (NC → SPS)

9.2 Signalbeschreibung

9.2.1 Ausgangssignale (SPS → NC)

Allgemeines

Je nach Funktion wertet das CNC-System MTX die Signale als **statische Signale** mit den Signalpegeln Logisch 0 und Logisch 1 aus oder als **flankengesteuerte Signale** mit den Signalfanken 0 → 1 oder 1 → 0.



Der unterstützte Funktionsumfang einer Spindel ist von der Art der Spindel (Maschinenparameter SP/SpFunc/Sp[1]/Base/SpType "Spindelart" (1040 00001)), anderen Einstellungen in den Maschinenparametern und den Antrieben abhängig.

Für Spindeln vom Typ "Externe Spindel" wird nur das Signal "qSp_Reset" ausgewertet.

C-Achse Ein qSp_CAxOn

Mit diesem Signal kann die SPS eine Spindel auf C-Achsbetrieb umschalten.

Anwendung:

In Sonderfällen (Spindel/Werkzeugrevolver) kann es notwendig sein, direkt nach Steuerungshochlauf den C-Achsbetrieb einzuschalten.

Signalfanke 0 → 1: Die laufende Spindel wird abgebremst und in den C-Achsbetrieb geschaltet.

Der NC-interne Ablauf der Umschaltung wird von den Einstellungen im Maschinenparameter **/SP/SpFunc/Sp[1]/SpTurrAx/MeasSysSpTurrAx** "C-Achse: Werkzeugrevolver" (1040 00060) beeinflusst.

C-Achse Aus qSp_CAxOff

Mit diesem Signal schaltet die SPS eine Spindel zurück in den Spindelbetrieb.

Signalflanke	0 → 1:	Der Antrieb wird in den Spindelbetrieb zurückgeschaltet. Der NC-interne Ablauf der Umschaltung wird von den Einstellungen im Maschinenparameter /SP/SpFunc/Sp[1]/SpTurrAx/MeasSysSpTurrAx "C-Achse: Werkzeugrevolver" (1040 00060) beeinflusst.
	1 → 0:	Keine Funktion.

Spindel tippen M3 qSp_JogPlusSpindel tippen M4 qSp_JogMinus

Diese Signale wirken in der NC-Betriebsart **Einrichten** und stehen im Funktionszusammenhang mit den codierten Signalen Drehzahl Tippen Bit 0 ... Bit 2.

Signalpegel	1:	Voraussetzung ist die Drehzahlvorgabe über die codierten Signale Drehzahl Tippen Bit 0 ... Bit 2. Weiterhin muss der Regelkreis geschlossen sein (Antrieb in Betrieb gesetzt), die Spindelsperre muss zurückgesetzt und eine Getriebestufe muss aktiviert sein. Solange Spindel tippen M3 oder Spindel tippen M4 gesetzt ist, dreht die Spindel mit der entsprechenden Drehzahl in der vorgewählten Drehrichtung.
	0:	Die Spindel wird bis zum Stillstand gebremst, der Regelkreis bleibt geschlossen.

Beispiel:

Die Spindel soll im Einrichtbetrieb mit einer vordefinierten Drehzahl im Uhrzeigersinn verfahren werden.

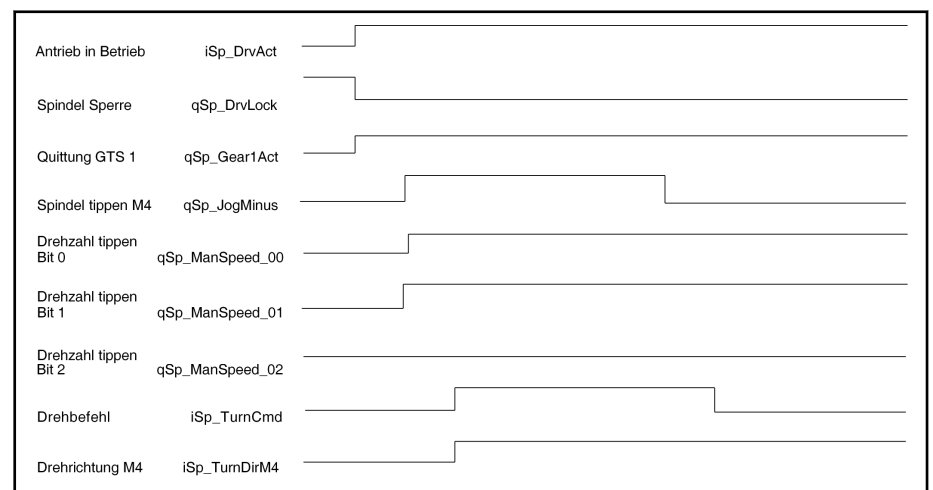


Abb. 9-1: Spindel tippen

Der Regelkreis ist geschlossen (**Antrieb in Betrieb** ist gesetzt), **Spindelsperre** zurückgesetzt und die Getriebestufe 1 (**Quittung Getriebestufe 1**) aktiviert.

Mit Setzen des Signals **Spindel tippen M3** muss parallel eine Drehzahl vorgegeben werden (**Drehzahl Tippen Bit 0 ... Drehzahl Tippen Bit 2**). Die Tipp-Drehzahl wird spindelspezifisch im Maschinenparameter **/SP/SpFunc/Sp[1]/Jog/Speed/JogSpeed[1]** "Tipp-Spindeldrehzahl" (1040 00015) definiert.

Der **Drehbefehl** wird von der NC-Steuerung gesetzt, und in Verbindung mit dem Drehbefehl wird die **Drehrichtung M4** ausgegeben.

Da die Spindelsperre bereits zurückgesetzt ist, beschleunigt die Spindel auf die vorgewählte Drehzahl.

Steht das Signal **Spindelsperre** bei der Ausgabe des Drehbefehls noch an, wird die Drehrichtung M4 erst nach Rücksetzen von Spindelsperre ausgegeben.

Mit Rücksetzen des Signals **Spindel tippen** wird der Bremsvorgang an der Spindel eingeleitet. Ist die Spindel zum Stillstand gekommen, wird **Drehbefehl** zurückgesetzt.



Im Normalbetrieb wird die Spindel in der Regel nicht im Lagerregelkreis sondern nur gesteuert verfahren. Daher ist es möglich, dass der Drehbefehl zurückgesetzt ist, die Spindel aber noch ausläuft. Um sicher zu sein, dass die Spindel steht, muss zusätzlich das Signal "Spindel steht" abgefragt werden.

Spindelgrundstellung qSp_Reset

Mit diesem Signal kann die aktuelle Bewegungsvorgabe bzw. ein aktives Kommando (z. B. Spindel richten) der betreffenden Spindel abgebrochen werden. Die Spindel quittiert das Erhalten des Abbruchauftrags mit Setzen des Signals **Spindel grundgestellt** (iSp_Reset).

Signalflanke	0 → 1:	Das Signal Spindel grundgestellt (iSp_Reset) wird gesetzt.
	1 → 0:	Keine Auswirkung.

Spindel M3 manuell qSp_TurnCWSpindel M4 manuell qSp_TurnCCW

Diese Signale bestimmen alternativ zum NC-Teileprogramm (Programmierung M3, M4) die Drehrichtung der Spindel:

- Spindel M3 manuell:
Spindel Rechtslauf
- Spindel M4 manuell:
Spindel Linkslauf.

In diesem Zusammenhang ist auch das Signal **Spindel M5 manuell** von Bedeutung (siehe Beispiel [Spindel M5 manuell qSp_Stop](#), Seite 100).

Signalflanke	0 → 1:	Mit dieser Flanke wird dem Spindelantrieb eine Spindeldrehzahl und die betreffende Drehrichtung vorgegeben. Ist der Regelkreis geschlossen (Antrieb in Betrieb) und keine Spindelsperre ansteht, beginnt die Spindel, zu drehen. Die Spindeldrehzahl entspricht der zuletzt programmierten Drehzahl.
	1 → 0:	Keine Auswirkung.

Spindel M5 manuell qSp_Stop

Dieses Signal aktiviert alternativ zum NC-Teileprogramm (Programmierung M5) die Funktion Spindel Stopp.

In diesem Zusammenhang sind auch die Signale Spindel M3 manuell und Spindel M4 manuell von Bedeutung.

Signalflanke	0 → 1:	Die Spindel stoppt mit der parametrisierten Bremsbeschleunigung. Die zuvor aktive Getriebestufe und Spindeldrehzahl

bleiben steuerungsintern gespeichert. Der Regelkreis bleibt geschlossen und der Spindelantrieb unter Drehmoment.

1 → 0: Keine Auswirkung.

Beispiel:

Die Spindel soll manuell im Gegenuhrzeigersinn auf die zuletzt programmierte Drehzahl beschleunigt werden.

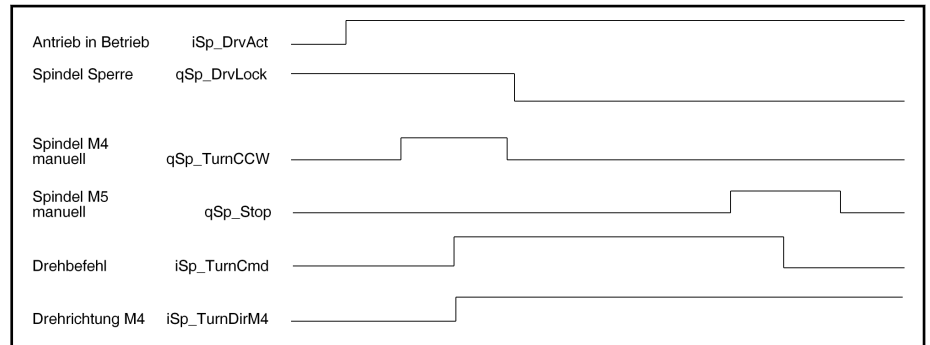


Abb. 9-2: Spindel M5 manuell

Das Signal **Antrieb in Betrieb** sagt aus, dass der Regelkreis geschlossen ist und die Spindel unter Drehmoment steht.

Mit Setzen des Signals **Spindel M4 manuell** wird dem Spindelantrieb eine Drehzahl übergeben, die der zuletzt programmierten Drehzahl entspricht.

Aufgrund der Drehzahlvorgabe werden von der NC **Drehbefehl** und **Drehrichtung M4** gesetzt.

Mit der Ausgabe des **Drehbefehls** kann das Signal **Spindel M4 manuell** zurückgesetzt werden. Solange die **Spindelsperre** gesetzt ist, dreht die Spindel nicht.

Erst mit Rücksetzen der **Spindelsperre** dreht die Spindel und **Spindel dreht** wird gesetzt. Durch Vorgeben des Signals **Spindel M5 manuell** wird die Drehbewegung der Spindel gestoppt, **Drehbefehl** und **Drehrichtung M4** werden von der NC zurückgesetzt.

Spindel M19 manuell qSp_Orientate

Dieses Signal ruft alternativ zum NC-Teilprogramm (M19) die Funktion Spindel richten auf. Dazu muss der Regelkreis geschlossen und der Antrieb in Betrieb sein.

In diesem Zusammenhang sind die Signale **Spindel in Position**, **Spindel richten aktiv** und **Spindel gerichtet** von Bedeutung.



Diese Funktion steht nur für Sercos Spindeln (Maschinenparameter **/SP/SpFunc/Sp[1]/Base/SpType** "Auswahl der Spindelart" (1040 00001)) zur Verfügung.

Signalflanke	0 → 1:	Spindel richten wird eingeleitet. Die Position ergibt sich aus dem aktuellen Inhalt des Systemdatums SysSpCmdData[n]/PlcOriPos. Die Drehrichtung ist abhängig vom Sercos Parameter S-0-0154.
	1 → 0:	Keine Auswirkung.

Beispiel:

Die Spindel soll manuell gerichtet werden, der Regelkreis ist geschlossen.

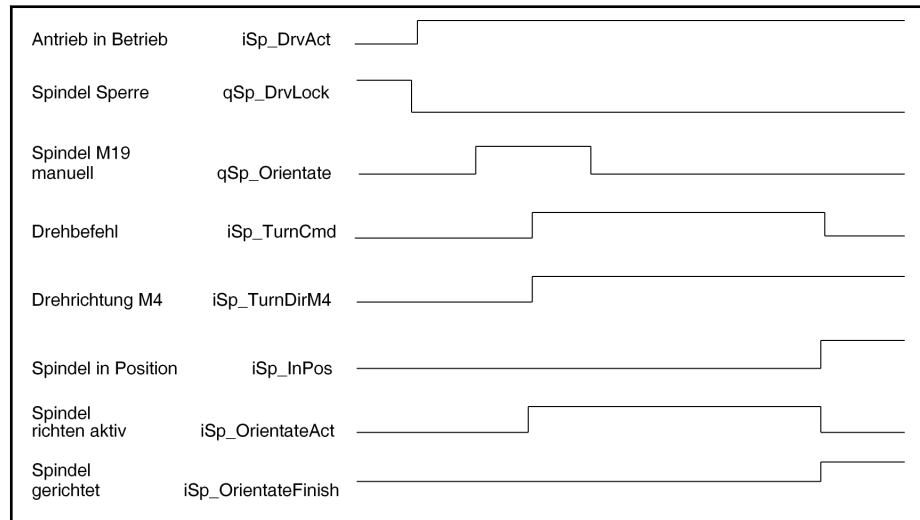


Abb. 9-3: Spindel M19 manuell

Der Regelkreis ist geschlossen (**Antrieb in Betrieb** ist gesetzt) und die **Spindelsperre** ist zurückgesetzt.

Mit Setzen des Signals **Spindel richten manuell** erhält die Spindel das Kommando zum Drehen, **Drehbefehl** wird gesetzt.

In Abhängigkeit der Drehrichtung wird **Drehrichtung M4** gesetzt, wenn die Spindel im Gegenuhrzeigersinn (M4) dreht.

Mit der Ausgabe des Signals **Drehbefehl** kann das Signal **Spindel richten manuell** zurückgesetzt werden.

Dreht die Spindel, wird **Spindel richten aktiv** gesetzt.

Nach Positionierung auf die Markerposition wird **Spindel gerichtet** und **Spindel in Position** gesetzt und der **Drehbefehl** zurückgesetzt.

Betriebsartenwahl (BA) qSp_SafOpModeSwitch

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive.

Als Kanal 1 wird hier die Betriebsartenwahl der SPS über Sercos Interface in Echtzeit an den Antrieb übertragen.

Signalpegel	1:	Normalbetriebsart NO
	0:	Sonderbetriebsart SO (Sonderbetrieb mit Stillstand/mit Bewegung)

Antriebssperre qSp_SafDrvLock

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive.

Als Kanal 1 wird hier das Signal "Antriebssperre" per SPS über Sercos Interface in Echtzeit an den Antrieb übertragen. Bei Vorgabe der Antriebssperre schaltet sich der Antrieb momentenfrei.

Signalpegel	1:	Antriebssperre ein.
	0:	Antriebssperre aus.

Zustimmtaste (ZT) qSp_SafEnablCtrl

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive.

Als Kanal 1 wird hier die Umschaltung in "Sonderbetrieb mit Bewegung" per SPS über Sercos Interface in Echtzeit an den Antrieb übertragen.

Signalpegel	1:	Sonderbetrieb mit Bewegung
	0:	Sonderbetrieb mit Stillstand

Si-Schalter 1 (S1) qSp_SafSwitch1

Si-Schalter 2 (S2) qSp_SafSwitch2

Diese Signale sind Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive.

Für den "Sonderbetrieb mit Bewegung" können im Antrieb bis zu 4 verschiedene Parametersätze abgelegt sein. Als Kanal 1 wird hier der, per SPS angewählte Parametersatz über Sercos in Echtzeit an den Antrieb übertragen.

Bis MTX14V06 unterstützt die MTX maximal 2 sichere Bewegungen. Ab MTX14V08 können in Verbindung mit der Sicherheitstechnik nach IEC 61508 maximal 4 sichere Bewegungen verwendet werden.

Beachten sie bitte die Hinweise in der Antriebsdokumentation zu "Zustandsmaschine der integrierten Sicherheitstechnik" und zum Sercos Parameter "P-0-3212 (SI-Steuerwort, Kanal 1) Bit 3 und Bit 4".



Wenn der SI-Schalter 2 "P-0-3212 (SI-Steuerwort, Kanal 1) Bit 4" zyklisch zum Antrieb übertragen wird, dann erwartet die MTX 4 konfigurierte sichere Bewegungen!

Wenn nur maximal 2 sichere Bewegungen verwendet werden, dann sollte nur der SI-Schalter 1 "P-0-3212 (SI-Steuerwort, Kanal 1) Bit 3" zyklisch zum Antrieb übertragen werden.

	Bit 1 (qSp_SafSwitch2)	Bit 0 (qSp_SafSwitch1)
SMM1	0	0
SMM2	0	1
SMM3	1	0
SMM4	1	1

Tab. 9-3: Anwahl der sicheren Bewegungen

Prüfeingang Si-Status qSp_SafCheckInputState

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive.

Als Kanal 1 wird hier ein per SPS gesetztes sicheres Statussignal über Sercos Interface in Echtzeit an den Antrieb übertragen.

Status Si-Signale qSp_SafTechState

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive.

Die Steuersignale der SPS (qSp_Saf...), die über den Kanal 1 zum Antrieb gehen, müssen zur Überprüfung in gewissen Zeitabständen dynamisiert werden. Das heißt, die Signale werden von der SPS für einen Zyklus auf "0" ge-

setzt. Die SPS setzt gleichzeitig den Si-Signalstatus von gültig auf dynamisiert und teilt dadurch dem Antrieb mit, dass die Steuersignale ungültig sind.

Danach werden die Signale neu gesetzt.

Signalpegel	1:	Die Steuersignale der Sicherheitstechnik werden dynamisiert.
	0:	Die Steuersignale der Sicherheitstechnik sind gültig.

Überführzeit abkürzen qSp_SafRedTransTime

Dieses Signal ist Bestandteil der Sicherheitstechnik der Antriebsbaureihe IndraDrive.

Dieses Signal bietet der SPS die Möglichkeit zum Verkürzen der Überführzeit von Normalbetrieb nach Sonderbetrieb bzw. von Sonderbetrieb nach Sonderbetrieb.

Bei Verwendung der Funktion "Integrierte Sicherheitstechnik nach IEC 61508":

- Beachten sie bitte die Hinweise in der Antriebsdokumentation zu "Zustandsmaschine der integrierten Sicherheitstechnik" und zum Sercos Parameter "P-0-3212 (SI-Steuerwort, Kanal 1) Bit 11".

Bei Verwendung der Funktion "Integrierte Sicherheitstechnik Safe Motion":

- Beachten sie bitte die Hinweise in der Antriebsdokumentation zu "Überföhrfunktionen" und zum Sercos Parameter "P-0-3261.0.1 (SMO: Steuerwort Zustandsmaschine, funktional) Bit 0".

Signalpegel	1:	Meldet dem Antrieb, dass die Anpassung des Sollwertsystems abgeschlossen ist und der Antrieb aktiviert die Überwachungen der angewählten Sonderbetriebsart.
	0:	Keine Wirkung.

Quittung GTS 1...4 qSp_Gear1Act ... 4ActQuittung Leerlauf qSp_GearIdleAct

Mit diesen Signalen wird die aktuelle Getriebestufe gemeldet. Liegt keine Getriebestufenübereinstimmung vor, wird beim nächsten Spindelauftrag "Trudeln" ausgelöst.

Im Steuerungshochlauf wird die quitierte Getriebestufe in der NC übernommen.

Getriebestufenwechsel wird für Analog-Spindeln (/SP/SpFunc/Sp[1]/Base/SpType "Auswahl der Spindelart" (1040 00001)) nicht unterstützt. Für Analog-Spindeln muss GTS 1 quitiert werden.

Drehzahl tippen Bit 0 ... Bit 2 qSp_ManSpeed_00 ... 02

Diese Signale wirken ausschließlich in Verbindung mit den Signalen **Spindel tippen M3** oder **Spindel tippen M4**.

Abhängig von der Codierung wird die im Maschinenparameter /SP/SpFunc/Sp[1]/Jog/Speed/JogSpeed[1] "Tipp-Spindeldrehzahl" (1040 00015) definierte Drehzahl aktiviert. Liegt die vorgewählte Drehzahl nicht im Bereich der aktuellen Getriebestufe, wird sie auf die minimale bzw. maximale Drehzahl der aktivierten Getriebestufe begrenzt.

Maschinenparameter	Drehzahl tippen		
	Bit 2 (qSp_Speed_02) Man-	Bit 1 (qSp_Speed_01) Man-	Bit 0 (qSp_Speed_00) Man-
.../JogSpeed [1]	0	0	0
.../JogSpeed [2]	0	0	1
.../JogSpeed [3]	0	1	0
.../JogSpeed [4]	0	1	1
.../JogSpeed [5]	1	0	0
.../JogSpeed [6]	1	0	1
.../JogSpeed [7]	1	1	0
.../JogSpeed [8]	1	1	1

Tab. 9-4: Drehzahl Tippen

Drehzahlbegrenzung SD qSp_SpeedLimit

Mit diesem Signal aktiviert die SPS die in den Systemdaten "/SysSpCmdData[n]/MinSpeedLimit" und "/SysSpCmdData[n]/MaxSpeedLimit" eingetragenen Drehzahlbegrenzungen.

- Signalflanke**
- 0 → 1:** Übernimmt den aktuellen Inhalt der Systemdaten und aktiviert die Drehzahlbegrenzung.
 - 1 → 0:** Die Drehzahlbegrenzung wird ausgeschaltet.

S-Wert-Vorgabe über SD qSp_SValueSD

Dieses Signal ermöglicht die Spindel-S-Wert-Vorgabe über die Systemdaten "/SysSpCmdData[n]/Speed" und "/SysSpCmdData[n]/OriPos".

Bei gesetztem Interface-Signal "qSp_SValueSD" ergeben sich folgende Einschränkungen der Spindelfunktionalität:

- G96 funktioniert nicht!
Die Kombination G96 und "Drehzahlvorgabe über SD" erzeugt einen Laufzeitfehler!
- M40 funktioniert nicht!
Nur im Kanal programmierte Drehzahlwerte veranlassen einen automatischen Getriebestufenwechsel.
- SD(205,2,x) (Programmierte Spindeldrehzahl) liefert immer den letzten im Teileprogramm programmierten Wert.

- Signalpegel**
- 1:** Der Inhalt des Systemdatums "SysSpCmdData[n]/Speed" wird als Drehzahlwert verwendet.
Der Inhalt des Systemdatums "SysSpCmdData[n]/OriPos" wird als Richtposition verwendet.
 - 0:** Die im Teileprogramm vorgegebenen S-Werte wirken.

Referenzschalter qSp_HomeSwitch

Überträgt das Signal des Referenzschalters zum Antrieb, wenn dieser nicht direkt am Antrieb verdrahtet ist. Voraussetzung ist, dass dies im Antrieb entsprechend konfiguriert ist.



- Das Signalsteuerwort S-0-0145 muss im Master-Daten-Telegramm (S-0-1050.0.6) enthalten sein
- Bit 0 von S-0-0400 muss im Signal-Steuerwort (S-0-0145) konfiguriert sein

Signalpegel	1:	Der Referenzschalter ist betätigt.
	0:	Der Referenzschalter ist nicht betätigt.

Spindeloverride 100% qSp_Override100

Dieses Signal blendet das Spindeloverride aus. Die Spindel dreht dann mit der programmierten Drehzahl.

Signalpegel	1:	Der Spindeloverride ist ausgeblendet.
	0:	Der Spindeloverride ist wirksam.

Override Bit 0 ... Bit 15 qSp_Override_00 ... 15

Die aktuelle Spindeldrehzahl kann über die **Override-Funktion** beeinflusst werden. Die Override-Funktion wirkt dabei auf die, im NC-Teileprogramm programmierten Drehzahlen und auf die Drehzahl im Einrichtbetrieb.

Über dieses Interface ist eine stufenlose 16 Bit-Vorgabe möglich. Das CNC-System MTX interpretiert den Wert direkt als Override-Wert in 0,01%.

Kundeneingang 1 ... 8 qSp_Custom1 ... 8

NC-seitig konfigurierbares Interface, das je nach Bedarf belegt werden kann.

(Diese Möglichkeit ist z. Zt. noch nicht aktiv)

Verzögerte Fehlerreaktion abkürzen qSp_RedDelayErrClass1

Dieses Signal steht in Zusammenhang mit iSp_DelayErrClass1 (Verzögerte Fehlerreaktion).

Dieses Signal bietet der SPS die Möglichkeit die Zeit bis zum Stillsetzen der Spindel durch die NC zu verkürzen.



Weitere Informationen sind dem Kapitel "Verzögerte NC-Fehlerreaktion bei nichtfatalen Antriebsfehlern" im Handbuch "Funktionsbeschreibung" zu entnehmen.

Signalflanke	0 → 1:	Die verzögerte Fehlerreaktion wird beendet. Die NC stoppt die Spindel.

Antrieb Ein qSp_DrvOn

Dieses Signal schaltet im Antrieb das Drehmoment ein, wenn er betriebsbereit ist (iSp_DrvReady gesetzt). Der Regelkreis ist geschlossen.

Das Rücksetzen von **Antrieb Ein** leitet einen Ausschaltvorgang ein. Ist die Spindel zu diesem Zeitpunkt noch in Bewegung, wird sie mit der voreingestellten Bremsrampe zum Stillstand gebracht und in den Zustand Spindel Stopp (M5) geschaltet.

Im Stillstand wird der Regelkreis geöffnet und das Signal **Antrieb in Betrieb** zurückgesetzt. Der Antrieb ist jetzt drehmomentfrei.



Der gesamte Ablauf ist für Analog-Spindeln ohne Bedeutung.

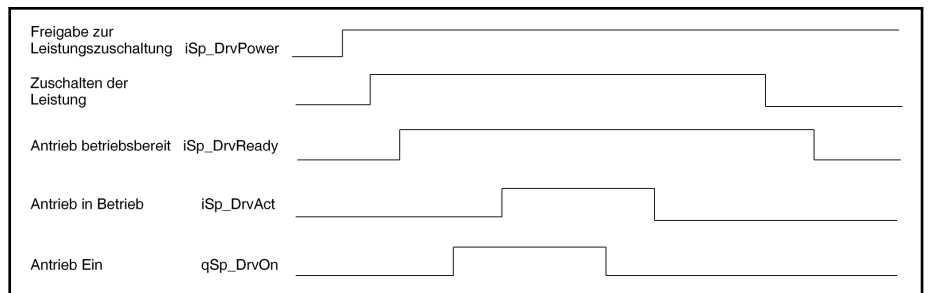


Abb. 9-4: Antrieb ein

Beispiel:**Funktionsweise des Ein- und Ausschaltvorgangs**

Nach dem Einschalten der Maschine bzw. Anlegen der Spannung durchlaufen NC-Steuerung, Servo- und Spindelantrieb eine Initialisierungsphase.

Während der Initialisierungsphase der einzelnen Gerätekompenten wird u. a. die digitale Schnittstelle zur Kommunikation zwischen NC-Steuerung und Antrieben aufgebaut.

Diese digitale Schnittstelle entspricht den Festlegungen des **Sercos Interface**.

Die Initialisierung des **Sercos Interface** erfolgt in 4 Phasen systemintern und automatisch. Nachdem die Schnittstelle initialisiert und der zyklische Betrieb aktiv ist, wird das Signal **Freigabe zur Leistungszuschaltung** ausgegeben.

Die Leistung an den betreffenden Antrieben kann zugeschaltet werden. Nachdem die Leistung zugeschaltet wurde, wird das Signal **Antrieb betriebsbereit** ausgegeben.

Mit Setzen von **Antrieb Ein** wird der Lageregelkreis geschlossen. Der Antrieb steht unter Drehmoment und das Signal **Antrieb in Betrieb** wird ausgegeben.

Das Einleiten des Ausschaltvorgangs bzw. Abschalten der Leistung erfolgt, indem das SPS-Ablaufprogramm das Signal **Antrieb Ein** zurücksetzt.

Ist die Achse zu diesem Zeitpunkt noch in Bewegung, wird sie mit der voreingestellten Bremsrampe zum Stillstand gebracht.

Im Stillstand wird der Lageregelkreis geöffnet und das Signal **Antrieb in Betrieb** zurückgesetzt. Der Antrieb ist jetzt drehmomentfrei.

Im nächsten Schritt kann die Leistung abgeschaltet werden.

Das Signal **Freigabe zur Leistungszuschaltung** bleibt anstehen, solange die Gerätekompenten (NC und Antriebe) weiterhin initialisiert sind.

Spindel Sperre qSp_DrvLock

Dieses Signal ist in jeder Betriebsart wirksam, wenn der Regelkreis geschlossen ist und die Spindel unter Drehmoment steht.

Es verhindert, dass eine programmierte oder manuell vorgegebene Spindelbewegung ausgeführt wird. Auch die Trudeldrehzahl wird nicht ausgeführt, wenn Spindel Sperre gesetzt ist.

Drehbefehl, Drehrichtung M4 oder das Signal Spindel in Position werden nicht beeinflusst.

Signalpegel	1:	Die Spindel kann nicht gedreht werden. Ist eine Drehbewegung bereits aktiv, wird die Spindel mit der parametrisierten Bremsbeschleunigung gestoppt.
	0:	Die Drehbewegung der Spindel ist nicht gesperrt. Wurde ein Kommando zum Drehen der Spindel gegeben (über das NC-Teilprogramm oder manuell) und die entsprechende Getrie-

bestufe aktiv ist, kann die Spindel unmittelbar drehen. Wurde eine drehende Spindel zuvor mit Spindel Sperre zum Stillstand gebracht, beschleunigt sie jetzt wieder auf die ursprüngliche Drehzahl.

Beispiel:

Spindel Sperre

Die Maschine ist eingeschaltet und der Regelkreis ist bereits geschlossen. Über das NC-Teileprogramm ist eine Drehbewegung, z. B. M3 S1000, programmiert.

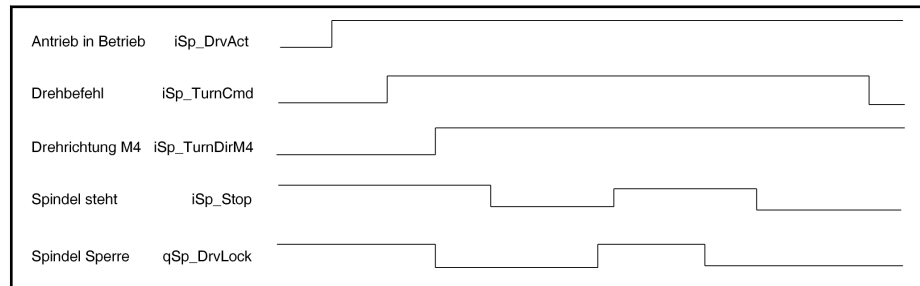


Abb. 9-5: Spindel Sperre

Mit dem Signal **Antrieb in Betrieb** wird dem SPS-Ablaufprogramm mitgeteilt, dass der Regelkreis geschlossen ist. Mit Ausführen des NC-Satzes, in dem die Drehbewegung programmiert ist, wird das Signal **Drehbefehl** ausgegeben.

Solange **Spindel Sperre** gesetzt ist, wird die Drehbewegung noch nicht ausgeführt, **Spindel steht** ist gesetzt, **Drehrichtung M4** ist nicht gesetzt. Die Drehrichtung ist nur dann abzufragen, wenn ein Drehbefehl ansteht.

Mit Rücksetzen von **Spindel Sperre** wird die programmierte Drehbewegung ausgeführt, **Drehrichtung M4** entsprechend der Bewegung gesetzt und **Spindel steht** zurückgesetzt.

Wird während der Drehbewegung **Spindel Sperre** wieder gesetzt, stoppt die Drehbewegung und **Spindel steht** wird gesetzt, nachdem die Spindel zum Stillstand gekommen ist.

Mit Rücksetzen von **Spindel Sperre** beschleunigt die Spindel auf die zuvor gültige Drehzahl und **Spindel steht** wird zurückgesetzt.

Das Signal **Drehbefehl** wird zurückgesetzt, wenn die Spindel über die Spindelstopp-Funktion **M5** (NC-Teileprogramm oder Interface-Signal) gestoppt wird.

9.2.2 Eingangssignal (NC --> SPS)

Allgemeines



Der unterstützte Funktionsumfang einer Spindel ist von der Art der Spindel (Maschinenparameter SP/SpFunc/Sp[1]/Base/SpType "Spindelart" (1040 00001)), anderen Einstellungen in den Maschinenparametern und den Antrieben abhängig. Für Spindeln vom Typ "Externe Spindel" werden die Signale nicht gesetzt.

C-Achse ist aktiv iSp_CAxAct

Dieses Signal zeigt an, ob die betreffende Spindel im C-Achsbetrieb (Lage-schnittstelle) oder im Spindelbetrieb (Drehzahlschnittstelle) arbeitet. Spindeln, die auf C-Achsbetrieb umgeschaltet werden können, besitzen neben dem Spindel-Interface **zusätzlich ein normales Achs-Interface** für den C-

Achsbetrieb. Das Signal C-Achse ist aktiv zeigt damit an, welches Interface relevant ist.

Signalpegel	1:	Die Spindel ist im C-Achsbetrieb: das Achs-Interface ist gültig, das Spindel-Interface wird von der NC nicht bedient.
	0:	Die Spindel ist im Spindelbetrieb: das Spindel-Interface wird von der NC bedient, das Achs-Interface ist nicht gültig.

C-Achsumschaltung iSp_CAxSwitch

Dieses Signal zeigt an, dass die Spindel sich gegenwärtig in der Umschaltphase zwischen Spindel- und C-Achsbetrieb befindet.

Zeitgleich mit diesem Befehl wird immer das Signal **Drehbefehl** ausgegeben.

Signalpegel	1:	Spindel befindet sich in der Umschaltphase.
	0:	Spindel befindet sich nicht in der Umschaltphase.

Drehbefehl iSp_TurnCmd

Die Signale **Drehbefehl**, **Drehrichtung M4**, **Spindel steht** und **Spindel In Position** beeinflussen sich wechselseitig.

In den Signaldiagrammen und Erläuterungen zu den Signalen Spindel M3 manuell, Spindel M4 manuell, Spindel M5 manuell (siehe [Spindel M3 manuell qSp_TurnCW, Seite 100](#)) und Spindelsperre (siehe [Spindel Sperre qSp_DrvLock, Seite 107](#)) sind die Zusammenhänge beispielhaft dargestellt.

In Verbindung mit dem **Drehbefehl** wird das Signal **Drehrichtung M4** ausgegeben.

Signalpegel	1:	Die Spindel soll eine Drehbewegung ausführen, die über eine manuelle Vorgabe (z. B. Spindel M3 manuell) oder über das NC-Teileprogramm (z. B. M3 S1500) vorgegeben wird. Voraussetzung ist, dass der Regelkreis geschlossen ist, d. ., das Signal Antrieb in Betrieb gesetzt ist.
	0:	Die Drehbewegung wurde über einen programmierten Spindel Stopp M5 oder über das Signal Spindel M5 manuell (qSp_Stop) beendet. Drehbefehl ist außerdem nicht gesetzt, wenn die Spindel gerichtet , positioniert oder die Getriebestufe gewechselt wird.

Drehrichtung M4 iSp_TurnDirM4

Dieses Signal wirkt nur im Zusammenhang mit dem Signal **Drehbefehl**, d. h., **Drehrichtung M4** und **Drehbefehl** müssen im SPS-Ablaufprogramm als UND-Verknüpfung realisiert sein.

Ist kein Drehbefehl gesetzt, hat das Signal Drehrichtung M4 keine Bedeutung.

Bei den Funktionen "Getriebestufenwechsel", "Trudeldrehzahl" und "Spindel richten" wird das Signal Drehrichtung M4 nicht ausgegeben bzw. der Signalzustand bleibt unverändert.

Signalpegel	1:	Drehrichtung M4 gesetzt und Drehbefehl gesetzt bedeutet, dass die Spindel eine Drehbewegung im Gegenuhrzeigersinn (M4) ausführen soll.
	0:	Drehrichtung M4 nicht gesetzt und Drehbefehl gesetzt bedeutet, dass die Spindel eine Drehbewegung im Uhrzeigersinn (M3) ausführen soll.

Spindel in Position iSp_InPos

Dieses Signal ist nur von Bedeutung in Verbindung mit der Funktion **Spindel richten** oder **Spindel positionieren**.

Signalpegel	Voraussetzung:	Der Regelkreis ist geschlossen und die Spindel steht unter Drehmoment (Signal Antrieb in Betrieb ist gesetzt).
	1:	Die Spindel ist gerichtet oder positioniert und befindet sich im Inpositionsfenster, das im Antrieb definiert ist.
	0:	Die Spindel wurde nicht gerichtet und/oder befindet sich nicht im parametrisierten Inpositionsfenster.

Lageregelung aktiv iSp_PosCtrl

Signalpegel	1:	Die Spindel wird über die Lageschnittstelle angesteuert.
	0:	Die Spindel wird über die Drehzahlschnittstelle angesteuert.

Spindel grundgestellt iSp_Reset

Dieses Signal zeigt an, dass der Grundstellungsauftrag von der Spindel angenommen wurde und die Spindel wieder bereit ist, neue Bewegungsvorgaben anzunehmen.

Drehzahl erreicht iSp_ProgSpReach

Signalpegel	1:	Die Drehzahl an der Spindel entspricht der programmierten Spindeldrehzahl und/oder befindet sich innerhalb des Drehzahl-Toleranzbandes, dass durch die Überlagerung der Maschinenparameter /SP/SpFunc/Sp[1]/SpeedWin/AbsWin "Drehzahl erreicht-Fenster in U/min" (1040 00020) und /SP/SpFunc/Sp[1]/SpeedWin/PercWin "Drehzahl erreicht-Fenster in %" (1040 00021) spindelspezifisch definiert wird. Ist die Drehzahl über programmierte Drehzahlgrenzwerte oder Getriebestufendrehzahlgrenzwerte begrenzt, wird dieses Signals ausgegeben, wenn der Grenzwert erreicht ist. Weiterhin wird die Stellung des Override-Schalters berücksichtigt.
	0:	Die Drehzahl an der Spindel befindet sich nicht im Drehzahl-Toleranzband.

Drehzahl begrenzt iSp_SpLim

Dieses Signal zeigt an, dass die programmierte Drehzahl durch die aktive Getriebestufe bzw. durch die programmierte Drehzahlbegrenzung (SMin, SMax) beschränkt wird.

Signalpegel	1:	Die programmierte Drehzahl wird beschränkt.
	0:	Die programmierte Drehzahl wird nicht beschränkt.

Spindel steht iSp_Stop

Dieses Signal bezieht sich auf die Drehzahl der Spindel.

Für Sercos Spindeln wird dieses Signal nur gebildet wenn das Bit 0 der Sercos Identnummer **S-0-0331, Meldung n_ist = 0** zyklisch via S-0-144 vom Antrieb in die NC übertragen wird.

Signalpegel	1:	M5 wurde ausgeführt (über NC-Teileprogramm oder Interface), oder über das NC-Teileprogramm wurde die Drehzahl 0(S 0) vorgegeben, oder Spindel Sperre oder Override 0 wurde gesetzt.
	0:	Die Spindel dreht.



Für Sercos Spindeln beachten Sie bitte die Sercos Identnummer **S-0-0124, Stillstandsfenster**.

Für externe und analoge Spindeln mit Geber beachten Sie bitte den Maschinenparameter **SP/SpFunc/Sp[]/SpeedWin/Standstill-Win** "Stillstandsfenster" (1040 00022).

Spindel gerichtet iSp_OrientateFinish

Zusammen mit diesem Signal sind die Signale **Spindel richten aktiv** und **Spindel in Position** von Bedeutung. Der Funktionszusammenhang ist im Signaldiagramm in [Spindel M19 manuell qSp_Orientate, Seite 101](#) dargestellt.

Signalpegel	1:	Die Funktion Spindel richten ist beendet und die Spindel steht auf der Richtposition.
	0:	Der Richtvorgang ist beendet und die Spindel steht nicht mehr auf der Position, die zuvor als "Richtposition" angefahren wurde.

Spindel richten aktiv iSp_OrientateAct

Dieses Signal wirkt nur in Verbindung mit der Funktion **Spindel richten**. Der Funktionszusammenhang ist im Signaldiagramm (siehe [Kap. 9.2.1 "Ausgangssignale \(SPS --> NC\)" auf Seite 98](#)) dargestellt.

Signalpegel	1:	Die Funktion Spindel richten wurde aktiviert über das NC-Teilprogramm (z. B. Programmierung M19) oder über das Signal qSp_Orientate.
	0:	Spindel richten ist nicht aktiviert.

Si-Betriebsart Bit 0 ... Bit 3 iSp_SafOpMode_00 ... 03

Diese Signale sind Bestandteil der Sicherheitstechnik der Antriebsbaureihe IndraDrive.

Die Steuerung liest die angewählte Betriebsart des Antriebs (P-0-3215 oder P-0-3262) und gibt sie über diese Ausgänge an die SPS weiter.



Die sicheren Bewegungen SMM14 bis SMM16 werden im Bit-Interface nicht angezeigt.

Sicherheitsfunktion	Si-Betriebsart			
	Bit 3 (iSp_SafOpMode_03)	Bit 2 (iSp_SafOpMode_02)	Bit 1 (iSp_SafOpMode_01)	Bit 0 (iSp_SafOpMode_00)
Normalbetrieb NO	0	0	0	0
Sicherer Stopp 1 (NOT-HALT) (SS1ES)	0	0	0	1
Sicherer Stopp 1 (SS1) / Sicherer Stopp 2 (SS2)	0	0	1	0
Sichere Bewegung (SMM1)	0	0	1	1
Sichere Bewegung (SMM2)	0	1	0	0

Sicherheitsfunktion	Si-Betriebsart			
	Bit 3 (iSp_SafOpMode_03)	Bit 2 (iSp_SafOpMode_02)	Bit 1 (iSp_SafOpMode_01)	Bit 0 (iSp_SafOpMode_00)
Sichere Bewegung (SMM3)	0	1	0	1
Sichere Bewegung (SMM4)	0	1	1	0
Sichere Bewegung (SMM5)	0	1	1	1
...				
Sichere Bewegung (SMM13)	1	1	1	1

Tab. 9-5: Si-Betriebsart

Status sichere Lage iSp_SafStatePos

Dieses Signal ist Bestandteil der Sicherheitstechnik der Antriebsbaureihe IndraDrive.

Als Voraussetzung für die Sicherheitsfunktion "Sichere Absolutlage" meldet der Antrieb über dieses Signal, ob sicher referenziert ist.

Signalpegel 1: Der Antrieb ist sicher referenziert.
0: Keine sichere Referenzierung.

Statusausgang Regler iSp_SafCtrlOutputState

Dieses Signal ist Bestandteil der Funktion "Integrierte Sicherheitstechnik nach IEC 61508" der Antriebsbaureihe IndraDrive.

Über dieses Signal gibt der Antrieb Rückmeldung über seinen sicheren Zustand.

Signalpegel 1: Sicherer Zustand des Antriebs.
0: Kein sicherer Zustand des Antriebs.

Anwahl GTS 1 ... 4 iSp_Gear1Sel ... 4Sel

In Abhängigkeit der Getriebestufenanwahl (manuell oder über NC-Teilprogramm) wird die vorgewählte Getriebestufe ausgegeben.

Über das NC-Teilprogramm kann die Getriebestufe direkt durch die Programmierung von **M41...M44** Getriebestufe 1...Getriebestufe 4) angewählt werden. In Verbindung mit der **automatischen Getriebestufenerkennung M40** wird die Getriebestufe von der programmierten **Drehzahl S** abgeleitet.

Sind die Drehzahlbereiche in den einzelnen Getriebestufen überlappend, wird die Getriebestufe ausgegeben, in der die Drehzahl des Spindelantriebs höher ist.

Im Zusammenhang mit der Getriebeumschaltung ist das Signal **Trudeldrehzahl erreicht** von Bedeutung.

Verhalten nach dem Einschalten: Nach dem Einschalten bzw. Hochlauf der Steuerung wird die Getriebestufe ausgegeben, die als **Quittung** vom SPS-Programm über die entsprechenden Eingänge gemeldet wird. Wird keine Getriebestufe quittiert, wählt die NC die höchste in den Maschinenparametern konfigurierte Stufe an.

Sofern nach dem Einschalten die **automatische Getriebestufenerkennung M40** aktiv ist, wird die Getriebestufe erst aus-

Signalflanke 0 → 1:

gegeben, nachdem eine **Drehzahl S...** in Verbindung mit einer **Drehrichtung M3** oder **M4** programmiert wurde.
Für SP/SpFunc/Sp[1..32]/Gear/ChangeMethod = Getriebe gilt:

Mit dieser Flanke soll eine Umschaltung des Getriebes erfolgen. Um den Schaltvorgang möglichst schonend durchzuführen, wird im Spindelantrieb über Sercos Interface die Funktion **Spindel Trudeln** aktiviert (sofern im Spindelantrieb diese Funktion zur Verfügung steht). Sobald die Trudeldrehzahl erreicht ist, setzt die NC das Signal **Trudeldrehzahl erreicht** und der mechanische Schaltvorgang kann ausgeführt werden. Nach Beenden des Schaltvorgangs muss die aktuelle Getriebestufe auf das betreffende Interface **Quittung GTS 1 ... Quittung GTS 4** zurückgemeldet werden. Mit Aktivieren des betreffenden Parametersatzes über Sercos Interface ist die Getriebeumschaltung abgeschlossen.

Für SP/SpFunc/Sp[1..32]/Gear/ChangeMethod = Wicklung gilt:

Mit dieser Flanke soll eine Umschaltung der Wicklung erfolgen. Nach Beenden des Schaltvorgangs muss die aktuell eingeschaltete Wicklung auf das betreffende Interface **Quittung GTS 1 ... Quittung GTS 4** zurückgemeldet werden.

GTS Umschaltung iSp_GearChange

Dieses Signal wirkt nur in Verbindung mit der Funktion Getriebeumschaltung, ausgelöst durch die **M-Funktionen M40, 41, 42**.

Signalpegel	1:	Ein Wechsel der Getriebestufe wird durchgeführt.
	0:	Die Umschaltung der Getriebestufe ist beendet.

Trudeldrehzahl erreicht iSp_IdleSpeed

Dieses Signal wirkt in Verbindung mit der Funktion Getriebeumschaltung und der Rückmeldung der aktuellen Getriebestufe.

Bei der Signalausgabe wird die Stellung des Spindeloverride nicht berücksichtigt.

Signalpegel	1:	Die Spindeldrehzahl entspricht der Trudeldrehzahl. Mit diesem Signal wird üblicherweise das Getriebe umgeschaltet.
	0:	Die Trudeldrehzahl ist noch nicht erreicht.

Anwahl Getriebe-Leerlauf iSp_GearIdleSel

Abhängig von der Getriebestufenanwahl (manuell oder über das NC-Teilprogramm) wird die vorgewählte Getriebestufe bzw. die Leerlaufstellung des Getriebes ausgegeben.

Über das NC-Teilprogramm kann die Leerlaufstellung des Getriebes direkt durch die Programmierung der Funktion **M48** angewählt werden. In Verbindung mit **M48** darf keine Drehrichtung **M3** oder **M4**, kein **Spindel richten M19** sowie keine Drehzahl **S...** programmiert werden.

Im diesem Zusammenhang ist auch das Signal **Trudeldrehzahl erreicht** von Bedeutung.

Signalflanke 0 → 1:

Mit dieser Signalflanke wird das Getriebe in Leerlaufstellung umgeschaltet. Um den Schaltvorgang möglichst schonend durchzuführen, wird im Spindelantrieb über Sercos Interface die Funktion **Spindel Trudeln** aktiviert, sofern im Spindelantrieb diese Funktion zur Verfügung steht.

Sobald die Trudeldrehzahl (eventuell in Verbindung mit einer Pendelbewegung) erreicht ist, setzt die NC das Signal **Trudeldrehzahl erreicht** und der mechanische Schaltvorgang kann ausgeführt werden.

Nach Beenden des Schaltvorgangs muss die Leerlaufstellung des Getriebes am Interface **Quittung Leerlauf** zurückgemeldet werden, die Getriebeumschaltung ist abgeschlossen.

1 → 0: Die Funktion Getriebe in Leerlaufstellung umschalten ist ausgewählt.

Werkzeugbegrenzungen aktiv iSp_ToolLimAct

Dieses Signal zeigt an, dass werkzeugspezifische Begrenzungen aktiv sind.



Weitere Informationen finden Sie in der Dokumentation "MTX Funktionsbeschreibung Grundlagen" im Kapitel "Spindelfunktionen" unter "Werkzeugspezifische Begrenzungen".

Signalpegel	1:	Werkzeugbegrenzungen sind aktiv.
	0:	Werkzeugbegrenzungen sind nicht aktiv.

Spindeloverride 0% iSp_Override0

Dieses Signal zeigt an, dass der Override der Spindel auf 0% geschaltet ist und kein Override 100% über Interface (qSp_Override100) angewählt ist.

Signalpegel	1:	Auf die Spindel wirken 0%.
	0:	Auf die Spindel wirkt ein Wert ungleich 0%.

Spindeloverride 100% iSp_Override100

Dieses Signal zeigt für eine Spindel an, dass ihr Override auf 100% geschaltet ist und/oder Override 100% über Interface (qSp_Override100) angewählt ist.

Signalpegel	1:	Auf die Spindel wirkt ein Wert von 100%.
	0:	Auf die Spindel wirken keine 100%.

SCS-Signalstatus 0 ...15 iSp_ScsState00 ... 15

Im Signalstatuswort (Sercos Parameter S-0-0144) werden die konfigurierten Bit-Signale aus dem zyklischen Telegramm des Spindelantriebs in die NC übertragen.

Die Statussignale lassen sich im Parameter S-0-0026 konfigurieren. Für die Übertragung der Signale im zyklischen Telegramm muss der Parameter S-0-0144 in der Konfigurationsliste des Antriebstelegramm (S-0-0016) aufgenommen werden.

Beispiel:

Konfiguration:

- S-0-0026 = (330, 331, 0, 333, 0, 335, 336, 337, 0, 0, 0, 0, 0, 0, 0):
Konfiguration des Signalstatuswortes
- S-0-0016 = (40, 144):
Konfiguration des Antriebs-Telegramms

Damit werden den einzelnen Signalen folgende Statusmeldungen zugeordnet:

iSp_ScsState00: S-0-0330 Meldung $n_{ist} = n_{soll}$

iSp_ScsState01: S-0-0331 Meldung $n_{ist} = 0$
iSp_ScsState02: -
iSp_ScsState03: S-0-0333 Meldung $M_d \geq M_{dx}$, als "Symbol_D"
iSp_ScsState04: -
iSp_ScsState05: S-0-0335 Meldung $n_{soll} > n_{Grenz}$
iSp_ScsState06: S-0-0336 Meldung In-Position
iSp_ScsState07: S-0-0337 Meldung $P \geq P_x$, als "Symbol_D"
NC-A 7.0: -
NC-A 7.1: -
NC-A 7.2: -
NC-A 7.3: -
NC-A 7.4: -
NC-A 7.5: -
NC-A 7.6: -
NC-A 7.7: -



Eine ausführliche Beschreibung aller zur Verfügung stehenden Statusmeldungen siehe Sercos Interface-Spezifikation.

Kundenausgang 1 ... 8 iSP_Custom1 ... 8

NC-seitig konfigurierbares Interface, das je nach Bedarf belegt werden kann.
(Diese Möglichkeit ist z. Zt. noch nicht aktiv)

Nr. der Kopplung Bit 0 ... Bit 2 iSp_CoupleIndex_00 ... 02

Bei aktiver Spindelkopplung wird mit diesem Signal für jedes Mitglied eines Koppelverbandes die Nummer der Kopplung ausgegeben.

Spindel ist Master iSp_Master

Dieses Signal kennzeichnet den Master eines Koppelverbandes.

1: Spindel ist Master.
0: Spindel ist Slave.

Kopplungsfehler iSp_CoupleErr

Dieses Signal liefert bei aktiver Spindelkopplung (siehe iSp_CoupleIndex_00...02) für Master-Spindeln (iSp_Master gesetzt) eine Information über den aktuellen Zustand des Koppelverbandes.

Signalpegel 1: Der Koppelverband befindet sich in einem Fehlerzustand. Durch "Spindelgrundstellung" der Masterspindel oder "Gesamtgrundstellung" kann dieser Zustand verlassen werden.
0: Der Koppelverband befindet sich im Normalzustand.

Synchronlauf 1 iSp_Synchr1

Dieses Signal hat bei aktiver Spindelkopplung (siehe iSp_CoupleIndex_00...02) für Slave-Spindeln (iSp_Master nicht gesetzt) folgenden Informationsgehalt bezüglich des **Winkelversatzes**.

Signalpegel 1: Das Synchronlaufenster wird nicht verletzt und ein programmierter Winkelversatz ist eingefahren.

- 0: Das Synchronlauffenster wird verletzt oder ein programmierter Winkelversatz wird zur Zeit eingefahren.

Synchronlauf 2 iSp_Synchr2

Dieses Signal hat bei aktiver Spindelkopplung (siehe iSp_CoupleIndex_00...02) für Slave-Spindeln (iSp_Master nicht gesetzt) folgenden Informationsgehalt bezüglich des **Fehlerfensters**.

- Signalpegel** 1: Das Synchronlauffehlerfenster wird nicht verletzt.
0: Das Synchronlauffehlerfenster wird verletzt.

Verzögerte Fehlerreaktion iSp_DelayErrClass1

Dieses Signal steht in Zusammenhang mit iSp_DrvErrClass1 (Fehler Zustandsklasse-1).

Bei aktivierter Funktion "Verzögerte NC-Fehlerreaktion bei nichtfatalen Antriebsfehlern" wird dieses Signal zusammen mit iSp_DrvErrClass1 gesetzt.



Weitere Informationen sind dem Kapitel "Verzögerte NC-Fehlerreaktion bei nichtfatalen Antriebsfehlern" im Handbuch "Funktionsbeschreibung" zu entnehmen.

- Signalpegel** 1: Ein nichtfataler Antriebsfehler ist aufgetreten und der Antrieb räumt der SPS und der NC Zeit zum koordinierten Stillsetzen der Bewegung ein.
0: Es ist keine verzögerte Fehlerreaktion aktiv oder die verzögerte Fehlerreaktion ist beendet.

Fehler Zustandsklasse-1 iSp_DrvErrClass1

Dieses Signal meldet einen Fehler der Zustandsklasse-1 (Antriebsabschaltung mit Verriegelung) und ist ein direktes Abbild des entsprechenden Signals im Sercos Antriebs-Statuswort des betreffenden Antriebs. Die Verriegelung kann nur durch das Kommando "Reset Zustandsklasse 1", das bei Grundstellung automatisch von der NC gesetzt wird, wieder aufgehoben werden.

Eine Auswertung des anstehenden Fehlers kann manuell über die NC-Bedienoberfläche erfolgen oder automatisch durch Lesen von S-0-0011 direkt von der SPS.

Der Parameter S-0-0011 besteht aus 2 Byte und hat folgenden Aufbau:

Bit 0:	-	
Bit 1:	Verstärker-Abschalttemperatur	S-0-0203
Bit 2:	Motor-Abschalttemperatur	S-0-0204
Bit 3:	-	
Bit 4:	Steuerspannungsfehler	
Bit 5:	Feedbackfehler (Geber, Messsystem)	
Bit 6:	Fehler im elektronischen Kommutierungssystem	
Bit 7:	Überstrom	
Bit 8:	Überspannung	
Bit 9:	Unterspannungsfehler	
Bit 10:	Phasenfehler der Leistungsversorgung	

Bit 11:	exzessive Reglerabweichung	S-0-0159
Bit 12:	Kommunikationsfehler	
Bit 13:	Lagegrenzwert überschritten	S-0-0049
Bit 14:	-	
Bit 15:	Herstellerspezifischer Fehler	S-0-0129

Tab. 9-6: Parameter S-0-0011 - Aufbau



Dieses Signal wird nur für Sercos Spindeln (/SP/SpFunc/Sp[1]/Base/SpType "Spindelart" (1040 00001)) gebildet.

Signalpegel	1:	Ein Fehler der Zustandsklasse-1 steht an.
	0:	Es steht kein Fehler der Zustandsklasse-1 an.



Nähere Informationen zu Fehlern der Zustandsklasse 1 siehe Sercos Interface-Spezifikation.

Änderung Zustandsklasse-2 iSp_DrvChangeClass2

Dieses Signal meldet eine Änderung der Zustandsklasse 2 (Abschaltvorwarnung) und ist ein direktes Abbild des entsprechenden Signals im Sercos Antriebs-Statuswort des betreffenden Antriebs.

Mit einer Änderung der Zustandsklasse-2 wird das Bit gesetzt; nach Lesen des Status über den Parameter S-0-0012 wird es wieder zurückgesetzt. Das Lesen kann manuell über die NC-Bedienoberfläche oder automatisch durch Lesen von S-0-0012 direkt von der SPS erfolgen.

Der Parameter S-0-0012 besteht aus 2 Byte und enthält folgende Warnungen:

Bit 0:	-	
Bit 1:	Verstärker-Übertemperatur-Warnung	S-0-0311
Bit 2:	Motor-Übertemperatur-Warnung	S-0-0312
Bit 3 bis 14:	-	
Bit 15:	Herstellerspezifische Warnung	S-0-0181

Tab. 9-7: Parameter S-0-0012 - Warnungen

Über die Maske der Zustandsklasse 2 (S-0-0097) können die Abschaltvorwarnungen bezüglich ihrer Wirkung auf das Änderungsbit ausmaskiert werden.



Dieses Signal wird nur für Sercos Spindeln (/SP/SpFunc/Sp[1]/Base/SpType "Spindelart" (1040 00001)) gebildet.

Änderung Zustandsklasse-3 iSp_DrvChangeClass3

Dieses Signal meldet eine Änderung der Zustandsklasse-3 (Meldungen der Betriebszustände) und ist ein direktes Abbild des entsprechenden Signals im Sercos Antriebs-Statuswort des betreffenden Antriebs. Mit einer Änderung der Zustandsklasse-3 wird das Bit gesetzt, nach Lesen des Status über den Parameter S-0-0013 wird es wieder zurückgesetzt. Das Lesen kann manuell

Spindelbezogene Interface-Signale

über die NC-Bedienoberfläche oder automatisch durch Lesen von S-0-0013 direkt von der SPS erfolgen.

Der Parameter S-0-0013 besteht aus 2 Byte und enthält folgende Meldungen:

Bit 0:	$n_{ist} = n_{soll}$	S-0-0330
Bit 1:	$n_{ist} = 0$	S-0-0331
Bit 2:	$ n_{ist} < n_x $	S-0-0332
Bit 3:	$ Md \geq Md_x $	S-0-0333
Bit 4:	$ Md \geq Md_{grenz} $	S-0-0334
Bit 5:	$ n_{soll} > n_{grenz} $	S-0-0335
Bit 6:	In Position	S-0-0336
Bit 7:	$ P \geq P_x $	S-0-0337
Bit 8:	-	
Bit 9:	$n_{ist} \geq \text{min. Spindeldrehzahl}$	S-0-0339
Bit 10:	$ n_{ist} \geq \text{max. Spindeldrehzahl}$	S-0-0340
Bit 11:	In Position grob	S-0-0341
Bit 12:	Zielposition erreicht	S-0-0342
Bit 13:	Interpolator angehalten	S-0-0343
Bit 14:	-	
Bit 15:	Herstellerspezifische Betriebszustände	S-0-0182

Tab. 9-8: Parameter S-0-0013 - Meldungen

Über die Maske der Zustandsklasse 3 (S-0-0098) können die Betriebszustände bezüglich ihrer Wirkung auf das Änderungsbit ausmaskiert werden.



Dieses Signal wird nur für Sercos Spindeln (/SP/SpFunc/Sp[1]/Base/SpType "Spindelart" (1040 00001)) gebildet.

Testbetrieb iSp_DryRun

Dieses Signal zeigt an, ob eine Spindel auf Testbetrieb geschaltet ist. Im Testbetrieb wird eine Spindel zwar weiterhin intern interpoliert, die berechneten Sollwerte gelangen jedoch lediglich zur Anzeige und werden nicht zum Antrieb übertragen.

Testbetrieb kann über Softkey, automatisch bei nicht angeschlossenen Antrieben (keine Sercos Ring vorhanden) oder über den Maschinenparameter /MAIN/Dr[1]/Enab/VirtMode "Virtueller Antrieb" (1001 00010) eingeschaltet werden.

Freigabe zur Leistungszuschaltung iSp_DrvPower

Dieses Signal steht in engem Zusammenhang mit den Signalen **Anlauf-sperre**, **Antrieb Ein**, **Antrieb betriebsbereit** und **Antrieb in Betrieb**.

Der Funktionszusammenhang ist im Signaldiagramm für das Signal **Anlauf-sperre** ([Antrieb Ein qSp_DrvOn](#), Seite 106) ausführlich dargestellt.

Signalpegel 1: Die Initialisierung des Sercos Interface, des Spindelantriebs und der NC-Steuerung sind abgeschlossen und es liegt kein

Fehler vor. Dieses Signal ist Voraussetzung für das Zuschalten der Leistung an den Antrieb.

0: Die Leistung am Antrieb kann nicht zugeschaltet werden.



Dieses Signal wird nur für Sercos Spindeln (/SP/SpFunc/Sp[1]/Base/SpType "Spindelart" (1040 00001)) gebildet.

Antrieb betriebsbereit iSp_DrvReady

Dieses Signal steht in engem Zusammenhang mit den Signalen **Anlauf-sperre**, **Antrieb Ein**, **Freigabe zur Leistungszuschaltung** und **Antrieb in Betrieb**.

Der Funktionszusammenhang ist im Signaldiagramm für das Signal **Anlauf-sperre** ([Antrieb Ein qSp_DrvOn](#), Seite 106) ausführlich dargestellt.

Signalpegel 1: Die Leistung am betreffenden Antrieb ist zugeschaltet. Das Signal bleibt gesetzt, solange die Leistung nicht abgeschaltet wird.

0: Am Antrieb ist keine Leistung zugeschaltet.



Dieses Signal wird nur für Sercos Spindeln (/SP/SpFunc/Sp[1]/Base/SpType "Spindelart" (1040 00001)) gebildet.

Antrieb in Betrieb iSp_DrvAct

Dieses Signal steht in engem Zusammenhang mit den Signalen **Anlauf-sperre**, **Antrieb Ein**, **Freigabe zur Leistungszuschaltung** und **Antrieb betriebsbereit**.

Der Funktionszusammenhang ist im Signaldiagramm für das Signal **Anlauf-sperre** ([Antrieb Ein qSp_DrvOn](#), Seite 106) ausführlich dargestellt.

Signalpegel 1: Der Regelkreis ist geschlossen und der Antrieb steht unter Drehmoment. Sofern keine weiteren Verriegelungen, wie z. B. **Spindel Sperre** vorliegen, kann die Spindel gedreht werden.

0: Der Regelkreis ist geöffnet, der Antrieb steht nicht unter Drehmoment.



Beim Wechsel von Logisch 1 auf Logisch 0 wird NC-intern Spindel Stopp (M5) ausgelöst.

10 Hilfsfunktionen

10.1 Allgemeines

Die Hilfsfunktionen werden im NC-Teileprogramm programmiert.

Alle in einem NC-Satz programmierten **bitcodierten Hilfsfunktionen** werden parallel auf Merker abgebildet.

Ein Ausgeben von **nicht bitcodierten Hilfsfunktionen** kann über den Maschinenparameter **/AUXF/OutBehav/OutBehavMode** "Ausgabeverhalten der Hilfsfunktionen" (3010 00015) vorgegeben werden:

- 0:** Nur bitcodierte Hilfsfunktionen werden parallel ausgegeben.
- 1:** Alle Hilfsfunktionen eines NC-Satzes werden parallel ausgegeben.

Durch die parallele Übergabe werden Stillstandszeiten an der Maschine vermieden.

Für die Hilfsfunktionen gelten folgende Festlegungen:

- Jeder Kanal verfügt über mehrere Hilfsfunktionen. Die Namen (Hilfsfunktionstypen) der Hilfsfunktionen (M, S, T, ...) werden im Maschinenparameter **/AUXF/ClassDef/Class[1]/AuxFuncName** "Namen der Hilfsfunktionen" (3010 00030) festgelegt.
- *Eine Hilfsfunktion kann als*
 - **Bitcodierte Hilfsfunktion**
 - **32-Bit BCD-codierte Hilfsfunktion**
 - **64-Bit BCD-codierte Hilfsfunktion**
 - **INTEGER-codierte Hilfsfunktion (32 Bit)**
 - **FLOAT-codierte Hilfsfunktion (32 Bit)**
 - **DOUBLE-codierte Hilfsfunktion (64 Bit)**

auf einen Operandenbereich abgebildet werden (Maschinenparameter **/AUXF/ClassDef/Class[1]/EnablAuxFunc**, **/AUXF/ClassDef/Class[1]/AuxFuncChDep**, **/AUXF/ClassDef/Class[1]/Bit**, **/AUXF/ClassDef/Class[1]/Bcd32**, **/AUXF/ClassDef/Class[1]/Int**, **/AUXF/ClassDef/Class[1]/Float**, **/AUXF/ClassDef/Class[1]/Double** "Klassen der Hilfsfunktionen" (3010 00040)). Diese Hilfsfunktionen können sowohl kanalabhängig als auch kanalunabhängig definiert werden.

- Als bitcodierte Hilfsfunktionen können pro Kanal mehrere Hilfsfunktionstypen vereinbart werden (Maschinenparameter **/AUXF/ClassDef/Class[1]/AuxFuncName** "Namen der Hilfsfunktionen" (3010 00030)), z. B. die Hilfsfunktion mit der Kennzeichnung **M** und die Hilfsfunktion mit der Kennzeichnung **H**.
- Die Anfangsadressen der bitcodierten Hilfsfunktionen werden über den Maschinenparameter **/AUXF/Plclf/ChDep/Chlff[1]/AddrChDepBit** "Adressen der kanalspezifischen bitcodierten Hilfsfunktionen" (2060 00009) für jeden Kanal einzeln eingestellt.
- Die Ausgabe der Hilfsfunktionen erfolgt asynchron bzgl. der Bearbeitung der IEC-Tasks. D. h., auch innerhalb der SPS-Bearbeitung kann eine Hilfsfunktion von der CNC an die SPS im Merkerbereich übergeben werden.

Wird eine Hilfsfunktion innerhalb von einem SPS-Zyklus ausgegeben und bereits quittiert, kann die Hilfsfunktionsausgabe nicht mittels der Trace-Funktion aufgezeichnet werden, da diese Aufzeichnung nur am Ende der SPS-Task erfolgt.

- Aus diesem Grund müssen Hilfsfunktionen des Typs **Double** immer quittungspflichtig definiert werden, um die Konsistenz des Hilfsfunktionswertes zu gewährleisten.
Außerdem darf nur vor der Quittierung auf die Werte der Hilfsfunktion zugegriffen werden. Nach der Quittierung kann sich der Wert inkonsistent ändern !
- Zur Quittierung von nicht bitcodierten Hilfsfunktionen darf nur das Änderungssignal (Bit 0) zurückgesetzt werden.
- Werden mehrere nicht bitcodierte Hilfsfunktionen eines Typs definiert, liegen sie alle direkt hintereinander im Merkerbereich ab. An welcher Stelle die Hilfsfunktion eines Typs zu liegen kommt, ist über den Parameter Index (Macoda-Id 3010 00080) definiert. Der Index beginnt jeweils bei 0.

Zuordnung von Hilfsfunktionstypen und Adressen im Merkerbereich:

NC	SPS	IEC-Typ	Macoda-Id für Merkeradresse	
			Kanalabh.	Kanalunabh.
32-Bit BCD	DWORD	MT_AuxFuncBCD32_t	2060 00010	2060 00003[6]
64-Bit BCD	DWORD1, DWORD2	MT_AuxFuncBCD64_t	2060 00011	2060 00003[7]
INTEGER	DINT	MT_AuxFuncInt_t	2060 00012	2060 00020[1]
FLOAT	REAL	MT_AuxFuncReal_t	2060 00013	2060 00020[2]
DOUBLE	LREAL	MT_AuxFuncLReal_t	2060 00014	2060 00020[3]

Tab. 10-1: Hilfsfunktionstypen

10.2 Bitcodierte Hilfsfunktionen

Am häufigsten werden im SPS-Ablaufprogramm die Hilfsfunktionen in Form von Einzelsignalen benötigt. Auf diese Weise wird z. B. das Kühlmittel ein-/ausgeschaltet, eine Klemmung geöffnet/geschlossen oder ein Zylinder aus-/eingefahren.

Um diese Funktionen auszulösen, wird im NC-Teilprogramm eine Hilfsfunktion programmiert. Die decodierten Hilfsfunktionen werden auf einem Operandenbereich (z. B. Merker) abgebildet. Die absolute Größe dieses Bereiches ist von der Anzahl der Hilfsfunktionen abhängig, die decodiert werden sollen sowie von der Anzahl der Kanäle (Maschinenparameter **/MAIN/NofCh** "Kanalanzahl" (9040 00001)).

Die Startadresse des Operandenbereiches wird im Maschinenparameter **/AUXF/Plclf/ChDep/Chlff[1]/AddrChDepBit** "Adressen der kanalspezifischen bitcodierten Hilfsfunktionen" (2060 00009) für jeden Kanal definiert.

Beispiel:

Bitcodierte Hilfsfunktionen

In der NC sind 3 Kanäle definiert, für jeden Kanal sind 128 (Maschinenparameter **/AUXF/NofAuxFunc/ChDep/ChDepBit** "Anzahl der Hilfsfunktionen [2]" (3010 00010)) decodierte Hilfsfunktionen vom Typ **M** deklariert (Maschinenparameter **/AUXF/ClassDef/Class[1]/AuxFuncName** "Namen der Hilfsfunktionen" (3010 00030)), der Operandenbereich soll bei Merker M200.0 für Kanal 0 (Maschinenparameter **/AUXF/Plclf/ChDep/Chlff[1]/AddrChDepBit** "Adressen der kanalspezifischen bitcodierten Hilfsfunktionen [1]" (2060 00009)) liegen. Die Hilfsfunktionen der restlichen Kanäle liegen jeweils direkt hintereinander

(Maschinenparameter `/AUXF/Plclf/ChDep/Chlff[2]/AddrChDepBit` "Adressen der kanalspezifischen bitcodierten Hilfsfunktionen [2]" (2060 00009) -> M240, Maschinenparameter `/AUXF/Plclf/ChDep/Chlff[3]/AddrChDepBit` "Adressen der kanalspezifischen bitcodierten Hilfsfunktionen [3]" (2060 00009) -> M280).

Das Merkerfeld erhält aufgrund dieser Parameter folgende Struktur:

Kanal	M-Funktion	Merkerfeld
0	M31... M0	M203.7 ... M200.0
	M128... M96	M215.7 ... M212.0
1	M31... M0	M243.7 ... M240.0
	M128... M96	M255.7 ... M252.0
2	M31... M0	M283.7 ... M280.0
	M128... M96	M295.7 ... M292.0

Tab. 10-2: Struktur des Merkerfelds

Für jeden Kanal wird in diesem Beispiel ein 16 Byte großes Merkerfeld definiert, in dem **jede bitcodierte Hilfsfunktion einen Merker** belegt.

Die bitcodierten Hilfsfunktionen können **quittungspflichtig** oder **nicht quittungspflichtig** sein, wie in Maschinenparameter `/AUXF/BitAckn/Byte000_015/Byte000 ... /AUXF/BitAckn/Byte176_191/Byte191` "Quittungspflicht bitcodierter Hilfsfunktionen" (3010 00020) definiert.

Die Quittierung erfolgt durch das Rücksetzen der betreffenden bitcodierten Hilfsfunktion durch das SPS-Anwenderprogramm.

Solange eine quittungspflichtige Hilfsfunktion durch das SPS-Anwenderprogramm nicht zurückgesetzt ist, wird der folgende NC-Satz nicht abgearbeitet.

Hilfsfunktionen, die nicht quittungspflichtig sind, müssen ebenfalls durch das SPS-Anwenderprogramm zurückgesetzt werden, um bei einer erneuten Ausgabe der Hilfsfunktion den Wechsel des Signalzustands von Logisch 0 auf Logisch 1 erkennen zu können.

Die bitcodierten Hilfsfunktionen sind in erster Linie kanalbezogen, d. h., es wird in dem jeweiligen Kanal das zugeordnete Bit gesetzt. Bei M10 wird z. B. das 11. Bit des Kanals, innerhalb dessen die Hilfsfunktion ausgegeben wurde, gesetzt.

Je nach Applikation kann das SPS-Anwenderprogramm vereinfacht werden, wenn einige bitcodierten Hilfsfunktionen **global** wirken.

Zu diesem Zweck wird im Merkerfeld eine bestimmte Hilfsfunktion eines beliebigen Kanals der globalen Hilfsfunktion zugewiesen.

10.3 Nicht bitcodierte kanalunabhängige Hilfsfunktionen

Außer den bitcodierten Hilfsfunktionen (z. B. vom Typ M) können Hilfsfunktionen auch im BCD-Code mit 32 Bit Länge oder mit 64 Bit Länge oder als Binärwert im Format INTEGER (32 Bit), FLOAT (32Bit) und DOUBLE (64Bit) ausgegeben werden (u. a. die Hilfsfunktionen vom Typ S).

Nicht bitcodierte kanalunabhängige Hilfsfunktionen werden keinem Kanal zugeordnet und wirken damit in jedem Kanal gleich.

Für jeden nicht bitcodierten Hilfsfunktionstyp wird ein Operandenbereich definiert, der jeweils eine Länge von 8 Byte (32-Bit-BCD, INTEGER und FLOAT) bzw. 12 Byte (64-Bit-BCD und DOUBLE) hat.

Die Adresslage dieser Bereiche wird in dem Maschinenparameter **/AUXF/Plclf/ChIndep/AddrChIndepBcd32** "Operandenadresse (allgemein)" (2060 00003) bzw. **/AUXF/Plclf/ChIndep/AddrChIndepInt**, **/AUXF/Plclf/ChIndep/AddrChIndepFloat**, **/AUXF/Plclf/ChIndep/AddrChIndepDouble** "Operandenadressen: INTEGER, FLOAT, DOUBLE" (2060 00020) als Offset im Merkerbereich definiert.

Beispiel:

Nicht bitcodierte kanalunabhängige Hilfsfunktionen

Die Hilfsfunktion "S" ist als 32 bitcodierte Hilfsfunktion vereinbart.

Die Startadresse für das Merkerfeld der Hilfsfunktion "S" soll bei Merkerbyte M20 beginnen.



Abb. 10-1: Startadresse für das Merkerfeld

Der Steuercode in Merkerbyte M20 hat folgende Bedeutung:

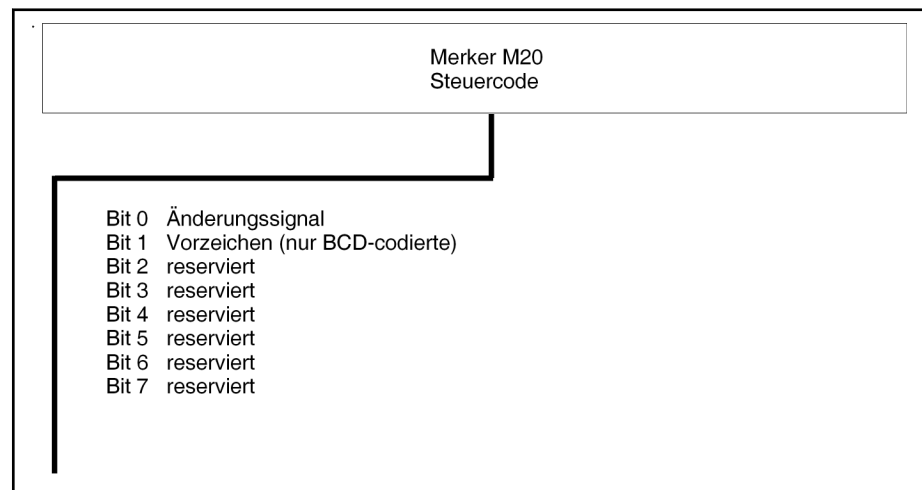


Abb. 10-2: Steuercode Merkerbyte M20

Im vorangehenden Beispiel wird mit der Ausgabe einer BCD-codierten Hilfsfunktion auf das Merkerfeld Byte M24 bis M27 der BCD-Wert übergeben und zusätzlich das Änderungssignal M20.0 gesetzt. Das SPS-Anwenderprogramm muss die Ausgabe der Hilfsfunktion quittieren, indem das Änderungssignal auf Merker M20.0 wieder zurückgesetzt wird.

Sofern im Teileprogramm die Hilfsfunktion mit negativem Vorzeichen programmiert ist, wird zusätzlich der Merker M20.1 gesetzt.

Die Struktur (siehe Beispiel) ist bei jeder BCD-codierten Hilfsfunktion identisch; bei den Hilfsfunktionen mit 64 Bit Länge beträgt die Gesamtlänge des entsprechenden Merkerfeldes 12 Byte statt 8 Byte.

10.4 Nicht bitcodierte kanalabhängige Hilfsfunktionen

Neben den nicht bitcodierten kanalunabhängigen Hilfsfunktionen können diese Hilfsfunktionen auch als kanalabhängig definiert werden, wobei dann jedem Kanal ein Merkerbereich mit den entsprechenden Hilfsfunktionen zugewiesen ist.

Die Adresslage dieser Bereiche wird in Maschinenparametern für jeden Kanal separat als Offset im Merkerbereich definiert (-1: Hilfsfunktionsbereich nicht genutzt):

- **/AUXF/Plclf/ChDep/Chlff[1]/AddrChDepBcd32** "Adressen der kanalspez. 32 Bit BCD-codierten Hilfsfunktionen" (2060 00010)
- **/AUXF/Plclf/ChDep/Chlff[1]/AddrChDepBcd64** "Adressen der kanalspez. 64 Bit BCD-codierten Hilfsfunktionen" (2060 00011)
- **/AUXF/Plclf/ChDep/Chlff[1]/AddrChDepInt** "Adressen der kanalspez. INTEGER-codierten Hilfsfunktionen" (2060 00012)
- **/AUXF/Plclf/ChDep/Chlff[1]/AddrChDepFloat** "Adressen der kanalspez. FLOAT-codierten Hilfsfunktionen" (2060 00013)
- **/AUXF/Plclf/ChDep/Chlff[1]/AddrChDepDouble** "Adressen der kanalspez. DOUBLE-codierten Hilfsfunktionen" (2060 00014)

Beispiel:

Nicht bitcodierte kanalabhängige Hilfsfunktionen

Die Hilfsfunktion "T" ist als kanalabhängige INTEGER-codierte Hilfsfunktion vereinbart.

Die Startadresse für das Merkerfeld der Hilfsfunktion "T" soll beginnen:

- in Kanal 0 bei Merkerfeld M112
- in Kanal 1 bei Merkerfeld M120
- in Kanal 2 bei Merkerfeld M128.

Merker M115 reserviert	Merker M114 reserviert	Merker M113 reserviert	Merker M112 Steuercode
Merker M119 MSB	32 Bit INTEGER-Wert		Merker M116 LSB

Abb. 10-3: Kanalabhängige INTEGER-Hilfsfunktion von Kanal 0

Merker M123 reserviert	Merker M122 reserviert	Merker M121 reserviert	Merker M120 Steuercode
Merker M127 MSB	32 Bit INTEGER-Wert		Merker M124 LSB

Abb. 10-4: Kanalabhängige INTEGER-Hilfsfunktion von Kanal 1

Merker M131 reserviert	Merker M130 reserviert	Merker M129 reserviert	Merker M128 Steuercode
Merker M135 MSB	32 Bit INTEGER-Wert		Merker M132 LSB

Abb. 10-5: Kanalabhängige INTEGER-Hilfsfunktion von Kanal 2

Der Steuercode in Merkerbyte M20 hat folgende Bedeutung:

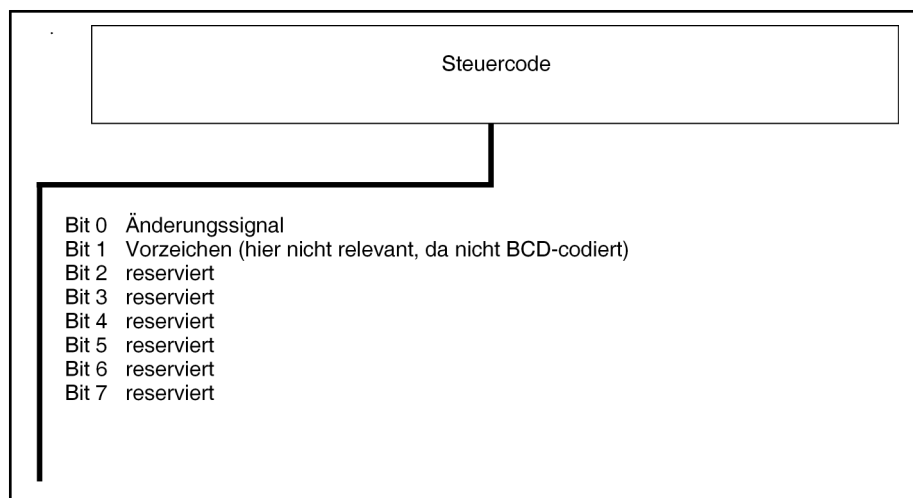


Abb. 10-6: Steuercode Merkerbyte M20

Im vorangehenden Beispiel wird mit der Ausgabe der INTEGER-codierten Hilfsfunktion "T" im Kanal 1 auf das Merkerfeld Byte M124 bis M127 der Binärwert übergeben und zusätzlich das Änderungssignal M120.0 gesetzt. Das SPS-Anwenderprogramm muss die Ausgabe der Hilfsfunktion quittieren, indem das Änderungssignal auf Merker M120.0 wieder zurückgesetzt wird.

Die Struktur (siehe Beispiel) ist bei jeder nicht bitcodierten Hilfsfunktion identisch; bei den Hilfsfunktionen mit 64 Bit Länge beträgt die Gesamtlänge des entsprechenden Merkerfeldes 12 Byte an Stelle von 8 Byte.

11 Anbindung digitale I/Os

Zugriffe auf digitale I/Os aus dem SPS-Anwenderprogramm werden in den Maschinenparametern der Gruppe 4075 angepasst. Hierbei wird die Zuordnung der I/O-Signale in der SPS und die Verwendung der SPS-Adressen eingestellt.

Für das CNC-System MTX ist eine Erweiterungskarte mit digitalen I/Os erhältlich. Darüber hinaus sind dezentrale I/Os über PROFIBUS-DP möglich.

PROFIBUS-DP-Schnittstelle

Das CNC-System MTX verfügt über eine PROFIBUS-DP Schnittstelle, die als **DP-Master Schnittstelle** verwendet werden kann.

Diese Schnittstelle dient zur Anbindung dezentraler PROFIBUS-DP Peripheriebaugruppen.

Die I/O-Konfiguration des PROFIBUS-DP wird über die IndraLogic-Bedienoberfläche erstellt. Zur Analyse von Fehlern kann das Diagnose-Tool DPConf verwendet werden.



Ein Fehler, den der DP-Master im DP-Master-Status meldet, führt zwangsläufig nicht zum STOPP der SPS. Ist eine Fehlerreaktion erforderlich, muss diese im SPS-Anwenderprogramm programmiert werden.

12 Programmbausteine

12.1 Genereller Aufbau von Programmbausteinen

Die integrierten Programmbausteine können vom SPS-Ablaufprogramm zyklisch und parametrisiert aufgerufen werden.

Wird ein Baustein mehrmals aufgerufen, ohne dass die Daten des vorhergehenden Aufrufes bereits zurückgeliefert wurden, ist darauf zu achten, dass eine andere Instanz des Bausteins verwendet wird.

Ist bei der Ausführung einer Funktion ein Fehler entstanden, und die Funktion kann nicht ordnungsgemäß ausgeführt werden, wird ein Fehlercode ausgegeben. Fehler können u. a. durch fehlerhafte Parametrierung, wie z. B. eine unzulässige Kanalnummer entstehen. Weiterhin können systeminterne Fehler auftreten, die ebenfalls mit einem entsprechenden Fehlercode gemeldet werden.

Der **Fehlercode** hat die Länge von 2 Byte.

Im Fehlercode wird die Art des Fehlers oder Zustandes in Form einer vorzeichenbehafteten Zahl dargestellt.

Die Fehler oder Zustände können in zwei Kategorien eingeteilt werden:

1. Allgemeine Fehler und Zustände

Dies sind Fehler oder Zustände, die **beim Aufrufen** einer Funktion durch falsche Parametrierung oder NC-interne Fehler entstanden sind.

2. Funktionsabhängige Fehler und Zustände

Dies sind Fehler oder Zustände, die **als Quittung** nach Ausführen einer Funktion gemeldet werden.



Die Fehlermeldungen sind teilweise funktionsabhängig und werden in der Beschreibung der jeweiligen Funktion erläutert.

Die Fehlercodes werden von Rexroth definiert und in Abhängigkeit der Weiterentwicklungen entsprechend aktualisiert.

Die Funktion startet, wenn "Start" gesetzt wird und "Active" nicht gesetzt ist. Sofern "Start" nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist. "Active" wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und "Active" wird zurückgesetzt. Die Funktion kann innerhalb eines Aufrufes abgeschlossen werden.

Das Programmierbeispiel zum ersten Programmbaustein ist in den SPS-Programmiersprachen KOP, FUP, AWL und ST dargestellt. Alle weiteren Programmierbeispiele sind grundsätzlich in KOP dargestellt.

12.2 Achsenistwerte lesen (MT_ScsPos)

12.2.1 Funktion

Dieser Programmbaustein liefert die zum Zeitpunkt der Abfrage gültigen **Istwerte aller Achsen**.

Der zurückgelieferte Achsenistwert ist auf 0.1 µm normiert und wird als Integerzahl mit 4 Byte Länge dargestellt.

Beim zyklischen Aufruf des Bausteins sollte darauf geachtet werden, dass der Baustein nicht unnötig oft aufgerufen und somit das Restsystem unnötig belastet wird. Dies kann zum Beispiel mittels Timer-Programmierung, zeitge-

gesteuerte Bausteine oder durch den Aufruf des Bausteins nur in jedem n-ten Zyklus erreicht werden.

Der Baustein kann im SPS-Zyklus ein oder mehrmals, absolut oder bedingt aufgerufen werden.

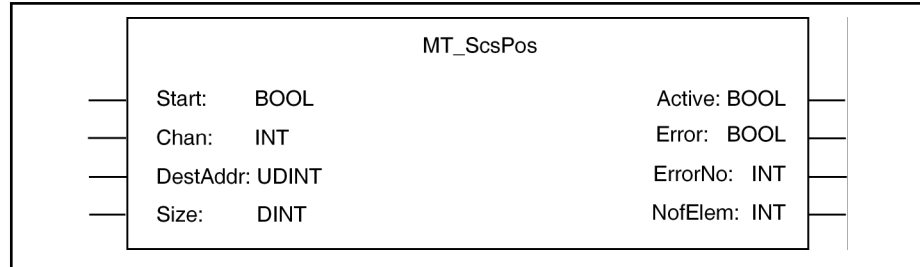


Abb. 12-1: Baustein MT_ScsPos

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung
Chan:	Gewünschte Kanalnummer
DestAddr:	Adresse für die Daten (ADR(<Variable>))
Size:	Max. Größe des Speichers für die Daten in Bytes (SIZEOF(<Variable>))

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE
NofElem:	Anzahl der zurückgelieferten Elemente



Unabhängig von der übergebenen Kanalnummer liefert der Baustein die Istwerte aller Achsen.

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

12.2.2 Projektierungsbeispiel

Im Kanal 1 sind 3 Achsen definiert. Die Achsenistwerte sollen mit der positiven Flanke des SPS-Eingangs %IX40.2 abgefragt werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX40.2 ausgelöst.

Mit diesen Vorgaben könnte der Aufruf des Programmbausteins **MT_ScsPos** wie folgt gestaltet werden.

Programmbeispiel in ST (Deklaration)

```
VAR
  R_TRIG_Inst :   R_Trig;
  StartGetAxPos : BOOL;
  waitForData  :   BOOL;
  Channel      :   INT;
  AxesPos      :   ARRAY[0..2] OF DINT;
  GetAxesPos   :   MT_ScsPos;
```



```

AxPos_X :      DINT;
AxPos_Y :      DINT;
AxPos_Z :      DINT;
END_VAR

```

Programmbeispiel in ST (Implementation)

```

(* Definitions *)
Channel := 1;
AxPos_X := 0;
AxPos_Y := 0;
AxPos_Z := 0;

(* Requesting the axis positions via a rising edge of IX 40.2 *)
R_TRIG_Inst ( CLK := %IX40.2 );
StartGetAxPos := R_TRIG_Inst.Q;

(* Requesting the axis positions *)
GetAxesPos (Start:=StartGetAxPos, Chan:=Channel, DestAddr:=ADR(AxesPos), Size:=SIZEOF(AxesPos));

(* Set waitForData flag as long as the function is active *)
IF ( GetAxesPos.Active = TRUE ) THEN
  waitForData := GetAxesPos.Active;
END_IF
IF ( GetAxesPos.Active OR NOT waitForData ) THEN
  RETURN;
END_IF

(* Reset waitForData flag *)
waitForData := FALSE;

(* Error handling when the error occurs *)
IF ( GetAxesPos.Error = TRUE ) THEN
  (* Error handling *)
  RETURN;
ELSE
  IF ( GetAxesPos.NofElem >= 3 ) THEN
    (* Save new axis position *)
    AxPos_X := AxesPos[0];
    AxPos_Y := AxesPos[1];
    AxPos_Z := AxesPos[2];
  END_IF
END_IF

```

12.2.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

1:	Ungültige Kanalnummer
-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-3:	Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
-1:	allgemeiner Parameterfehler

12.3 Programmanwahl/-abwahl (MT_ProgSel)

12.3.1 Funktion

Dieser Programmbaustein ermöglicht die **Anwahl und die Abwahl eines NC-Teilprogramms** über das SPS-Ablaufprogramm.

Mit dem Mode 3 (**Programmwechsel**) kann ein neues Programm angewählt werden, ohne dass die modalen Funktionen zurückgesetzt werden. Hiermit kann z. B. bei laufender Spindel ein neues Programm angewählt werden,

wenn im Einschaltsatz "M5" entfernt wird. Voraussetzung ist hierbei, dass das letzte Programm fertig abgearbeitet ist oder noch nicht gestartet wurde.

Der Baustein Externe Programmanwahl/-abwahl kann absolut oder bedingt aufgerufen werden. Sind mehrere Kanäle aktiviert, kann der Baustein pro SPS-Zyklus auch mehrmals aufgerufen werden.

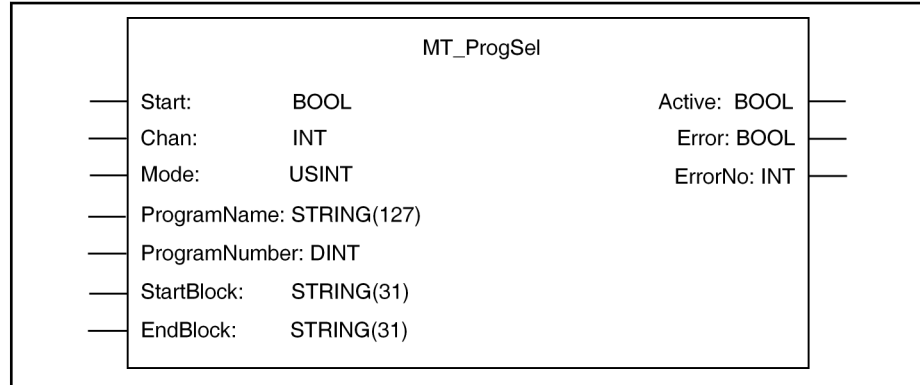


Abb. 12-2: Baustein Programmanwahl/-abwahl

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung
Chan:	Gewünschte Kanalnummer
Mode:	1 Programmanwahl 2 Programmanwahl mit automatischer Abwahl des alten Programms 3 Programmwechsel 4 Programmabwahl
ProgramName:	Programmname (incl. Pfad)

Wird im Parameter **ProgramName** ein Leerstring übergeben, wird als Programmname die Programmnummer mit einem vorangestellten "P" verwendet. Der Baustein MT_ProgSel wandelt die Programmnummer in ASCII-Zeichen um, da die Teileprogramme im CNC-System MTX nicht numerisch, sondern zeichenorientiert abgelegt sind.

Wird die Auswahl des Programms über den Programmnamen (ASCII-String) vorgegeben, ist darauf zu achten, dass der Programmname nicht mehr als 30 Zeichen beinhaltet (incl. dem ASCII-Zeichen NUL als Endekennung).

Bei Programmen, die gelinkt werden müssen (Programme oder Unterprogramme, die CPL-Anweisungen enthalten) sind nur 28 Zeichen zulässig, da für die Extension bei der Linktabelle noch zusätzlich 2 Zeichen benötigt werden.

Soll das Programm bei der Programmanwahl nicht nach dem, in den Maschinenparametern definierten Suchpfad innerhalb der NC angewählt werden, kann der Programmnamen mit dem gewünschten Verzeichnis übergeben werden. Die Gesamtlänge von 127 Zeichen darf hierbei jedoch nicht überschritten werden. Um zum Beispiel das Programm "test.cpl" aus dem Benutzer-FEPROM anzuwählen, muss in ProgramName "/usrfep/test.cpl" übergeben werden.

ProgramNum- ber:	Programmnummer
StartBlock:	Startsatz (Leerstring: kein Startsatz)
EndBlock:	Endsatz (Leerstring: kein Endsatz)

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE

Um ein NC-Teilprogramm anwählen zu können, muss die Betriebsart **Automatik** aktiv sein.

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Programmanwahl/-abwahl durchgeführt ist, ist die Funktion beendet, und Active wird zurückgesetzt.

Der Parameter **Mode** legt fest, ob ein Programm angewählt oder ein bereits aktives Programm abgewählt werden soll.

Bei der Abwahl eines Programms muss lediglich die Kanalnummer angegeben werden. Die Kanalnummer wird immer als Integerzahl angegeben.

Wird im Parameter **ProgramName** ein Leerstring übergeben, wird als Programmname die Programmnummer mit einem vorangestellten "P" verwendet. Der Baustein MT_ProgSel wandelt die Programmnummer in ASCII-Zeichen um, da die Teilprogramme im CNC-System MTX nicht numerisch, sondern zeichenorientiert abgelegt sind.

Wird die Auswahl des Programms über den Programmnamen (ASCII-String) vorgegeben, ist darauf zu achten, dass der Programmname nicht mehr als 29 Zeichen beinhaltet und die Zeichenkette mit dem ASCII-Zeichen NUL abgeschlossen wird. Soll das Programm bei der Programmanwahl nicht nach dem, in den Maschinenparametern definierten Suchpfad innerhalb der NC angewählt werden, kann der Programmname mit dem gewünschten Verzeichnis übergeben werden. Die Gesamtlänge von 30 Zeichen darf hierbei jedoch nicht überschritten werden. Um zum Beispiel das Programm "test.cpl" aus dem Benutzer-FEPROM anzuwählen, muss als Programmname "/usrfep/test.cpl" übergeben werden.

Soll das Teilprogramm bei einem bestimmten Satz gestartet werden, kann neben dem Programmnamen auch ein **Startsatz** vorgegeben werden. Der Startsatz kann ein NC-Satz nach DIN 66025 (beginnt immer mit "N") oder ein CPL-Satz (kein "N" vor der Satznummer) sein. Der Startsatz wird daher als Zeichenkette vorgegeben. Die Länge des Startsatzes kann max. 9 ASCII-Zeichen betragen und muss am Ende mit dem ASCII-Zeichen NUL abgeschlossen werden.

Bearbeitet das Teilprogramm nicht bis zu Ende (M30), sondern soll bereits früher beendet werden, kann ein **Endsatz** definiert werden. Der Endsatz wird nach den gleichen Kriterien wie der Startsatz definiert.

Ist ein Programm noch aktiv, während ein anderes Programm angewählt werden soll, genügt es, eine Programmanwahl über den Mode 2 oder 3 durchzu-

führen. Voraussetzung für die damit verbundene automatische Programmauswahl ist, dass sich die NC im Zustand **Bereit** oder **Hält** befindet.

12.3.2 Projektierungsbeispiel

Im Kanal 1 soll das NC-Teilprogramm mit dem Programmnamen "Labor.npg" aktiviert werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX40.3 ausgelöst.

Der Aufruf des Bausteins **MT_ProgSel** könnte folgendermaßen ausgeführt werden:

Programmbeispiel in ST (Deklaration)

```
VAR
  R_TRIG_Inst :    R_Trig;
  StartProgSelect : BOOL;
  Mode :          USINT;
  Channel :       INT;
  ProgName :      STRING(31);
  ProgramSelect : MT_ProgSel;
  activeProgSelect : BOOL;
END_VAR
```

Programmbeispiel in ST (Implementation)

```
(* Definitions *)
Mode := 1;
Channel := 1;
ProgName := 'Labor.npg';

(* Rising edge -->StartProgSelect *)
R_TRIG_Inst ( CLK := %IX40.3 );
StartProgSelect := R_TRIG_Inst.Q;

(* Call ProgramSelect *)
ProgramSelect (Start:=StartProgSelect, Chan:=Channel, Mode:= Mode,
               Programname:=ProgName, StartBlock:='', EndBlock := '');
activeProgSelect := ProgramSelect.Active;

(* Error handling *)
IF ( ProgramSelect.Error = TRUE ) THEN
  (* Error handling *)
END_IF
```

12.3.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-
-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-1:	allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

-
1:	Ungültige Kanalnummer

Allgemeine Fehler und Zustände bei Programmanwahl:

-
2:	Kanal ist belegt
3:	Im Kanal wird z. Zt. die Funktion Grundstellung ausgeführt
4:	Programmname ist zu lang

- 5: Keine Referenzpunkte angefahren
- 6: Programm kann nicht geöffnet werden
- 7: Fehler während der Programmanwahl
- 8: Es ist keine oder eine falsche Betriebsart angewählt
- 25: Syntaxfehler im angewählten Programm

Allgemeine Fehler und Zustände bei Programmbwahl:

-
- 3: Im Kanal wird zur Zeit die Funktion Grundstellung ausgeführt
 - 10: Es ist kein Programm angewählt
 - 11: Programm läuft und Vorschub Halt steht nicht an

12.4 NC-Satzvorgabe (MT_NcBlk)

12.4.1 Funktion

Dieser Programmbaustein ermöglicht die Vorgabe eines beliebigen NC-Satzes über das SPS-Ablaufprogramm. Der NC-Kanal muss sich hierbei in der Betriebsart Handeingabe oder Automatik befinden. Die Vorgabe eines NC-Satzes ist auch während eines laufenden Programms möglich, wenn hierbei keine Achsadressen vorgegeben werden, die Auswirkungen auf die laufende Bearbeitung haben. Auf diese Weise können z. B. Sollwerte für Hilfsachsen oder Hilfsfunktionen an die NC übergeben werden.

Wird ein NC-Satz vorgegeben, der sich auf die laufende Bearbeitung auswirken würde, wie z. B. Vorgeben eines Vorschubs oder Programmieren einer synchronen Achse, führt dies zu einem Syntaxfehler.

Der vorgegebene Satz muss vollständig abgearbeitet sein, bevor im gleichen Kanal erneut ein Satz vorgegeben werden kann.

Der Baustein kann absolut oder bedingt aufgerufen werden.

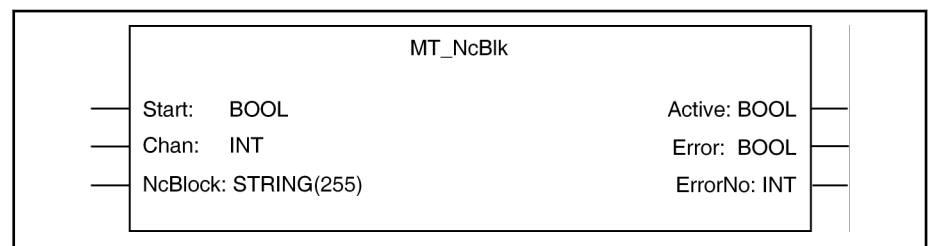


Abb. 12-3: Baustein MT_NcBlk

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

-
- Start:** Flag für eine neue Anforderung
 - Chan:** Gewünschte Kanalnummer
 - NcBlock:** Abzuarbeitender NC-Satz

Die Rückgabeparameter lauten:

-
- Active:** Flag, das anzeigt, ob die Funktion noch aktiv ist
 - Error:** Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
 - ErrorNo:** Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern **Start** nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist, d. h., der vorgegebene Satz aktiv geworden ist. Die Bewegung muss aber noch nicht abgeschlossen sein.

Bei der Vorgabe einer asynchronen Bewegung, deren Endposition noch nicht erreicht ist, wird auf diese Weise sicher gestellt, dass der Fahrbefehl für die Achse zu dem Zeitpunkt, in dem die Funktion beendet ist, bereits gesetzt ist.

Mit Beenden der Funktion wird der Status Active zurückgesetzt.

Die NC arbeitet den NC-Satz sofort nach dem Empfang ohne zusätzliches NC-Start-Signal ab.

12.4.2 Projektierungsbeispiel

Im Kanal 1 soll der NC-Satz "G91 X10 Y5 Z1 F10000" übergeben werden.

Der Aufruf des Bausteines **MT_NcBlk** könnte folgendermaßen ausgeführt werden:

Programmbeispiel in ST (Deklaration)

```
VAR
  R_TRIG_Inst :    R_Trig;
  StartNcBlockReq : BOOL;
  Channel :       INT;
  NcSatz :        STRING(255);
  NcBlockReq :    MT_NcBlk;
  activeNcBlockReq : BOOL;
END_VAR
```

Programmbeispiel in ST (Implementation)

```
(* Definitions *)
Channel := 1;
NC block := 'G91 X10 Y5 Z1 F10000';

(* Rising edge --> StartNcBlockReq *)
R_TRIG_Inst ( CLK := %IX41.3 );
StartNcBlockReq := R_TRIG_Inst.Q;

(* Call NcBlockReq *)
NcBlockReq (Start:=StartNcBlockReq, Chan:=Channel, NcBlock:=NcSatz);
activeNcBlockReq := NcBlockReq.Active;

(* Error handling *)
IF ( NcBlockReq.Error = TRUE ) THEN
  (* Error handling *)
END_IF
```

12.4.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-
-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-1:	allgemeiner Parameterfehler

Funktionsabhängige Fehler und Zustände als Quittung nach Ausführen der Funktion:

1:	Unzulässige Kanalnummer
2:	Kanal ist belegt
3:	Im Kanal wird z. Zt. die Funktion Grundstellung ausgeführt
4:	NC-Satz ist zu lang
5:	Keine Referenzpunkte angefahren
7:	NC-Satzvorgabe wurde durch Grundstellung abgebrochen
8:	Es ist keine oder eine falsche Betriebsart angewählt
25:	Bei NC-Satzvorgabe wurde ein Laufzeitfehler erkannt, z. B. ein Syntaxfehler

12.5 Erweiterte NC-Satzvorgabe (MT_NcBlkExt)

12.5.1 Funktion

Der Programmbaustein **Erweiterte NC-Satzvorgabe** ermöglicht die Vorgabe eines beliebigen NC-Satzes über das SPS-Ablaufprogramm. Über den Eingangsparameter **Options** kann die Bearbeitung der Vorgabe auf NC-Seite beeinflusst werden. Der NC-Kanal muss sich hier bei in der Betriebsart Handeingabe oder Automatik befinden. Die Vorgabe eines NC-Satzes ist auch während eines laufenden Programms möglich, wenn hierbei keine Achsadressen vorgegeben werden, die Auswirkungen auf die laufende Bearbeitung haben. Auf diese Weise können z. B. Sollwerte für Hilfsachsen oder Hilfsfunktionen an die NC übergeben werden.

Wird ein NC-Satz vorgegeben, der sich auf die laufende Bearbeitung auswirken würde, wie z. B. Vorgeben eines Vorschubs oder Programmieren einer synchronen Achse, führt dies zu einem Syntaxfehler.

Der vorgegebene Satz muss vollständig abgearbeitet sein, bevor im gleichen Kanal erneut ein Satz vorgegeben werden kann.

Der Baustein kann absolut oder bedingt aufgerufen werden.

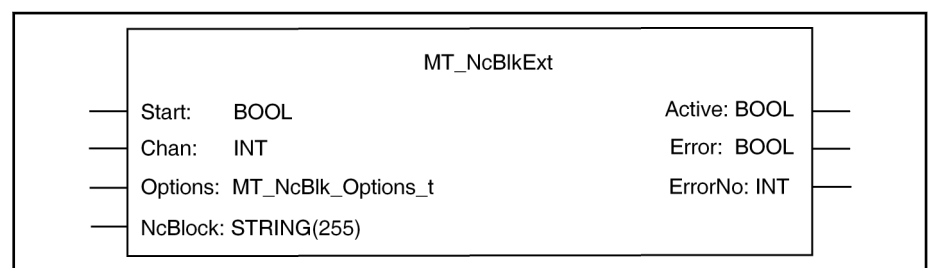


Abb. 12-4: Aufruf des Bausteins MT_NcBlkExt

Bausteinparameter

Der Baustein wird mit den Parametern aufgerufen:

Start:	Flag, das eine neue Anforderung durchführt
Chan:	Gewünschte Kanalnummer

Options: Options ist vom Typ "MT_NcBlk_Options_t" und kann zur Steuerung der Satzvorgabe folgende Werte annehmen:

NO_AUTO_START

Kein automatischer Start nach der Aktivierung des NC-Satzes. Die Bearbeitung des NC-Satzes muss über das Signal qCh_Nc Start durch die SPS gestartet werden.

NO_CHAN_CHANGE

Keine automatische Umleitung des Auftrages in den AutoStartkanal (Hilfskanal), falls der übergebene Kanal belegt ist.

PROCESS_DELETE

Falls im Kanal, innerhalb dessen die Satzvorgabe durchgeführt wird, ein Programm aktiv ist (Bereit) oder Vorschub Halt aktiv ist, wird dieses ausgewählt und der vorgegebene NC-Satz aktiv

NcBlock: Abzuarbeitender NC-Satz

Die Rückgabeparameter lauten:

Active: Flag, das anzeigt, ob die Funktion noch aktiv ist

Error: Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde

ErrorNo: Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn das Bit **Start** gesetzt wird und Active nicht gesetzt ist. Sofern das Bit Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist, d. h., der vorgegebene Satz aktiv geworden ist. Die Bewegung muss aber noch nicht abgeschlossen sein.

Bei der Vorgabe einer asynchronen Bewegung, deren Endposition noch nicht erreicht ist, wird auf diese Weise sicher gestellt, dass der Fahrbefehl für die Achse zu dem Zeitpunkt, in dem die Funktion beendet ist, bereits gesetzt ist. Das Ende der Verfahrbewegung kann über das Interfacesignal "iAx_InPos" abgefragt werden.

Mit Beenden der Funktion wird der Status Active zurückgesetzt.

Ist das Optionbit NO_AUTO_START nicht gesetzt, arbeitet die NC den NC-Satz sofort nach dem Empfang ohne zusätzliches NC-Start-Signal ab.

12.5.2 Projektierungsbeispiel

Der NC-Satz "G91 X10 Y5 Z1 F10000" soll im Kanal 1 übergeben werden und es soll **keine Umleitung** in den Autostartkanal erfolgen.

Der Aufruf des Bausteins **MT_NcBlkExt** könnte folgendermaßen ausgeführt werden:

Programmbeispiel in ST (Deklaration)

```
VAR
  R_TRIG Inst :    R_Trig;
  StartNcBlockReq : BOOL;
  Channel :       INT;
  NcSatz :        STRING(255);
  Options :       MT_NcBlk_Options_t;
  NcBlockReq :    MT_NcBlkExt;
  activeNcBlockReq : BOOL;
END_VAR
```


Programmbeispiel in ST (Implementation)

```

(* Definitions *)
Channel := 1;
NC_block := 'G91 X10 Y5 Z1 F10000';
Options := Options.NO_CHAN_CHANGE;

(* Rising edge --> StartNcBlockReq *)
R_TRIG_Inst ( CLK := %IX41.3 );
StartNcBlockReq := R_TRIG_Inst.Q;

(* Call NcBlockReq *)
NcBlockReq (Start:=StartNcBlockReq, Chan:=Channel, Options:=Options, NcBlock:=NcSatz);
activeNcBlockReq := NcBlockReq.Active;

(* Error handling *)
IF ( NcBlockReq.Error = TRUE ) THEN
  (* Error handling *)
END_IF

```

12.5.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-
-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-1:	allgemeiner Parameterfehler

Funktionsabhängige Fehler und Zustände als Quittung nach Ausführen der Funktion:

-
1:	Unzulässige Kanalnummer
2:	Kanal ist belegt
3:	Im Kanal wird z.Z. die Funktion Grundstellung ausgeführt
4:	NC-Satz ist zu lang
5:	Keine Referenzpunkte angefahren
7:	NC-Satzvorgabe wurde durch Grundstellung abgebrochen
8:	Es ist keine oder eine falsche Betriebsart ausgewählt
25:	Bei NC-Satzvorgabe wurde ein Laufzeitfehler erkannt, z. B. ein Syntaxfehler

12.6 Kanal-Neustart (MT_ChanRestart)

12.6.1 Funktion

Der Programmbaustein **Kanal-Neustart** startet den Kanal vom Beginn des angewählten Hauptprogramms neu. Es wird keine Grundstellung ausgeführt, d. h. auch der Einschaltzustand bei Grundstellung wird nicht übernommen.

ACHTUNG: Die modalen Zustände werden nicht modifiziert! Am Anfang des Hauptprogramms müssen alle im Verlauf des Programms benutzten modalen Funktionalitäten zurückgesetzt werden, da alle zum Zeitpunkt des Kanal-Neustarts aktiven modalen Funktionen weiterhin aktiv sind.

Der Baustein kann absolut oder bedingt aufgerufen werden.



Abb. 12-5: Baustein MT_ChanRestart

Bausteinparameter

Der Baustein wird mit den Parametern aufgerufen:

- Start:** Flag, das eine neue Anforderung durchführt
- Chan:** Gewünschte Kanalnummer
- Options:** Für zukünftige Erweiterungen.
Options ist vom Typ "MT_ChanRestart_Options_t".

Die Rückgabeparameter lauten:

- Active:** Flag, das anzeigt, ob die Funktion noch aktiv ist
- Error:** Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde.
Genauere Diagnosen sind über ErrorID und ErrorIdent möglich
- ErrorId:** Fehlercode: nur gültig, wenn Error = TRUE
- ErrorIdent:** Detaillierte Fehlerbeschreibung (für künftige Erweiterungen)

Die Funktion startet, wenn das Bit **Start** gesetzt wird und Active nicht gesetzt ist. Sofern das Bit Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active ist während der Bearbeitung des Kanal-Neustarts gesetzt.

Mit Beenden der Funktion wird der Status Active zurückgesetzt.

12.6.2 Projektierungsbeispiel

Der in ChanNo eingetragene Kanal soll neu gestartet werden.

Der Aufruf des Bausteins **MT_ChanRestart** könnte folgendermaßen ausgeführt werden:

Programmbeispiel in ST (Deklaration)

```
VAR
  R_TRIG_Inst :      R_Trig;
  StartChanRestart : BOOL;
  ChanNo :          INT;
  Options :         MT_ChanRestart_Options_t;
  ChanRestartReq :  MT_ChanRestart;
  activeChanRestartReq : BOOL;
END_VAR
```

Programmbeispiel in ST (Implementation)

```
(* Definitions *)
Channel := 1;
Options := 0;

(* Increasing edge --> StartChanRestart *)
R_TRIG_Inst ( CLK := %IX41.2 );
StartChanRestart := R_TRIG_Inst.Q;

(* Call ChanRestartReq *)
ChanRestartReq (Start:=StartChanRestartReq, Chan:=ChanNo, Options:=Options);
activeChanRestartReq := ChanRestartReq.Active;

(* Error handling *)
```

```

IF ( ChanRestartReq.Error = TRUE ) THEN
  (* Error handling *)
END_IF

```

12.6.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorId" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

- 6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
- 1: allgemeiner Parameterfehler

Funktionsabhängige Fehler und Zustände als Quittung nach Ausführen der Funktion:

- 2: Kanal-Neustart ist bereits aktiv
- 37: Der Kanal-Neustart ist wegen eines Kanal-Fehlers nicht möglich
- 38: Kanal-Neustart ist aufgrund der aktuellen Betriebsart oder wegen des aktuellen Zustands des Kanals unzulässig

12.7 Temperaturkompensation (MT_TempComp)

12.7.1 Funktion

Dieser Programmbaustein ermöglicht die Vorgabe eines Korrekturwertes zur Kompensation des Temperatureinflusses über das SPS-Ablaufprogramm. Die Temperaturkompensation übergibt das SPS-Programm für eine bestimmte Anzahl von Achsen und wird sofort nach der Übergabe aktiv. Der Korrekturwert wirkt nicht positionsabhängig.

Um Servo-Fehler zu vermeiden, dürfen die Differenzen der Korrekturwerte das Achssprungvermögen der Achsen nicht überschreiten.

Der Baustein kann absolut oder bedingt aufgerufen werden.

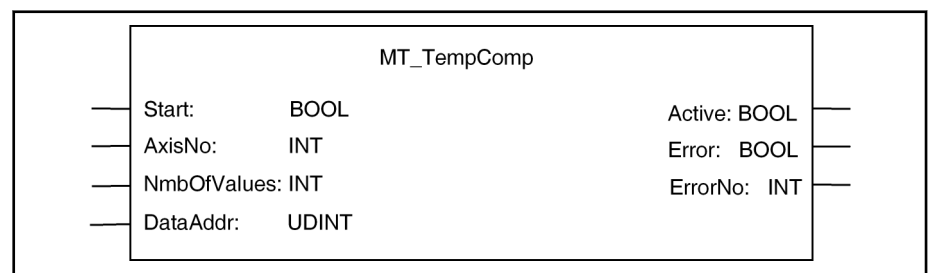


Abb. 12-6: Aufruf Baustein MT_TempComp

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

- Start:** Flag für eine neue Anforderung
- AxisNo:** Physikalischer Achsindex, ab der die Werte übernommen werden
- NmbOfValues:** Anzahl der Korrekturwerte
- DataAddr:** Adresse der Daten (Feld von LREAL-Werten)

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die NC die Funktion quittiert hat, ist die Funktion beendet, und Active wird zurückgesetzt.

In **AxisNo** wird der physikalische Achsindex und in **NmbOfValues** die Anzahl der Korrekturwerte vorgegeben. Auf NC-Seite werden die übergebenen Temperaturkompensationswerte ab der Achse mit dem übergebenen Achsindex *v* errechnet. Der Achsindex beginnt mit 1 für die 1. Achse.

Die Temperaturkompensation wird von der NC sofort übernommen.

12.7.2 Projektierungsbeispiel

Für die 4. Achse soll zur Kompensation des Temperatureinflusses eine Korrektur von 123 µm, für die 5. Achse eine Korrektur von 234 µm vorgegeben werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX41.3 ausgelöst.

Der Aufruf des Bausteines **MT_TempComp** könnte folgendermaßen ausgeführt werden:

Programmbeispiel in ST (Deklaration)

```

VAR
  R_TRIG_Inst :      R_Trig;
  StartTempComp :    BOOL;
  AxesNumber :      INT;
  TempValues :       ARRAY[0..5] OF LREAL;
  TempCompReq :      MT_TempComp;
  activeTempCompReq : BOOL;
END_VAR

```

Programmbeispiel in ST (Implementation)

```

(* Definitions *)
AxesNumber := 2;
TempValues[4] := 0.123;
TempValues[5] := 0.234;

(* Rising edge --> StartTempComp *)
R_TRIG_Inst ( CLK := %IX41.3 );
StartTempComp := R_TRIG_Inst.Q;

(* Call TempCompReq *)
TempCompReq (Start:=StartTempComp, AxisNo:=AxesNumber, NmbOfValues:=2, DataAddr:=AD,
activeTempCompReq := TempCompReq.Active;

(* Error handling *)
IF ( TempCompReq.Error = TRUE ) THEN
  (* Error handling *)
END_IF

```

12.7.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-1:	allgemeiner Parameterfehler

Funktionsabhängige Fehler und Zustände als Quittung nach Ausführen der Funktion:

16:	Die Anzahl der Korrekturwerte ist zu hoch (eventuell in Verbindung mit der Achsnummer)
24:	Unzulässige Achsnummer
33:	Der letzte Korrekturwert wurde vom Interpolator noch nicht übernommen

12.8 Maschinenparameter lesen (MT_CfgData)

12.8.1 Funktion

Dieser Programmbaustein liefert die Werte eines Maschinenparameterblocks.

Die Variable für die Werte muss groß genug gewählt werden, um alle Einzelparameter des Blocks aufnehmen zu können.

Das Format der Werte wird vom Baustein nicht gewandelt, so dass reelle Werte auch in diesem Format zurückgeliefert werden.

Der Baustein kann im SPS-Zyklus ein oder mehrmals, absolut oder bedingt aufgerufen werden.

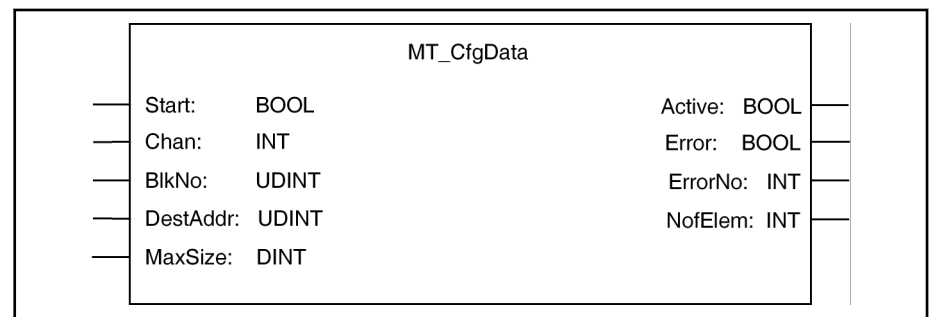


Abb. 12-7: Aufruf Baustein MT_CfgData

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung
Chan:	Gewünschte Kanalnummer bei kanalspezifischen Maschinenparametern
BlkNo:	Nummer des Maschinenparameterblocks
DestAddr:	Adresse für die Daten
Size:	Max. Größe des Speichers für die Daten in Bytes

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE
NofElem:	Anzahl der zurückgelieferten Elemente

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

Beim Lesen eines kanalabhängigen Blocks werden die Werte gelesen, die sich auf den in **Chan** übergebenen Kanal beziehen. Ist der in Chan übergebene Wert -1, wird die Grundeinstellung des Blocks zurückgeliefert.

Bei kanalunabhängigen Parametern wird der Wert in Chan ignoriert.

12.8.2 Projektierungsbeispiel

Die Werte des Maschinenparameters **/SP/SpFunc/Sp[1]/Gear/GearStep[1]/MaxSpAcc** "Maximale Spindelbeschleunigung" (1040 00031) sollen mit der positiven Flanke des SPS-Eingangssignals %IX40.7 abgefragt werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX40.7 ausgelöst.

Mit diesen Vorgaben könnte der Aufruf des Programmbausteins **MT_CfgData** folgendermaßen gestaltet werden:

Programmbeispiel in ST (Deklaration)

```

VAR
  R_TRIG_Inst :      R_Trigger;
  StartCfgData :    BOOL;
  Channel :         INT;
  BlockNumber :     UDINT;
  BlockValue :      ARRAY[0..2] OF DINT;
  GetConfigData :   MT_CfgData;
  activeGetConfigData : BOOL;
  MaxSpeedUp :      UDINT;
END_VAR

```

Programmbeispiel in ST (Implementation)

```

(* Definitions *)
Channel := 1;
BlockNumber := 104000031;

(* Rising edge --> StartCfgData *)
R_TRIG_Inst ( CLK := %IX40.7 );
StartCfgData := R_TRIG_Inst.Q;

(* Call GetConfigData *)
GetConfigData (Start:=StartGetCfgData, Chan:=Channel, BlkNo:=BlockNumber,
               DataAddr:=ADR(BlockValue), MaxSize:=SIZEOF(BlockValue));
activeGetConfigData := GetConfigData.Active;

(* Error handling *)
IF ( GetConfigData.Error = TRUE ) THEN
  (* Error handling *)
END_IF

(* Transfer: Save maximum spindle acceleration of the first spindle *)
MaxSpeedUp := BlockValue[1];

```

12.8.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-3:	Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
-1:	allgemeiner Parameterfehler

Funktionsabhängige Fehler und Zustände als Quittung nach Ausführen der Funktion:

100:	Unzulässige Maschinenparameter-Nummer

12.9 Handraddaten (MT_Handw)

12.9.1 Funktion

Für Handräder in Verbindung mit PROFIBUS-DP übergibt dieser Programmbaustein den aktuellen Wert des Handrads incl. Handradnummer an die NC. Da die Daten über PROFIBUS-DP übertragen werden, müssen die Daten aus dem Eingangsbereich mit Hilfe dieses Programmbausteins an die NC übergeben werden.

Vor dem Übertragen der Daten muss ein Handrad, mittels steigender Flanke am entsprechenden Achs-Interface (Handradauswahl Bit 0, 1), angewählt werden.

Der Baustein kann im SPS-Zyklus ein oder mehrmals, absolut oder bedingt aufgerufen werden.

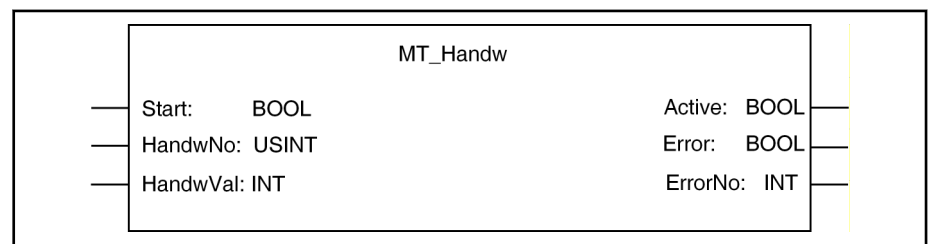


Abb. 12-8: Aufruf Baustein MT_Handw

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Statisches Signal für eine neue Anforderung
HandwNo:	Nummer des Handrads
HandwVal:	Zählerstand des Handrads

Die Rückgabeparameter lauten:

Active:	Signal, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten übergeben sind, ist die Funktion beendet, und Active wird zurückgesetzt.

12.9.2 Projektierungsbeispiel

Die Inkremente für das Handrad liegen ab Eingang 80 ab und sie sollen an die NC übertragen werden, solange der Eingang %IX40.7 TRUE ist.

Der Baustein **MT_Handw** könnte folgendermaßen aufgerufen werden:

Programmbeispiel in ST (Deklaration)

```

VAR
  R_TRIG_Instance :    R_Trigger;
  StartHandw      :    BOOL;
  HandwValue      :    INT;
  HandWheelData   :    MT_Handw;
  activeHandWheelData : BOOL;
END_VAR

```

Programmbeispiel in ST (Implementation)

```

(* Definition *)
HandwValue := %IW80;

(* Rising edge --> StartHandw *)
R_TRIG_Instance ( CLK := %IX40.7 );
StartHandw := R_TRIG_Instance.Q;

(* Call HandWheelData *)
HandWheelData (Start:=StartHandw, HandwNo:=1, HandwVal:=HandwValue);
activeHandWheelData := HandWheelData.Active;

(* Error handling *)
IF ( HandWheelData.Error = TRUE ) THEN
  (* Error handling *)
END_IF

```

12.9.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-1:	allgemeiner Parameterfehler

Funktionsabhängige Fehler und Zustände als Quittung nach Ausführen der Funktion:

1:	Unzulässige Handradnummer

12.10 Sercos Parameter (MT_ScsData)

12.10.1 Funktion

Der Programmbaustein ermöglicht das Lesen und Vorgeben eines Parameterwertes innerhalb der Antriebe. Der zu lesende bzw. vorzugebende Wert ist vom Format einer 4 Byte großen, vorzeichenlosen Zahl (UDINT), wie er innerhalb der Antriebe abliegt. Eine Wandlung in ein anderes Format muss durch das Anwenderprogramm erfolgen.

Beim Lesen eines Wertes liefert die Funktion zusätzlich das Attribut, das den Parameter beschreibt. Beim Vorgeben eines Wertes an die Antriebe muss das Attribut mit übergeben werden.

Beim Lesen aus dem zyklischen Telegramm (Mode 4 und 5) liefert der Baustein die Werte der Identnummer von allen applizierten Antrieben in Form eines Array of UDINT, wobei bei Antrieben, bei denen die Identnummer nicht im zyklischen Telegramm enthalten ist, der Wert 16#8000000 geliefert wird.

Der Baustein kann absolut oder bedingt aufgerufen werden.

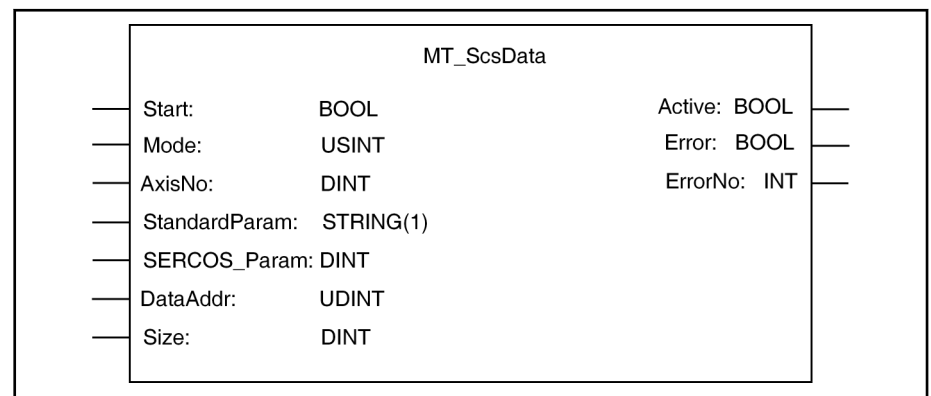


Abb. 12-9: Aufruf Baustein MT_ScsData



Bei mehrmaligem Bausteinaufruf mit Datenanforderung/-vorgabe direkt über den Servicekanal (nichtzyklische Daten) wird der Servicekanal belastet! Empfohlener Abstand zwischen zwei Bausteinaufrufen ca. 500 ms!



Die Datenübertragung kann beschleunigt werden, wenn der Sercos Multiplex-Kanal verwendet wird.

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung
Mode:	1: Lesen von Attribut und Inhalt des Parameters
	2: Vorgeben eines neuen Wertes für den Parameter
	3: Lesen des Attributes
	4: Lesen des Parameterinhaltes aus dem zyklischen Telegramm
	Die Werte werden mit einem zeitlichen Mindestabstand von 100 ms geliefert. Es werden die Werte aller Achsen geliefert
	5: Lesen des Parameterinhaltes aus dem zyklischen Telegramm ohne zeitliche Verzögerung
	Es werden die Werte aller Achsen geliefert

AxisNo:	Physikalische Achsnummer; nur bei Mode 1, 2 und 3 relevant Der Achsindex beginnt mit 1 für die erste Achse
StandardParam:	"S" = Standardparameter "P" =Produktspezifischer Parameter
Sercos_Param:	Parameternummer
DataAddr:	Adresse der Daten. Das Format der Daten ist abhängig vom Mode , der beim Anfordern übergeben wird: - Beim Lesen/ Vorgeben von Einzelwerten (Mode1, 2 und 3) werden die Daten vom Typ MT_ScsData_t zurückgeliefert: <pre> TypeMT_ScsData_t Struct scsAttr: UDINT; (*Attribut des Parameters*) scsVal: UDINT; (*Wert des Parameters*) End_Struct End_type </pre> - Beim Lesen von Daten aus dem zyklischen Telegramm (Mode 4 und 5) werden die Werte aller Achsen als Array of UDINT zurückgeliefert. Bei Antrieben, für die die übergebene Identnummer nicht im zyklischen Telegramm definiert ist, wird der Wert 16#80000000 zurückgeliefert.
Size:	Größe des Speichers für die Daten in Bytes

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird Start zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert/vorgegeben sind, ist die Funktion beendet, und Active wird zurückgesetzt.

12.10.2 Projektierungsbeispiel

Der aktuelle Wert des Sercos Parameters S-0-0108 Feedrate-Override der physikalischen Achse 1 soll mit 50% vorgegeben werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX45.0 ausgelöst.

Der Aufruf des Bausteins **MT_ScsData** könnte folgendermaßen erfolgen:

Programmbeispiel in ST (Deklaration)

```

VAR
    instScsData : MT_ScsData;
    StartScsData : BOOL;
    ScsDataMode45 : ARRAY[1..250] OF DINT;
    ScsDataMode123 : MT_ScsData_t; // Attribute must be read before writing
    Mode : USINT;
    AxNo : DINT; // -1 = all axes for mode 4 or 5,
                // otherwise the number of the axis
    ParaType : STRING(1);

```

```
ParaNo :           DINT;  
DataAdr :          UDINT;  
DataSize :         DINT;  
END_VAR
```

Programmbeispiel in ST (Implementation)

```
(* Definitions *)  
AxNo := 1;  
ParaType := "S"; // Feedrate override  
ParaNo := 108;  
ScsValueModel23.scsVal := 5000; // 50%  
Mode := 2;  
  
StartScsData := ...;  
  
(* Determine DataAddr and DataType *)  
IF StartScsData THEN  
    IF AxNo = -1 THEN  
        DataAdr:= ADR(ScsDataMode45); DataSize:= SIZEOF(ScsDataMode45);  
    END_IF  
    IF AxNo > 0 AND AxNo <= 250 THEN  
        DataAdr:= ADR(ScsDataMode123); DataSize:= SIZEOF(ScsDataMode123);  
    END_IF  
END_IF  
  
(* Call instScsData *)  
instScsData(Start:= StartScsData, Mode:= Mode, AxisNo:= AxNo, StandardParam:= ParaType,  
            SERCOS_Param:= ParaNo, DataAddr:= DataAdr, Size:= DataSize,  
            Active=> , Error=> , ErrorNo=> );  
  
IF instScsData.Active THEN  
    StartScsData := FALSE;  
END_IF  
  
(* Error handling *)  
IF instScsData.Error THEN  
    // error handling *)  
END_IF
```

Programmbausteine

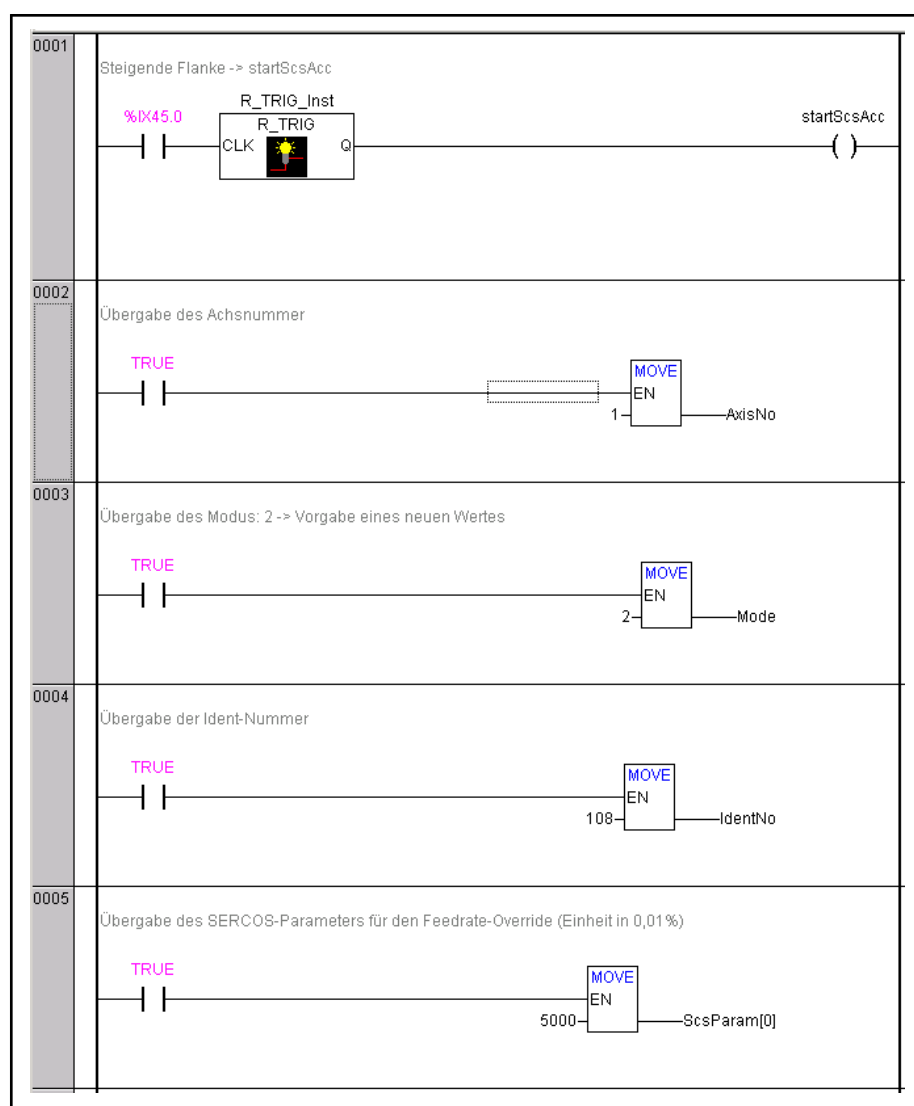


Abb. 12-10: Aufruf Baustein MT_ScsData

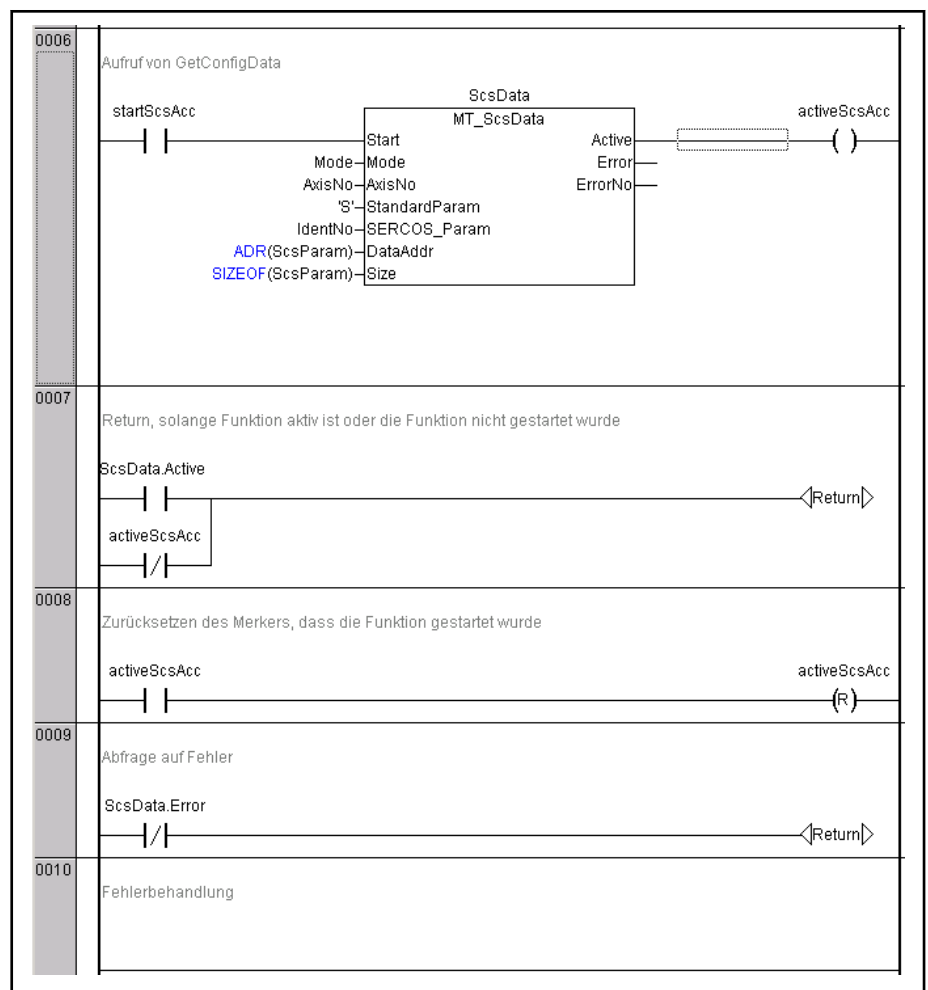


Abb. 12-11: Aufruf Baustein MT_ScsData

12.10.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

- 6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
- 3: Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
- 1: Allgemeiner Parameterfehler

Funktionsabhängige Fehler und Zustände als Quittung nach Ausführen der Funktion:

- 3: NCS-Kanal momentan belegt
- 6: Parameternummer unbekannt oder unzulässig
- 9: Zugriffskonflikt auf dem Service-Kanal
- 14: Handshake-Timeout

15:	Busy-Timeout
22:	Übertragenes Datum zu kurz
23:	Übertragenes Datum zu lang
24:	Element nicht änderbar
25:	Element z. Zt. nicht änderbar (falsche Hochlaufphase)
26:	Datum kleiner als Min-Wert
27:	Datum größer als Max-Wert
28:	Betriebsdatum unzulässig (z. B. Bitkombination)
29:	Sercos Ring z. Zt. nicht verfügbar (Hochlauf)
30:	Fehlerhaftes Attribut empfangen
31:	Systemachse unzulässig oder diese existiert nicht als Sercos Achse (Dummy-Achse)
33:	Datenlänge übersteigt Service-Container-Größe
40:	Max-Anzahl der gleichzeitig aktiven Kommandos überschritten
41:	Wartezeit Kommandoänderungsbit überschritten
50:	Konvertierungsfehler (Wandlung in Sercos Darstellung oder umgekehrt)
109	Element ist durch ein Passwort geschützt
110	Element wird zyklisch übertragen
112	Element nicht änderbar
114	Element z. Zt. nicht änderbar (PM)
115	Element z. Zt. nicht änderbar (OM)
117	Kommando kann nicht unterbrochen werden
118	Kommando kann nicht gestartet werden
119	Kommando kann nicht ausgeführt werden
500:	Systemachse nicht als Antrieb vorhanden
501:	Konfigurationsfehler (keine Übereinstimmung zwischen Systemachs-Nr. und Achs-Nr. im Ring)
503:	Ungültige Sercos Phase für angeforderte Funktion

12.11 Werkzeuglisten bearbeiten (MT_P_DbRecList)

12.11.1 Funktion



Nur in Verbindung mit "alten" Werkzeugdatenbanktabellen, die nicht XML-basiert sind.

Für XML-basierte Tabellen muss der Baustein "MT_DbRecList" verwendet werden.

Dieser Programmbaustein bietet die Möglichkeit, eine Liste von Werkzeugdatensätzen anzufordern oder zu löschen:

- **Suche nach Werkzeugdatensätzen**

Bei der Suche nach Werkzeugdatensätzen können bis zu 7 Suchkriterien, jeweils mit logisch UND verknüpft, angegeben werden. Für nicht benutzte Suchkriterien muss in "Field" 0 eingegeben werden.

- **Löschen von Werkzeugdatensätzen**

Es kann eine Liste von Datensätzen zum Löschen vorgegeben werden. Der Baustein löscht jeden dieser Datensätze, indem er den Inhalt mit Ausnahme von Sektor- und Platznummer mit "0" überschreibt. Die Werkzeugliste, die mit bestimmten Suchkriterien angefordert wurde, kann auch zum Löschen benutzt werden. Die Anzahl der zu löschenden Datensätze wird in **NofElem** übergeben.

Der Baustein kann im SPS-Zyklus ein oder mehrmals, absolut oder bedingt aufgerufen werden.

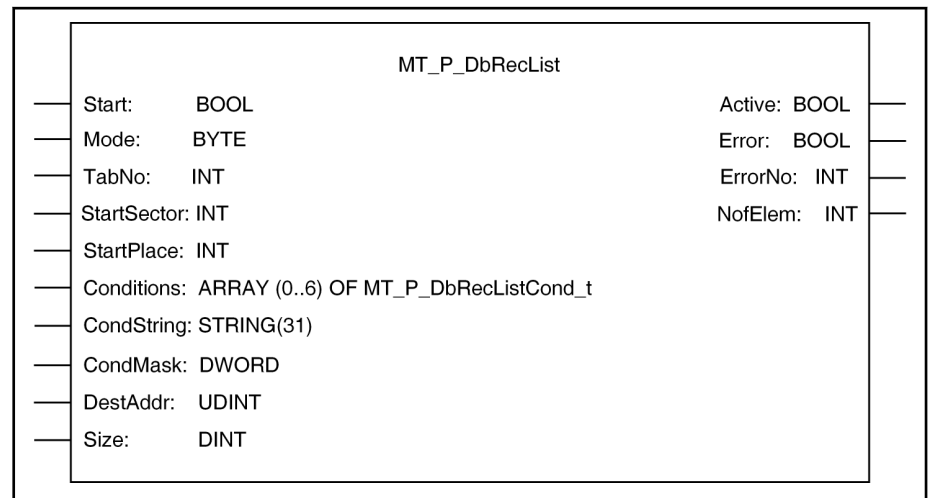


Abb. 12-12: Aufruf Baustein MT_P_DbRecList

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung
Mode:	1: Werkzeugdatensätze suchen ab Tabellenanfang 2: Werkzeugdatensätze suchen ab StartSector/StartPlace 3: Inhalt der Werkzeugdatensätze gemäß Liste löschen
TabNo:	Tabellen-Nummer
StartSector:	Sektornummer, ab der die Datensätze gesucht werden (nur bei Mode 2)
StartPlace:	Platznummer, ab der die Datensätze gesucht werden (nur bei Mode 2)

Conditions: Bis zu 7 Suchbedingungen in Form eines Feldes mit der folgenden Struktur:

Type MT_P_DbRecListCond_t

Struct

Field: INT; (* Vergleichselement für die Suche:

0 - Suchbedingung wird nicht berücksichtigt

1 - Sektor

2 - Query_int1

3 - Query_int2

4 - Query_int3

5 - Query_int4

6 - Query_string

7 - Query_bitfield *)

Value: INT; (* Vergleichswert für die Suche bei Field = 1 - 5 *)

End_struct

End_type

CondString: Vergleichsstring für die Suche bei Field = 6 Query_string

CondMask: Bitmaske nur bei Field = 7 Query_bitfield

DestAddr: Adresse der Daten in Form eines Feldes mit der folgenden Struktur:

Type MT_P_DbRecListEntry

Struct

Sector: INT;

Place: INT;

End_struct

End_type

Size: Größe des Speichers für die Daten in Bytes

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE
NofElem:	Anzahl der zurückgelieferten Elemente

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

Suchbegriff:

Zur Auswahl des Suchbegriffs können folgende Werte vorgegeben werden:

0	Suchbegriff ignorieren	
1	Sektor	
2	Werkzeug-Nummer	(query_int 1)
3	Duplo-Nummer	(query_int 2)
4		(query_int 3)
5		(query_int 4)
6	Werkzeug-Name	(query_string)
7	Werkzeugkennung	(bitfield)

Tab. 12-1: Auswahl Suchbegriff

12.11.2 Projektierungsbeispiel

Die Liste aller Werkzeuge mit der Werkzeugnummer 123, die sich im Magazin (Sektor 2) befinden und deren Standzeit abgelaufen ist, sollen gesucht werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX40.0 ausgelöst.

Der Aufruf des Bausteines MT_P_DbRecList könnte folgendermaßen ausgeführt werden:

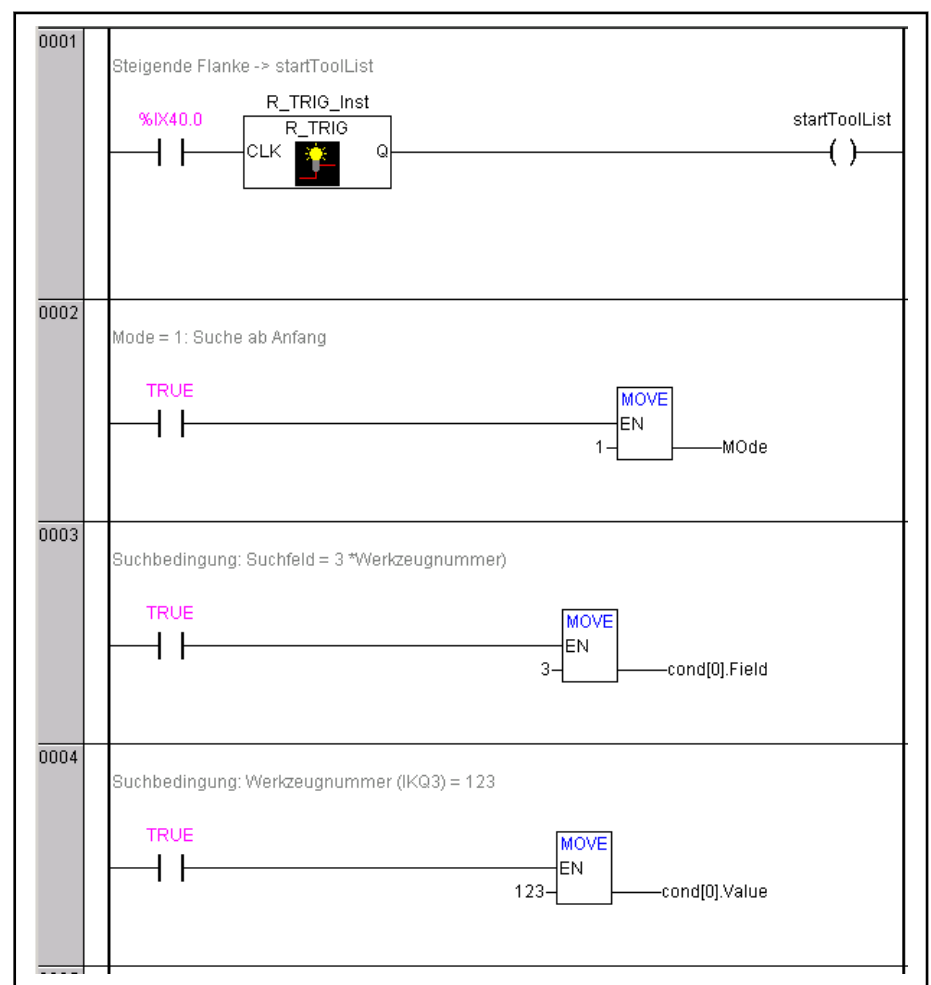


Abb. 12-13: Aufruf des Bausteins MT_P_DbRecList

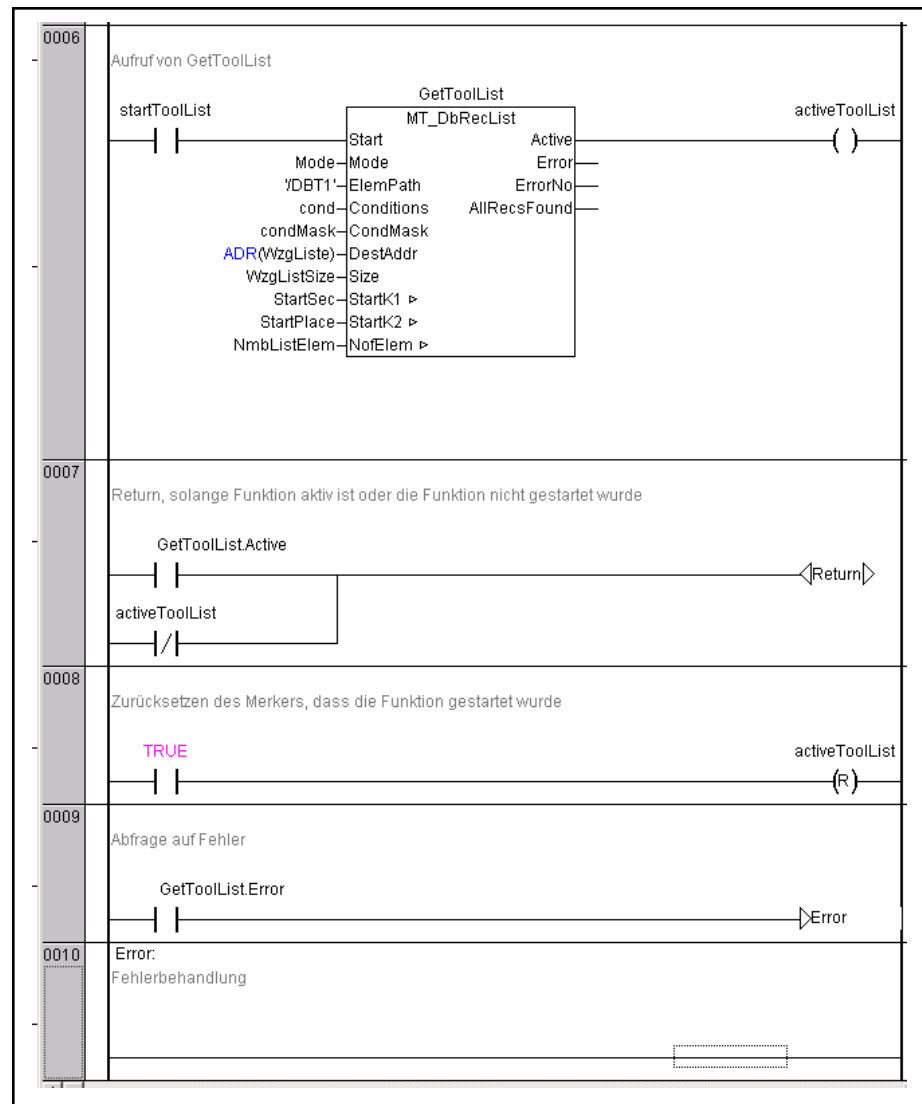


Abb. 12-14: Aufruf des Bausteins MT_P_DbRecList

12.11.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

- 6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
- 3: Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
- 1: allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

- 68: Werkzeugtabelle ist leer
- 1002: Unzulässige Tabellenummer

12.12 Datensätze bearbeiten (MT_P_DbData)

12.12.1 Funktion



Nur in Verbindung mit "alten" Werkzeugdatenbanktabellen, die nicht XML-basiert sind.

Für XML-basierte Tabellen muss der Baustein "MT_DbData" verwendet werden.

Dieser Programmbaustein bietet verschiedene Funktionen zum Lesen und Ändern von Werkzeugdaten.

Der Baustein kann im SPS-Zyklus ein oder mehrmals, absolut oder bedingt aufgerufen werden.

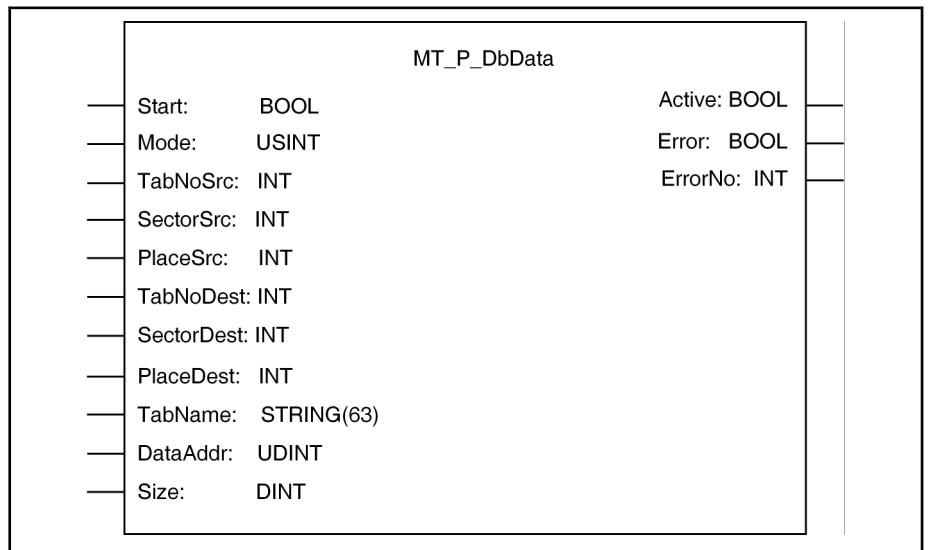


Abb. 12-15: Aufruf Baustein MT_DbData

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

- Start:** Flag für eine neue Anforderung
- Mode:**
- 1 Werkzeugdaten lesen
 - 2 Werkzeugdaten lesen mit Zugriffssperre für andere Prozesse
 - 3 Werkzeugdaten schreiben
 - 4 Werkzeugdatensatz freigeben
 - 5 Werkzeugdatensatz vertauschen
 - 6 Werkzeugdatensatz kopieren
 - 7 Werkzeugdatensatz löschen
 - 8 Einlesen einer ASCII-Datei der Werkzeugdaten
 - 9 Erzeugen einer ASCII-Datei der Werkzeugdaten
- TabNoSrc:** Tabellennummer (Quelle)
- SectorSrc:** Sektornummer (Quelle)
- PlaceSrc:** Platznummer (Quelle)
- TabNo-Dest:** Tabellennummer (Ziel) [nur bei Mode 5 und 6 relevant]

**Sector-
Dest:** Sektornummer (Ziel) [nur bei Mode 5 und 6 relevant]

PlaceDest: Platznummer (Ziel) [nur bei Mode 5 und 6 relevant]

TabName: Tabellename [nur bei Mode 8 und 9 relevant]

DestAddr: Adresse der Daten.
Die Daten werden als Typ MT_P_DbData zurückgeliefert:

Type MT_P_DbData

Struct

Sector: INT

Place: INT

Query_int_1: INT

Query_int_2: INT

Query_int_3: INT

Query_int_4: INT

Query_bitfield: INT

Query_int: ARRAY[1...40]OFINT

Query_string: STRING(31)

End_struct

End_type

Size: Größe des Speichers für die Daten in Bytes

Die Rückgabeparameter lauten:

Active: Flag, das anzeigt, ob die Funktion noch aktiv ist

Error: Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde

ErrorNo: Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

12.12.2 Projektierungsbeispiel

Die Daten des Werkzeugs in Tabelle1, Sektor 2 auf Platz 5 sollen gelesen werden incl. Sperre gegen weitere Zugriffe.

Der Auftrag wird mit steigender Flanke am Eingang %IX41.0 ausgelöst.

Der Aufruf des Bausteines **MT_P_DbData** könnte folgendermaßen ausgeführt werden:

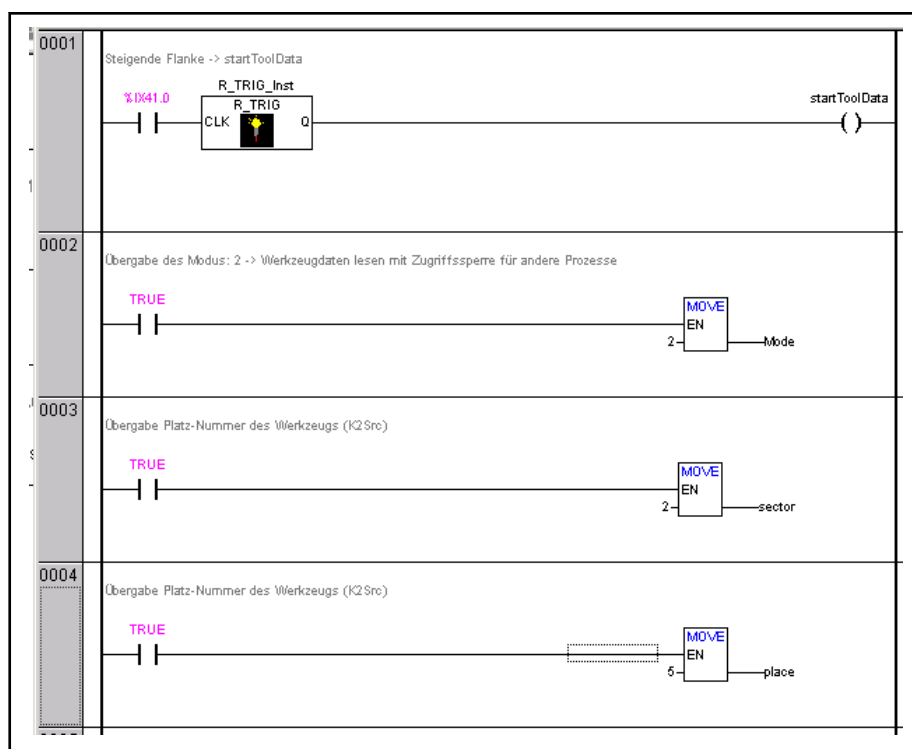


Abb. 12-16: Aufruf des Bausteins **MT_P_DbData**

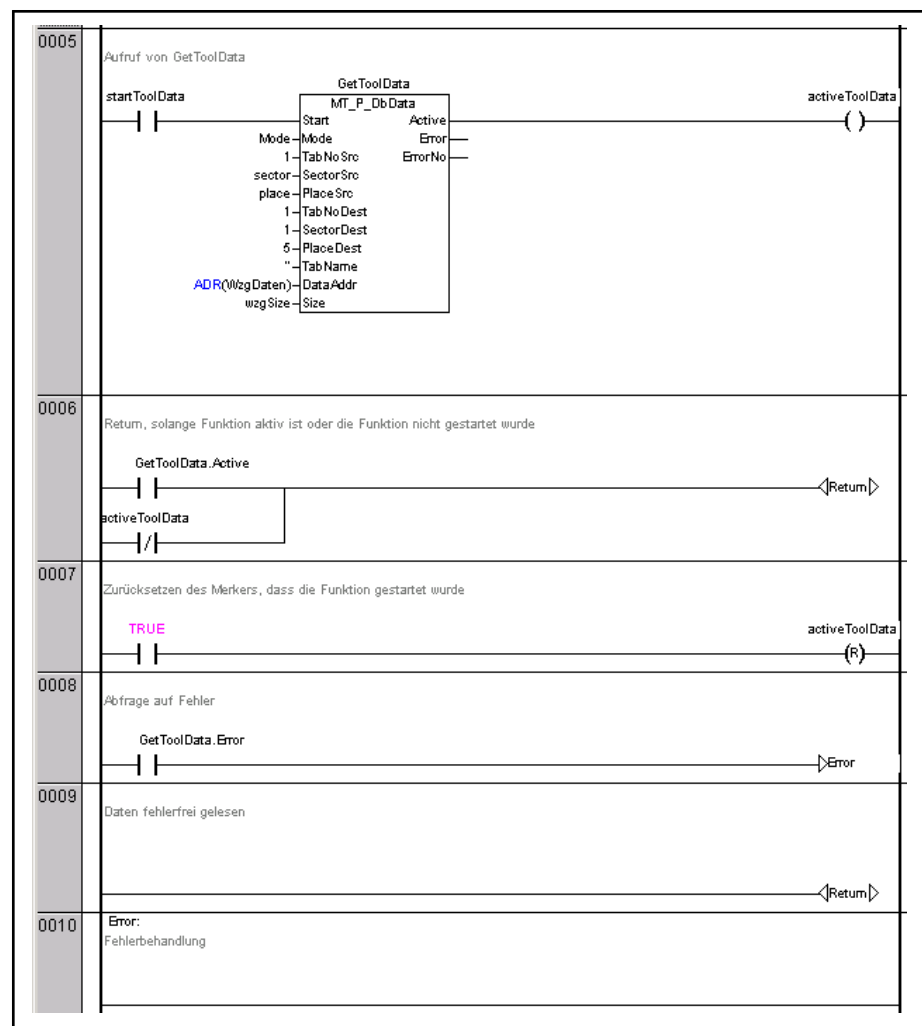


Abb. 12-17: Aufruf des Bausteins MT_P_DbData

12.12.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-3:	Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
-1:	allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

68:	Werkzeugtabelle ist leer
72:	Der Datensatz wurde nicht gefunden
108:	Der letzte gelockte Datensatz ist noch nicht freigegeben

- 115: Der Datensatz wurde vor dem Zurückschreiben oder Freigeben nicht gesperrt
- 1002: Unzulässige Tabellennummer

12.13 Datensatzlisten bearbeiten (MT_DbRecList)

12.13.1 Funktion

Dieser Programmbaustein bietet die Möglichkeit, innerhalb einer Datenbanktabelle nach Datensätzen zu suchen oder Datensätze gemäß einer Liste zu löschen.

- **Suche nach Datensätzen**

Bei der Suche nach Datensätzen können bis zu 12 Suchkriterien, jeweils mit logisch UND verknüpft, angegeben werden. Die Auswahl, welche Suchbedingungen berücksichtigt werden, erfolgt über den Parameter **CondMask**.

- **Löschen von Datensätzen**

Es kann eine Liste von Datensätzen zum Löschen vorgegeben werden. Der Baustein löscht jeden dieser Datensätze, in dem er den Inhalt mit Ausnahme von Sektor- und Platznummer mit "0" überschreibt. Die Liste, die mit bestimmten Suchkriterien angefordert wurde, kann auch zum Löschen benutzt werden. Die Anzahl der zu löschenden Datensätze wird in **NofElem** übergeben.

Im Fehlerfall wird in **NofElem** der Index des Feldelementes zurückgeliefert, bei dem der Fehler aufgetreten ist.

Der Baustein kann im SPS-Zyklus ein oder mehrmals, absolut oder bedingt aufgerufen werden.

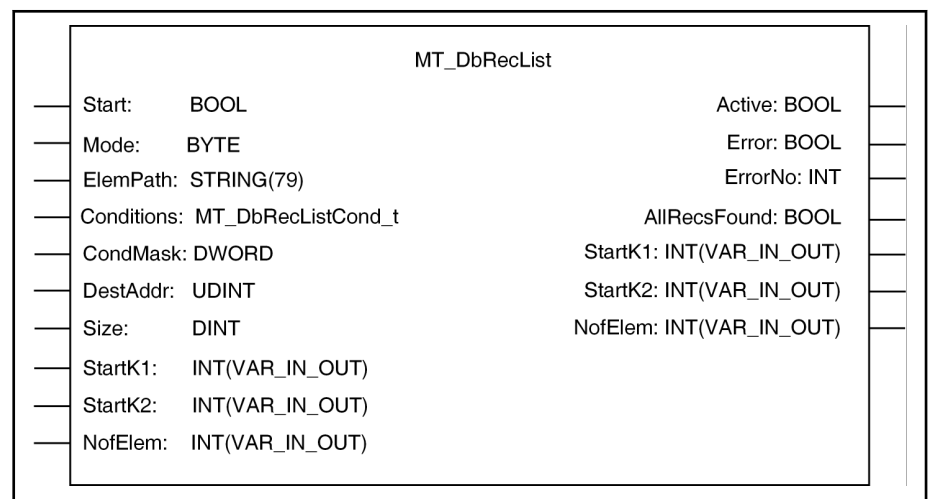


Abb. 12-18: Aufruf des Bausteins MT_DbRecList

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

- Start:** Flag für eine neue Anforderung
- Mode:**
- 1 Datensätze suchen ab Tabellenanfang.
 - 2 Datensätze suchen ab K1, K2.
 - 3 Inhalt der Datensätze gemäß Liste löschen
- ElemPath:** Identifikation der Daten incl. Pfadangaben (z. B. "/DBT1")

Conditions: Suchbedingungen vom Typ MT_DbRecListCond_t:

Type MT_DbRecList Cond_t

Struct

K1: INT; Key1

K2: INT; Key2

SKQ: INT; String

IKQ1: DINT; Integerwert 1 (Schlüsselwert)

IKQ2: DINT; Integerwert 2 (Schlüsselwert)

IKQ3: DINT; Integerwert 3 (Schlüsselwert)

IQ1: DINT; Integerwert 1 (kein Schlüsselwert)

IQ2: DINT; Integerwert 2 (kein Schlüsselwert)

IQ3: DINT; Integerwert 3 (kein Schlüsselwert)

BQ1: DWORD; Bitfeld 1

BQ1_Mask: DWORD; Maske für Bitfeld 1

BQ2: DWORD; Bitfeld 2

BQ2_Mask: DWORD; Maske für Bitfeld 2

BQ3: DWORD; Bitfeld 3

BQ3_Mask: DWORD; Maske für Bitfeld 3

End_struct

End_type

CondMask: Maske zur Auswahl der Conditions, die bei der Suche berücksichtigt werden:

16#00000001: K1

16#00000002: K2

16#00000004: SKQ

16#00000008: IKQ1

16#00000010: IKQ2

16#00000020: IKQ3

16#00000040: IQ1

16#00000080: IQ2

16#00000100: IQ3

16#00000200: BQ1

16#00000400: BQ2

16#00000800: BQ3

DestAddr: Adresse der Daten in Form eines Feldes mit der folgenden Struktur:

Type MT_DBT1Hd

Struct

K1: INT

K2: INT

SKQ: STRING(31)

IKQ1: DINT

IKQ2: DINT

IKQ3: DINT

IQ1: DINT

IQ2: DINT

IQ3: DINT

BQ1: DWORD

BQ2: DWORD

BQ3: DWORD

End_struct

End_type

Size: Größe des Speichers für die Daten in Bytes

StartK1: Bei einer Suche mit Mode 2 (Suche ab K1, K2) wird hier der Wert für K1 übergeben, ab dem die Suche beginnen soll

StartK2: Bei einer Suche mit Mode 2 (Suche ab K1, K2) wird hier der Wert für K2 übergeben, ab dem die Suche beginnen soll



Können bei einer Suche nicht alle gefundenen Elemente zurückgeliefert werden, da nicht genügend Speicherplatz zur Verfügung gestellt wurde (AllRecsFound= FALSE), werden in StartK1 und StartK2 die Werte zurückgeliefert, mit denen anschließend die Suche fortgesetzt werden kann.

NofElem: Nur beim Löschen:
Anzahl der zu löschenden Elemente

Die Rückgabeparameter lauten:

Active: Flag, das anzeigt, ob die Funktion noch aktiv ist

Error: Fehlerkennung:
FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde

ErrorNo: Fehlercode:
nur gültig, wenn Error = TRUE

NofElem: Nur beim Suchen:
Anzahl der zurückgelieferten Elemente

AllRecs-Found:	TRUE: Alle Datensätze, die den Suchbedingungen entsprechen, wurden zurückgeliefert. FALSE: Es existieren weitere Datensätze, die aus Speichermangel nicht zurückgeliefert werden können
StartK1 / StartK2:	Können bei einer Suche nicht alle gefundenen Elemente zurückgeliefert werden, da nicht genügend Speicherplatz zur Verfügung gestellt wurde (AllRecsFound=FALSE), werden in StartK1 und StartK2 die Werte zurückgeliefert, mit denen anschließend die Suche fortgesetzt werden kann
NofElem:	Suchen: Anzahl der zurückgelieferten Elemente. Löschen: Nur im Fehlerfall: Index des Datensatzes innerhalb der Liste, bei dem der Fehler auftrat. Beim Löschen der Datensätze wird die Liste vom Ende her abgearbeitet. Deshalb sind im Fehlerfall alle Datensätze vom Anfang bis zum fehlerhaften Feldelement nicht gelöscht

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

12.13.2 Projektierungsbeispiel

Die Liste aller Werkzeuge innerhalb der Werkzeugtabelle DBT1 mit der Duplo-Nummer 123, die sich im Magazin (K1=2) befinden, soll gesucht werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX40.0 ausgelöst.

Der Aufruf des Bausteins **MT_DbRecList** könnte folgendermaßen ausgeführt werden:

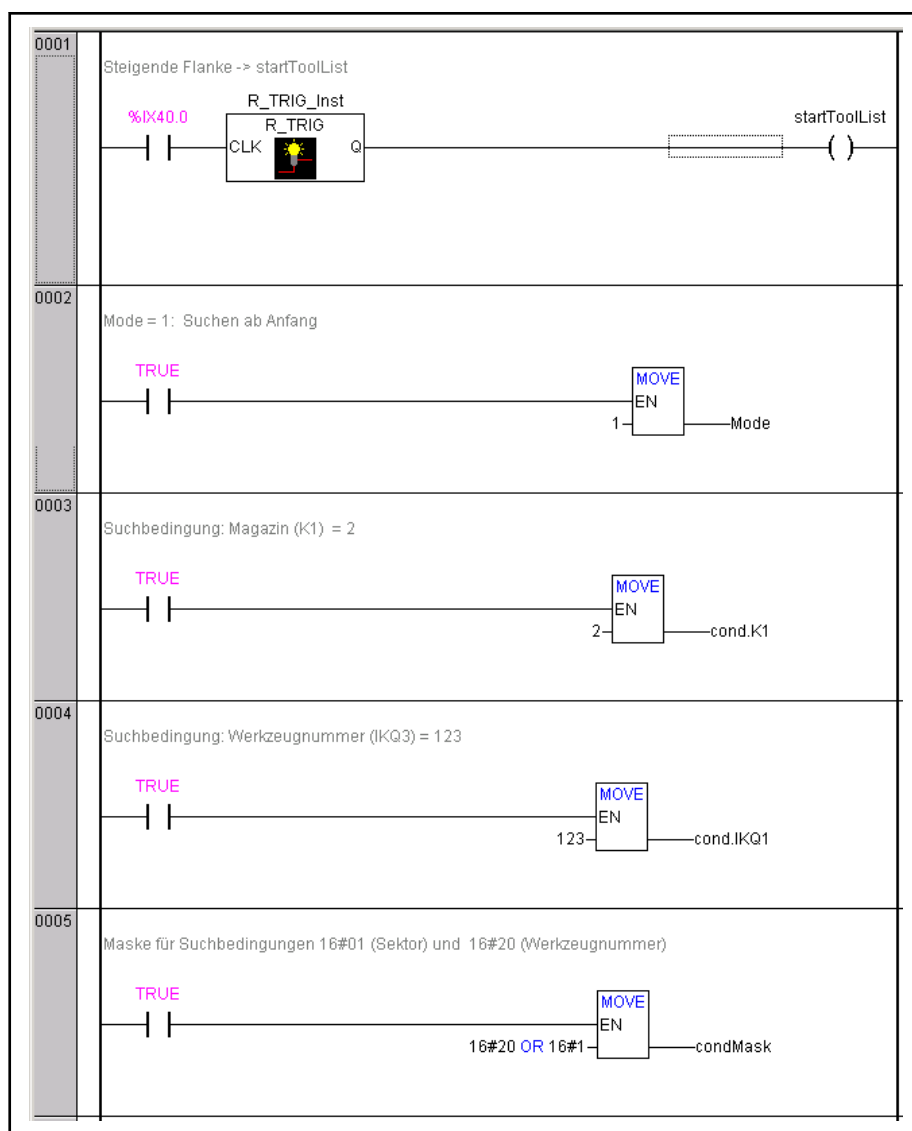


Abb. 12-19: Aufruf des Bausteins MT_DbRecList

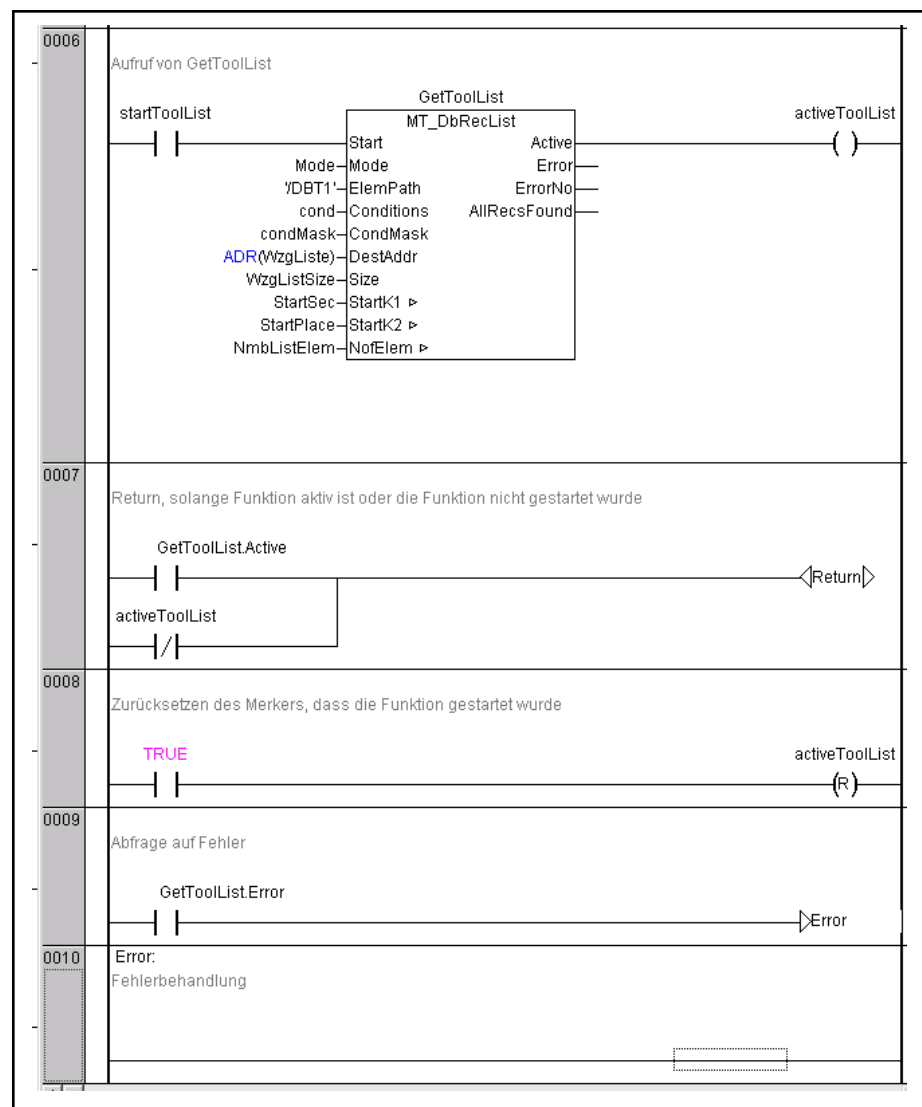


Abb. 12-20: Aufruf des Bausteins MT_DbRecList

12.13.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

- 6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
- 3: Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
- 1: allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

- 499: Fehler beim Interpretieren der XML-Datei
- 510: Syntaxfehler in ElemPath (falscher Name)

512:	Syntaxfehler in ElemPath (falsche Elementposition)
516:	Das in ElemPath spezifizierte Element ist nicht im Schema deklariert
518:	Element nicht in Schema deklariert
519:	Übergebene Datengröße stimmt nicht mit der Elementgröße überein
520:	Übergebene Typ-ID passt nicht zum Element (ElemPath)
521:	Unbekannte Typ-ID
525:	Ungültiger REAL oder LREAL Wert
1001:	Fehler beim Interpretieren der XML-Datei
1002:	Datenbankserver nicht bereit
1003:	Der angegebene Name ist zu lang
1007:	Übergebener Datenbereich zu klein
1008:	Puffer für den Pfad zu klein (Events)
1102:	Ungültiges Root-Element in ElemPath
1104:	Datensatz ist bereits gesperrt
1109:	Datensatz ist zu groß
1112:	Ungültige Suchbedingungen
1204:	Keine Datensätze mit diesem Schlüssel gefunden
1205:	Datenbank ist leer
1206:	Letzter gesperrter Datensatz ist nicht freigegeben
1208:	Kein gesperrter Datensatz vorhanden

12.14 Datensätze bearbeiten (MT_DbData)

12.14.1 Funktion

Dieser Programmbaustein bietet verschiedene Funktionen zum Lesen und Ändern von Datensätzen (z. B. Werkzeugdaten).

Es können sowohl komplette Datensätze als auch Teil- Datensätze bearbeitet werden. Die Auswahl erfolgt über die Eingangsvariable **ElemPath** (z. B. / DBT1). Die Daten werden in dem Format geliefert, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden. Für den korrekten Zugriff auf die Daten, müssen entsprechende Strukturen innerhalb des SPS-Programms definiert werden. Für die Standardkonfiguration stehen die entsprechenden Typen innerhalb der MTX-Bibliothek zur Verfügung (s. u.).

Beim Schreiben von Datenbankdaten muss in **Typeld** die Id des Typs übergeben werden, von dem die Variable auf NC-Seite definiert ist. Beim Lesen von Variablen dieses Typs wird diese zurückgeliefert. Bei mehrfachen Schreibzugriffen muss diese Id nur einmal vor dem ersten Zugriff bestimmt und kann dann bei jedem weiteren Zugriff übergeben werden.

Der Baustein kann im SPS-Zyklus ein oder mehrmals, absolut oder bedingt aufgerufen werden.

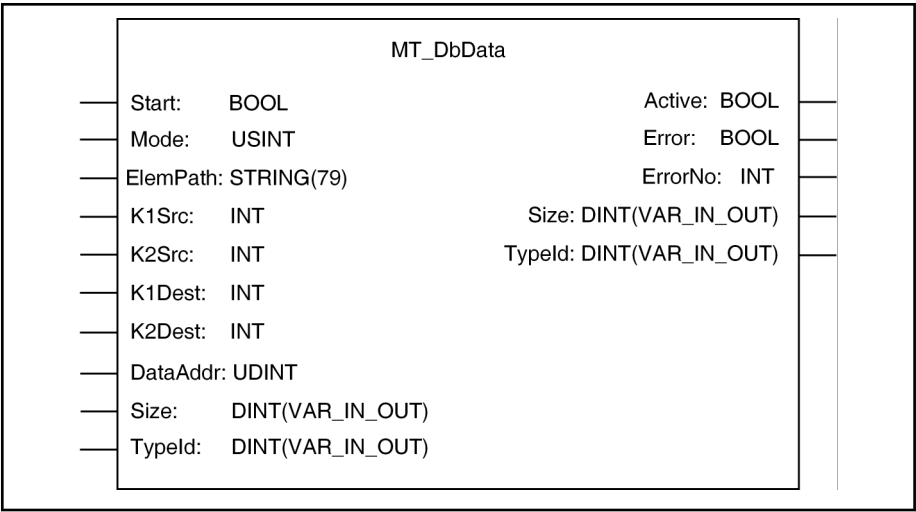


Abb. 12-21: Aufruf des Bausteins MT_DbData

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung
Mode:	DbData_Read (Wert 1) Datensatz lesen DbData_ReadLock (Wert 2) Datensatz lesen, mit Zugriffssperre für andere Prozesse DbData_Write (Wert 3) Datensatz schreiben DbData_Unlock (Wert 4) Gesperrten Datensatz freigeben DbData_Move (Wert 5) Datensatz verschieben DbData_Copy (Wert 6) Datensatz kopieren DbData_Delete (Wert 7) Inhalt des Datensatzes löschen DbData_WriteWithoutLock (Wert 11) Datensatz schreiben ohne vorheriges Sperren DbData_Interchange (Wert 12) Datensätze vertauschen <i>Bei den Funktionscodes "DbData_WriteWithoutLock" und "DbData_Write" können über den Eingangsparameter "Mode" folgende Optionen durch das Setzen von Einzelbits zusätzlich übergeben werden. Diese zusätzlichen Bits werden mittels OR-Verknüpfung hinzugefügt:</i> • Level des Checks der Übergabedaten im Server (ist keines der Bits gesetzt, wird die Minimalüberprüfung durchlaufen, bei der nur Variablen vom Typ float, double und string auf gültiges Format überprüft werden): • Mode = ModeORDbData_DisableBasicCheck Keine Überprüfung • Mode = ModeORDbData_EnableFullCheck Komplette Überprüfung der Daten • Übernahme nur von Teilen des Datensatzes: Mode = ModeORDbData_WrOnlyToolspecData nur die werkzeugspezifischen Teile des Datensatzes werden übernommen • Mode = ModeORDbData_WrOnlyPlacespecData nur die platzspezifischen Teile des Datensatzes werden übernommen
ElemPath:	Identifikation der Daten incl. Pfadangaben (z. B. "/DBT1")
K1Src:	Key 1 (Quelle)
K2Src:	Key 2 (Quelle)
K1Dest:	Key 1 (Ziel), nur bei Mode 5, 6 und 12 relevant
K2Dest:	Key 2 (Ziel), nur bei Mode 5, 6 und 12 relevant
DataAddr:	Adresse der Daten. Für Zugriffe auf Datensätze der Default-Konfiguration der Werkzeugta- belle hat ein Datensatz folgende Struktur:

Type MT_DBT1Rec_t

Struct

Hd: MT_DBT1Hd_t

Ud: MT_DBT1Ud_t

End_struct

End_type

Type MT_DBT1Ud_t

Struct

TI: MT_TI_t

Ed: ARRAY[1...2]OFMT_Ed_t

End_struct

End_type

Type MT_TI_t

Struct

Descr: STRING(31)

LifeU: STRING(3)

RemTILife: REAL

End_struct

End_type

Type MT_Ed_t

Struct

Geo: MT_Geo_t

Wear: MT_Wear_t

Life: MT_EdLife_t

End_struct

End_type

Type MT_Geo_t

Struct

L1: LREAL

L2: LREAL

L3: LREAL

Rad: LREAL

Ori: DINT

End_struct

End_type

TypeMT_Wear_t**Struct**

L1: LREAL

L2: LREAL

L3: LREAL

Rad: LREAL

End_struct**End_type****TypeMT_EdLife_t****Struct**

MonOn: BOOL

RemLife: REAL

WarnLim: REAL

MaxLife: REAL

End_struct**End_type****Size:****Lesen:**

Größe des Speichers für die Daten in Bytes

Schreiben:

Größe der zu schreibenden Daten in Bytes. Die übergebene Größe muss mit der Größe des zu beschreibenden Datensatzes übereinstimmen

Typeld:**Schreiben:**

Typeld des Datensatzes

(Wird beim Lesen eines Datensatzes zurückgeliefert)

Ist der Typ der zu beschreibenden Variablen ein Basistyp, kann auch die Typeld als Konstante von MT_basicTypeld vorgegeben werden. Andernfalls muss die Typeld vorgegeben werden, die beim Lesen einer Variablen dieses Typs zurückgeliefert wurde

Die Rückgabeparameter lauten:

Active: Flag, das anzeigt, ob die Funktion noch aktiv ist

Error: Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde

ErrorNo: Fehlercode: nur gültig, wenn Error = TRUE

Size: **Lesen:**

Größe der gelesenen Werkzeugdaten in Bytes bzw. im Fehlerfall die benötigte Speichergröße

Schreiben:

Größe der geschriebenen Werkzeugdaten in Bytes bzw. im Fehlerfall die korrekte Größe des Datensatzes

Typeld: **Lesen:**

Typeld des Datensatzes

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

12.14.2 Projektierungsbeispiel

Die Daten des Werkzeugs in der Werkzeugdatenbank -DBT1, Sektor 2 (Key1) auf Platz 5 (Key2) sollen gelesen werden, incl. Sperre gegen weitere Zugriffe.

Der Auftrag wird mit steigender Flanke am Eingang %IX41.0 ausgelöst.

Der Aufruf des Bausteines **MT_DbData** könnte folgendermaßen ausgeführt werden:

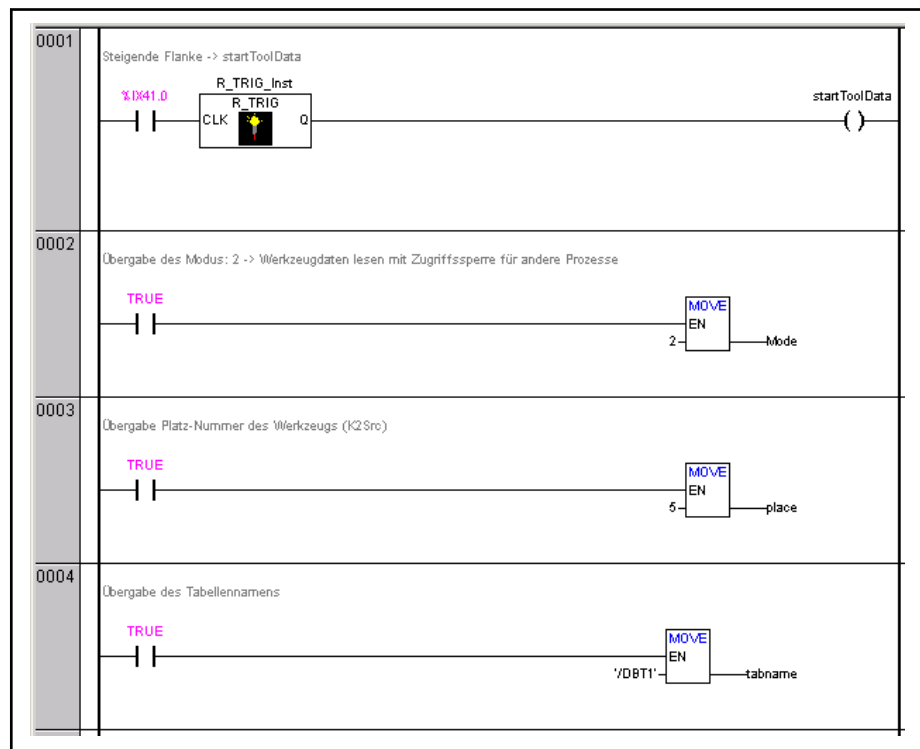


Abb. 12-22: Aufruf des Bausteins MT_DbData

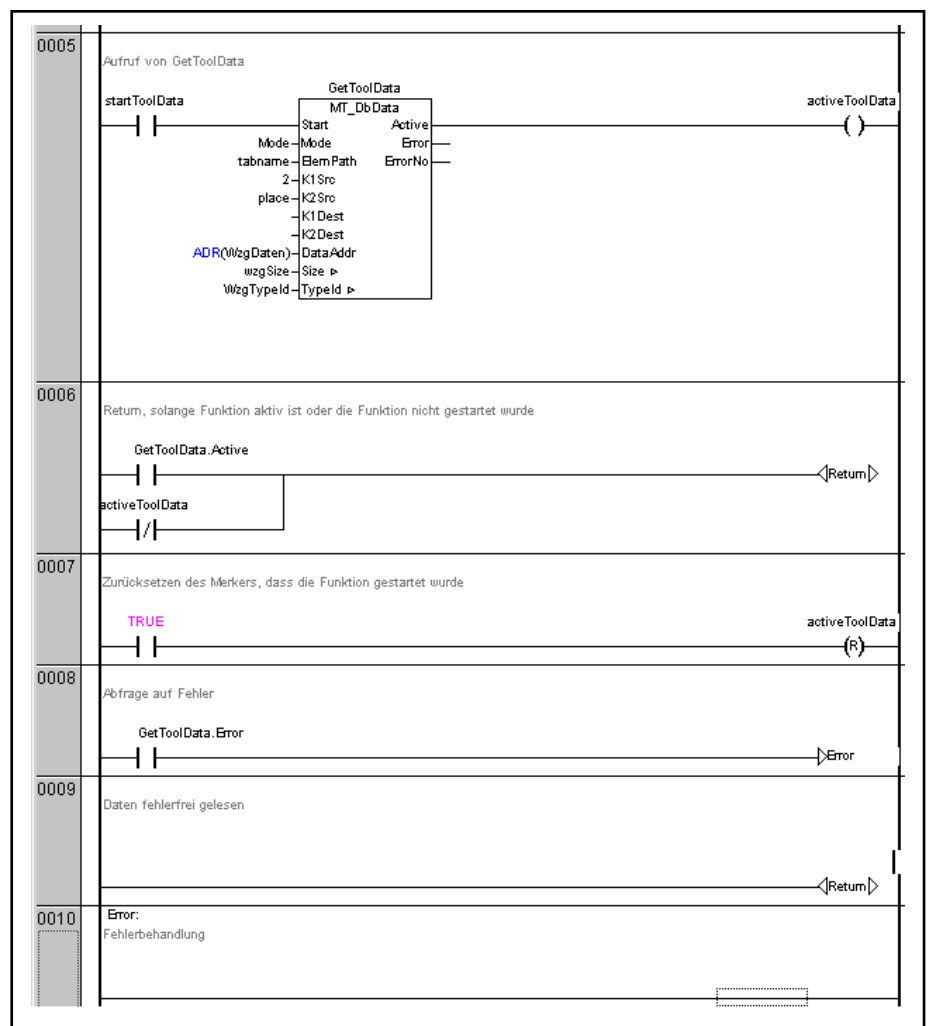


Abb. 12-23: Aufruf des Bausteins MT_DbData

12.14.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

- 6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
- 3: Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
- 1: allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

- 499: Fehler beim Interpretieren der XML-Datei
- 506: Mehr Elemente vorhanden, als laut maxOccurs zulässig
- 507: Inhalt des Elements entspricht nicht dem Basis-Datentyp laut Schema
- 508: Inhalt des Elements entspricht nicht den Facets laut Schema

510:	Syntaxfehler in ElemPath (falscher Name)
512:	Syntaxfehler in ElemPath (falsche Elementposition)
516:	Das in ElemPath spezifizierte Element ist nicht im Schema deklariert
518:	Element nicht in Schema deklariert
519:	Übergebene Datengröße stimmt nicht mit der Elementgröße überein
520:	Übergebene Typ-ID passt nicht zum Element (ElemPath)
521:	Unbekannte Typ-ID
525:	Ungültiger REAL oder LREAL Wert
1001:	Fehler beim Interpretieren der XML-Datei
1002:	Datenbankserver nicht bereit
1003:	Der angegebene Name ist zu lang
1007:	Übergebener Datenbereich zu klein
1008:	Puffer für den Pfad zu klein (Events)
1102:	Ungültiges Root-Element in ElemPath
1104:	Datensatz ist bereits gesperrt
1109:	Datensatz ist zu groß
1112:	Ungültige Suchbedingungen
1204:	Keine Datensätze mit diesem Schlüssel gefunden
1205:	Datenbank ist leer
1206:	Letzter gesperrter Datensatz ist nicht freigegeben
1208:	Kein gesperrter Datensatz vorhanden

12.15 Datenbanktabelle sichern (MT_DbSave)

12.15.1 Funktion

Über diesen Programmbaustein kann eine Datenbanktabelle ganz oder teilweise in eine Datei gesichert werden. Dabei besteht die Möglichkeit:

- die Datei zunächst zu löschen und anschließend die Datensätze zu sichern (Mode=1)
- oder -
- die Datensätze an das Ende der Datei anzufügen (Mode = 2).

Über den Parameter **SelMask** kann in Verbindung mit K1Sel und K2Sel vorgegeben werden, dass nur bestimmte Datensätze gesichert werden.

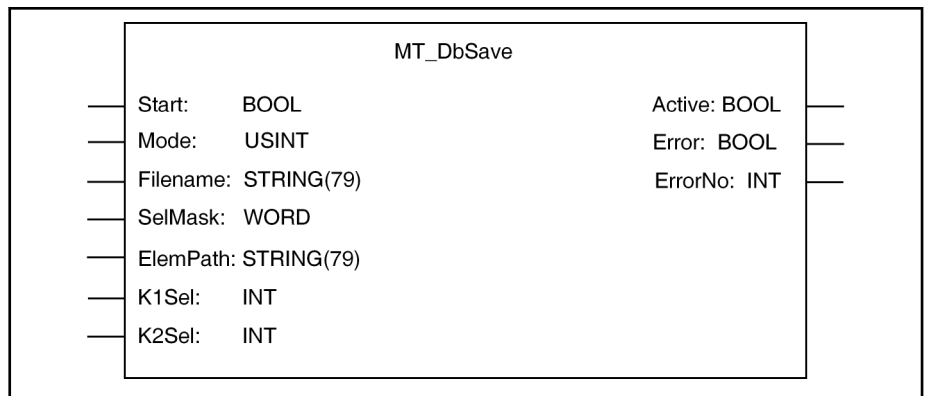


Abb. 12-24: Datenbanktabelle sichern

Bausteinparameter**Der Baustein wird aufgerufen mit den Parametern:**

Start:	Flag für eine neue Anforderung
Mode:	1: Datenbanktabelle in eine XML-Datei sichern, nach vorherigem Löschen des Dateiinhaltes 2: Datenbanktabelle in eine XML-Datei sichern, anhängen an den Dateiin- halt
ElemPath:	Identifikation der Daten incl. Pfadangaben (z. B. "/DBT1")
SelMask:	Vorgaben zum Sichern ausgewählter Datensätze: 16#00: Keine ausgewählten Datensätze; K1Sel und K2Sel werden nicht berück- sichtigt 16#01: Alle Datensätze sichern, bei denen K1 der Vorgabe von K1Sel ent- spricht 16#02: Alle Datensätze sichern, bei denen K1 und K2 der Vorgabe von K1Sel und K2Sel entspricht
K1Sel:	Auswählen der Datensätze über Key1
K2Sel:	Auswählen der Datensätze über Key2
Filename:	Name der Sicherungsdatei

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. So-
fern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es
zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

12.15.2 Projektierungsbeispiel

Alle Datensätze der Datenbanktabelle " \DBT1" sollen in der Datei" toolt-ab.dbf" gesichert werden, wobei der Dateiinhalt zuvor gelöscht werden soll (Mode=1).

Der Auftrag wird mit steigender Flanke am Eingang %IX41.0 ausgelöst.

Der Aufruf des Bausteins **MT_DbSave** könnte folgendermaßen ausgeführt werden:

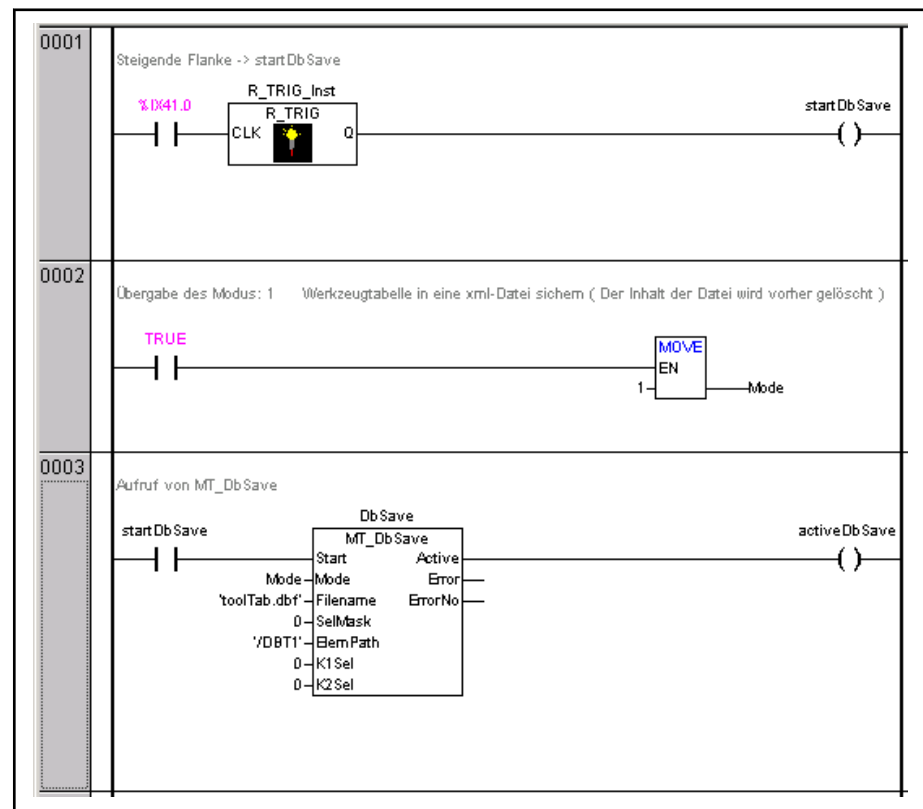


Abb. 12-25: Aufruf des Bausteins MT_DbSave

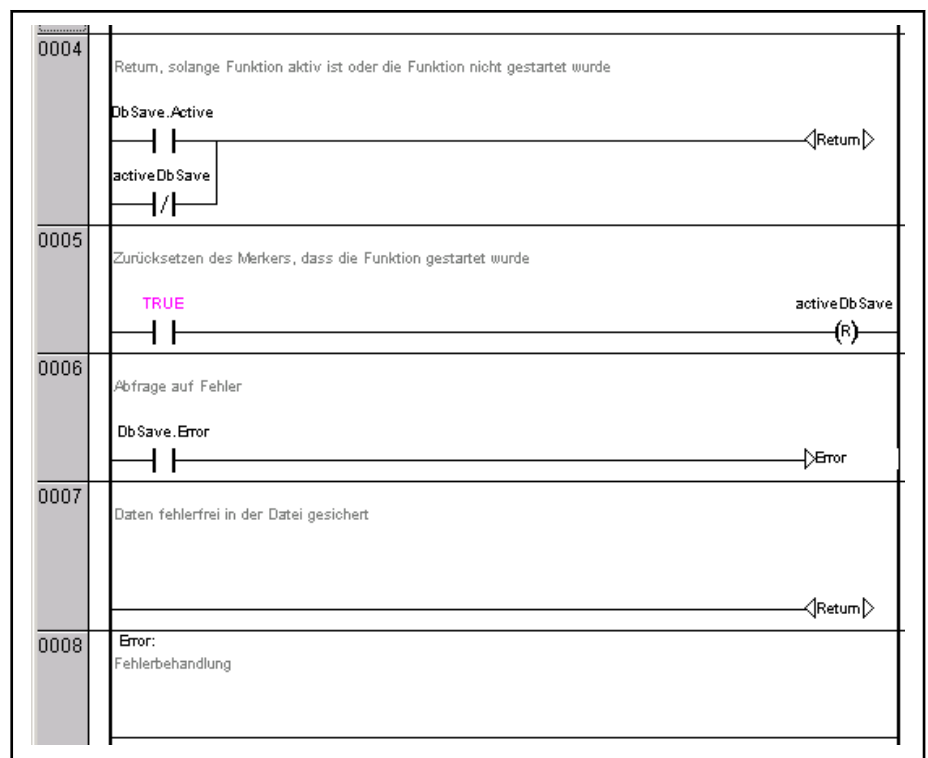


Abb. 12-26: Aufruf des Bausteins MT_DbSave

12.15.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

- 6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
- 3: Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
- 1: allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

- 499: Fehler beim Interpretieren der XML-Datei
- 510: Syntaxfehler in ElemPath (falscher Name)
- 512: Syntaxfehler in ElemPath (falsche Elementposition)
- 516: Das in ElemPath spezifizierte Element ist nicht im Schema deklariert
- 518: Element nicht in Schema deklariert
- 519: Übergebene Datengröße stimmt nicht mit der Elementgröße überein
- 520: Übergebene Typ-ID passt nicht zum Element (ElemPath)
- 521: Unbekannte Typ-ID
- 1001: Fehler beim Interpretieren der XML-Datei

1002:	Datenbankserver nicht bereit
1003:	Der angegebene Name ist zu lang
1007:	Übergebener Datenbereich zu klein
1008:	Puffer für den Pfad zu klein (Events)
1102:	Ungültiges Root-Element in ElemPath
1104:	Datensatz ist bereits gesperrt
1109:	Datensatz ist zu groß
1112:	Ungültige Suchbedingungen
1204:	Keine Datensätze mit diesem Schlüssel gefunden
1205:	Datenbank ist leer
1206:	Letzter gesperrter Datensatz ist nicht freigegeben
1208:	Kein gesperrter Datensatz vorhanden

12.16 Datenbanktabelle laden (MT_DbLoad)

12.16.1 Funktion

Über diesen Programmbaustein kann eine Datenbanktabelle ganz oder teilweise aus einer XML-Datei eingelesen werden.

Über den Parameter **SelMask** kann in Verbindung mit K1Sel und K2Sel vorgegeben werden, dass nur bestimmte Datensätze geladen werden.

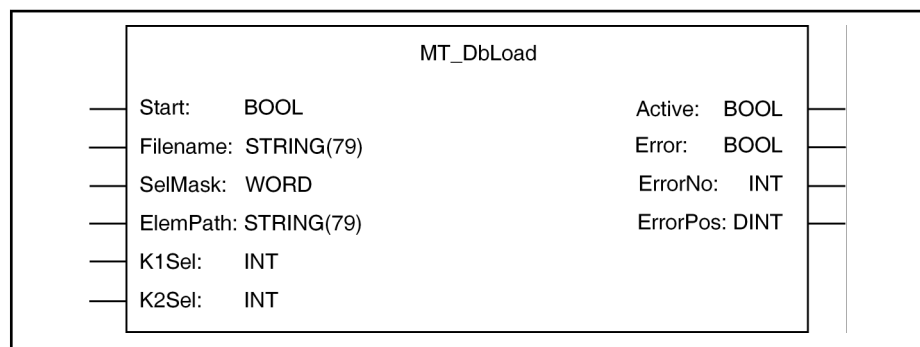


Abb. 12-27: Datenbanktabelle laden (MT_DbLoad)

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung
ElemPath:	Identifikation der Daten incl. Pfadangaben (z. B. "/DBT1")
SelMask:	Vorgaben zum Laden ausgewählter Datensätze:
16#00:	Keine ausgewählten Datensätze, K1Sel und K2Sel werden nicht berücksichtigt
16#01:	Alle Datensätze laden, bei denen K1 der Vorgabe von K1Sel entspricht
16#02:	Alle Datensätze laden, bei denen K1 und K2 der Vorgabe von K1Sel und K2Sel entspricht

K1Sel: Auswählen der Datensätze über Key1
K2Sel: Auswählen der Datensätze über Key2
Filename: Name der Datei, die geladen wird

Die Rückgabeparameter lauten:

Active: Flag, das anzeigt, ob die Funktion noch aktiv ist
Error: Fehlerkennung:
FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo: Fehlercode:
nur gültig, wenn Error = TRUE

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

12.16.2 Projektierungsbeispiel

Alle Datensätze der Datenbanktabelle "DBT1" sollen aus der Datei "toolt-ab.dbf" geladen werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX41.0 ausgelöst.

Der Aufruf des Bausteines **MT_DbLoad** könnte folgendermaßen ausgeführt werden:

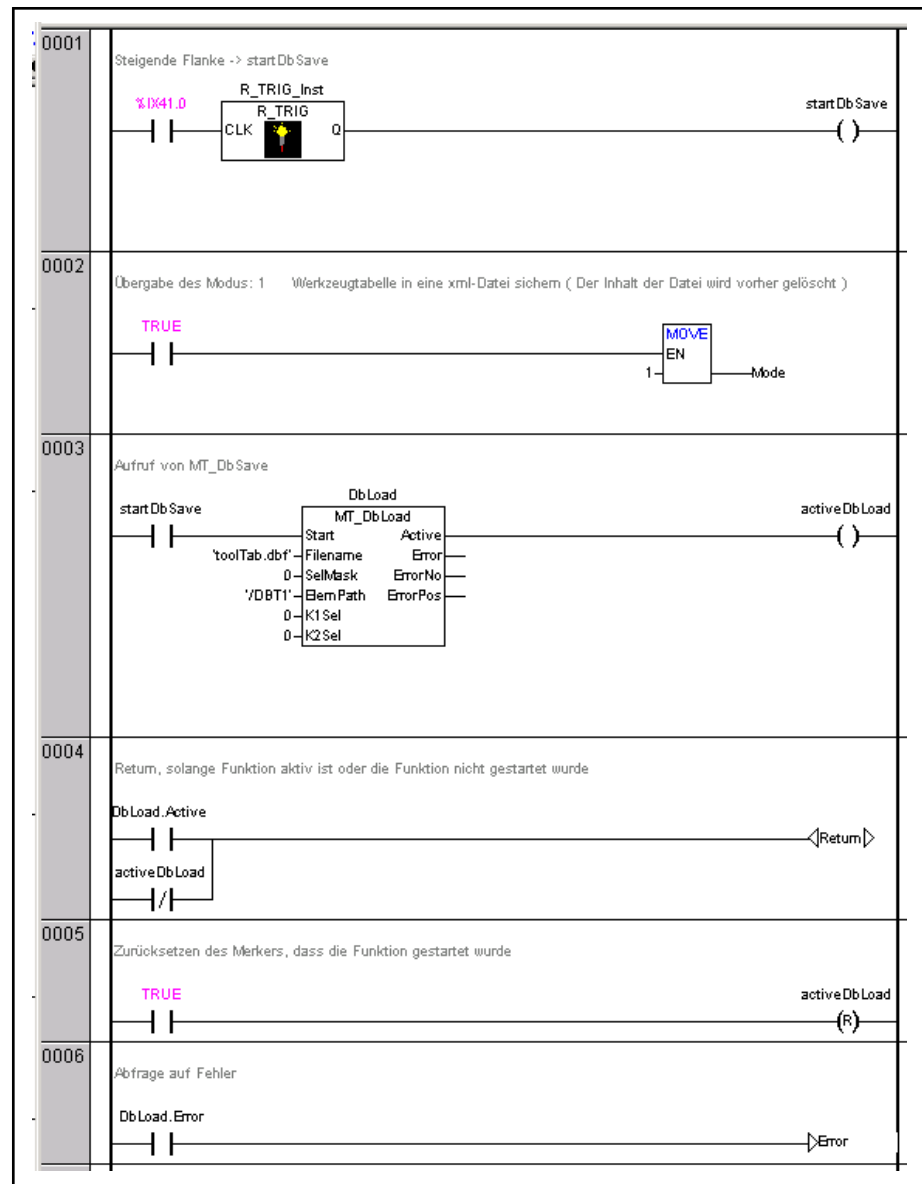


Abb. 12-28: Aufruf des Bausteins MT_DbLoad

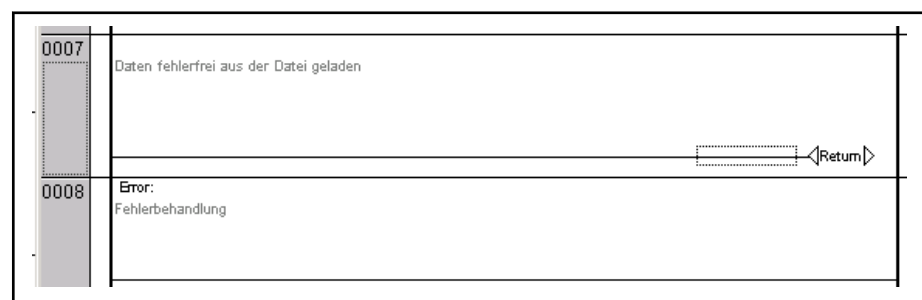


Abb. 12-29: Aufruf des Bausteins MT_DbLoad

12.16.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-3:	Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
-1:	allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

499:	Fehler beim Interpretieren der XML-Datei
510:	Syntaxfehler in ElemPath (falscher Name)
512:	Syntaxfehler in ElemPath (falsche Elementposition)
516:	Das in ElemPath spezifizierte Element ist nicht im Schema deklariert
518:	Element nicht in Schema deklariert
519:	Übergebene Datengröße stimmt nicht mit der Elementgröße überein
520:	Übergebene Typ-ID passt nicht zum Element (ElemPath)
521:	Unbekannte Typ-ID
1001:	Fehler beim Interpretieren der XML-Datei
1002:	Datenbankserver nicht bereit
1003:	Der angegebene Name ist zu lang
1007:	Übergebener Datenbereich zu klein
1008:	Puffer für den Pfad zu klein (Events)
1102:	Ungültiges Root-Element in ElemPath
1104:	Datensatz ist bereits gesperrt
1109:	Datensatz ist zu groß
1112:	Ungültige Suchbedingungen
1204:	Keine Datensätze mit diesem Schlüssel gefunden
1205:	Datenbank ist leer
1206:	Letzter gesperrter Datensatz ist nicht freigegeben
1208:	Kein gesperrter Datensatz vorhanden

12.17 Werkzeugkorrektur (MT_TCorr)

12.17.1 Funktion

Dieser Programmbaustein ermöglicht das Vorgeben von Korrekturwerten. Es können bis zu 16 Korrekturen mit einem Aufruf übergeben werden.

Die externen Werkzeugkorrekturen werden über die G-Funktionen "G47 EDx" (für x = 1 - 16) eingeschaltet und mit G48 wieder abgeschaltet.



Die Korrekturwerte werden beim nächsten, vorzubereitenden Programmsatz aktiv.

Soll die Korrektur unmittelbar nach der Eingabe wirksam werden, muss die Vorbereitung von NC-Sätzen angehalten werden (z. B. durch **WAIT** oder **Ein-**

lesesperre) oder die bereits vorbereiteten NC-Sätze erneut vorbereitet werden (**Restweg löschen**).

Der Baustein kann absolut oder bedingt aufgerufen werden.

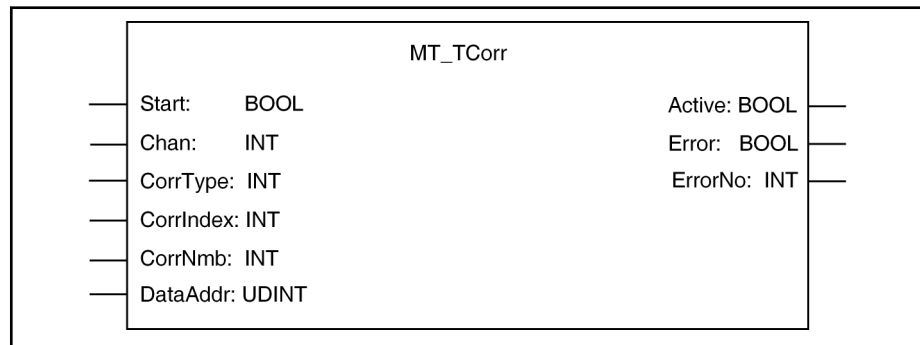


Abb. 12-30: Werkzeugkorrektur (MT_TCorr)

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

-
- Start:** Flag für eine neue Anforderung
- Chan:** Nummer des Kanals, innerhalb dessen die Korrekturwerte wirken
- CorrType:** 1:
Korrektur mit je 1 Radius-, 3 Längenkorrekturwerten und der Schneidenlage.
2:
Korrektur mit je 1 Radius-, 3 Längenkorrekturwerten, Schneidenlage und den Winkeln Phi, Theta und Psi
- CorrIndex:** Auswahl der Korrektur: zulässige Werte 1 - 16
- CorrNmb:** Anzahl der Korrekturen
- DataAddr:** Adresse der Daten.
Je nach Korrekturtyp werden die Korrekturwerte als Feld in folgendem Format übergeben:

CorrType 1:**Type** MT_Geo_t**Struct**

Len1: LREAL

Len2: LREAL

Len3: LREAL

Rad: LREAL

Ori: DINT

End_struct**End_type****CorrType 2:****Type** MT_GeoExtended_t**Struct**

Len1: LREAL

Len2: LREAL

Len3: LREAL

Rad: LREAL

Ori: DINT

Phi: LREAL

The: LREAL

Psi: LREAL

End_struct**End_type****Die Rückgabeparameter lauten:**

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Die Korrekturwerte werden sofort zur NC gesendet.

Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die NC die Korrekturvorgabe quittiert hat.

12.17.2 Projektierungsbeispiel

Als 2. Korrektur (G47 ED 2) im Kanal 1 soll für ein Fräswerkzeug (Korrekturtyp 2) eine Radiuskorrektur von 3.1234 mm und eine Längenkorrektur von 124 mm vorgegeben werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX42.0 ausgelöst.

Der Aufruf des Bausteines **MT_TCorr** könnte folgendermaßen ausgeführt werden:

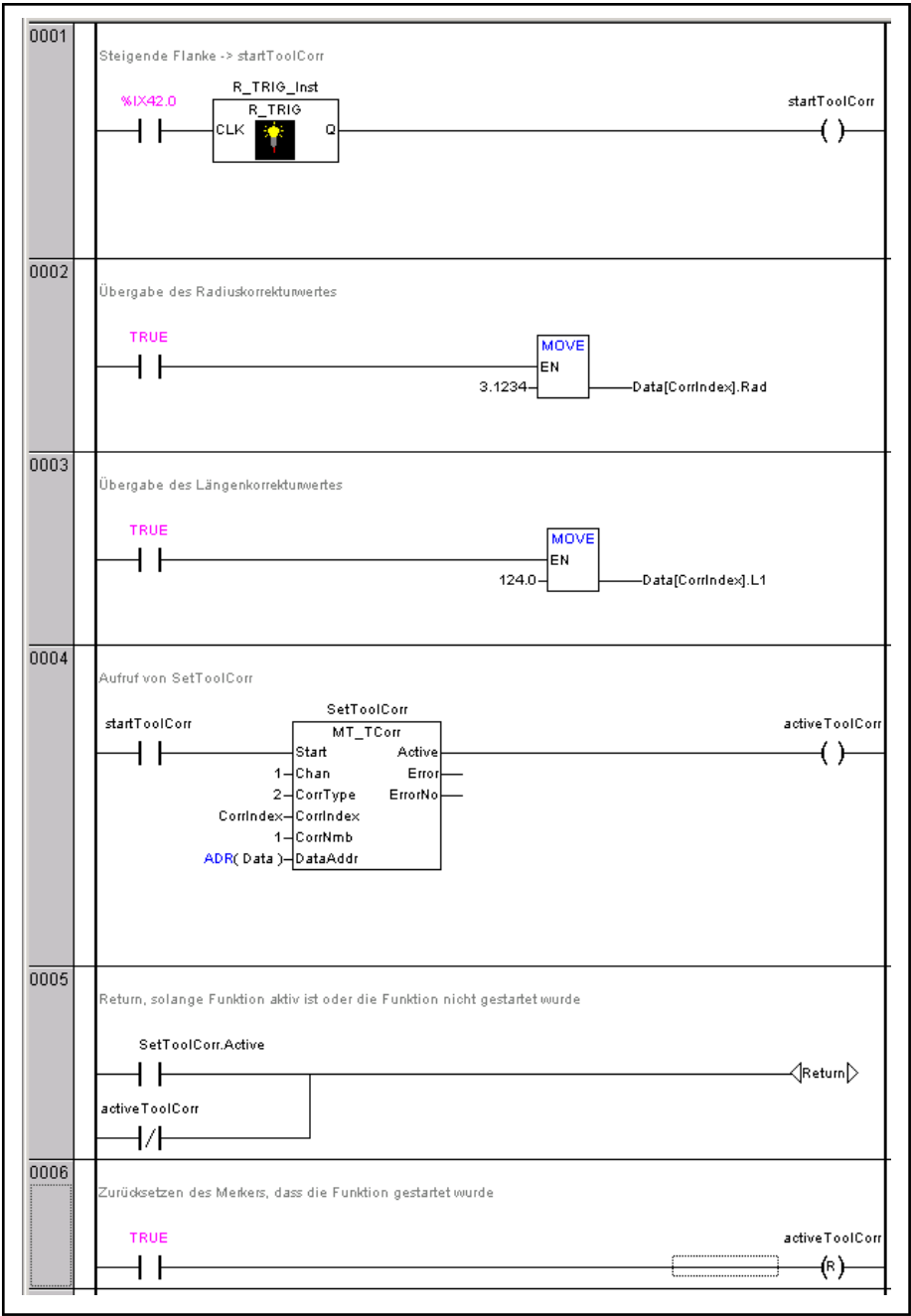


Abb. 12-31: Aufruf des Bausteins MT_TCorr

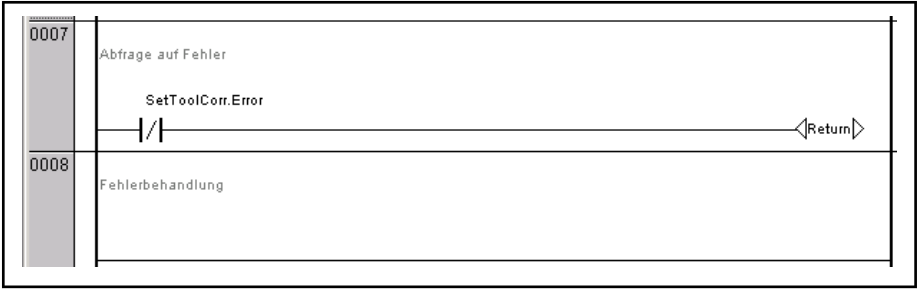


Abb. 12-32: Aufruf des Bausteins MT_TCorr

12.17.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-3:	Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
-1:	allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

1:	Unzulässige Kanalnummer
19:	Unzulässiger Korrekturindex
20:	Die Anzahl der Korrekturen ist zu hoch (eventuell in Verbindung mit dem Korrekturindex)
21:	Unzulässiger Korrekturtyp
22:	Unzulässige Schneidenlage
36:	Nicht lesbarer, nicht initialisierter oder nicht sinnvoller Korrekturwert (z. B. 10^{100})

12.18 Lesen/Schreiben von permanenten CPL-Variablen (MT_CplData)

12.18.1 Funktion

Dieser Programmbaustein liefert den zum Zeitpunkt der Abfrage gültigen Wert einer permanenten CPL-Variablen oder kann diese CPL-Variable beschreiben.

Werden Arrays bearbeitet, ist die Anzahl der zu übertragenden Elemente auf 100 begrenzt. Müssen mehr Elemente übertragen werden, sind mehrere Anforderungen mit entsprechenden Offsets zu stellen.

Der Baustein kann absolut oder bedingt aufgerufen werden.

Der Baustein führt keine Typkonvertierung durch. Der Quell- und Zieldatentyp müssen übereinstimmen.

Für den Zugriff auf CPL-Variablen mit Typkonvertierung steht der Baustein MT_CplDataConv zur Verfügung.

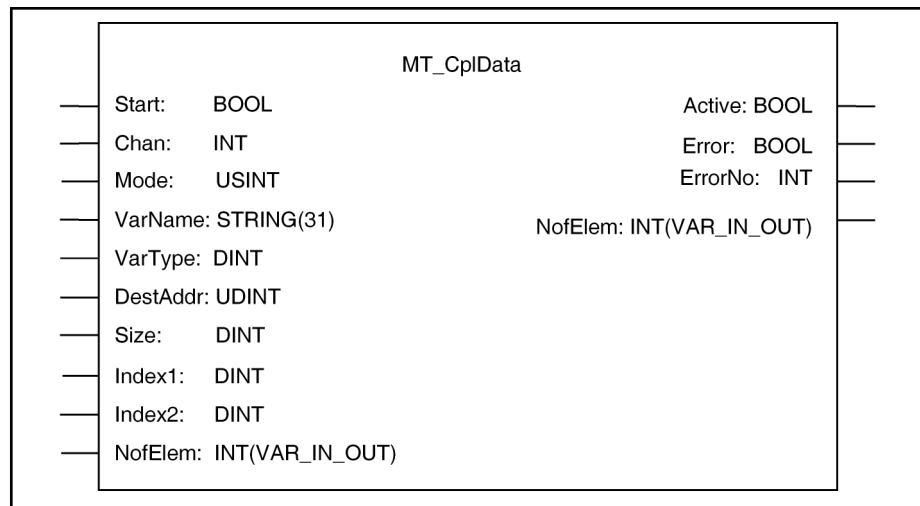


Abb. 12-33: Aufruf des Bausteins MT_CplData

Bausteinparameter**Der Baustein wird aufgerufen mit den Parametern:**

Start:	Flag für eine neue Anforderung
Chan:	Gewünschte Kanalnummer
Mode:	0: Lesen eines CPL-Werts 1: Schreiben eines CPL-Werts
VarName:	Name der CPL-Variablen
VarType:	Typdefinition der CPL-Variablen: 0 int 1 float 2 double 3 boolean 4 char
DestAddr:	Adresse der Daten
Size:	Größe des Speichers für die Daten in Bytes
Index1:	Feld-Index 1 (nur für Felder)
Index2:	Feld-Index 2 (nur für zweidimensionale Felder)
NofElem:	Lesen: Anzahl der angeforderten Werte Schreiben: Anzahl der vorgegebenen Werte

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass Funktion fehlerfrei abgearbeitet wurde

ErrorNo: Fehlercode:
nur gültig, wenn Error = TRUE

NofElem: **Lesen:**
Anzahl der gelesenen Werte
Schreiben:
Anzahl der übernommenen Werte

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

Beim Parameter **NofElem** ist zu beachten, dass er sowohl ein Eingangs- als auch ein Ausgangsparameter ist: Wird als Eingangswert 0 vorgegeben, werden auch keine Daten gelesen bzw. übernommen. Der Baustein kehrt dann auch ohne Fehlerkennung zurück

12.18.2 Projektierungsbeispiel

Im Kanal 1 soll die CPL-Variable "@Werkzeugplatz" vom Typ Integer ausgelesen werden.

Der Auftrag wird mit steigender Flanke am Eingang %IX42.5 ausgelöst.

Der Aufruf von **MT_CplData** könnte folgendermaßen erfolgen:

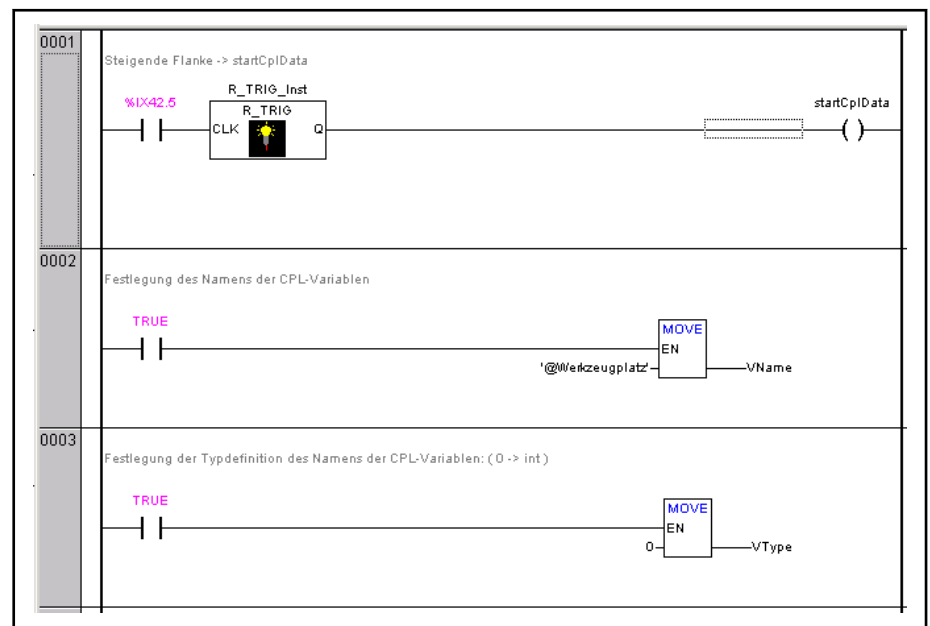


Abb. 12-34: Aufruf von MT_CplData

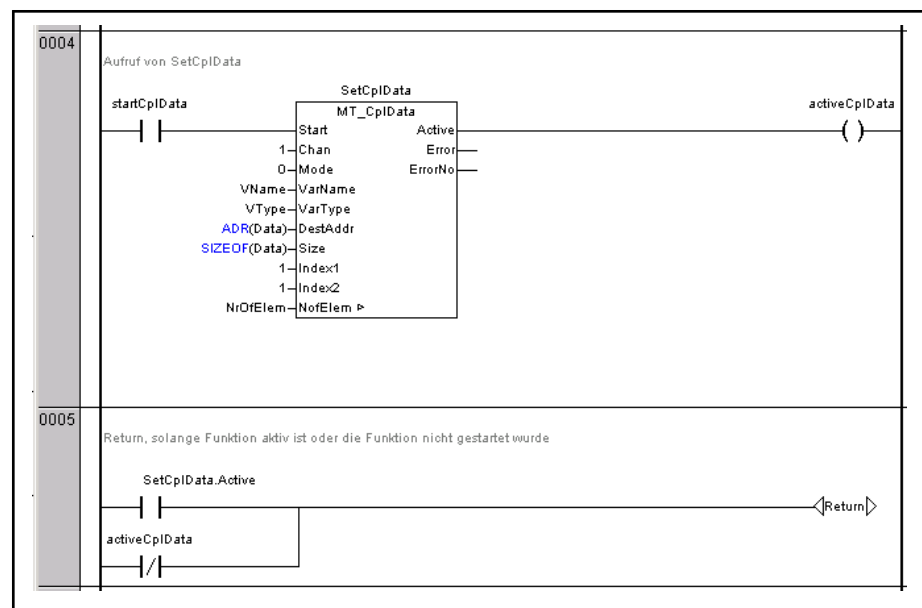


Abb. 12-35: Aufruf von MT_CplData Graphic

12.18.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-3:	Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
-1:	allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

1:	Variable nicht gefunden
2:	Variablentyp unterschiedlich
3:	Feldindex fehlerhaft
4:	Variablentyp existiert nicht

12.19 Lesen/Schreiben von permanenten CPL-Variablen mit Typkonvertierung (MT_CplDataConv)

12.19.1 Funktion

Dieser Programmbaustein liefert den zum Zeitpunkt der Abfrage gültigen Wert einer permanenten CPL-Variablen oder kann diese CPL-Variable beschreiben. Der Datentyp von CPL- und SPS-Variable kann sich unterscheiden.

Der Baustein führt eine Konvertierung zwischen unterschiedlichen Datentypen durch. Ein dabei auftretende Informationsverlust bzw. Wertebereichsüberlauf wird nicht signalisiert.

Der Baustein kann absolut oder bedingt aufgerufen werden.

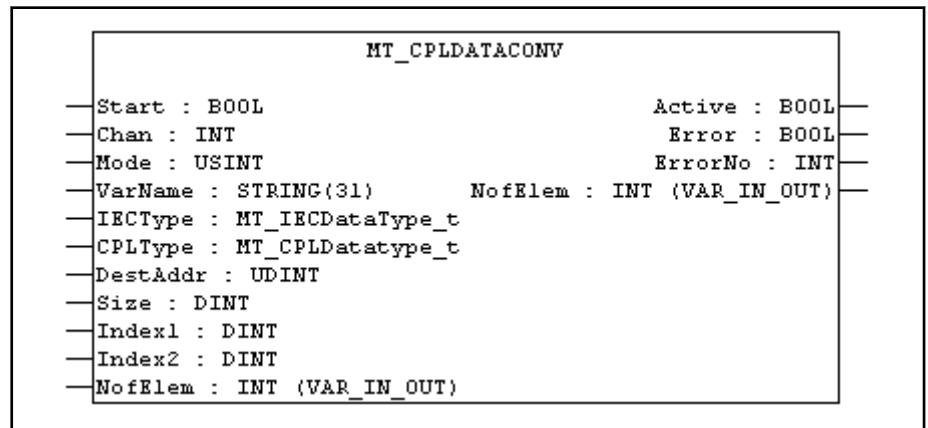


Abb. 12-36: Aufruf des Bausteins MT_CplDataConv

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung
Chan:	Gewünschte Kanalnummer
Mode:	0: Lesen eines CPL-Werts 1: Schreiben eines CPL-Werts
VarName:	Name der CPL-Variablen
IECType:	Typdefinition der SPS-Variablen: IEC_DINT (0) DINT IEC_REAL (1) REAL IEC_LREAL (2) LREAL IEC_BOOL (3) BOOL IEC_SINT (4) SINT IEC_INT (5) INT IEC_UDINT (6) UDINT IEC_DWORD (7) DWORD IEC_USINT (8) USINT IEC_BYTE (9) BYTE IEC_STRING (10) STRING IEC_UINT (11) UINT IEC_WORD (12) WORD
CPLType:	Typdefinition der CPL-Variablen: CPL_INT (0) int CPL_REAL (1) float CPL_LREAL (2) double CPL_BOOL (3) boolean CPL_CHAR (4) char
DestAddr:	Adresse der Daten

Size:	Größe des Speichers für die Daten in Bytes
Index1:	Feld-Index 1 (nur für Felder)
Index2:	Feld-Index 2 (nur für zweidimensionale Felder)
NofElem:	Lesen: Anzahl der angeforderten Werte
	Schreiben: Anzahl der vorgegebenen Werte

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE
NofElem:	Lesen: Anzahl der gelesenen Werte
	Schreiben: Anzahl der übernommenen Werte

Die Funktion startet, wenn **Start** gesetzt wird und **Active** nicht gesetzt ist. Sofern Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Active wird gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und Active wird zurückgesetzt.

Beim Parameter NofElem ist zu beachten, dass er sowohl ein Eingangs- als auch ein Ausgangsparameter ist: Wird als Eingangswert 0 vorgegeben, werden auch keine Daten gelesen bzw. übernommen. Der Baustein kehrt dann auch ohne Fehlerkennung zurück

12.19.2 Projektierungsbeispiel

Im Kanal 1 soll die CPL-Variable "@Vorschub_X" vom Typ Integer ausgelesen werden. SPS-seitig wird der Wert im Real-Format benötigt.

Der Auftrag wird mit steigender Flanke am Eingang %IX42.5 ausgelöst.

Der Aufruf von **MT_CplDataConv** könnte folgendermaßen erfolgen:

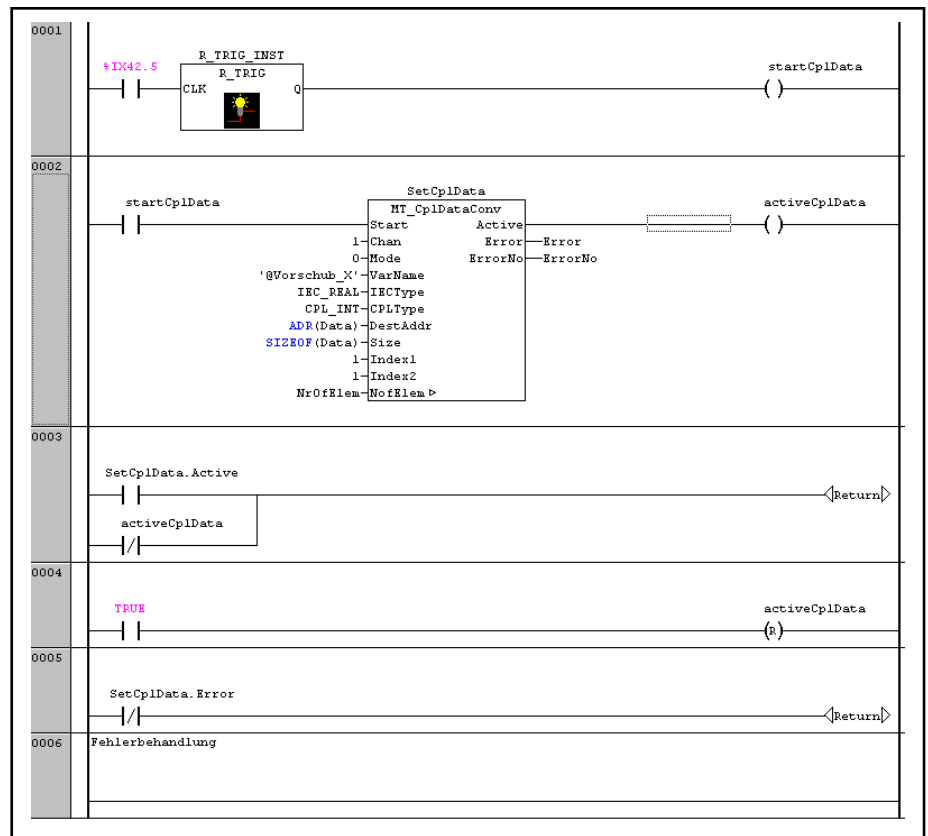


Abb. 12-37: Aufruf von MT_CplDataConv

12.19.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden
-3:	Datenüberlauf: Es werden nur so viele Daten zurückgeliefert, wie in den verfügbaren Speicher passen
-1:	allgemeiner Parameterfehler

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

1:	Variable nicht gefunden
2:	Variablentyp unterschiedlich
3:	Feldindex fehlerhaft
4:	Variablentyp existiert nicht

12.20 Systemdaten lesen (MT_SD_RD)

12.20.1 Funktion

Der Baustein **Systemdaten lesen** bietet die Möglichkeit, beliebige Systemdaten zu lesen. Es können sowohl komplette Variablen als auch Teil-Variablen gelesen werden. Die Auswahl erfolgt über die Eingangsvariable **XPath** (z. B. /VAR_1). Die Daten werden in dem Format geliefert, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden. Für den korrekten Zugriff auf die Daten, müssen entsprechende Strukturen innerhalb des SPS-Programms definiert werden.

Einschränkungen:

- Auf unterster Ebene können keine kompletten Arrays gelesen werden (z. B. /ArrayOfInts). Arrays, die Unterelemente einer Systemvariablen sind, können dagegen auch komplett gelesen werden (z. B. /ArrayOfInts).
- Die Größe der angeforderten Daten ist auf 4 KByte begrenzt.

Im SPS-Zyklus können mehrere Instanzen des Bausteins aufgerufen werden.

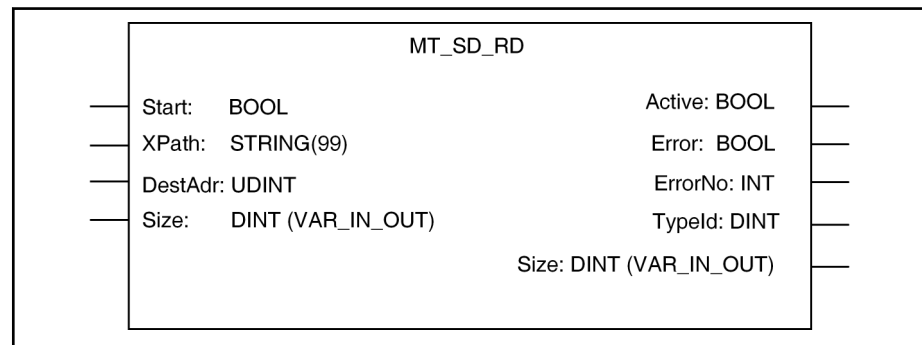


Abb. 12-38: Aufruf des Bausteins MT_SD_RD

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag für eine neue Anforderung Eine neue Anforderung wird durchgeführt, wenn Start = TRUE und Active = FALSE ist
XPath:	Identifikation der Systemdaten incl. Pfadangaben (z. B. "/Var_1")
DataAdr:	Adresse der Daten Die Daten müssen in dem Format übergeben werden, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden
Size:	Max. Größe des Datenbereichs für die angeforderten Daten in Bytes (SIZEOF(<variable>))

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE

Size: Größe der Daten der angeforderten Systemdaten in Bytes

Typeld: ID des Typs des Systemdatums

Diese ID muss beim Schreiben von Systemdaten vorgegeben werden

Die Funktion startet, wenn das Bit **Start** gesetzt wird und Active nicht gesetzt ist. Sofern das Bit Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Im Status wird **Active** gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und **Active** wird zurückgesetzt.

12.20.2 Projektierungsbeispiel

Die Daten des Systemdatums "/Var_1" sollen gelesen werden.

Der Auftrag startet mit steigender Flanke am Eingang **E43.0**.

Der Aufruf des Bausteins **MT_SD_Rd** könnte folgendermaßen ausgeführt werden:

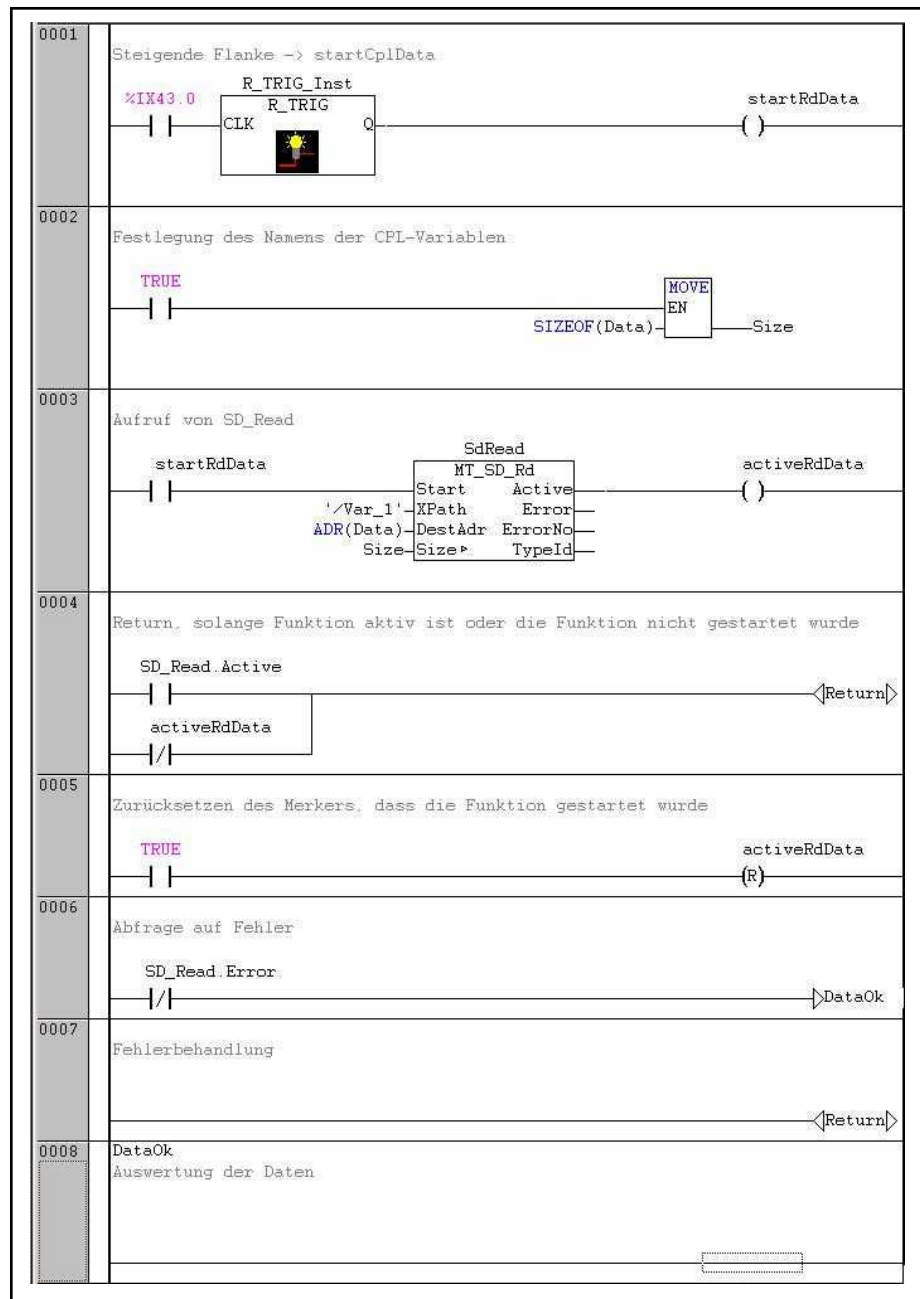


Abb. 12-39: Aufruf des Bausteins MT_SD_Rd

12.20.3 Statusmeldungen

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

1: Fehler innerhalb der XML-Lib

502: Interner Fehler des XML-Servers

507: Inhalt des Elements entspricht nicht dem Basis-Datentyp lt. Schema

508:	Inhalt des Elements entspricht nicht den Facets lt. Schema
509:	Interner Fehler beim Validieren
510:	Syntax-Fehler im Path-String
511:	Segment Path liegt nicht innerhalb des RootPath (intern)
512:	Element-Position laut Schema nicht zulässig (Path)
513:	Über Path spezifiziertes Element kann nicht angelegt werden
514:	Über Path spezifiziertes Element kann nicht gelöscht werden
515:	Über Path spezifiziertes Element ist nicht in den XML-Daten enthalten
516:	Über Path spezifiziertes Element ist nicht im Schema deklariert
517:	Letztes Element im Path stimmt nicht mit erstem Element in Data überein
518:	Element aus XML-Datei oder XML-String nicht im Schema deklariert
519:	Datengröße passt nicht zum spezifizierten Element
520:	Datentyp passt nicht zum spezifizierten Element
521:	Es wurden zu viele binäre Daten angefordert
522:	Unbekannter Datentyp
523:	Ungültige Datengröße
524:	Unbekanntes Root-Element
4001:	Interner Fehler des Systemdaten-Servers
4004:	Datenpuffer zu klein
4101:	Ungültiger Mode (intern)
4102:	Ungültiger Xpath

12.21 Systemdaten schreiben (MT_SD_WR)

12.21.1 Funktion

Der Programmbaustein bietet die Funktion zum Schreiben von Systemdaten.

Es können sowohl komplette Variablen als auch Teil-Variablen geschrieben werden. Die Auswahl erfolgt über die Eingangsvariable **XPath** (z. B. /VAR_1). Die Daten müssen in dem Format übergeben werden, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden und auch die Größe in **Size** muss mit der Größe des Typs übereinstimmen.

In **TypeId** muss die Id des Typs übergeben werden, von dem die Variable auf NC-Seite definiert ist. Diese wird beim Lesen von Variablen dieses Typs zurückgeliefert. Bei mehrfachen Schreibzugriffen muss diese Id nur einmal vor dem ersten Zugriff bestimmt und kann dann bei jedem weiteren Zugriff übergeben werden.

Einschränkungen:

- Auf unterster Ebene können keine kompletten Arrays geschrieben werden (z. B. /ArrayOfInts). Arrays, die Unterelemente einer Systemvariablen sind, können dagegen auch komplett geschrieben werden (z. B. /ArrayOfInts).
- Die Größe der angeforderten Daten ist auf 4 KByte begrenzt.

Im SPS-Zyklus können mehrere Instanzen des Bausteins aufgerufen werden.

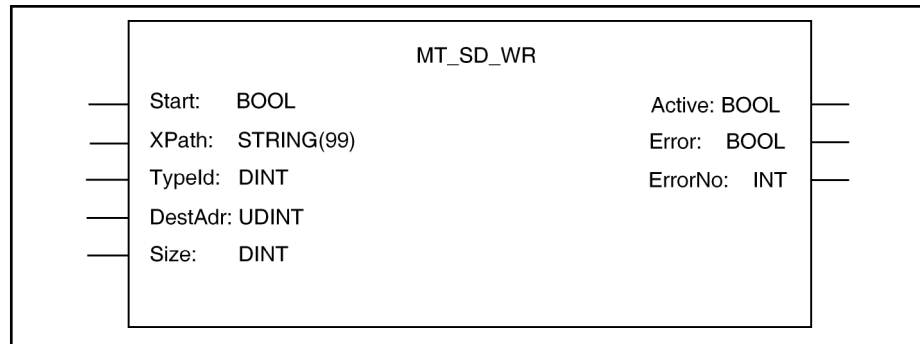


Abb. 12-40: Aufruf des Bausteins MT_SD_WR

Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Start:	Flag, das eine neue Anforderung durchführt. Eine neue Anforderung wird durchgeführt, wenn Start =TRUE und Active = FALSE ist
XPath:	Identifikation der angeforderten Systemdaten incl. Pfadangaben (z. B. "/Var_1")
Typeld:	Id des Typs des Systemdatums
DataAdr:	Adresse der Daten (ADR(<variable>)) Die Daten müssen in dem Format übergeben werden, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden
Size:	Größe der übergebenen Daten in Bytes Die Größe muss hierbei exakt mit der Größe des Systemdatums übereinstimmen

Die Rückgabeparameter lauten:

Active:	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error:	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorNo:	Fehlercode: nur gültig, wenn Error = TRUE

Die Funktion startet, wenn das Bit **Start** gesetzt wird und Active nicht gesetzt ist. Sofern das Bit Start nicht über das SPS-Ablaufprogramm zurückgesetzt wird, wird es zurückgesetzt, wenn die Funktion beendet ist.

Im Status wird **Active** gesetzt, bis die Funktion ausgeführt ist. Sobald die Daten zurückgeliefert sind, ist die Funktion beendet, und **Active** wird zurückgesetzt.

12.21.2 Projektierungsbeispiel

Die Variable "/Var_1.Int_2" soll mit dem Wert 1234 beschrieben werden.

Der Auftrag startet mit steigender Flanke am Eingang **E43.4**.

Der Aufruf des Bausteins **MT_SD_Wr** könnte folgendermaßen ausgeführt werden:

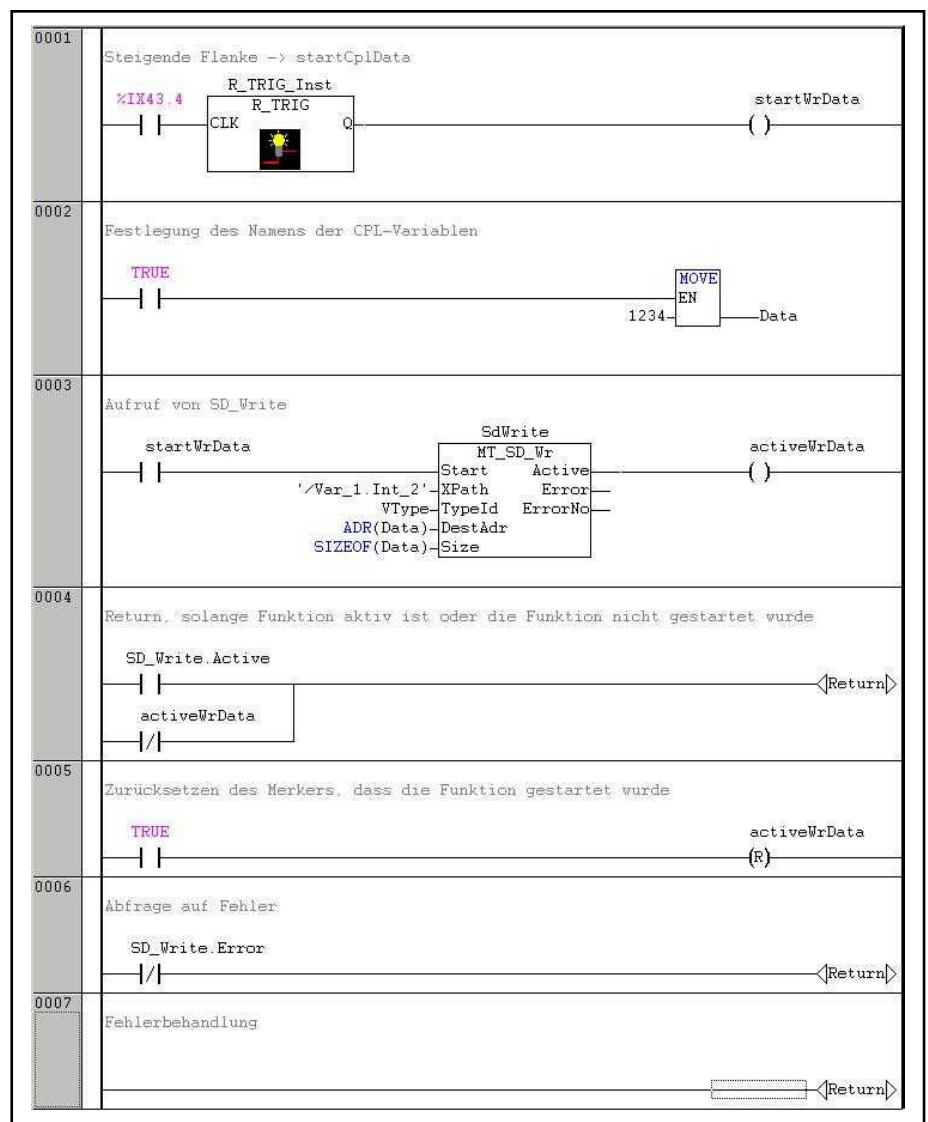


Abb. 12-41: Aufruf des Bausteins MT_SD_Wr

12.21.3 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

-6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden

Allgemeine Fehler und Zustände als Quittung nach Ausführen der Funktion:

1: Fehler innerhalb der XML-Lib
 502: Interner Fehler des XML-Servers
 507: Inhalt des Elements entspricht nicht dem Basis-Datentyp lt. Schema
 508: Inhalt des Elements entspricht nicht den Facets lt. Schema

509:	Interner Fehler beim Validieren
510:	Syntax-Fehler im Path-String
511:	Segment Path liegt nicht innerhalb des RootPath (intern)
512:	Element-Position laut Schema nicht zulässig (Path)
513:	Über Path spezifiziertes Element kann nicht angelegt werden
514:	Über Path spezifiziertes Element kann nicht gelöscht werden
515:	Über Path spezifiziertes Element ist nicht in den XML-Daten enthalten
516:	Über Path spezifiziertes Element ist nicht im Schema deklariert
517:	Letztes Element im Path stimmt nicht mit erstem Element in Data überein
518:	Element aus XML-Datei oder XML-String nicht im Schema deklariert
519:	Datengröße passt nicht zum spezifizierten Element
520:	Datentyp passt nicht zum spezifizierten Element
521:	Es wurden zu viele binäre Daten angefordert
522:	Unbekannter Datentyp
523:	Ungültige Datengröße
524:	Unbekanntes Root-Element
4001:	Interner Fehler des Systemdaten-Servers
4004:	Datenpuffer zu klein
4101:	Ungültiger Mode (intern)
4102:	Ungültiger Xpath

12.22 Diagnosedaten lesen (MT_DiagCode)

12.22.1 Allgemeines

Der Funktionsbaustein "MT_DiagCode" liefert ein Abbild des Inhaltes der NC - Fehler- und Meldungsdatenbank. Über eine entsprechende Eingangsbeschaltung können die Diagnosemeldung auf relevante Bearbeitungskanäle und/oder relevante Meldungsklassen eingeschränkt werden. Die Auswahl der relevanten Klassen erfolgt über eine ODER Verknüpfung der einzelnen Konstanten wie zum Beispiel **16#00000004 OR 16#00000008** für die Auswahl von Antriebs- und Interpolatorfehler.

Ein Diagnoseeintrag wird abgebildet über einer Struktur mit Informationen über:

- Datum
- Uhrzeit
- Fehlerklasse
- Diagnosenummer

Vom Anwender wird der Speicher für eine Liste dieser Strukturen zur Verfügung gestellt. Der FB trägt die Diagnoseeinträge in absteigender zeitlicher Reihenfolge (Neueste zuerst) in die Liste sortiert ein. Ist die vom Anwender zur Verfügung gestellte Liste zu klein für die aufgelaufenen Fehler, sind alle außerhalb der Liste liegende Fehler für den Anwender nicht sichtbar. Ausgangsseitig erfolgt durch den FB eine Benachrichtigung über neue Meldun-

gen und die Angabe deren Anzahl. Zu beachten ist, dass die Diagnosenummer über alle Diagnoseklassen hinweg, mit Ausnahme der MZA-Meldungen, eindeutig ist. MZA-Meldungen besitzen die in der SPS projektierten Meldungsnummer (Bit-Position). NC-Programm- und Antriebsdiagnosen werden in der NC über jeweils eine oder mehrere Sammeldiagnosenummern abgebildet. Eine Unterscheidung einzelner NC-Programm- oder Antriebsdiagnosen ist somit nicht möglich.

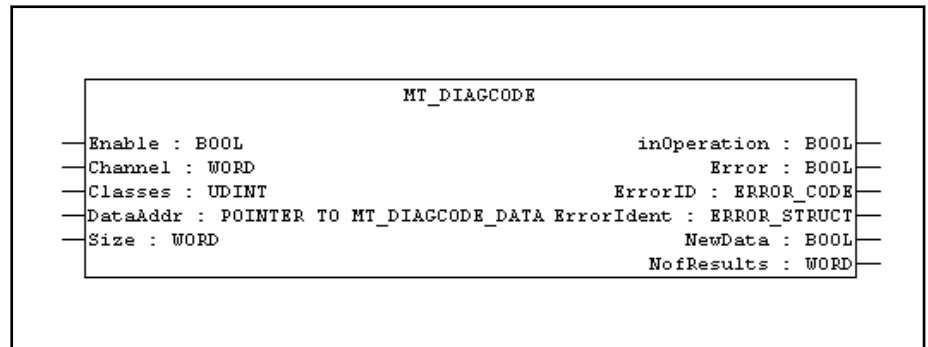


Abb. 12-42: Aufruf des Bausteins MT_DiagCode

12.22.2 Bausteinparameter



Ab Version 14V04: Änderung des Parameters Channel

Als Maske zur Auswahl von mehreren Kanälen kann der Parameter nur für die Kanäle 1-12 verwendet werden. Für das Auslesen der Informationen von Kanalnummern > 12 muss als Parameter die Kanal-Nr. und zusätzlich das höchstwertige Bit gesetzt werden. Daher können bei Kanalnummern > 12 immer nur die Daten eines einzelnen Kanals angefordert werden. Sollen bei der Vorgabe einer Kanalnummer auch alle kanalunabhängigen Informationen geliefert werden, muss zusätzlich zu dem höchstwertigen Bit 15 auch das Bit 14 gesetzt werden. Dies kann über eine ODER-Verknüpfung mit 16#8000 bzw. 16#4000 erfolgen.

Der Baustein wird aufgerufen mit den Parametern:

Enable:	Aktivierungseingang Alle Eingangsparameter werden bei einer steigender Flanke an diesem Eingang übernommen
Channel:	Filter für Bearbeitungskanäle: Bit 0 = 1 kanalunabhängige Diagnosemeldungen Bit m = 1 ($1 \leq m \leq n$) Aktiviert die Diagnose für den Kanal m, n: Anzahl der Kanäle im System (max. 12)

Ab Version 14V04:

Als Maske zur Auswahl von mehreren Kanälen kann der Parameter nur für die Kanäle 1-12 verwendet werden. Für das Auslesen der Informationen von Kanalnummern > 12 muss als Parameter die Kanal-Nr. und zusätzlich das höchstwertige Bit gesetzt werden. Daher können bei Kanalnummern > 12 immer nur die Daten eines einzelnen Kanals angefordert werden. Sollen bei der Vorgabe einer Kanalnummer auch alle kanalunabhängigen Informationen geliefert werden, muss zusätzlich zu dem höchstwertigen Bit 15 auch das Bit 14 gesetzt werden.

Beispiel:

Anforderung der Daten von Kanal 1, 2, 3 und 6 mittels Kanalmaske:

Channel := 2#1001110 ;

Anforderung der Daten von Kanal 27:

Channel := 27 OR 16#8000;

Anforderung der Daten von Kanal 27 inkl. der kanalunabhängigen Informationen:

Channel := 27 OR 16#8000 OR 16#4000;

Classes:	Filter für Meldungsklassen: 16#00000001 - schwerer Systemfehler 16#00000002 - leichter Systemfehler 16#00000004 - Antriebsfehler 16#00000008 - Interpolatorfehler 16#00000010 - Hardwarefehler 16#00000020 - ICL-Fehler 16#00000040 - Teileprogrammfehler 16#00000080 - CPL-Fehler 16#00000100 - BOF-Fehler 16#00000200 - DNC-Fehler 16#00000400 - Ein/Ausgabefehler 16#00000800 - Maschinenfehler 16#00001000 - Kernsystemwarnungen 16#00002000 - Peripheriewarnung 16#00004000 - Schnittstellenwarnung 16#00008000 - Laufzeitwarnung 16#00010000 - Laufzeithinweis 16#00020000 - DNC-Hinweis 16#00040000 - Bediener-Hinweis 16#00080000 - Allgemeiner-Hinweis 16#00100000 - Diagnose-Hinweis 16#00200000 - Maschinenwarnung 16#00400000 - Maschinen-Hinweis
DataAddr:	Pointer auf ein oder mehrere Strukturen des Types MT_DIAGCODE_DATA. Die Datenstrukturen werden vom FB mit den Diagnoseinformationen gefüllt. Der Inhalt der Datenstrukturen ist nur gültig, wenn NewData TRUE ist
Size:	Größe des in DestAddr bereitgestellten Speicher in Bytes
inOperation:	Signalisiert über TRUE, dass der Baustein die interne Diagnose-datenbank auf Änderungen überwacht. Der Ausgang wird im Fehlerfall (Error=TRUE) bzw. bei Deaktivierung des Bausteins über Enable=FALSE (nach Abschluss eines evtl. laufenden Leseprozesses) auf FALSE gesetzt
Error:	Signalisiert über TRUE Fehlerzustände im Baustein. Genauere Diagnosen sind über ErrorID und ErrorIdent möglich
ErrorID:	grobe Fehlerbeschreibung
ErrorIdent:	Detaillierte Fehlerbeschreibung (für künftige Erweiterungen)

NewData: Signalisiert über TRUE, dass unter Berücksichtigung der Filterkriterien (Channel und Classes) neue Diagnosemeldungen aufgetreten sind, oder Diagnosemeldungen gelöscht wurden: der Inhalt des durch DataAddr adressierten Speicherbereiches hat sich geändert

NofResults: Anzahl der Diagnosedatensätze in DataAddr.
Wert ist nur gültig, wenn DataAddr = TRUE ist

Definition der MT_DIAGCODE_DATA Struktur:

```
STRUCT
    DateTime :DATE_AND_TIME; //Date and time of the message in UTC
    Chan :UINT; //Channel in which the diagnostics message occurred
    Class :UDINT; //Diagnostics class
    Number :DINT; //Diagnostics number
END_STRUCT
```

12.22.3 Projektierungsbeispiel

Es sollen alle Diagnosemeldungen des Kanals 1 und alle kanalunabhängigen Meldungen kontinuierlich überwacht werden. Der zur Verfügung gestellte Speicher zur Aufnahme der Diagnoseinformationen ist 32 Strukturelemente groß. Ändert sich der Inhalt des Diagnosespeichers, so wird dieser zur Weiterverarbeitung kopiert.

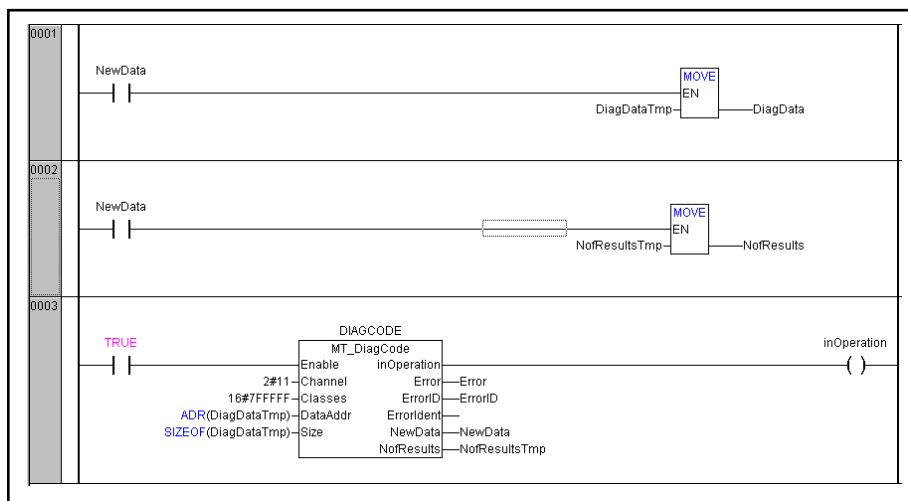


Abb. 12-43: Projektierungsbeispiel

12.22.4 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

- 6: Parameterfehler
- 2: Interner Fehler

12.23 Diagnosetexte lesen (MT_DiagText)

12.23.1 Allgemeines

Der Funktionsbaustein MT_DiagText liefert ein Abbild des Inhaltes der NC-Fehler- und Meldungsdatenbank. Über eine entsprechende Eingangsbeschaltung können die Diagnosemeldungen auf relevante Bearbeitungskanäle und/oder relevante Meldungsklassen eingeschränkt werden. Die Auswahl der relevanten Klassen erfolgt über eine ODER Verknüpfung der einzelnen Konstanten wie zum Beispiel **16#00000004 OR 16#00000008** für die Auswahl von Antriebs- und Interpolatorfehler.

Im Unterschied zu dem Funktionsbaustein MT_DiagCode werden auch die Diagnosetexte mitgeliefert.

Voraussetzung Die sprachabhängigen Textdateien müssen im UTF8 Format in das Verzeichnis "text" des Benutzerfeproms geladen werden.

Ein Diagnoseeintrag wird abgebildet über eine Struktur mit Informationen über:

- Datum, Uhrzeit
- Kanal
- Fehlerklasse
- Diagnosenummer
- Diagnosetext

Vom Anwender wird der Speicher für eine Liste dieser Strukturen mit Angabe der Anzahl der Listenelemente dem FB zur Verfügung gestellt. Der FB trägt die Diagnoseeinträge in absteigender zeitlicher Reihenfolge (den neuesten Eintrag zuerst) in die Liste sortiert ein. Ist die vom Anwender zur Verfügung gestellte Liste zu klein für die aufgelaufenen Fehler, so sind alle außerhalb der Liste liegende Fehler für den Anwender nicht sichtbar. Ausgangsseitig erfolgt durch den FB eine Benachrichtigung über neue Meldungen und die Anzahl der Meldungen. Zu beachten ist, dass die Diagnosenummer über alle Diagnoseklassen hinweg, mit Ausnahme der MZA-Meldungen, eindeutig ist. MZA-Meldungen besitzen die in der SPS projizierten Meldungsnummer (Bit-Position). NC-Programm- und Antriebsdiagnosen werden in der NC über jeweils eine oder mehrere Sammeldiagnosenummern abgebildet. Eine Unterscheidung einzelner NC-Programm- oder Antriebsdiagnosen ist somit nicht möglich.

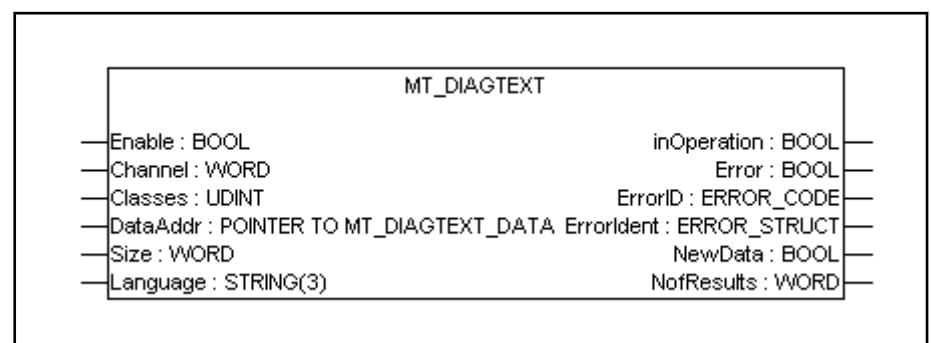


Abb. 12-44: Aufruf des Bausteins MT_DiagText

Einschränkung Es können maximal 3 Instanzen des FBs pro SPS-Task angelegt werden.

12.23.2 Bausteinparameter



Ab Version 14V04: Änderung des Parameters Channel

Als Maske zur Auswahl von mehreren Kanälen kann der Parameter nur für die Kanäle 1-12 verwendet werden. Für das Auslesen der Informationen von Kanalnummern > 12 muss als Parameter die Kanal-Nr. und zusätzlich das höchstwertige Bit gesetzt werden. Daher können bei Kanalnummern > 12 immer nur die Daten eines einzelnen Kanals angefordert werden. Sollen bei der Vorgabe einer Kanalnummer auch alle kanalunabhängigen Informationen geliefert werden, muss zusätzlich zu dem höchstwertigen Bit 15 auch das Bit 14 gesetzt werden. Dies kann über eine ODER-Verknüpfung mit 16#8000 bzw. 16#4000 erfolgen.

Enable	Aktivierungseingang, Alle Eingangsparameter werden bei einer steigender Flanke an diesem Eingang übernommen.

Channel	Filter für Bearbeitungskanäle: Bit 0 = 1 kanalunabhängige Diagnosemeldungen Bit m = 1 (1 ≤ m ≤ n) Aktiviert die Diagnose für den Kanal m, n: Anzahl der Kanäle im System (max. 12)

Ab Version 14V04:

Als Maske zur Auswahl von mehreren Kanälen kann der Parameter nur für die Kanäle 1-12 verwendet werden. Für das Auslesen der Informationen von Kanalnummern > 12 muss als Parameter die Kanal-Nr. und zusätzlich das höchstwertige Bit gesetzt werden. Daher können bei Kanalnummern > 12 immer nur die Daten eines einzelnen Kanals angefordert werden. Sollen bei der Vorgabe einer Kanalnummer auch alle kanalunabhängigen Informationen geliefert werden, muss zusätzlich zu dem höchstwertigen Bit 15 auch das Bit 14 gesetzt werden.

Beispiel:

Anforderung der Daten von Kanal 1, 2, 3 und 6 mittels Kanalmaske:

Channel := 2#1001110 ;

Anforderung der Daten von Kanal 27:

Channel := 27 OR 16#8000;

Anforderung der Daten von Kanal 27 inkl. der kanalunabhängigen Informationen:

Channel := 27 OR 16#8000 OR 16#4000;

Classes	Filter für Meldungsklassen: 16#00000001 - schwerer Systemfehler 16#00000002 - leichter Systemfehler 16#00000004 - Antriebsfehler 16#00000008 - Interpolatorfehler 16#00000010 - Hardware Fehler 16#00000020 - ICL-Fehler 16#00000040 - Teileprogrammfehler 16#00000080 - CPL-Fehler 16#00000100 - BOF-Fehler 16#00000200 - DNC-Fehler 16#00000400 - Ein/Ausgabefehler 16#00000800 - Maschinenfehler 16#00001000 - Kernsystem-Warnungen 16#00002000 - Peripherie Warnung 16#00004000 - Schnittstellenwarnung 16#00008000 - Laufzeitwarnung 16#00010000 - Laufzeithinweis 16#00020000 - DNC-Hinweis 16#00040000 - Bediener-Hinweis 16#00080000 - Allgemeiner-Hinweis 16#00100000 - Diagnose-Hinweis 16#00200000 - Maschinen-Warnung 16#00400000 - Maschinen-Hinweis
DataAddr	Pointer auf ein oder mehrere Strukturen des Types MT_DIAG-TEXT_DATA. Die Datenstrukturen werden vom FB mit den Diagnoseinformationen gefüllt. Der Inhalt der Datenstrukturen ist nur gültig, wenn NewData TRUE ist.
Size	Größe des in DestAddr bereitgestellten Speicher in Bytes
Language	Länderkennung für die gewählte Sprache z. B. "049" für Deutsch
inOperation	Signalisiert über TRUE, dass der Baustein die interne Diagnose-datenbank auf Änderungen überwacht. Der Ausgang wird im Fehlerfall (Error=TRUE) bzw. bei Deaktivierung des Bausteins über Enable=FALSE (nach Abschluss eines evtl. laufenden Lese-prozesses) auf FALSE gesetzt.
Error	Signalisiert über TRUE Fehlerzustände im Baustein. Genauere Diagnosen sind über ErrorID und ErrorIdent möglich.
ErrorID	grobe Fehlerbeschreibung
ErrorIdent	Detaillierte Fehlerbeschreibung (für künftige Erweiterungen)

NewData	Signalisiert über TRUE, dass unter Berücksichtigung der Filterkriterien (Channel und Classes) neue Diagnosemeldungen aufgetreten sind, oder Diagnosemeldungen gelöscht wurden: der Inhalt des durch DataAddr adressierten Speicherbereiches hat sich geändert.
NofResults	Anzahl der Diagnosedatensätze in DataAddr. Wert ist nur gültig, wenn DataAddr = TRUE ist.

Definition der MT_DIAGTEXT_DATA Struktur:

Programm:

```
STRUCT
  DateTime :DATE_AND_TIME; //Date and time of the message in UTC
  Chan :UINT; //Channel in which the diagnostics message occurred
  Class :UDINT; //Diagnostics class
  Number :DINT; //Diagnostics number
END_STRUCT
```

12.23.3 Projektierungsbeispiel

siehe [Kap. 12.22.3 "Projektierungsbeispiel" auf Seite 202.](#)

12.23.4 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

3:	Es können keine weiteren Instanzen des FBs angelegt werden
6:	Parameterfehler
-2:	Interner Fehler

12.24 Lesen der Motion Control Data-Services (MT_MCoDS)

12.24.1 Allgemeines

Der Funktionsbaustein MT_MCoDS ermöglicht den lesenden Zugriff auf die internen Datendienste der CNC. Die Parametrierung des Bausteins basiert weitestgehend auf der Beschreibung des MCoDS-Befehls im Programmierhandbuch (Kapitel: CPL-Funktionen / NCS-Ankopplung) und sind diesem zu entnehmen.

Die Ergebniswerte werden in einem, vom Anwender zur Verfügung gestellten Speicherbereich geschrieben. Ausgehend von der Beschreibung im Programmierhandbuch ist dabei folgendes Typemapping zu beachten:

CPL Datentypen	SPS-Datentypen
Double	LREAL
Real	LREAL
Integer	DINT
Character	STRING

Tab. 12-2: Typemapping für Funktionsbaustein MT_MCoDS

Bei der Verwendung des Funktionsbausteins ist zu beachten, dass der Zugriff auf die internen Datendienste der CNC nicht zu jedem beliebigen Zeitpunkt gewährleistet ist. In diesen Fällen kann der Funktionsbaustein keine gültigen Daten bereitstellen. Der Ausgang Error wird dabei auf True gesetzt, der Ergebnisspeicher wird mit Nullen aufgefüllt.

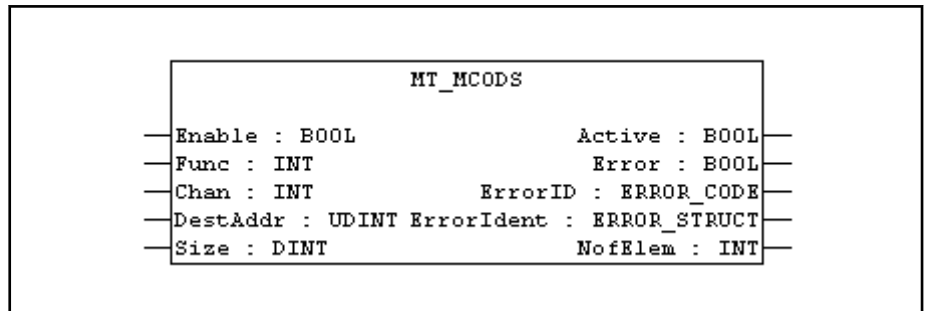


Abb. 12-45: Aufbau der Bausteins MT_MCoDS

12.24.2 Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Enable	Aktivierungseingang
Func	Funktionsnummer für angeforderten Datendienst (siehe MCoDS-Beschreibung im Programmierhandbuch) Die Funktionen 42, 55 56, 57, 62, 94, und 95 werden nicht unterstützt.
Chan	Angabe des Kanals, auf den die Funktion wirken soll
DataAddr	Pointer auf den Ergebnissdatenbereich
Size	Größe des in DestAddr bereitgestellten Speichers in Bytes
Active	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error	Signalisiert über TRUE Fehlerzustände im Baustein. Genauere Diagnosen sind über ErrorID und ErrorIdent möglich
ErrorID	grobe Fehlerbeschreibung
ErrorIdent	Detaillierte Fehlerbeschreibung (für künftige Erweiterungen)
NofElem	Anzahl der zurückgelieferten Elemente

12.24.3 Projektierungsbeispiel

Im folgendem Beispiel werden kontinuierlich die Sollpositionen aller Vorschub- und Hilfsachsen in aufsteigender, kanalunabhängiger Reihenfolge ermittelt. Für die Ergebnisse wird ein Feld von LREAL-Werten (AchsPos) bereitgestellt.

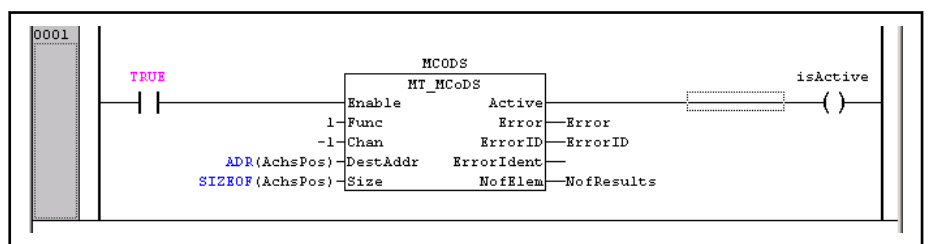


Abb. 12-46: Projektierungsbeispiel für Funktionsbaustein MT_MCoDS

12.24.4 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

6:	Parameterfehler
-2:	Interner Fehler

12.25 Automatischer Sercos III I/O-Tausch (MT_S3AutoAdapt)

12.25.1 Allgemeines

Der Tausch eines einzelnen Sercos III I/O-Teilnehmers an einer sich in Produktion befindlichen Anlage ist jederzeit ohne Verwendung von Oberflächen-Tools möglich, sofern der Maschinenhersteller die Funktion zum "automatischen Sercos III I/O-Tausch" freigibt.

Die Funktion zum automatischen I/O-Tausch kann im SPS-Anwender-Programm jederzeit über den Funktionsbaustein MT_S3AutoAdapt ein- oder ausgeschaltet werden.

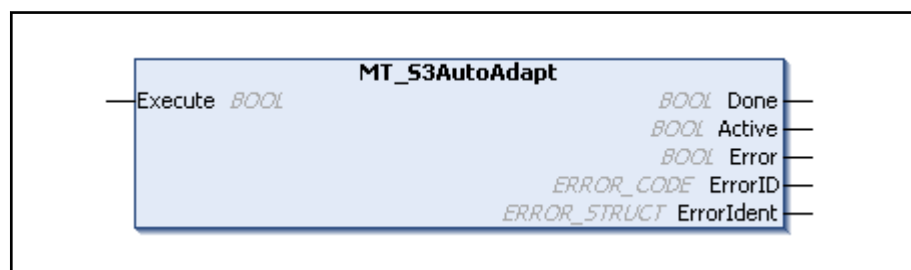


Abb. 12-47: Aufruf Baustein MT_S3AutoAdapt

12.25.2 Bausteinparameter

Der Baustein wird mit folgenden Parametern aufgerufen:

Execute	Flag für eine neue Anforderung
Active	Flag, das anzeigt, ob die Funktion noch aktiv ist
Error	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorID	Fehlercode: nur gültig, wenn Error = TRUE
ErrorIdent	Detaillierte Fehlerbeschreibung (für künftige Erweiterungen)

12.25.3 Projektierungsbeispiel

Das Aktivieren der Funktion erfolgt mit steigender Flanke am Eingang %IX41.3.

Der Aufruf des Bausteines **MT_S3AutoAdapt** könnte folgendermaßen ausgeführt werden:

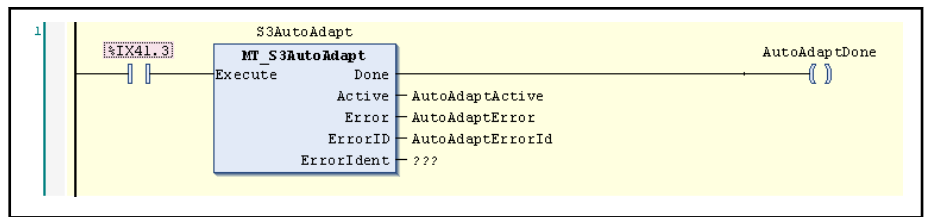


Abb. 12-48: Projektierungsbeispiel für Funktionsbaustein MT_S3AutoAdapt

12.25.4 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände bei Funktionsaufruf durch falsche Parametrierung oder NC-interne Fehler:

0:	Kein Fehler: Keine Adressanpassung notwendig oder die automatische Adressanpassung wurde fehlerfrei durchgeführt
506:	Es sind mehr als 2 I/O-Teilnehmer getauscht worden
507:	Die automatische Adressanpassung konnte nicht durchgeführt werden. Weitere Diagnose mit der IndraWorks-Oberfläche
508:	Die automatische Adressanpassung wird nicht unterstützt

12.26 Sercos Ringtopologie auslesen (MT_RingTop)

12.26.1 Allgemeines

Dieser Baustein liefert die Ist-Topologie des Sercos Rings und eine Information über den Online-Status der Sercos Geräte.

Zusätzlich wird bei einer Ringunterbrechung die Unterbrechungsstelle angezeigt.

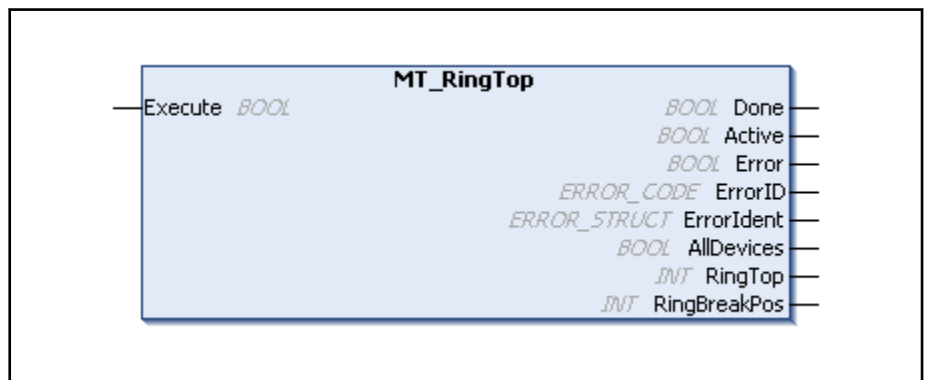


Abb. 12-49: Aufruf Baustein MT_RingTop

12.26.2 Bausteinparameter

Der Baustein wird mit folgenden Parametern aufgerufen:

Execute	Flag für eine neue Anforderung
Done	Anzeige, dass der Auftrag erfolgreich beendet wurde
Active	Flag, das anzeigt, dass der Baustein noch aktiv ist

Error	Fehlerkennung: FALSE bedeutet, dass die Funktion fehlerfrei abgearbeitet wurde
ErrorID	Fehlercode: nur gültig, wenn Error = TRUE
ErrorIdent	Detaillierte Fehlerbeschreibung (für künftige Erweiterungen)
AllDevices	TRUE - Alle Teilnehmer sind aktiv FALSE - Ein oder mehrere Teilnehmer sind ausgefallen
RingTop	Aktuelle Ringtopologie: 0 - Unbekannte Topologie. Die Sercos Kommunikation ist nicht aktiv 1 - Eine Linie an Port 1 ist aktiv 2 - Eine Linie an Port 2 ist aktiv 3 - Zwei Linien sind aktiv 4 - Ein geschlossener Ring ist aktiv
RingBreakPos	Unterbrechungsposition des Rings (Letzter Teilnehmer an Port 1): -1 - Ein geschlossener Ring ist aktiv (RingTop = 4) 0 - Der Ring ist zwischen Port 1 und dem ersten Teilnehmer unterbrochen oder es ist eine Linie an Port 2 angeschlossen sonst - Der Ring ist zwischen diesem Teilnehmer und Port 2 unterbrochen oder es ist eine Linie an Port 1 angeschlossen

12.26.3 Projektierungsbeispiel

Das Aktivieren der Funktion erfolgt mit steigender Flanke am Eingang %IX41.3.

Der Aufruf des Bausteines **MT_RingTop** könnte folgendermaßen ausgeführt werden:

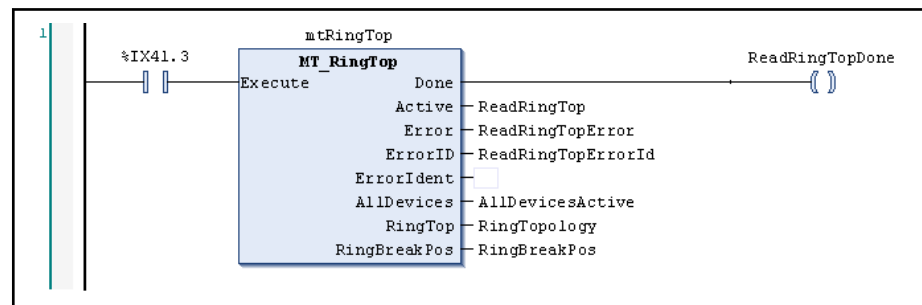


Abb. 12-50: Projektierungsbeispiel für Funktionsbaustein MT_RingTop

12.26.4 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, Die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt und in "ErrorID" eine Fehlernummer ausgegeben.

Allgemeine Fehler und Zustände:

-6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden

12.27 Log-Eintrag im Aktionsrekorder (MT_LogData)

12.27.1 Allgemeines

Mit Hilfe des Funktionsbausteins "MT_LogData" können Texte (LogText) und Inhalte von SPS-Variablen (VarAddr, VarSize) in das Logbuch des Aktionsrekorders geschrieben werden. Für eine spätere Nachbearbeitung kann dem Eintrag bei Bedarf eine Kennung (LogTextID) mitgegeben werden. Über einen Dateinamen (AddFiles) können auch mehrere auf der SPS hinterlegte Dateien dem Log-Eintrag zugeordnet werden. Der Aktionsrekorder kopiert die betreffenden Dateien in den Bereich des Aktionsrekorders auf dem mount-Verzeichnis. Innerhalb der Aktionsrekorder-Anzeige werden diese Daten bei Bedarf mit angezeigt. Über eine Bit-Maske (NcDataSel) kann die SPS vorgeben, welche NC-seitigen Daten zu dem jeweiligen Log-Eintrag in das Logbuch übernommen werden. Der Eingang ChannelSel ermöglicht das kanalabhängige Schreiben der NC-seitigen Daten. Der Typ des Log-Eintrages (EventType) dient innerhalb der Anzeige des Aktionsrekorders zur Klassifizierung der Maschinenereignisse. Mit Hilfe des Eingangs Component kann das Weiteren von der SPS aus das betreffende Maschinenereignis einer Komponente zugeordnet werden. Dies ist insbesondere für Kritische Events und Crashes relevant.

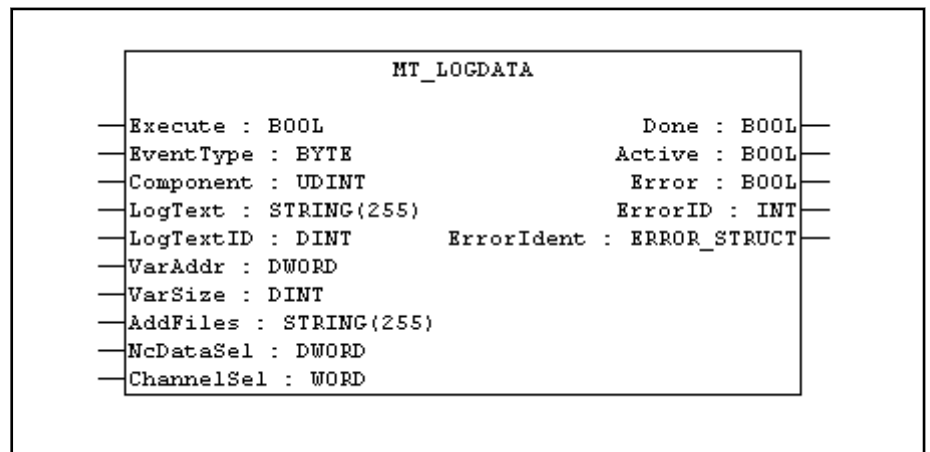


Abb. 12-51: Aufruf Baustein MT_LogData

12.27.2 Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Execute	Aktivierungseingang, Alle Eingangsparameter werden bei einer steigenden Flanke an diesem Eingang übernommen. Solange der Baustein noch aktiv ist (Active=TRUE), werden keine neuen Aufträge angenommen
EventType	Typ des Logbucheintrages: MT_LOGDATA_NORMAL (0) MT_LOGDATA_SPECIAL (1) MT_LOGDATA_CRITICAL (2) MT_LOGDATA_CRASH (3) (Änderungen / Erweiterungen unter der Verwendung der Sprachdatei „MTXacr_BaseLogData.xxx“ möglich)

Component	Frei definierbare ID der meldenden Komponente: (Änderungen / Erweiterungen unter Verwendung der Sprachdatei „MTXacr_BaseLogData.xxx“) Beispiel: 1: Spindel 2: Werkzeugwechsler 3: Werkstückspanner
LogText	Text für den Log-Eintrag (Leerstring, falls nicht benutzt)
LogTextID	Text-Kennung für die Nachbearbeitung unter Verwendung der Sprachdatei „MTXacr_AddLogData.xxx“ 0: keine Nachbearbeitung 1: "My Text %d" 2: "My Nice Text" 3: ...
VarAddr	Adresse der Variablen, deren Inhalt in den Text, der durch LogTextID ausgewählt ist, eingefügt wird. Falls LogTextID 0 ist, wird dieser Parameter ignoriert ACHTUNG ! Da die übergebenen Werte an Hand des Formatstrings im Text, ausgewählt durch LogTextID, interpretiert werden, muss der Typ der Variablen genau zu den Größen der Formatangaben im Text passen. Bei IL2G muss eine Struktur zusätzlich gepackt definiert werden !
VarSize	Gesamtgröße der Variablen in Bytes
AddFiles	Name und Pfad zusätzlicher Dateien, die der Aktionsrekorder ins Mountverzeichnis kopiert und in die ACR-Verwaltung einbindet . Es kann mehr als ein File vorgegeben werden; die Trennung erfolgt durch ",",
NcDataSel	Bitmaske zur Auswahl der NC-Daten, die mit diesem Eintrag ausgezeichnet werden: (Eine Mehrfachauswahl ist über eine OR-Verknüpfung möglich) MT_LOGDATA_NODATA (0): keine NC-Daten aufzeichnen MT_LOGDATA_AXPOS (1): Achspositionen aufzeichnen MT_LOGDATA_PRGLVL (2): Programmebeneninformation für alle Kanäle (Pfad und Dateiname der aktiven NC-Programme sowie die jeweilige aktuelle Position über alle aktiven Ebenen hinweg) MT_LOGDATA_MODGFUNC (4): Modale G-Funktionen aufzeichnen MT_LOGDATA_ACTTOOL (8): Aktives Werkzeug aufzeichnen (Aktives Werkzeug und vorausgewähltes Werkzeug mit jeweils Platz, Sektor, Schneidenanzahl, Name und T-NR sowie die aktive Schneide [L1, L2, L3, R, ORI, Theta, Phi]) MT_LOGDATA_TABLES (16): Dateiname der aktiven Korrekturtabelle (ZeroOffset , Placement , D-Corr table) MT_LOGDATA_AUXFUNC (32): Hilfsfunktionen

Channel-Sel	Auswahl der Kanäle für NcData MT_LOGDATA_ALLCHAN : NcData in allen Kanälen aufzeichnen MT_LOGDATA_CHAN01 : NcData in Kanal 1 aufzeichnen MT_LOGDATA_CHAN02 : NcData in Kanal 2 aufzeichnen : MT_LOGDATA_CHAN12 : NcData in Kanal 12 aufzeichnen (Note: Sind Achs-spezifische ausgewählt, werden die Daten aller Achsen in die Log-Datei übernommen)
Active	Signalisiert über TRUE, dass der Baustein den Auftrag bearbeitet
Done	Anzeige, dass das Kommando erfolgreich beendet wurde
Error	Signalisiert über TRUE Fehlerzustände im Baustein. Genauere Diagnosen sind über ErrorID möglich.
ErrorID	Fehlercode (Nur gültig, falls Error = TRUE)
ErrorIdent	nicht benutzt

12.27.3 Projektierungsbeispiel

Im Projektierungsbeispiel wird ein Log-Eintrag vom Typ NORMAL an den Aktionsrekorder übergeben.

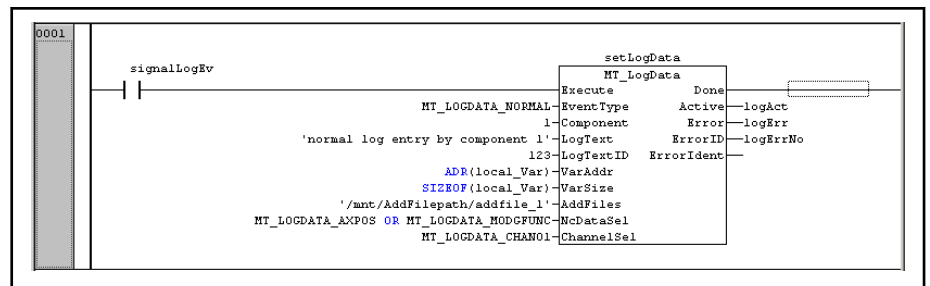


Abb. 12-52: Projektierungsbeispiel für Funktionsbaustein MT_LogData

12.27.4 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände:

6:	Parameterfehler
-2:	Interner Fehler

Funktionsabhängige Fehler und Zustände:

1:	Interner Fehler des Aktionsrekorders
2:	Aktionsrekorder nicht aktiv
3:	Unzulässiger Parameter
4:	Ungültige Ncs Nachricht
5:	Log-Einträge durch die SPS sind nicht aktiv
6:	Bereich für die Variablen zu groß (max. 512)

- 7: Text für den Log-Eintrag ist zu groß (max. 255 + abschließende '\$00')
- 8: Zusätzlichen Dateinamen sind zu groß (max. 255 + abschließende '\$00')

12.28 Systemdaten segmentiert lesen (MT_SD_Rd_Seg)

12.28.1 Allgemeines

Der Programmbaustein bietet die Möglichkeit zum Lesen von Systemdaten, die auch größer als 4K Byte sind. Es können sowohl komplette Variablen als auch Teil-Variablen gelesen werden. Die Auswahl erfolgt über die Eingangsvariable XPath (z. B. MyVarArr_1). Die Daten werden in dem Format geliefert, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden. Für den korrekten Zugriff auf die Daten müssen entsprechende Strukturen mit dem Pragma {attribute 'pack_mode':='1'} innerhalb des SPS-Programms definiert werden.

Einschränkungen:

Auf unterster Ebene können keine kompletten Arrays gelesen werden. Arrays, die Unter-elemente einer Systemvariablen sind, können dagegen auch komplett gelesen werden (z. B. /ArrayOfInts).

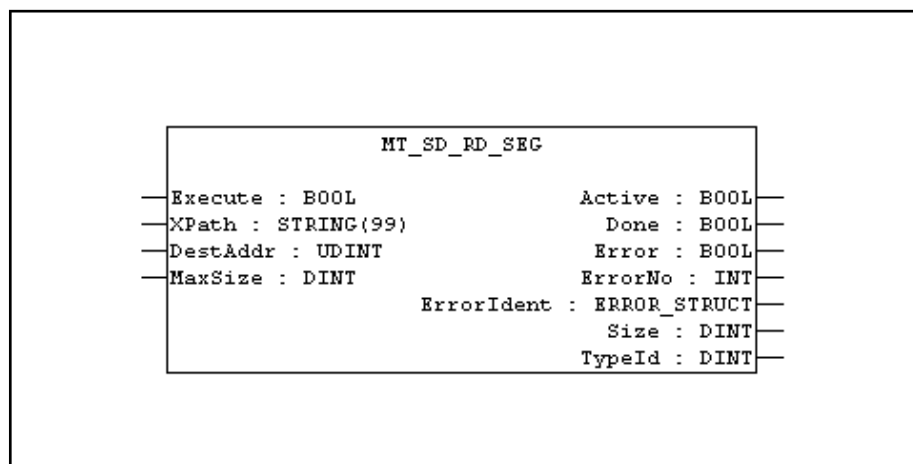


Abb. 12-53: Aufruf Baustein MT_SD_Rd_Seg

12.28.2 Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Execute	Aktivierungseingang, Alle Eingangsparameter werden bei einer steigender Flanke an diesem Eingang übernommen. Solange der Baustein noch aktiv ist (Active=TRUE), werden keine neuen Aufträge angenommen
XPath	Identifikation der Systemdaten incl. Pfadangaben (z. B. "MyVarArr_1")
DestAddr	Adresse der Daten Die Variable muss in demselben Format definiert sein, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden
MaxSize	Gesamtgröße der Variablen in Bytes
Active	Signalisiert über TRUE, dass der Baustein den Auftrag bearbeitet
Done	Anzeige, dass das Kommando erfolgreich beendet wurde

Error	Signalisiert über TRUE Fehlerzustände im Baustein. Genauere Diagnosen sind über ErrorID möglich.
ErrorNo	Fehlercode (Nur gültig, falls Error = TRUE)
ErrorIdent	nicht benutzt

12.28.3 Projektierungsbeispiel

Im Projektierungsbeispiel wird die Systemvariable **MyVarArr_1** gelesen und nach dem Empfang der Daten das Feldelement 10 in die lokale Variable **loc_Var** übernommen.

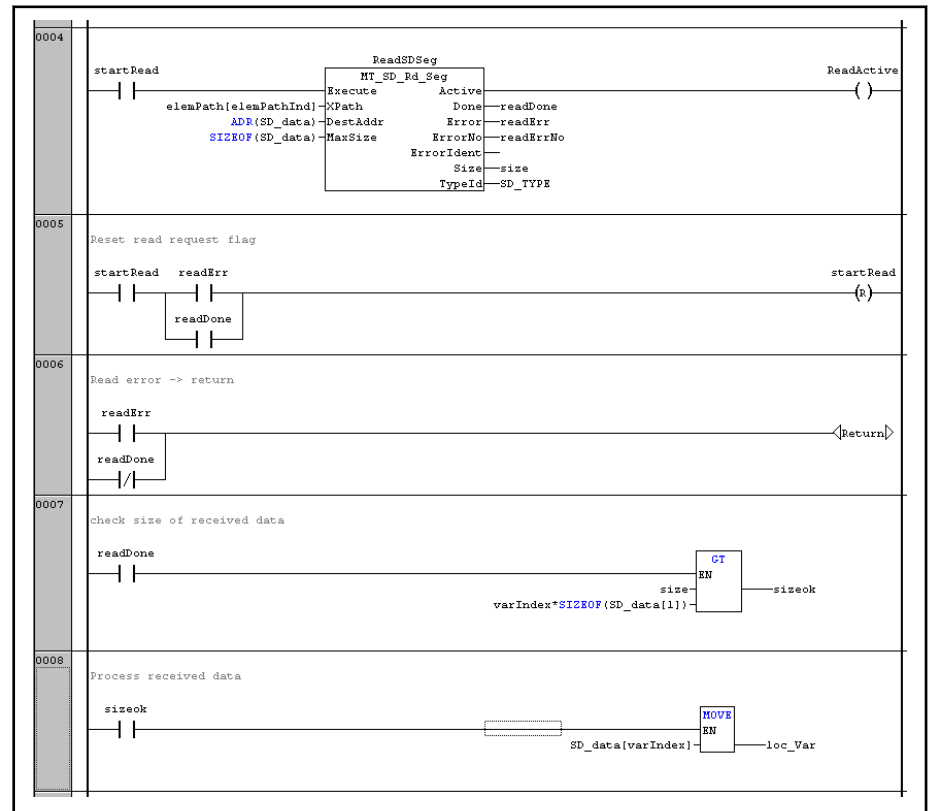


Abb. 12-54: Projektierungsbeispiel für Funktionsbaustein MT_SD_Rd_Seg

12.28.4 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände:

-6:	Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden

Funktionsabhängige Fehler und Zustände:

1:	Fehler innerhalb der XML-Lib
507:	Inhalt des Elements entspricht nicht dem Basis-Datentyp lt. Schema
508:	Inhalt des Elements entspricht nicht den Facets lt. Schema
509:	Interner Fehler beim Validieren

510:	Syntax-Fehler im Path-String
511:	Segment Path liegt nicht innerhalb des RootPath (intern)
512:	Element-Position laut Schema nicht zulässig (Path)
513:	Über Path spezifiziertes Element kann nicht angelegt werden
514:	Über Path spezifiziertes Element kann nicht gelöscht werden
515:	Über Path spezifiziertes Element ist nicht in den XML-Daten enthalten
516:	Über Path spezifiziertes Element ist nicht im Schema deklariert
517:	Letztes Element im Path stimmt nicht mit erstem Element in Data überein
518:	Element aus XML-Datei oder XML-String nicht im Schema deklariert
520:	Datentyp passt nicht zum spezifizierten Element
521:	Es wurden zu viele binäre Daten angefordert
522:	Unbekannter Datentyp
523:	Ungültige Datengröße
524:	Unbekanntes Root-Element
4001:	Interner Fehler des Systemdaten-Servers
4004:	Datenpuffer zu klein
4101:	Ungültiger Mode (intern)
4102:	Ungültiger Xpath

12.29 Systemdaten segmentiert schreiben (MT_SD_Wr_Seg)

12.29.1 Allgemeines

Der Programmbaustein bietet die Möglichkeit zum Schreiben von Systemdaten, die auch größer als 4K Byte sind. Es können sowohl komplette Variablen als auch Teil-Variablen geschrieben werden. Die Auswahl erfolgt über die Eingangsvariable XPath (z. B. MyVarArr_1). Die Daten müssen in dem Format übergeben werden, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden und auch die Größe in Size muss mit der Größe des Typs innerhalb der Systemdaten übereinstimmen. Für den korrekten Zugriff auf die Daten müssen entsprechende Strukturen mit dem Pragma {attribute 'pack_mode':='1'} innerhalb des SPS-Programms definiert werden. In Typeld muss die Id des Typs übergeben werden, von dem die Variable auf NC-Seite definiert ist. Diese wird beim Lesen von Variablen dieses Typs zurückgeliefert. Bei mehrfachen Schreibzugriffen muss diese Id nur einmal vor dem ersten Zugriff bestimmt werden und kann dann bei jedem weiteren Zugriff übergeben werden.

Einschränkungen:

Auf unterster Ebene können keine kompletten Arrays gelesen werden. Arrays, die Unterelemente einer Systemvariablen sind, können dagegen auch komplett gelesen werden (z. B. /ArrayOfInts).

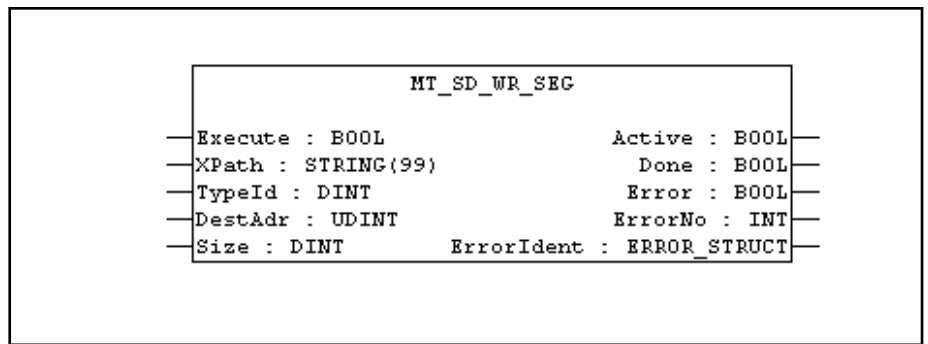


Abb. 12-55: Aufruf Baustein MT_SD_Wr_Seg

12.29.2 Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Execute	Aktivierungseingang, Alle Eingangsparameter werden bei einer steigenden Flanke an diesem Eingang übernommen. Solange der Baustein noch aktiv ist (Active=TRUE), werden keine neuen Aufträge angenommen
XPath	Identifikation der Systemdaten incl. Pfadangaben (z. B. "MyVarArr_1")
DestAddr	Adresse der Daten Die Variable muss in demselben Format definiert sein, wie sie auf NC-Seite durch die Schema-Dateien definiert wurden
MaxSize	Gesamtgröße der Variablen in Bytes
Active	Signalisiert über TRUE, dass der Baustein den Auftrag bearbeitet
Done	Anzeige, dass das Kommando erfolgreich beendet wurde
Error	Signalisiert über TRUE Fehlerzustände im Baustein. Genauere Diagnosen sind über ErrorID möglich.
ErrorNo	Fehlercode (Nur gültig, falls Error = TRUE)
ErrorIdent	nicht benutzt

12.29.3 Projektierungsbeispiel

Im Projektierungsbeispiel wird die Systemvariable **MyVarArr_1** komplett geschrieben. Zuvor wurde die Typeld des Systemdatums einmalig durch das Lesen der Variable bestimmt. Die Größe von **MyVarArr_1** muss genau mit der Größe des Systemdatums übereinstimmen

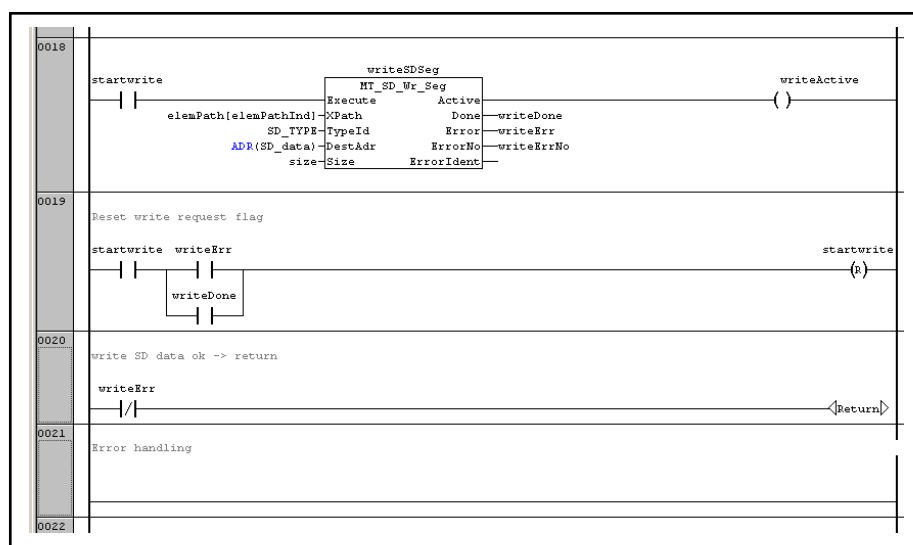


Abb. 12-56: Projektierungsbeispiel für Funktionsbaustein MT_SD_Wr_Seg

12.29.4 Statusmeldungen

Ist nach dem Aufrufen der Funktion ein Fehler oder Zustand entstanden, der es nicht erlaubt, die Funktion ordnungsgemäß durchzuführen, wird das Fehlerbit "Error" gesetzt. Weiterhin wird in "ErrorNo" die Art des Fehlers oder Zustandes in Form einer Zahl dargestellt:

Allgemeine Fehler und Zustände:

-6: Zur Zeit keine Betriebsmittel zum Versenden des Auftrags vorhanden

Funktionsabhängige Fehler und Zustände:

1:	Fehler innerhalb der XML-Lib
507:	Inhalt des Elements entspricht nicht dem Basis-Datentyp lt. Schema
508:	Inhalt des Elements entspricht nicht den Facets lt. Schema
509:	Interner Fehler beim Validieren
510:	Syntax-Fehler im Path-String
511:	Segment Path liegt nicht innerhalb des RootPath (intern)
512:	Element-Position laut Schema nicht zulässig (Path)
513:	Über Path spezifiziertes Element kann nicht angelegt werden
514:	Über Path spezifiziertes Element kann nicht gelöscht werden
515:	Über Path spezifiziertes Element ist nicht in den XML-Daten enthalten
516:	Über Path spezifiziertes Element ist nicht im Schema deklariert
517:	Letztes Element im Path stimmt nicht mit erstem Element in Data überein
518:	Element aus XML-Datei oder XML-String nicht im Schema deklariert
520:	Datentyp passt nicht zum spezifizierten Element
521:	Es wurden zu viele binäre Daten angefordert
522:	Unbekannter Datentyp
523:	Ungültige Datengröße

524:	Unbekanntes Root-Element
4001:	Interner Fehler des Systemdaten-Servers
4004:	Datenpuffer zu klein
4101:	Ungültiger Mode (intern)
4102:	Ungültiger Xpath

12.30 Fast Action Eingangsdaten lesen (MT_FaDataRead)

12.30.1 Allgemeines

Die Funktion dient zum Lesen der zyklisch übertragenen Daten vom Interpolator zur Fast Plc Task. Zu Beginn der Fast Plc Task müssen die Eingangsdaten unter Verwendung dieser Funktion gelesen werden. Nach dem Einlesen der Daten kann mit der eigentlichen Programmbearbeitung begonnen werden.

Die Größe der Struktur für die Eingangsdaten muss exakt diesselbe Größe aufweisen wie die auf NC-Seite konfigurierten Eingangsdaten für diese Fast Plc Task



Da die Daten auf NC-Seite gepackt übergeben werden, muss bei der Typ-Definition auf SPS-Seite die Struktur ebenfalls mit **{attribute 'pack_mode' := '1'}** gepackt definiert werden.

Eine ausführliche Beschreibung enthält die Dokumentation **MTX Funktionsbeschreibung - Sonderfunktionen** unter dem Kapitel **Schnelle Aktionen MTX fa**



Dieser Baustein darf nur aus einer Fast Plc Task aufgerufen werden (Die Task muss vom Typ **Extern ereignisgesteuert** und das Externe Ereignis **EVENT_FASTACTION_x** sein, wobei x für den Index 1 bis 4 steht)

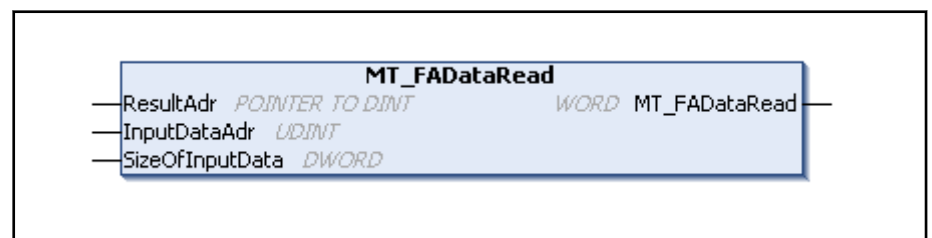


Abb. 12-57: Aufruf Baustein MT_FaDataRead

12.30.2 Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Funktionswert	Der Funktionswert enthält das Ergebnis der Funktion Es ist dies das Ergebnis der Bibliotheksfunktion, nicht das Ergebnis einer NC-seitigen Fast Action (s. u.)
ResultAdr	Reserviert für zukünftige Erweiterungen (Pointer auf den Return-Wert der NC-seitigen FA). Wird momentan noch nicht genutzt

InputDataAdr	Pointer auf die Daten, die vom Interpolator an die Fast Plc Task übertragen wurden
SizeOfInputData	Gesamtgröße des Datenbereichs der Eingangsdaten in Bytes

12.30.3 Projektierungsbeispiel

Im Projektierungsbeispiel ist 1 Fast Plc Task mit 2 Fast Actions konfiguriert. Das Beispiel enthält neben dem Aufruf der Funktion MT_FaDataRead zusätzlich auch den Aufruf der entsprechenden Funktion zum Schreiben der Ausgangsdaten:

1. Action: Die NC liefert die Sollpositionen zweier Achsen vom Typ LREAL (2*8 Byte)
2. Action: Die SPS beauftragt die NC, eine CPL-Variable vom Typ DINT zu schreiben (4 Byte)

Der Typ für die Eingangs- und Ausgangsdaten sind in faInp_t bzw. faOutp_t definiert.

Tasklogik: Überschreitet einer der Sollwerte eine Schranke, wird dieser Wert in die Ausgangsdaten (CPL-Variable) der 2. FA zurückgeliefert

```

1  {attribute 'pack_mode' := '1'}
2  TYPE faInp_t :
3  STRUCT
4      sollPos : ARRAY[0..1] OF LREAL ; (* 2 Sollpositionen vom Typ LREAL *)
5  END_STRUCT
6  END_TYPE
7

```

Abb. 12-58: Struktur für die Eingangsdaten

```

1  {attribute 'pack_mode' := '1'}
2  TYPE faOut_t :
3  STRUCT
4      CplVarVal : DINT ; (* Vorgabewert der CPL-Variablen *)
5  END_STRUCT
6  END_TYPE

```

Abb. 12-59: Struktur für die Ausgangsdaten

```

16 PROGRAM FAction
17 VAR
18
19     McResult      : DINT;          (* Ergebnis der NC-Seite *)
20     FaInputData   : faInp_t ;     (* Struktur der Eingangsdaten *)
21     FaOutputData  : faOut_t ;     (* Struktur der Ausgangsdaten *)
22     fbErr         : WORD;         (* Ergebnis des Bausteinaufrufes *)
23     maxVal        : LREAL := 123456.789 ;
24     err: BOOL;
25     greater: BOOL;
26 END_VAR
27

```

Abb. 12-60: Deklarationsteil des Programms

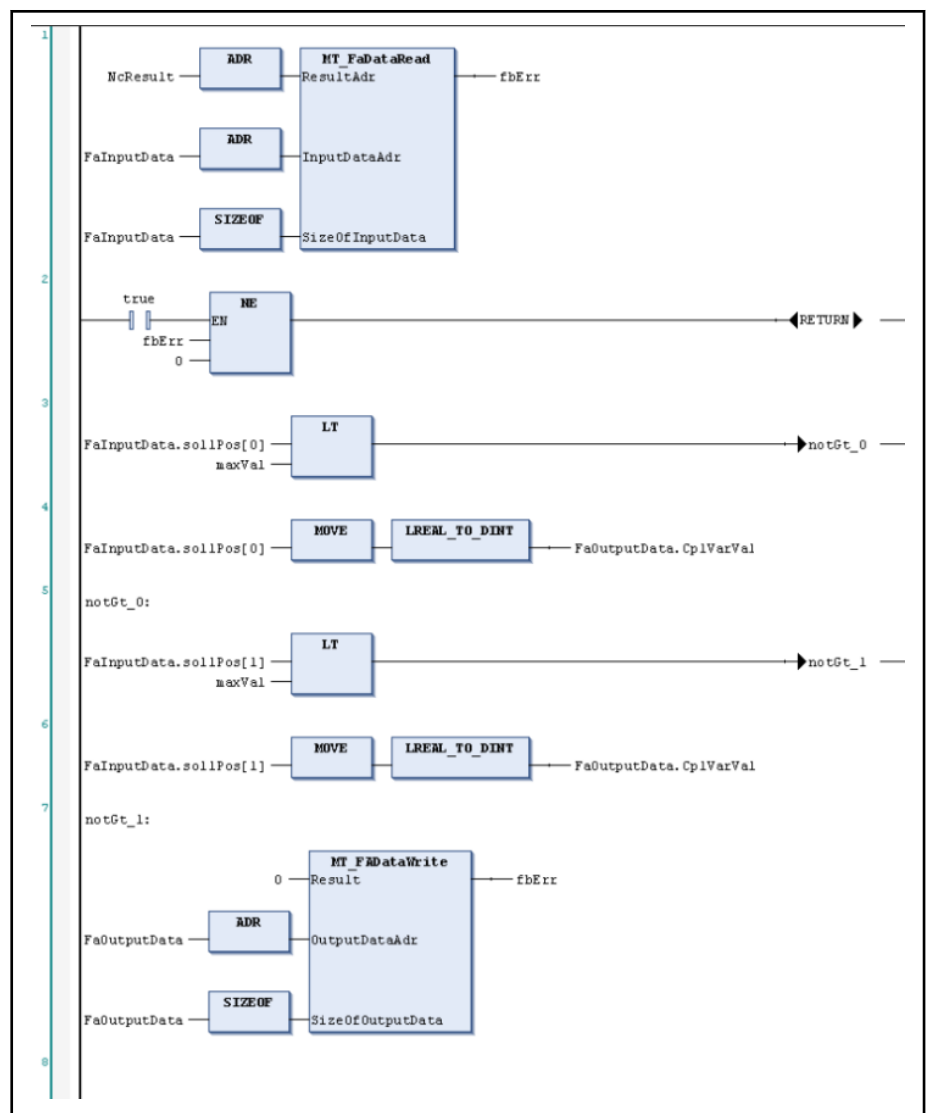


Abb. 12-61: Projektierungsbeispiel für die Funktionen MT_FaDataRead und MT_FaDataWrite

12.30.4 Statusmeldungen

Der Funktionswert kann folgende Werte annehmen:

0x0000:	Datenübertragung OK
0x000A:	Die Funktion wurde nicht ausgeführt, da der Aufruf nicht aus einer Fast Plc Task erfolgte
0x000B:	Konfigurationsfehler: Es sind keine Eingangsdaten für die Task konfiguriert oder die übergebene Größe in SizeOfInputData ist 0
0x000C:	Die Größe der konfigurierten Eingangsdaten stimmt nicht mit der übergebenen Größe in SizeOfInputData überein
0x000D:	Zieladresse in InputDataAdr ist ein Null-Pointer
0x7FFE:	Interner Fehler

12.31 Fast Action Ausgangsdaten schreiben(MT_FaDataWrite)

12.31.1 Allgemeines

Die Funktion dient zum Schreiben der zyklisch übertragenen Daten von der Fast Plc Task zum Interpolator. Am Ende der Fast Plc Task müssen die Ausgangsdaten unter Verwendung dieser Funktion zum Interpolator übertragen werden.

Die Größe der Struktur für die Ausgangsdaten muss exakt dieselbe Größe aufweisen wie die auf NC-Seite konfigurierten Ausgangsdaten für diese Fast Plc Task



Da die Daten auf NC-Seite gepackt übergeben werden, muss bei der Typ-Definition auf SPS-Seite die Struktur ebenfalls mit **{attribute 'pack_mode' := '1'}** gepackt definiert werden.

Eine ausführliche Beschreibung enthält das Handbuch **Funktionsbeschreibung 13VRS Sonderfunktionen** unter dem Kapitel **Schnelle Aktionen MTX fa**



Dieser Baustein darf nur aus einer Fast Plc Task aufgerufen werden (Die Task muss vom Typ **Extern ereignisgesteuert** und das Externe Ereignis **EVENT_FASTACTION_x** sein, wobei x für den Index 1 bis 4 steht)

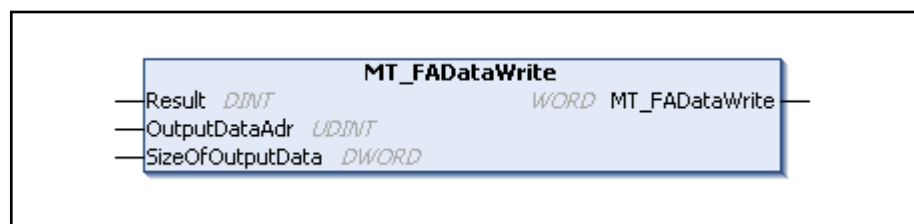


Abb. 12-62: Aufruf Baustein MT_FaDataWrite

12.31.2 Bausteinparameter

Der Baustein wird aufgerufen mit den Parametern:

Funktionswert	Der Funktionswert enthält das Ergebnis der Bibliotheksfunktion
OutputDataAdr	Pointer zu den Daten, die von der Fast Plc Task zum Interpolator übertragen werden
SizeOfOutputData	Gesamtgröße des Datenbereichs der Ausgangsdaten in Bytes
Result	Return-Wert der Fast Plc Task an den Interpolator

12.31.3 Statusmeldungen

Der Funktionswert kann folgende Werte zurückliefern:

0x0000:	Datenübertragung OK
0x000A:	Die Funktion wurde nicht ausgeführt, da der Aufruf nicht aus einer Fast Plc Task erfolgte
0x000B:	Konfigurationsfehler: Es sind keine Eingangsdaten für die Task konfiguriert oder die übergebene Größe in SizeOfInputData ist 0

- 0x000C:** Die Größe der konfigurierten Eingangsdaten stimmt nicht mit der übergebenen Größe in **SizeOfInputData** überein
- 0x000D:** Zieladresse in **InputDataAdr** ist ein Null-Pointer
- 0x7FFE:** Interner Fehler

12.31.4 Projektierungsbeispiel

Im Projektierungsbeispiel ist 1 Fast Plc Task mit 2 Fast Actions konfiguriert. Das Beispiel enthält neben dem Aufruf der Funktion MT_FaDataWrite zusätzlich auch den Aufruf der entsprechenden Funktion zum Lesen der Eingangsdaten:

1. Action: Die NC liefert die Sollpositionen zweier Achsen vom Typ LREAL (2*8 Byte)
2. Action: Die SPS beauftragt die NC, eine CPL-Variable vom Typ DINT zu schreiben (4 Byte)

Der Typ für die Eingangs- und Ausgangsdaten sind in **faInp_t** bzw. **faOutp_t** definiert.

Tasklogik: Überschreitet einer der Sollwerte eine Schranke, wird dieser Wert in die CPL-Variable der 2. FA zurückgeliefert

```

1  {attribute 'pack_mode' := '1'}
2  TYPE faInp_t :
3  STRUCT
4      sollPos : ARRAY[0..1] OF LREAL ; (* 2 Sollpositionen vom Typ LREAL *)
5  END_STRUCT
6  END_TYPE
7

```

Abb. 12-63: Struktur für die Eingangsdaten

```

1  {attribute 'pack_mode' := '1'}
2  TYPE faOut_t :
3  STRUCT
4      CplVarVal : DINT ; (* Vorgabewert der CPL-Variablen *)
5  END_STRUCT
6  END_TYPE
7

```

Abb. 12-64: Struktur für die Ausgangsdaten

```

16 PROGRAM FAction
17 VAR
18
19     McResult      : DINT;           (* Ergebnis der NC-Seite *)
20     FaInputData   : faInp_t ;      (* Struktur der Eingangsdaten *)
21     FaOutputData  : faOut_t ;      (* Struktur der Ausgangsdaten *)
22     fbErr         : WORD;          (* Ergebnis des Bausteinaufrufes *)
23     maxVal        : LREAL := 123456.789 ;
24     err: BOOL;
25     greater: BOOL;
26 END_VAR
27

```

Abb. 12-65: Deklarationsteil des Programms

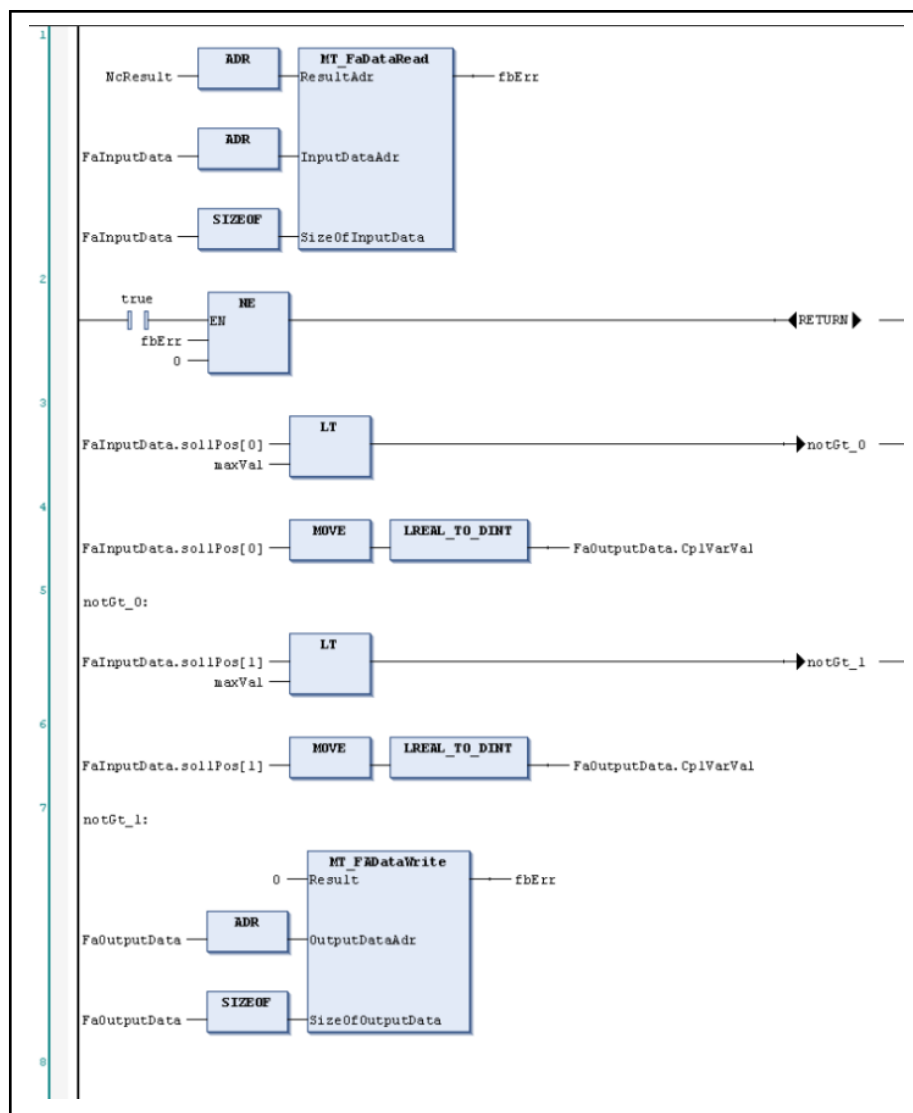


Abb. 12-66: Projektierungsbeispiel für die Funktionen **MT_FaDataRead** und **MT_FaDataWrite**

13 Konfiguration von SPS-spezifischen Daten in IndraWorks

13.1 Konfiguration der Maschinenzustandsanzeige (MZA)

13.1.1 Allgemeines

Die Maschinenzustandsanzeige ermöglicht die Diagnose von Maschinenfehlern, -warnungen und -hinweisen und unterstützt somit die schnelle Beseitigung von Störungen des Betriebsablaufs an der Maschine.

Die MZA-Diagnosen werden in der MTX-HMI im Diagnose-Header und im OP9 "Diagnose" angezeigt. MZA-Diagnosen können im Logbook mit protokolliert werden.

Die entsprechenden Meldungen werden im Klartext in der MZA-Datei "mza-texte.xxx" definiert und durch Setzen des betreffenden Signals am MZA-Bit-Interface im SPS-Programm aktiviert.

13.1.2 Parametrierung der Maschinenzustandsanzeige (MZA)

Der Austausch der MZA-Signale erfolgt zyklisch zwischen SPS und NC. Das Zeitintervall stellen Sie im Parameter "MachStatDiagCycTime" ein. Es sollte z. B. 300ms betragen, um die Kommunikationslast des Systems gering zu halten.

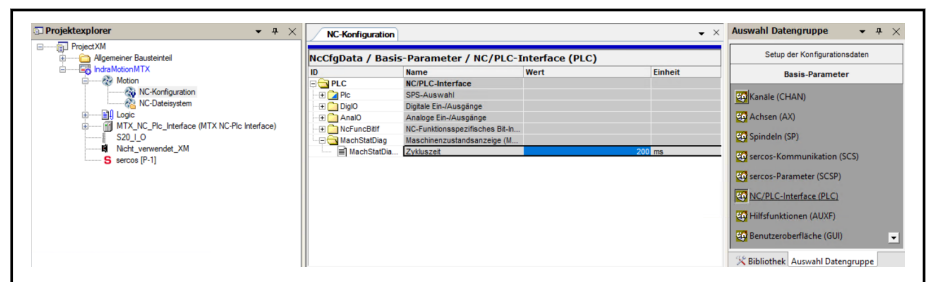


Abb. 13-1: MZA-Zykluszeit



Die Zykluszeit "0" schaltet die MZA-Diagnose aus.

13.1.3 Konfiguration des MZA-Bit-Interface

Je nach Anzahl der Maschinenfehler, -warnungen und -hinweise, können Sie die folgenden Bit-Interface-Datentypen mit unterschiedlicher Anzahl von Signalen verwenden:

- "qMZA" mit bis zu 2048 Signalen oder
- "qMZAext" mit bis zu 8192 Signale.

Konfiguration von SPS-spezifischen Daten in IndraWorks

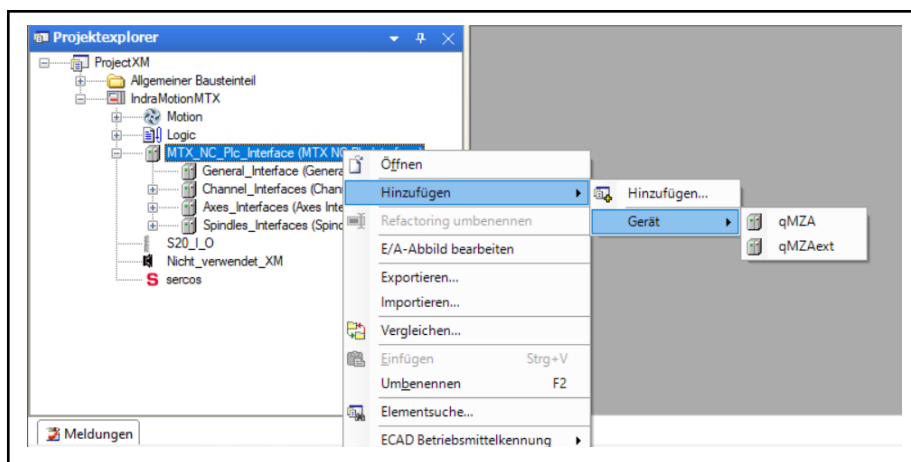


Abb. 13-2: Einfügen der MZA-Interfacesignale

Im MZA-Eigenschaften-Dialog (siehe nächste Abbildung) können die folgenden Einstellungen vorgenommen werden:

- Der Bezeichner für das MZA-Bit-Interface kann frei gewählt werden. Dazu muss der Bezeichner des MZA-Knotens unterhalb des Knotens "MTX_NC_Plc_Interface" mit der rechten Maustaste angeklickt werden. Im Kontextmenü kann jetzt der Befehl "Umbenennen" ausgewählt werden. Es empfiehlt sich den vorgeschlagenen Bezeichner "qMZA" bzw. "qMZAext" zu übernehmen.
- Die Speicheradresse, ab der der gewählte Datentyp im Bit-Interface beginnt, kann frei gewählt werden, sofern es sich nicht mit anderen Adressen im Bit-Interface überschneidet.

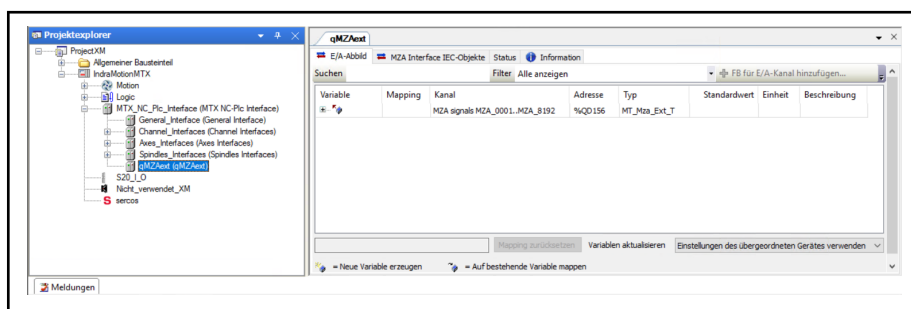


Abb. 13-3: MZA-Eigenschaften-Dialog

13.1.4 Aufbau der MZA-Datei

Die MZA-Datei muss unter dem Namen **"mzatexte.xxx"** im Wurzelverzeichnis "/" oder im Benutzer-FEPROM "/usrfep" der Steuerung abgelegt werden. Die Dateinamenserweiterung "xxx" steht für die verwendete Sprache, z. B. 049 für Deutsch.

Der MZA-Text unterteilt sich in einen Haupt-, Hilfs- und Behebungstext. Die Trennung erfolgt mit einem doppelten Nummernzeichen "##". Lange Texte können mit dem Zeichen "_" am Zeilenende in der nächsten Zeile fortgesetzt werden.

Beispiel:

MZA-Text

mzatexte.049

...

(E123, NOT-AUS Taster an Station ist betätigt! ## _

Der NOT-AUS-Taster an der Station ist noch betätigt! ## _
 Erst wenn diese NOT-AUS-Bedingung beseitigt ist, _
 darf die Leistung eingeschaltet werden!)
 (M124, Kühlmittel fehlt! ## Der Kühlmittelstand hat die Mindestmarke _
 unterschritten! ## _
 Füllen Sie gegebenenfalls Kühlmittel nach.)

...



Beenden Sie jede MZA-Definition - auch die letzte am Dateiende - nach der schließenden Klammer mit dem Zeilenvorschub, da sonst die Meldungstexte nicht wirksam werden.

Die Syntax der MZA-Definition in der Datei "mzatexte.xxx" stellt sich wie folgt dar:

(<Diagnoseklasse><Mza-Nummer>,<Haupttext>##<Hilfstext>##<Behebungstext>)

<Diagnoseklasse>	• E für Maschinenfehler • W für Maschinenwarnung • M für Maschinenhinweis
<Mza-Nummer>	Bit in der Struktur qMZA [1..2048] bzw. qMZA_Ext [1..8192] und gleichzeitig Meldungsnummer in der Diagnose.
<Haupttext>	Meist einzelzeiliger Meldungstext, der auch in der Statuszeile eingeblendet wird.
<Hilfstext>	Hilfstext zur näheren Beschreibung der Diagnose.
<Behebungstext>	Beschreibung zur Behebung der aufgetretenen Störung.

Die Texte die nicht im ASCII-Zeichensatz darstellbar sind, dürfen in der UTF8-Zeichenkodierung angegeben werden. Eine MZA-Datei im UTF8-Format muss am Dateianfang die hexadezimale 3-Byte-Dateikennung "EFBBBF" besitzen. Die Texte werden mit einer maximalen Länge von bis zu 1000 Zeichen übertragen. Bei Nutzung des UTF8-Format verringert sich die Textlänge entsprechend des Codierungsaufwands um bis zu 2/3 der darstellbaren Zeichen.

13.1.5 Aktivierung der Meldungen

Die MZA-Meldungen werden in der SPS durch Setzen der gewünschten Bit-Signale aktiviert.

Beispiel:

Strukturierter Text

```
qMZAext.MZA_0004:=TRUE;
```

Das Rücksetzen der MZA-Meldungen muss ebenfalls in der SPS programmiert werden.

Beispiel:

Strukturierter Text

qMZAext.MZA_0004:=FALSE;

13.1.6 Ergänzende Hinweise

Die steuerungsinterne Fehlerdatenbank kann nicht unbegrenzt Fehler aufnehmen.

*Aus dem Bereich MZA können **maximal** folgende Meldungen aufgenommen werden:*

- 16 Fehler
- 16 Warnungen und
- 16 Hinweise

Stehen mehr Meldungen an (d. h., beispielsweise mehr als 16 Maschinenfehler), werden nur die höchstprioritären Fehler in der internen Fehlerdatenbank gespeichert. Die Priorität ist durch die Meldungsnummer vorgegeben. Kleine Nummer bedeutet hohe Priorität.

Werden Meldungen gelöscht, kommt es vor, dass niederprioritäre Meldungen "nachrücken". Zu welchem Zeitpunkt die entsprechende Meldung gekommen ist, ist zum Zeitpunkt des "Nachrückens" allerdings unbekannt, da der Zeitstempel nicht gespeichert wird.

"Nachrückende" Meldungen bekommen daher den Zeitstempel 1.1.1970 00:00:00.000.

*Zum "Nachrücken" kommt es nur, wenn **mehr als***

- 16 Maschinenfehler gleichzeitig,
- 16 Maschinenwarnungen gleichzeitig oder
- 16 Maschinenhinweise gleichzeitig anstehen.

Den Bereich der überwachten MZA-Signale gibt die Datei "mzatexte.xxx" vor. Dementsprechend werden nur die SPS-MZA-Signale zwischen qMZA(ext).MZA_0001 und qMZA(_Ext).MZA_nnnn ausgewertet. "nnnn" ergibt sich aus der höchsten eingetragenen Fehler- (Ennnn,...), Warnungs- (Wnnnn,...) oder Hinweisnummer (Mnnnn,...) in der Datei "mzatexte.xxx". Signale außerhalb dieses Bereichs werden ignoriert. Dadurch kann für nicht benötigte Signale Übertragungszeit eingespart werden. Signale innerhalb des Bereichs ohne eine Textentsprechung erscheinen automatisch als Maschinenwarnung (<Nr.>).

14 MT_BaseAdditional.library

14.1 Einführung und Übersicht

Die Funktionen und Funktionsbausteine dieser Bibliothek ergänzen und vereinfachen die Basisfunktionalität, welche mit der Bibliothek MT_Base bereitgestellt wird.

Übersicht Kanalfunktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung
Allgemein	MT_ChNumberValidation, Seite 246	Prüft, ob die angegebene Kanalnummer in der NC-Konfiguration vorhanden ist
	MT_ChGetState, Seite 248	Aktuellen Kanalstatus ermitteln
Vorschub	MT_ChSetFeedOverride, Seite 261	Kanalvorschub ändern
	MT_GetFeedOverride, Seite 262	Aktiven Kanalvorschub ermitteln
NC-Programm	MT_ChSetReentryBreak-Point, Seite 258	NC-Programm Unterbrechungspunkt speichern
	MT_ChGetActiveNcProgram, Seite 250	Name des aktiven NC-Programms ermitteln
	MT_ChNcProgProgress, Seite 14.3.5	Aktuellen NC-Programmfortschritt ermitteln
Betriebsart	MT_ChOPMode, Seite 275	Ermitteln und Ändern der Betriebsarten eines Kanals (kombiniert MT_ChGetOPMode und MT_ChSetOPMode)
	MT_ChGetOPMode, Seite 263	Ermittelt die aktive Haupt- und Unterbetriebsart eines Kanals
	MT_ChSetOPMode, Seite 270	Ändert die Haupt- und Unterbetriebsarten eines Kanal
	MT_ChSetMainOPMode, Seite 266	Ändert die Hauptbetriebsarten eines Kanal

Tab. 14-1: Übersicht Kanalfunktionen

Übersicht Achsfunktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung
Allgemein	MT_DrvGetSafeState, Seite 281	Aktiven Sicherheitstechnik-Betriebszustand eines Antriebs ermitteln (auch für Spindel)
	MT_AxGetChIndex, Seite 285	Kanalzuordnung einer Achse prüfen
	MT_AxHandwheelACS, Seite 286	Handrad betreiben für eine Achse im Achskoordinatensystem
	MT_AxProbeMonitor, Seite 288	Überwachung einer Messtasterachse

Asynchrone Achsen	MT_AxAsyncSetOverride, Seite 308	Vorschub für asynchrone Achse ändern
	MT_AxAsyncOPMode, Seite 305	Ermittelt und Ändert die Betriebsart einer asynchronen Achse (kombiniert MT_AxAsyncGetOpMode und MT_AxAsyncSetOpMode)
	MT_AxAsyncGetOPMode, Seite 301	Aktive Betriebsart einer asynchronen Achse ermitteln
	MT_AxAsyncSetOPMode, Seite 302	Betriebsart einer asynchronen Achse ändern
Tippdistanz	MT_AxJogMode, Seite 296	Ermitteln und Ändern der aktiven Tippdistanz einer Achse (kombiniert MT_AxGetJogMode und MT_AxSetJogMode)
	MT_AxGetJogMode, Seite 291	Ermitteln der aktiven Tippdistanz für eine Achse
	MT_AxSetJogMode, Seite 292	Ändern der aktiven Tippdistanz für eine Achse
Leistung & Diagnose	MT_AxPower, Seite 316	Ermitteln des Antriebsstatus und Setzen/Rücksetzen der Reglerfreigabe (kombiniert MT_AxSetPowerOn, MT_AxSetPowerOff und MT_AxGetPowerStateExt)
	MT_AxSetPowerOn, Seite 312	Setzt die Reglerfreigabe für eine Achse
	MT_AxSetPowerOff, Seite 314	Rücksetzen der Reglerfreigabe für eine Achse
	MT_AxGetPowerState, Seite 309	Ermittelt den Antriebsstatus für eine Achse
	MT_AxGetPowerStateExt, Seite 311	Ermittelt den Antriebsstatus einer Achse inklusive Detailinformationen zu Fehler und Warnungen

Tab. 14-2: Übersicht Achsfunktionen

Übersicht Spindelfunktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung
Allgemein	MT_SpSetOverride, Seite 320	Vorschub für eine Spindel ändern
Tippgeschwindigkeit	MT_SpJogSpeed, Seite 325	Ermitteln und Ändern der aktiven Tippgeschwindigkeit einer Spindel (kombiniert MT_SpGetJogSpeed und MT_SpSetJogSpeed)
	MT_SpGetJogSpeed, Seite 321	Ermitteln der aktuellen Tippgeschwindigkeit für eine Spindel
	MT_SpSetJogSpeed, Seite 322	Ändern der aktiven Tippgeschwindigkeit für eine Spindel

Leistung & Diagnose	MT_SpPower, Seite 336	Ermitteln des Antriebsstatus und Setzen/Rücksetzen der Reglerfreigabe (kombiniert MT_SpSetPowerOn, MT_SpSetPowerOff und MT_SpGetPowerStateExt)
	MT_SpSetPowerOn, Seite 332	Setzt die Reglerfreigabe für eine Spindel
	MT_SpSetPowerOff, Seite 334	Rücksetzen der Reglerfreigabe für eine Spindel
	MT_SpGetPowerState, Seite 328	Ermittelt den Antriebsstatus für eine Spindel
	MT_SpGetPowerStateExt, Seite 330	Ermittelt den Antriebsstatus einer Spindel inklusive Detailinformationen zu Fehler und Warnungen

Tab. 14-3: Übersicht Spindelfunktionen

Übersicht Werkzeug-Management-Funktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung
Standzeit	MT_TmToolLife, Seite 339	Ermitteln der Standzeit für Werkzeuge

Tab. 14-4: Übersicht Werkzeug-Management-Funktionen

Übersicht Sonstige Funktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung
Sonstige	MT_DistanceAccumulator, Seite 346	Berechnet die Wegstrecke aus Positionsdaten
	MT_TimeAccumulator, Seite 349	Berechnet die verstrichene Zeit
	MT_GetRandomNumber, Seite 351	Generiert eine ganzzahlige Zufallszahl

Tab. 14-5: Übersicht Sonstige Funktionen

14.2 Standardschnittstellen an Funktionsbausteinen

14.2.1 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.

14.2.2 Bausteintypen

Mit Funktionsbausteinen ist es möglich, komplexe Aufgaben wiederverwendbar zu kapseln und diese Aufgaben über definierte Schnittstellen anzusprechen. Dabei kann die Bearbeitung entweder zustands- oder flankengesteuert erfolgen. Zudem unterscheidet man zwischen Aufträgen, die abschließend behandelt werden können und solchen die einmal angestoßen einen permanent Eingriff erfordern.

Zustandsgesteuert Wiederholt ein Funktionsbaustein nach dem Einschalten seinen Auftrag immer dann, wenn der Funktionsbaustein einen definierten Zustand erreicht hat, arbeitet der Funktionsbaustein "zustandsgesteuert".

Flankengesteuert Erfüllt ein Funktionsbaustein nach dem Einschalten seine Aufgabe genau einmal, so arbeitet der Funktionsbaustein "flankengesteuert".

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

Permanente Bearbeitung Kann ein Baustein seinen Auftrag nie endgültig abschließen, sondern ist dauerhaft im Eingriff, wird dies als "permanente Bearbeitung" definiert.

In Kombination ergeben sich somit vier mögliche Bausteintypen:

Steuerung	Bearbeitung	Beispiel
Zustandsgesteuert (Enable)	Abschließend (Done)	Dauerhaftes Lesen eines Parameters Ist das Lesen abgeschlossen, hat der Baustein seinen Auftrag beendet. Der Baustein wiederholt diesen Auftrag solange, wie der Steuereingang (Enable) gesetzt bleibt, z. B. MB_ReadParameter
Zustandsgesteuert (Enable)	Permanent (In....)	Beendbare Regelung Ist die Regelung aktiv, arbeitet die Regelung solange, bis der Steuereingang (Enable) zurückgesetzt wird und die Regelung damit deaktiviert
Flankengesteuert (Execute)	Abschließend (Done)	Einmaliges Schreiben eines Parameters Ist das Schreiben abgeschlossen, hat der Baustein seinen Auftrag beendet. Es ist eine erneute Flanke am Steuereingang (Execute) notwendig, um ein erneutes Schreiben zu starten, z. B. MB_WriteParameter
Flankengesteuert (Execute)	Permanent (In....)	Nicht beendbare Regelung Ist die Regelung einmal über den Steuereingang (Execute) gestartet, kann die Regelung nicht mehr über die Schnittstellen dieses Bausteines deaktiviert werden, z. B. MC_MoveVelocity

Tab. 14-6: Übersicht Bausteintypen

14.2.3 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 Funktionsbaustein verwendet.

- Enable = zustandsgesteuert
- Execute = flankengesteuert

Ausgänge Um nach außen kenntlich zu machen, ob ein Funktionsbaustein abschließend oder permanent arbeitet, werden verschiedene Variablennamen für die Ausgänge verwendet, welche den Bearbeitungszustand kennzeichnen.



- Done = Abschließende Bearbeitung
- In.... = Permanente Bearbeitung (z.B. InOperation, InSync, InVelocity, InGear, InTorque...)

Der optionale Ausgang Active kann bei abschließend und permanent arbeitenden Bausteinen verwendet werden, hat aber jeweils ein anderes Verhalten!

Alle Ausgänge zur Fehlerkennzeichnung (Error, ErrorID, ErrorIdent) finden sich bei abschließend und permanent arbeitenden Bausteinen und haben bei beiden das gleiche Verhalten.

Standardisierte Bezeichner

Bezeichner	Beschreibung
Eingänge	
Enable	Aktivierungseingang zustandsgesteuerter FBs
Execute	Aktivierungseingang flankengesteuerter FBs. Andere FBs können diesen FB unterbrechen
ExecuteLock	Aktivierungseingang flankengesteuerter FBs. Solange dieser Eingang TRUE ist, kann der FB nicht durch andere unterbrochen werden
Ausgänge	
Active	Ausgang, der den Bearbeitungszeitraum kennzeichnet
Done	Bearbeitung erfolgreich beendet und die Datenausgänge sind gültig
In.....	Ausgang signalisiert, dass der Baustein an seiner Aufgabe arbeitet und die Datenausgänge gültig sind
Shutdown (optional)	Ausgang signalisiert, dass sich der Baustein gerade in einen definierten Endzustand bringt (z. B. Achsen stoppt, Ressourcen freigibt, ...). Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird
CommandAborted	Ausgang zeigt an, dass der FB unterbrochen wurde (z. B. von einem anderen FB)
Error	Bearbeitung mit Fehler beendet
ErrorID	Ausgang für grobe Fehlerklassifikation
ErrorIdent	Ausgang für detaillierte Fehlerklassifikation

Tab. 14-7: Übersicht standardisierter Bezeichner

14.2.4 Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung

Steuerung / Bearbeitung	I/O	Variablenname	Beschreibung
Zustandsge- steuert / ab- schließend	I	Enable	Mit einer positiven Flanke von "Enable" werden die Eingangsvariablen erfasst. Neue Eingangswerte werden erst über eine erneute positive Flanke von "Enable" wirksam. "Enable" muss während der gesamten Bausteinbearbeitung TRUE sein! Das Deaktivieren von "Enable" unterbricht die Bearbeitung und setzt die Ausgänge "Done", "Active", "CommandAborted" und "Error" auf FALSE. Falls notwendig wird über "Shutdown" signalisiert, dass weitere Aufrufe notwendig sind, um den Baustein in einen definierten Endzustand zu bringen
	O	Done	Ist "Done" TRUE, hat der Funktionsbaustein seinen Auftrag erfolgreich beendet und ist in einem Endzustand. Datenausgänge sind jetzt gültig. "Active", "Error" und "CommandAborted" sind FALSE! "Done" bleibt für genau einen SPS-Zyklus TRUE. Ist anschließend "Enable" noch TRUE, startet der Baustein seine Bearbeitung erneut
	O	Active	Ist "Active" TRUE, arbeitet der Funktionsbaustein an seiner eigentlichen Aufgabe und ist in einem Zwischenzustand. Eventuelle Vorverarbeitungen werden nicht mit diesem Ausgang gekennzeichnet! "Done", "Error" und "CommandAborted" sind FALSE!
	O	Shutdown (optional)	Ist ein "Shutdown"-Ausgang vorhanden, benötigt der Baustein beim Deaktivieren von "Enable" noch mehrere Zyklen an Rechenzeit, um sich in einen definierten Endzustand zu bringen und verwendete Ressourcen freizugeben. Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird!
	O	CommandAborted	Ist "CommandAborted" TRUE, wurde der Funktionsbaustein unterbrochen und befindet sich in einem Endzustand. "Done", "Active" und "Error" sind FALSE! "CommandAborted" bleibt solange TRUE, bis der Steuereingang "Enable" deaktiviert wird. Ein zusätzlicher Reset-Eingang ist daher nicht notwendig. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "CommandAborted" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Error	Ist "Error" TRUE, wurde der Funktionsbaustein wegen eines Fehlers abgebrochen und befindet sich in einem Endzustand. "Error" bleibt solange TRUE, bis der Steuereingang "Enable" deaktiviert wird. Ein zusätzlicher Reset-Eingang ist daher nicht notwendig. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "Error" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird

Tab. 14-8: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung



Ist es notwendig, dass bestimmte Eingänge nicht nur mit der Flanke an "Enable" übernommen werden, sondern zyklisch während der Bausteinbearbeitung, so ist dies explizit zu dokumentieren und nicht über Variablennamen kenntlich zu machen!



Bei Bausteinen mit "Shutdown" muss genau definiert werden, wie sich der Baustein verhält, wenn "Enable" wieder auf TRUE gesetzt wird, solange "Shutdown" noch gesetzt ist (Retriggern).

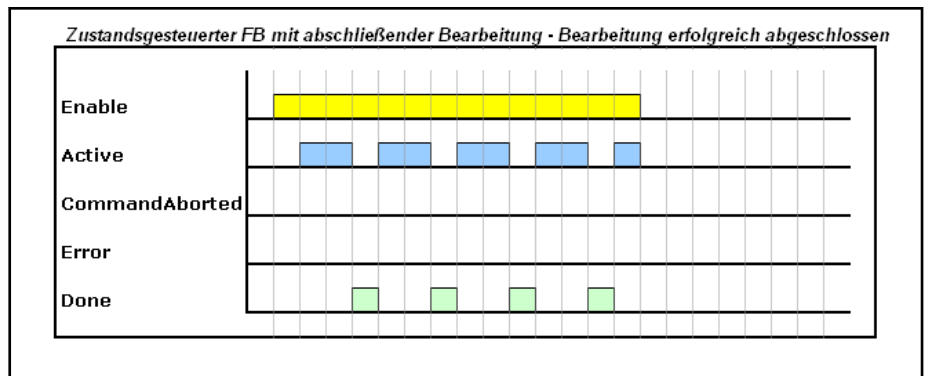


Abb. 14-1: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung erfolgreich abgeschlossen

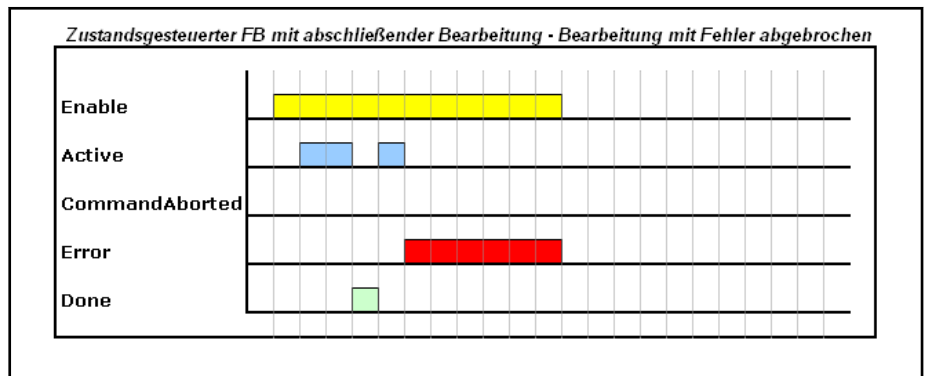


Abb. 14-2: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung mit Fehler abgeschlossen

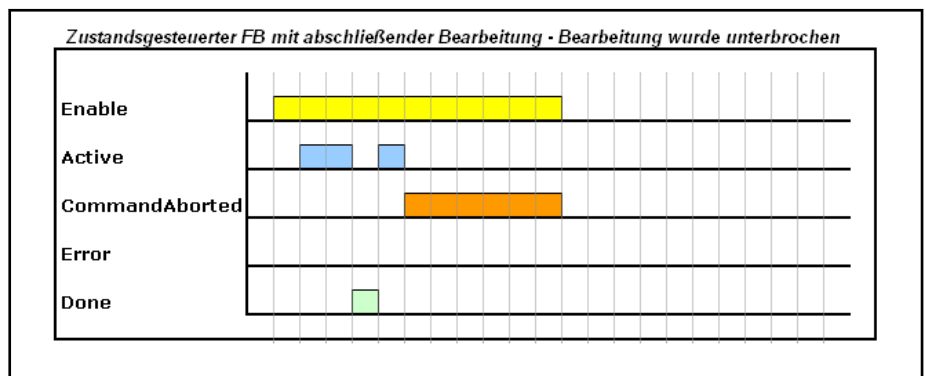


Abb. 14-3: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung unterbrochen

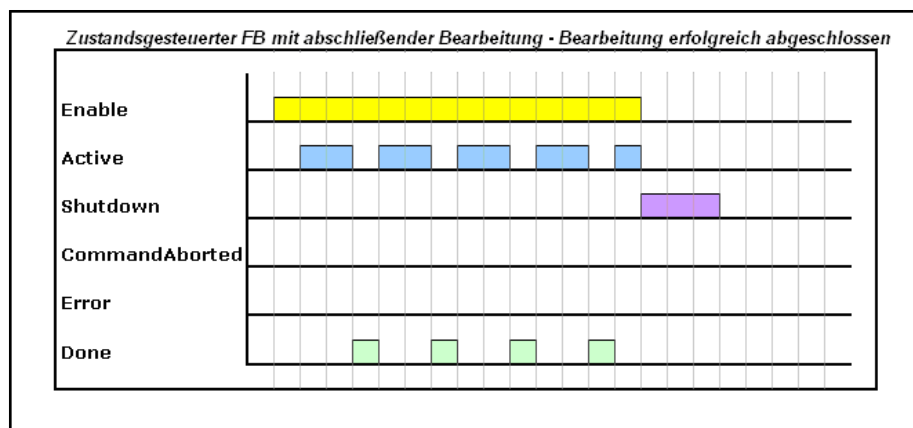


Abb. 14-4: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung erfolgreich abgeschlossen

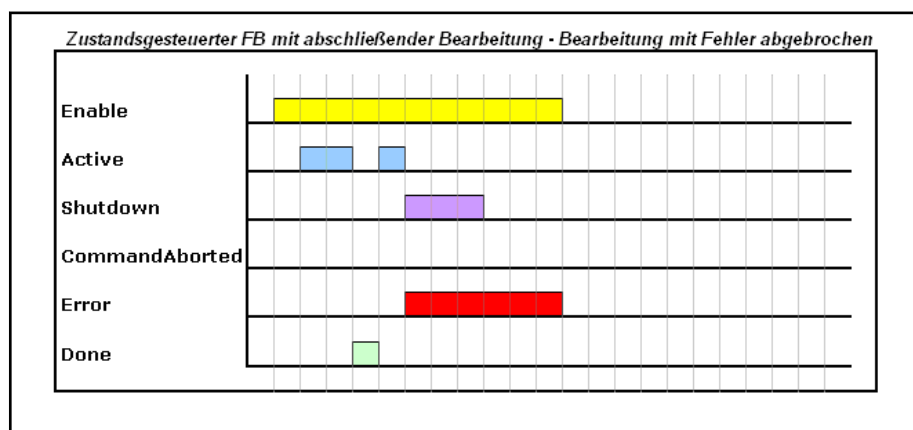


Abb. 14-5: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung mit Fehler abgeschlossen

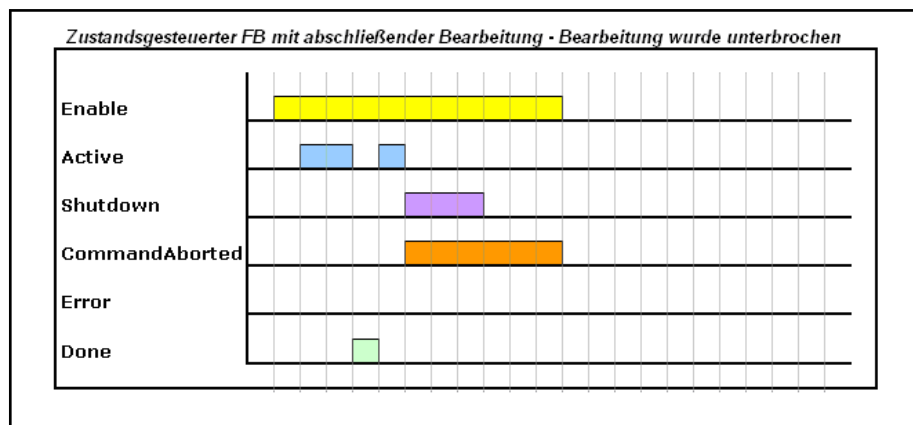


Abb. 14-6: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung wurde unterbrochen

14.2.5 Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung

Steuerung / Bearbeitung	I/O	Name	Beschreibung
Flankengesteuert / abschließend	I	Execute	Mit einer positiven Flanke von "Execute" werden die Eingangsvariablen erfasst. Neue Eingangswerte werden erst über eine erneute positive Flanke von "Execute" wirksam. Der Flankenwechsel an "Execute" reicht aus, um den Baustein zu starten. Zur weiteren Bearbeitung ist der Zustand von "Execute" nicht relevant. Durch einen erneuten Flankenwechsel noch während der Bearbeitung wird der alte Auftrag verworfen, die Eingänge erneut übernommen und mit den neuen Werten der Auftrag fortgesetzt (Retriggerung, Nachtriggern). Ist ein "Shutdown"-Ausgang vorhanden, wird Retriggering möglicherweise nicht unterstützt. Die Bausteindokumentation schafft hierüber Klarheit
	O	Done	Ist "Done" TRUE, hat der Funktionsbaustein seinen Auftrag erfolgreich beendet und ist in einem Endzustand. Datenausgänge sind jetzt gültig. "Active", "Error" und "CommandAborted" sind FALSE! Ist bei Auftragsabschluss der Steuerungseingang "Execute" FALSE, bleibt "Done" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "Done" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden wird dieser Ausgang gleichzeitig mit "Done" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Active	Ist "Active" TRUE, arbeitet der Funktionsbaustein an seiner eigentlichen Aufgabe und ist in einem Zwischenzustand. Eventuelle Vorverarbeitungen werden nicht mit diesem Ausgang gekennzeichnet! "Done", "Error", "Shutdown" und "CommandAborted" sind FALSE!
	O	Shutdown (optional)	Ist ein "Shutdown"-Ausgang vorhanden, benötigt der Baustein nach Beendigung der Aufgabe (signalisiert durch "Done") noch mehrere Zyklen an Rechenzeit, um sich in einen definierten Endzustand zu bringen und verwendete Ressourcen freizugeben. Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird!
	O	CommandAborted	Ist "CommandAborted" TRUE, wurde der Funktionsbaustein unterbrochen und befindet sich in einem Endzustand. "Done", "Active" und "Error" sind FALSE! Ist zum Unterbrechungszeitpunkt der Steuerungseingang "Execute" FALSE, bleibt "CommandAborted" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "CommandAborted" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "CommandAborted" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Error	Ist "Error" TRUE, wurde der Funktionsbaustein wegen eines Fehlers abgebrochen und befindet sich in einem Endzustand. Ist bei Auftreten des Fehlers der Steuerungseingang "Execute" FALSE, bleibt "Error" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "Error" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "Error" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird

Tab. 14-9: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung



Werden bestimmte Eingänge nicht nur mit der 0/1-Flanke an "Execute" übernommen, sondern zyklisch während der Bausteinbearbeitung, so ist dies explizit zu dokumentieren und nicht über Variablennamen kenntlich zu machen!



Bei Bausteinen mit "Shutdown" muss genau definiert werden, wie sich der Baustein verhält, wenn an "Execute" eine erneute Flanke erkannt wird, solange "Shutdown" noch gesetzt ist (Retriggern).

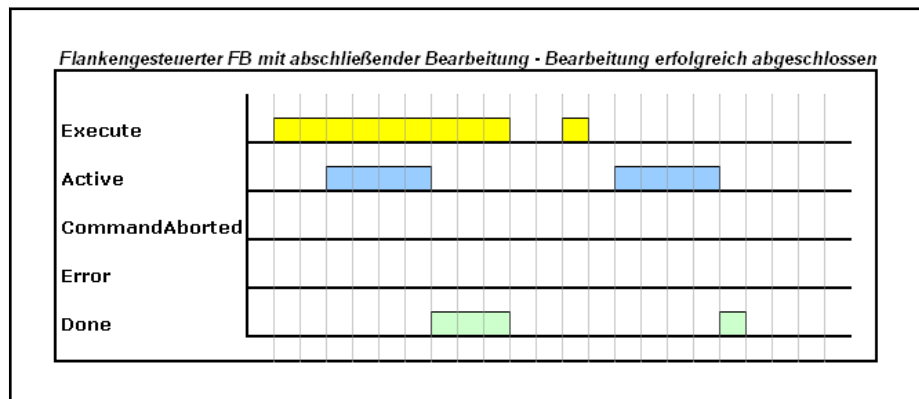


Abb. 14-7: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung erfolgreich abgeschlossen

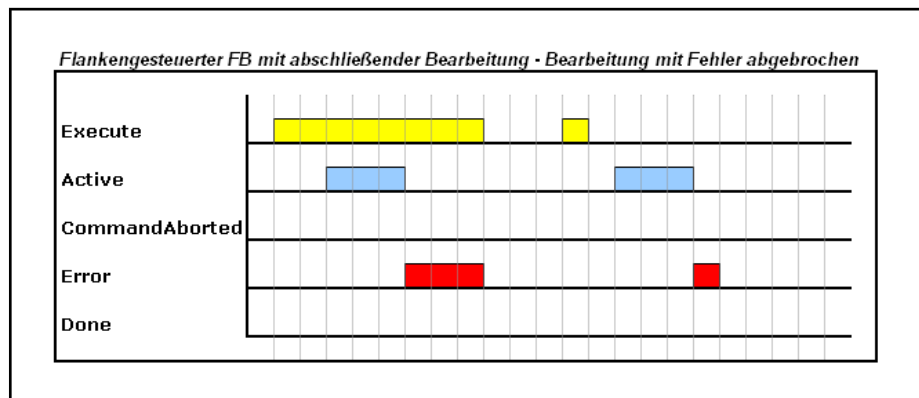


Abb. 14-8: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung mit Fehler abgeschlossen

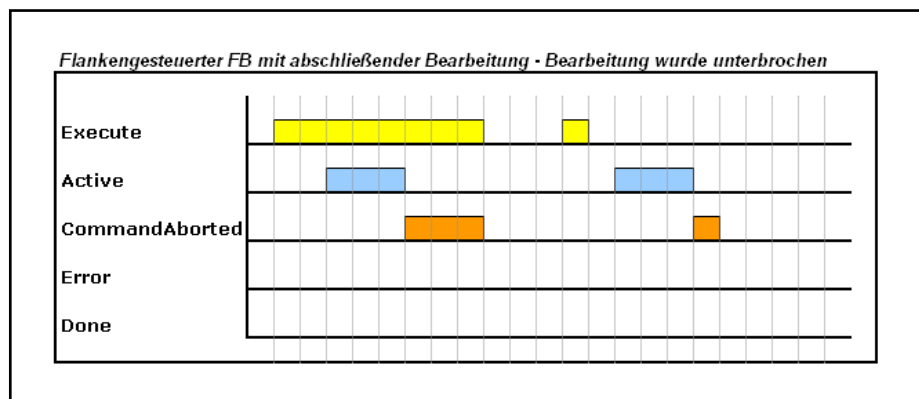


Abb. 14-9: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung ohne "Shutdown". Bearbeitung unterbrochen

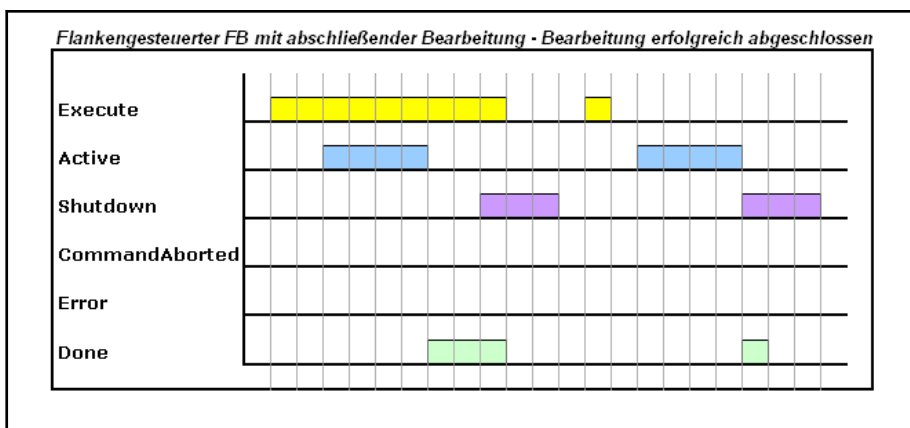


Abb. 14-10: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung erfolgreich abgeschlossen

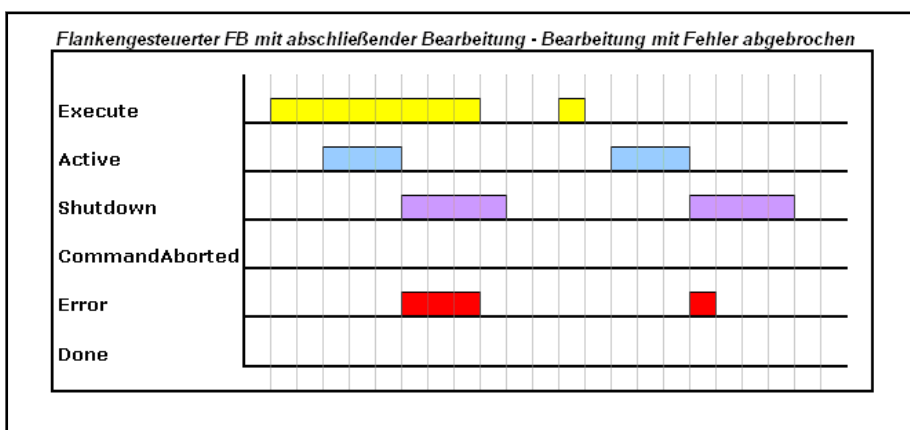


Abb. 14-11: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung mit Fehler abgeschlossen

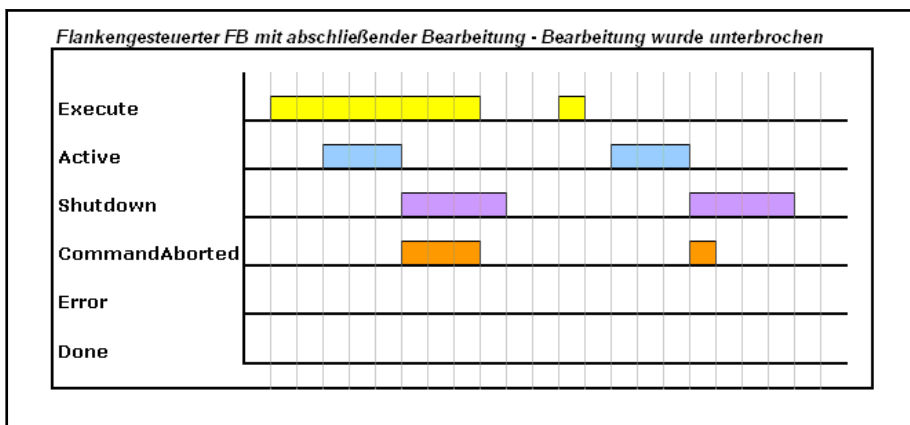


Abb. 14-12: I/O flankengesteuerter Funktionsbausteine, abschließende Bearbeitung mit "Shutdown". Bearbeitung unterbrochen

14.2.6 Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung

Steuerung / Bearbeitung	I/O	Variablenname	Beschreibung
Zustandsge- steuert / per- manent	I	Enable	Mit einer positiven Flanke von "Enable" werden die Eingangsvariablen erfasst. Neue Eingangswerte werden erst über eine erneute positive Flanke von "Enable" wirksam. "Enable" muss während der gesamten Bausteinbearbeitung TRUE sein! Das Deaktivieren von "Enable" unterbricht die Bearbeitung und setzt die Ausgänge "In....", "Active", "CommandAborted" und "Error" auf FALSE. Falls notwendig wird über "Shutdown" signalisiert, dass weitere Aufrufe notwendig sind, um den Baustein in einen definierten Endzustand zu bringen
	O	In....	Ist "In...." TRUE, hat der Funktionsbaustein sein Ziel erreicht, befindet sich aber weiter im Eingriff, um das Erreichte zu "bewahren" und befindet sich damit in einem dauerhaften Endzustand. Datenausgänge sind jetzt gültig. Da der FB weiter im Eingriff bleibt, muss auch "Active" TRUE bleiben. "Error" und "CommandAborted" sind FALSE! Solange "Enable" TRUE ist, bleibt "In...." ebenfalls TRUE
	O	Active	Ist Active TRUE, arbeitet der Funktionsbaustein an seiner eigentlichen Aufgabe. Eventuelle Vorverarbeitungen werden nicht mit diesem Ausgang gekennzeichnet! Da der FB permanent im Eingriff bleibt, bleibt "Active" TRUE bis der FB über den Steuereingang "Enable" abgeschaltet wird oder durch "Error" oder "CommandAborted" beendet wird. Solange "Active" TRUE ist, müssen "Error" oder "CommandAborted" FALSE sein!
	O	Shutdown (optional)	Ist ein "Shutdown"-Ausgang vorhanden, benötigt der Baustein beim Deaktivieren von "Enable" noch mehrere Zyklen an Rechenzeit, um sich in einen definierten Endzustand zu bringen und verwendete Ressourcen freizugeben. Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird!
	O	CommandAborted	Ist "CommandAborted" TRUE, wurde der Funktionsbaustein unterbrochen und befindet sich in einem Endzustand. "In....", "Active" und "Error" sind FALSE! "CommandAborted" bleibt solange TRUE, bis der Steuereingang "Enable" deaktiviert wird. Ein zusätzlicher Reset-Eingang ist daher nicht notwendig. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "CommandAborted" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Error	Ist "Error" TRUE, wurde der Funktionsbausteins wegen eines Fehlers abgebrochen und befindet sich in einem Endzustand. "Error" bleibt solange TRUE bis der Steuereingang "Enable" deaktiviert wird. Ein zusätzlicher Reset-Eingang ist daher nicht notwendig. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "Error" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird

Tab. 14-10: I/O zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung



Ist es notwendig, dass bestimmte Eingänge nicht nur mit der Flanke an "Enable" übernommen werden, sonder zyklisch während der Bausteinbearbeitung, so ist dies explizit zu dokumentieren und nicht über Variablennamen kenntlich zu machen!



Bei Bausteinen mit "Shutdown" muss genau definiert werden, wie sich der Baustein verhält, wenn "Enable" wieder TRUE gesetzt wird, solange "Shutdown" noch gesetzt ist (Retriggern).

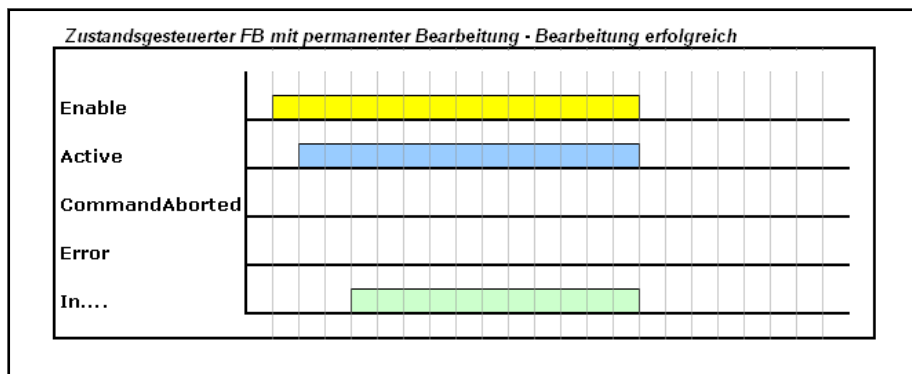


Abb. 14-13: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung erfolgreich

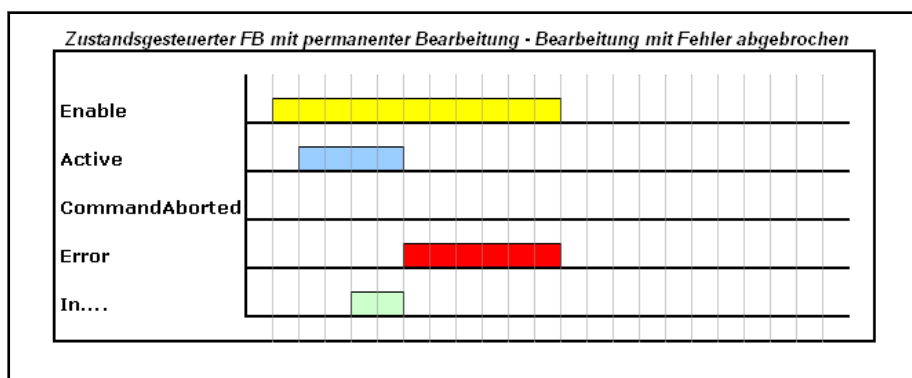


Abb. 14-14: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung mit Fehler abgebrochen

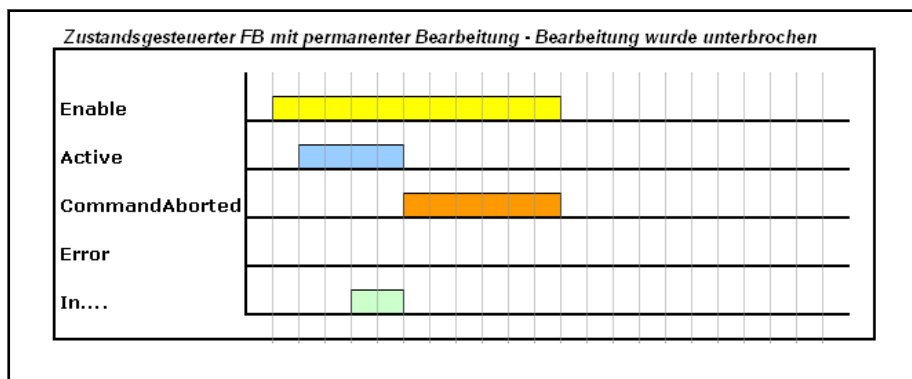


Abb. 14-15: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung unterbrochen

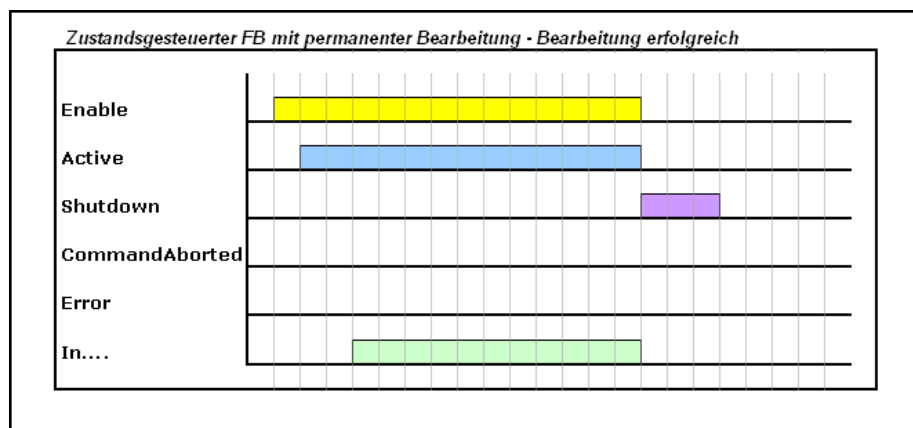


Abb. 14-16: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung erfolgreich

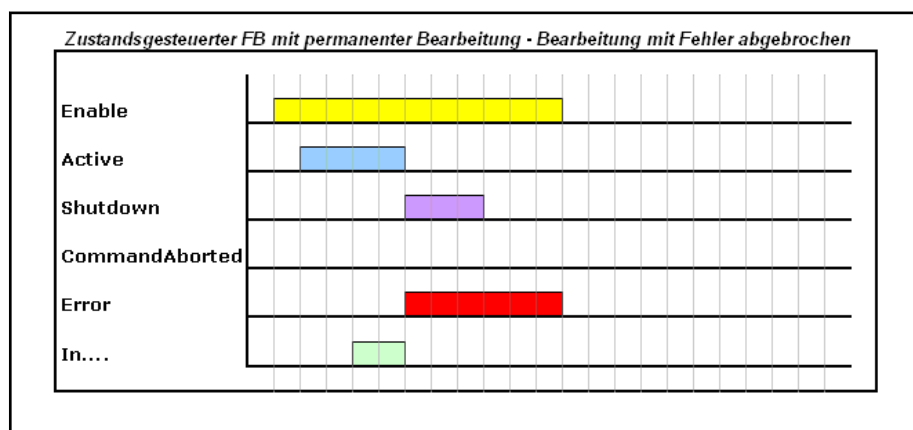


Abb. 14-17: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung mit Fehler abgebrochen

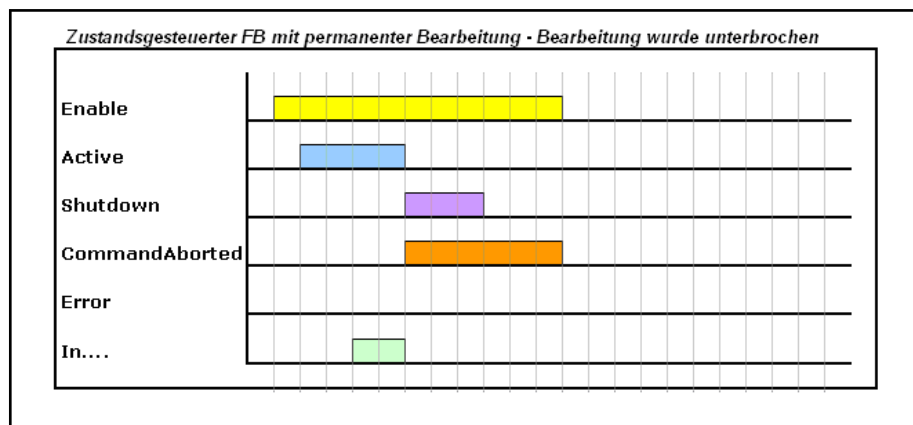


Abb. 14-18: I/O zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung unterbrochen

14.2.7 Ein- und Ausgänge flankengesteuerter Funktionsbausteine, permanente Bearbeitung

Steuerung / Bearbeitung	I/O	Variablenname	Beschreibung
Flankengesteuert / permanent	i	Execute	Mit einer positiven Flanke von "Execute" werden die Eingangsvariablen erfasst. Neue Eingangswerte werden erst über eine erneute positive Flanke von "Execute" wirksam. Der Flankenwechsel an "Execute" reicht aus, um den Baustein zu starten. Zur weiteren Bearbeitung ist der Zustand von "Execute" nicht relevant. Durch einen erneuten Flankenwechsel noch während der Bearbeitung werden der alte Auftrag verworfen, die Eingänge erneut übernommen und mit den neuen Werten der Auftrag fortgesetzt (Retriggung, Nachtriggern). Hat der Baustein einen "Shutdown"-Ausgang, ist dieses Retriggern eventuell nicht erlaubt
	O	In....	Ist "In...." TRUE, hat der Funktionsbaustein sein Ziel erreicht, befindet sich aber weiter im Eingriff, um das Erreichte zu "bewahren" und ist damit in einem dauerhaften Endzustand. Datenausgänge sind jetzt gültig. Da der FB weiter im Eingriff bleibt, muss auch "Active" TRUE bleiben. Das gilt auch dann, wenn "In...." wieder abfällt. "Error" und "CommandAborted" sind FALSE! Ist bei Auftragsabschluss der Steuerungseingang "Execute" FALSE, bleibt "In...." für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "In...." solange TRUE, bis "Execute" deaktiviert wird
	O	Active	Ist "Active" TRUE, arbeitet der Funktionsbaustein an seiner eigentlichen Aufgabe. Eventuelle Vorverarbeitungen werden nicht mit diesem Ausgang gekennzeichnet! Da der FB permanent im Eingriff bleibt, bleibt "Active" TRUE bis der Baustein durch "Error" oder "CommandAborted" beendet wird. Solange "Active" TRUE ist, müssen "Error" oder "CommandAborted" sowie „Shutdown“ FALSE sein!
	O	Shutdown (optional)	Ist ein "Shutdown"-Ausgang vorhanden, benötigt der Baustein, sobald der Baustein seine Aufgabe abbricht ("CommandAborted" oder "Error" = TRUE), noch mehrere Zyklen an Rechenzeit, um sich in einen definierten Endzustand zu bringen und verwendete Ressourcen freizugeben. Der Baustein muss solange aufgerufen werden, bis "Shutdown" FALSE wird!
	O	CommandAborted	Ist "CommandAborted" TRUE, wurde der Funktionsbaustein unterbrochen und befindet sich in einem Endzustand. "In....", "Active" und "Error" sind FALSE! Ist zum Unterbrechungszeitpunkt der Steuerungseingang "Execute" FALSE, bleibt "CommandAborted" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "CommandAborted" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "CommandAborted" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird
	O	Error	Ist "Error" TRUE, wurde der Funktionsbaustein wegen eines Fehlers abgebrochen und befindet sich in einem Endzustand. Ist bei Auftreten des Fehlers der Steuerungseingang "Execute" FALSE, bleibt "Error" für genau einen SPS-Zyklus TRUE. Ist "Execute" TRUE, bleibt "Error" solange TRUE bis "Execute" deaktiviert wird. Ist ein "Shutdown"-Ausgang vorhanden, wird dieser Ausgang gleichzeitig mit "Error" TRUE. Der Baustein muss dann solange weiterhin aufgerufen werden, bis "Shutdown" FALSE wird

Tab. 14-11: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung



Werden bestimmte Eingänge nicht nur mit der 0/1-Flanke an "Execute" übernommen, sondern zyklisch während der Bausteinbearbeitung, so ist dies explizit zu dokumentieren und nicht über Variablennamen kenntlich zu machen!



Bei Bausteinen mit "Shutdown" muss genau definiert werden, wie sich der Baustein verhält, wenn an "Execute" eine erneute Flanke erkannt wird, solange "Shutdown" noch gesetzt ist (Retriggeren).

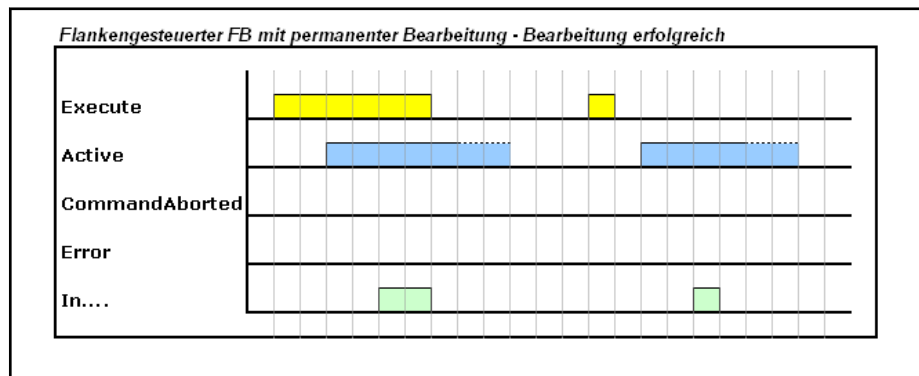


Abb. 14-19: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung erfolgreich

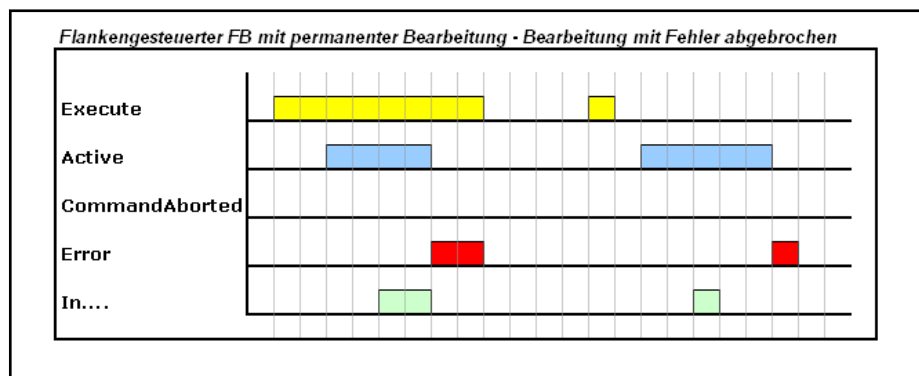


Abb. 14-20: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung mit Fehler abgebrochen

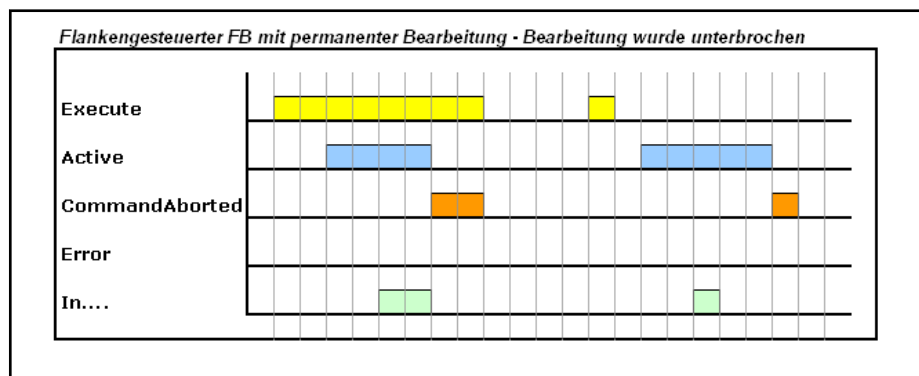


Abb. 14-21: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung ohne "Shutdown". Bearbeitung unterbrochen

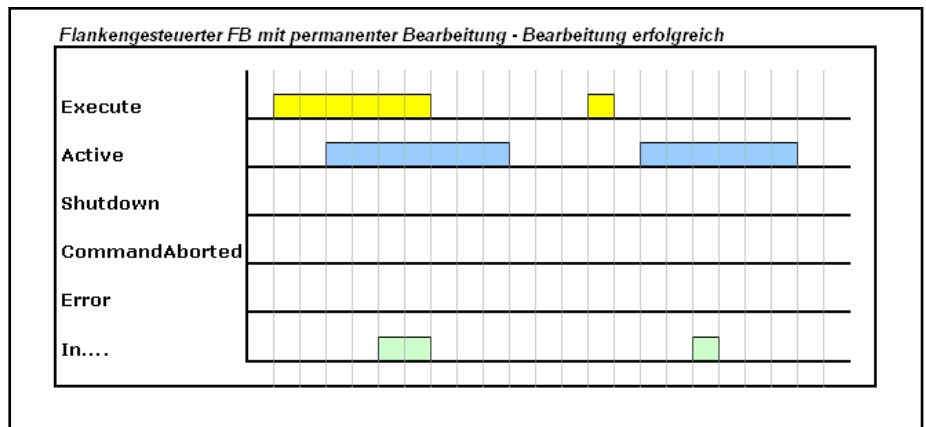


Abb. 14-22: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung erfolgreich

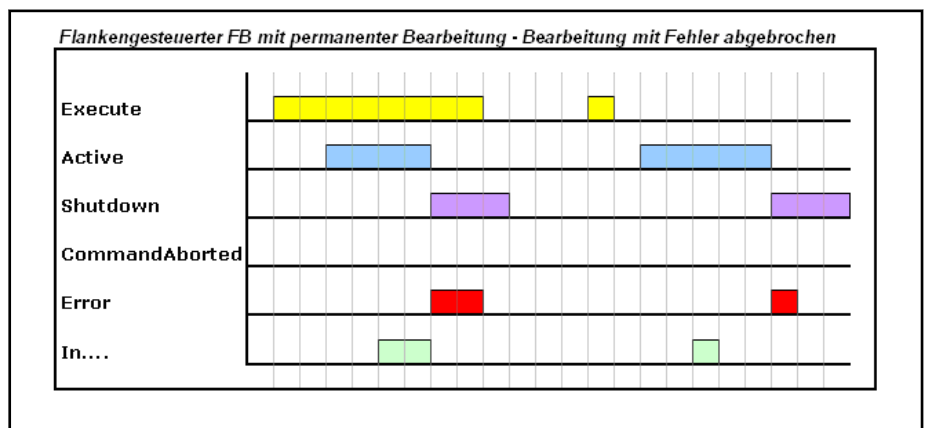


Abb. 14-23: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung mit Fehler abgebrochen

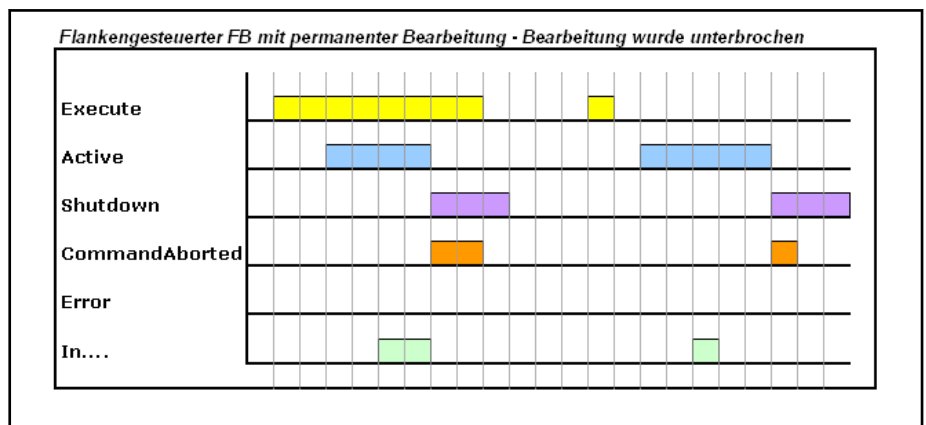


Abb. 14-24: I/O flankengesteuerter Funktionsbausteine, permanente Bearbeitung mit "Shutdown". Bearbeitung unterbrochen

14.3 NC-Kanal

14.3.1 Übersicht

Übersicht Kanalfunktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung

Allgemein	MT_ChNumberValidation, Seite 246	Prüft, ob die angegebene Kanalnummer in der NC-Konfiguration vorhanden ist
	MT_ChGetState, Seite 248	Aktuellen Kanalstatus ermitteln
Vorschub	MT_ChSetFeedOverride, Seite 261	Kanalvorschub ändern
	MT_GetFeedOverride, Seite 262	Aktiven Kanalvorschub ermitteln
NC-Programm	MT_ChSetReentryBreak-Point, Seite 258	NC-Programm Unterbrechungspunkt speichern
	MT_ChGetActiveNcProgram, Seite 250	Name des aktiven NC-Programms ermitteln
	MT_ChNcProgProgress, Seite 14.3.5	Aktuellen NC-Programmfortschritt ermitteln
Betriebsart	MT_ChOPMode, Seite 275	Ermitteln und Ändern der Betriebsarten eines Kanals (kombiniert MT_ChGetOPMode und MT_ChSetOPMode)
	MT_ChGetOPMode, Seite 263	Ermittelt die aktive Haupt- und Unterbetriebsart eines Kanals
	MT_ChSetOPMode, Seite 270	Ändert die Haupt- und Unterbetriebsarten eines Kanal
	MT_ChSetMainOPMode, Seite 266	Ändert die Hauptbetriebsarten eines Kanal

Tab. 14-12: Übersicht Kanalfunktionen

14.3.2 MT_ChNumberValidation, Gültigkeit der Kanalnummer prüfen

Kurzbeschreibung Der Funktionsbaustein prüft ob die angegebene Kanalnummer in der NC-Konfiguration vorhanden ist.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-13: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

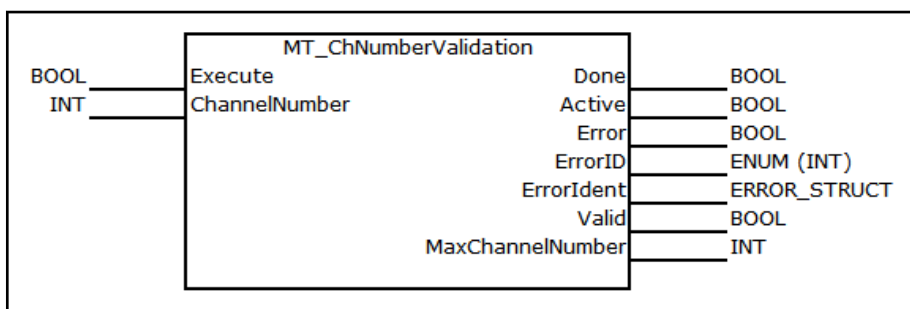


Abb. 14-25: Funktionsbaustein MT_ChNumberValidation

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Prüfung starten
	ChannelNumber	INT	Zu prüfende Kanalnummer

	Name	Typ	Beschreibung
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "Error" = TRUE)
	Valid	BOOL	TRUE = Kanalnummer gültig, FALSE = Kanalnummer ungültig, Wert von "Valid" nur gültig solange "Done" = TRUE
	MaxChannelNumber	INT	Maximale Anzahl in der NC konfigurierter Kanäle, Wert nur gültig solange "Done" = TRUE

Tab. 14-14: Ein-/Ausgänge Funktionsbaustein MT_ChNumberValidation

Signal-Zeit-Diagramm

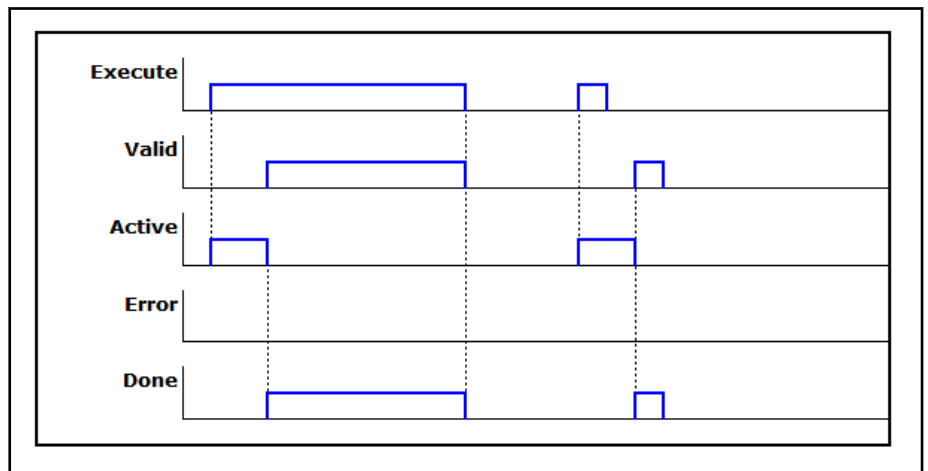


Abb. 14-26: Signal-Zeit-Diagramm MT_ChNumberValidation

Funktionsbeschreibung



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren die Prüfung genau einmal ausführt.

Ein Signalwechsel von Null nach Eins ($0 \rightarrow 1$) an "Execute" aktiviert den Funktionsbaustein und sichert den Zustand der Eingänge zu diesem Zeitpunkt. Sobald der Ausgang "Done" TRUE wird, ist die Prüfung der Kanalnummer erfolgreich abgeschlossen. Solange "Done" TRUE ist sind die Ausgänge "Valid" und "MaxChannelNumber" gültig.

Ist der Ausgang "Valid" TRUE so ist der Wert am Eingang "ChannelNumber" eine gültige Kanalnummer, welche in der NC-Konfiguration vorhanden (MaCoDa: 904000001) ist. Gibt es keine Entsprechung in der NC-Konfiguration ist der Ausgang "Valid" FALSE.

Der Ausgang "MaxChannelNumber" zeigt die Anzahl von Kanälen an die in der NC konfiguriert sind.

Konnte die Prüfung nicht durchgeführt werden, wird dies durch den Ausgang "Error" angezeigt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einer erneuten Prüfung, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute"

werden "Error" und "Done" gelöscht. Solange "Active" TRUE ist, haben Signaländerungen an den Eingängen keine Auswirkungen. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_ChNumberValidation"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00090415	Fehler von MT_CfgData, der Wert in Additional2 enthält der Fehlercode für MT_CfgData

Tab. 14-15: Fehler von "MT_ChNumberValidation", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel In diesem Beispiel geprüft ob die Nummer 3 eine gültige Kanalnummer ist. Das Ergebnis zeigt, das 3 eine gültige Kanalnummer ist und das 5 die maximal gültige Kanalnummer wäre.

```
1 PROGRAM prgTestChNumberValidation
2 VAR
3   fbChNumberValidation :MT_ChNumberValidation;
4   bExecute             :BOOL:=FALSE; (*execute setting operation mode*)
5   iChNumber            :UINT:=1;    (*channel number that should be validated*)
6   bDone               :BOOL:=FALSE; (*process successful finished*)
7   bActive             :BOOL:=FALSE; (*process active*)
8   bError              :BOOL:=FALSE; (*process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid*)
9   enErrorID           :ERROR_CODE;  (*basic error information, only valid when 'bError'==TRUE*)
10  stErrorIdent        :ERROR_STRUCT; (*detailed error information, only valid when 'bError'==TRUE*)
11  bValid              :BOOL:=FALSE;  (*TRUE=channel number ist valid, FALSE=channel number is invalid*)
12  iMaxChNumber        :UINT:=0;     (*maximum permissible channel number according to the NC-configuration*)
13 END_VAR
14 ..
```

Abb. 14-27: Deklaration MT_ChNumberValidation

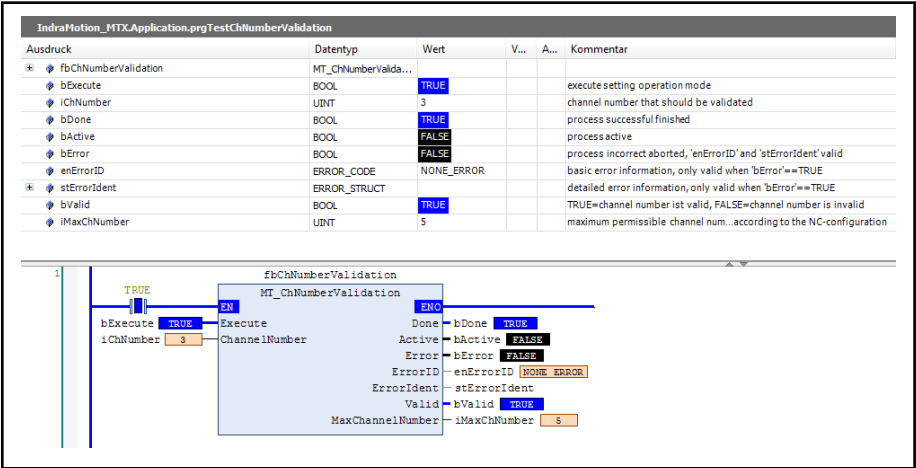


Abb. 14-28: Implementation MT_ChNumberValidation

14.3.3 MT_ChGetState, Aktuellen Kanalstatus ermitteln

Kurzbeschreibung Funktion zum Ermitteln des aktuellen Kanalstatus.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-16: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

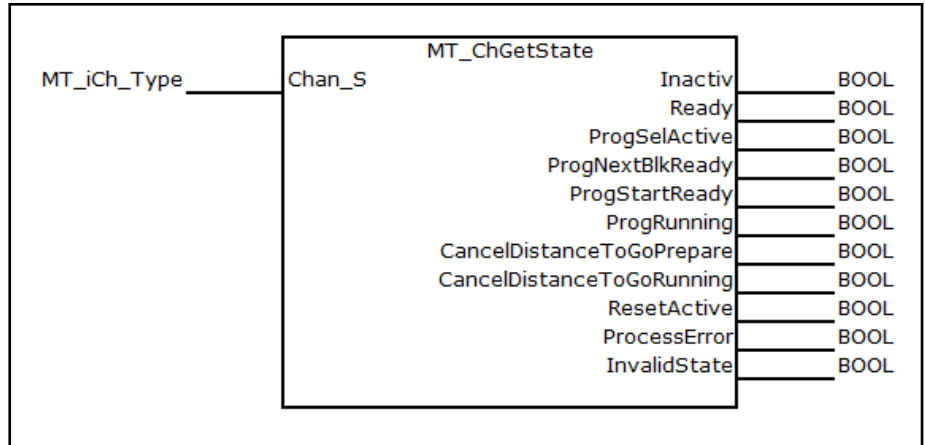


Abb. 14-29: Funktionsbaustein MT_ChGetState

	Name	Typ	Beschreibung
VAR_INPUT	Chan_S	MT_iCH_TYPE	Kanal-Interface: Kanal-Status-Signale
VAR_OUTPUT	Inactive	BOOL	Betriebsart ist nicht aktiv und ein Prozess kann angewählt werden
	Ready	BOOL	Betriebsart ist bereit und der Prozess kann gestartet werden
	ProgSelActive	BOOL	Ein angewähltes Programm wird vorbereitet (linken)
	ProgNextBlkReady	BOOL	Bei der gepufferten NC-Satzvorgabe sind alle Sätze abgearbeitet. Es wird auf die nächste Vorgabe gewartet
	ProgStartReady	BOOL	Betriebsart ist bereit, der Prozess steht am Programmanfang und kann gestartet werden
	ProgRunning	BOOL	Betriebsart ist aktiv und bearbeitet ein Programm oder einen NC-Satz
	CancelingDistanceToGoPrepare	BOOL	"Restweg löschen" wurde ausgelöst, Aufräumarbeiten dauern an
	CancelingDistanceToGoRunning	BOOL	"Restweg löschen" wurde ausgelöst, Betriebsart ist aktiv und bearbeitet vorhandene Puffer neu
	ResetActive	BOOL	Der Kanal führt gerade eine Grundstellung durch. Es dürfen keine Aufträge angenommen werden, bis der Zustand auf "inaktiv" übergeht
	ProcessError	BOOL	Es ist ein Fehler aufgetreten, der nur durch "Grundstellung" oder "Programm-Abwahl" behoben werden kann
	InvalidState	BOOL	Ungültiger Kanalstatus, die aktive Bitcodierung von "iCh_State_00..04" ist unbekannt

Tab. 14-17: Ein-/Ausgänge Funktionsbaustein MT_ChGetState

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

- Funktionsbeschreibung

Der aktuelle Kanalstatus wird bitcodiert über das Kanal-Interface (MT_iCh_Type) durch die Signale "iCh_State_00 .. 04" von der NC an die SPS übertragen. Diese Funktion dekodiert diese Information und setzt die entsprechenden Ausgänge der Funktion, um den Status des Kanals anzuzeigen. Es ist immer nur ein Status gültig.
- Fehlerbehandlung

Für diese Funktion ist keine Fehlerbehandlung notwendig.
- Implementationsbeispiel

Im Beispielprogramm wird die Betriebsart des ersten Kanals ermittelt.

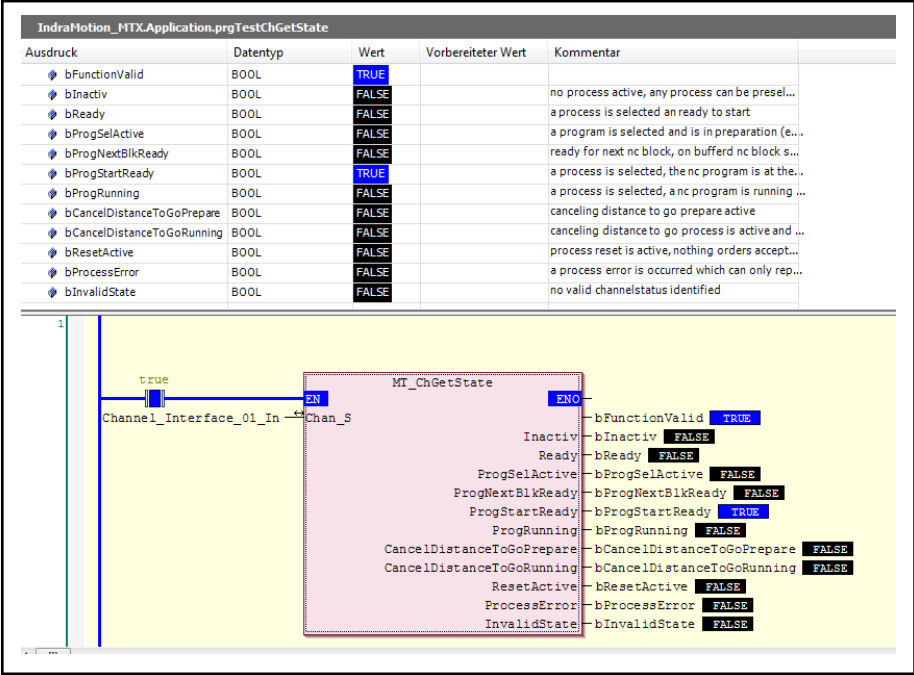


Abb. 14-30: Implementationsbeispiel MT_ChGetState

14.3.4 MT_ChGetActiveProgram, Aktives NC-Programm ermitteln

Kurzbeschreibung Funktionsbaustein zum Ermitteln des aktiven NC-Programms.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-18: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

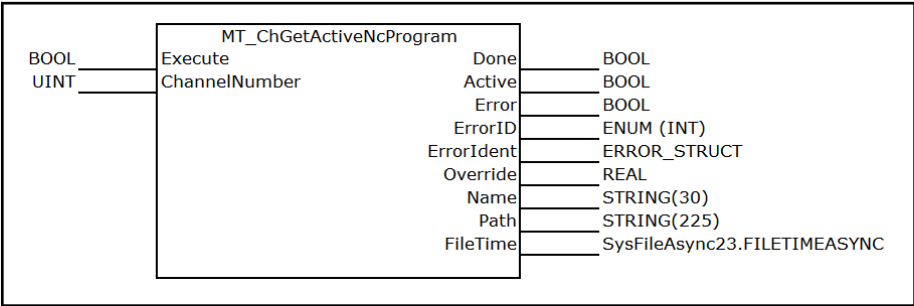


Abb. 14-31: Funktionsbaustein MT_ChGetActiveNcProgram

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Lesen des aktiven Nc-Programms starten
	ChannelNumber	UINT	Kanalnummer

	Name	Typ	Beschreibung
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig, wenn "Error" = TRUE)
	Name	STRING(30)	Dateiname des aktiven NC-Programms
	Path	STRING(225)	Ablagepfad der Datei des aktiven NC-Programms
	FileTime.dtCreation	DATE_AND_TIME	Datum und Uhrzeit zu der die Datei (aktives NC-Programm) erstellt wurde
	FileTime.dtLastAccess	DATE_AND_TIME	Datum und Uhrzeit als zuletzt auf die Datei (aktives NC-Programm) zugegriffen wurde
	FileTime.dtLastModification	DATE_AND_TIME	Datum und Uhrzeit als die Datei (aktives NC-Programm) zuletzt geändert wurde

Tab. 14-19: Ein-/Ausgänge Funktionsbaustein MT_ChGetActiveNcProgram

Signal-Zeit-Diagramm Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung Der Baustein ermittelt das aktuell im Kanal angewählte NC-Programm. Als Ergebnis wird der Dateiname (Name), der Ablagepfad (Path) sowie Datum und Uhrzeit der Erstellung, des letzten Zugriffs und der letzten Änderung (FileTime) ausgegeben.

Sobald Ausgang "Done" TRUE sind die ermittelten Werte am Ausgang "Name", "Path" und "FileTime" gültig.

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_ChGetActiveNcProgram"

Additional 1 (Hex)	Additional 2 (Hex)	Bedeutung
0x00000000	0x00000000	Kein Fehlercode verfügbar
16#00060401	MaxChannelNumber	Die angegebene Kanalnummer ist ungültig. Additional 2 enthält die höchste erlaubte Kanalnummer aus der NC-Konfiguration.

Additional 1 (Hex)	Additional 2 (Hex)	Bedeutung
0x00090426	ErrorCode MT_MCoDS	Fehler bei interner Verwendung von FB MT_MCoDS, Additional2 enthält Fehlercode dieses FB
0x00091401	ErrorCode SysFileAsync23.SysFileReadAsync	Fehler bei interner Verwendung von "SysFileAsync23.SysFileReadAsync" Additional 2 enthält den Fehlercode dieses FB

Tab. 14-20: Fehlercodes von "MT_ChGetActiveNcProgram", nur gültig wenn Ausgang"Error" == TRUE

Implementationsbeispiel

```
1 PROGRAM prgTestChGetActiveProgram
2 VAR
3     fbChGetActiveProgram : MT_ChGetActiveProgram;
4     bExecute             : BOOL:=FALSE;      (*execute setting operation mode*)
5     iChNumber            : UINT:=1;          (*channel number that should be validated*)
6     bDone                : BOOL:=FALSE;      (*process successful finished*)
7     bActive               : BOOL:=FALSE;      (*process active*)
8     bError                : BOOL:=FALSE;      (*process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid*)
9     enErrorID             : ERROR_CODE;       (*basic error information, only valid when 'bError'==TRUE*)
10    stErrorIdent           : ERROR_STRUCT;     (*detailed error information, only valid when 'bError'==TRUE*)
11    strName                : STRING(30);      (*nc program name*)
12    strPath                : STRING(225);     (*path to the nc program*)
13    stFileTime             : SysFileAsync23.FILETIMEASYNC; (*nc program file status (creation, last access, last modification*)
14 END_VAR
```

Abb. 14-32: Deklaration

IndraMotion_MTX.Application.prgTestChGetActiveProgram

Expression	Type	Value	Comment
fbChGetActiveProgram	MT_ChGetActiveProgram		
bExecute	BOOL	TRUE	execute setting operation mode
iChNumber	UINT	1	channel number that should be validated
bDone	BOOL	TRUE	process successful finished
bActive	BOOL	FALSE	process active
bError	BOOL	FALSE	process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid
enErrorID	ERROR_CODE	NONE_ERROR	basic error information, only valid when 'bError'==TRUE
stErrorIdent	ERROR_STRUCT		detailed error information, only valid when 'bError'==TRUE
strName	STRING(30)	'PYRAMIDE.ngp'	nc program name
strPath	STRING(225)	'/mnt/Pyramide/'	path to the nc program
stFileTime	SysFileAsync23.FILETIMEASYNC		nc program file status (creation, last access, last modification)
dtCreation	DATE_AND_TIME	DT#2018-8-28-11:37:58	
dtLastAccess	DATE_AND_TIME	DT#2018-8-28-11:37:58	
dtLastModification	DATE_AND_TIME	DT#2018-8-28-11:37:58	

Abb. 14-33: Implementation

14.3.5 MT_ChNcProgProgress, Aktuellen Programmfortschritt ermitteln

Kurzbeschreibung

Funktionsbaustein zum Ermitteln des aktuellen Fortschritt eines Nc-Programm.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-21: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

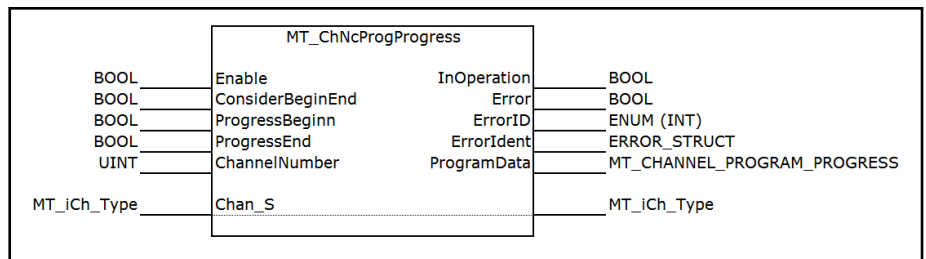


Abb. 14-34: Funktionsbaustein MT_ChNcProgProgress

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Lesen des aktiven Nc-Programm starten
	ConsiderBeginEnd	BOOL	Externe Ereignisse für ProgrammStart/-Stop berücksichtigen
	ProgressBeginn	BOOL	Externes Ereigniss, für den Fortschritt relevanter Programmteil wurde gestartet
	ProgressEnd	BOOL	Externes Ereignis, für den Fortschritt relevanter Programmteil ist zu Ende
	ChannelNumber	UINT	Kanalnummer
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig, wenn "Error" = TRUE)
	ProgramData.Name	STRING(30)	Dateiname des aktiven Nc-Programms
	ProgramData.Path	STRING(225)	Ablagepfad der Datei des aktiven Nc-Programm
	ProgramData.FileTime	DATE_AND_TIME	Datum und Uhrzeit zu der die Datei (aktives Nc-Programm)
	ProgramData.ProcessingTime	UDINT	aktuelle Bearbeitungszeit in Sekunden, Zeit welche das Programm bereits gelaufen ist
	ProgramData.RemainingTime	UDINT	aktuelle Restlaufzeit in Sekunden, Zeit die das Programm mit dem aktuellen Override vorraussichtlich noch laufen wird
	ProgramData.TotalTimeAct	UDINT	aktuelle Gesamtlaufzeit in Sekunden, Zeit welche das gesamte Programm mit den aktuellen Override vorraussichtlich laufen wird
	ProgramData.TotalTimeFile	UDINT	Gesamtlaufzeit in Sekunden, gelesen aus dem Nc-Programm (ProgRunTimeTotal)
	ProgramData.TotalTimeFileValid	BOOL	Gesamtlaufzeit, gelesen aus dem Nc-Programm (ProgRunTimeTotal) ist gültig
	ProgramData.TotalTimeMeasured	UDINT	Gesamtlaufzeit in Sekunden, gemessen von der SPS bei 100% Override
	ProgramData.TotalTimeMeasuredValid	BOOL	Gemessene Gesamtlaufzeit ist gültig

	Name	Typ	Beschreibung
	ProgramData.Progress	REAL	Programmfortschritt in Prozent
	ProgramData.ForecastAccuracy	UINT	Vorhersagegenauigkeit 1 = Gut, 2 = Mittel, 3 = Schlecht BYTE0 = Genauigkeit durch Änderung am Override beeinflusst BYTE1 = Genauigkeit durch Änderung der Betriebsart beeinflusst

Tab. 14-22: Ein-/Ausgänge Funktionsbaustein MT_ChGetActiveNcProgram

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Dieser Funktionsbaustein liefert Informationen zum Programmfortschritt für einen Kanal. Diese können mit der Anzeige im HMI-Header verknüpft werden.

Grundlagen:

Für die Fortschrittsberechnung wird die Gesamtlaufzeit des Nc-Programms benötigt (ermittelt bei 100% Override ohne Unterbrechung)! Wenn diese Information bekannt ist, kann sie in der ersten Zeile des Nc-Programms als Kommentar mit dem Schlüsselwort "ProgRunTimeTotal" angegeben werden. (z.B. ;ProgRunTimeTotal:=450s). Die Zeitangabe muss in Sekunden erfolgen! Auf Basis dieser Angabe wird dann der Programmfortschritt berechnet. Der im Nc-Programm angegebene Wert wird vom FB ausgelesen und findet sich in der Ausgangsstruktur im Element "ProgramData.TotalTimeFile".

Wenn der Funktionsbaustein einen unterbrechungsfreien Durchlauf des Nc-Programms bei 100% Override erkennt, wird diese Laufzeit gemessen und wird ab dann für jeden weiteren Programmlauf als Berechnungsgrundlage für den Programmfortschritt verwendet. Dies gilt auch, wenn in der ersten Zeile des Nc-Programms bereits eine Zeit angegeben wurde. Die von der SPS gemessene Programmlaufzeit wird in der Ausgangsstruktur "ProgramData.TotalTimeMeasured" angezeigt und ist nur gültig, wenn auch der Wert "ProgramData.TotalTimeMeasuredValid" TRUE ist.

Cache:

Normalerweise geht dieses Messergebnis mit der Abwahl des Nc-Programms verloren. Es besteht aber die Möglichkeit, einen Cache zu aktivieren, welcher alle Informationen zu einem Nc-Programm vorhält. Dazu muss in der globalen Variablenliste "pMtChannel" die Größe des Arrays "MT_NC_PROGRAM_PROGRESS_CACHE" auf größer 1 erhöht werden. (Bibliothekswart, Bibliothek MT_BaseAdditional)

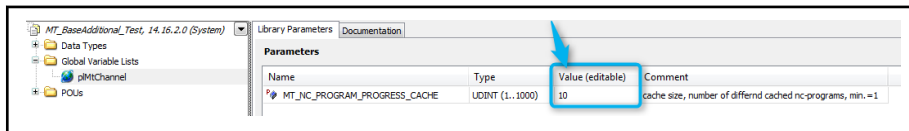


Abb. 14-35: Ein Cache von z.B. 10 speichert die Daten von maximal 10 verschiedenen Nc-Programmen. Ist der Cache voll, wird das Nc-Programm dessen Verwendung am längsten zurückliegt aus dem Cache entfernt und durch das neu angewählte ersetzt.



Cache benötigt Speicher! Für jeden Cacheplatz werden 381Byte benötigt.

Verwendung:

Sobald der FB durch Setzen von Eingang "Enable" aktiviert wurde wird laufend das aktive Programm, aus dem am Eingang "ChannelNumber" angegebenen Kanal, ermittelt. Sobald das Nc-Programm gestartet wurde, werden die aktuelle Gesamtlaufzeit (ProgramData.TotalTimeAct), die Restlaufzeit (ProgramData.RemainingTime) sowie der Fortschritt (ProgramData.Progress), in der Ausgangsstruktur permanent aktualisiert.

Programm-Start und Ende werden vom Baustein selbständig erkannt.

Sollen zusätzliche Kriterien für den Start und das Ende der Programmfortschrittsanzeige berücksichtigt werden, so kann dies über die Eingänge "ProgressBegin" und "ProgressEnd" realisiert werden. Damit der FB diese Eingänge berücksichtigt muss beim Einschalten des FB über "Enable" der Eingang "ConsiderBeginEnd" ebenfalls auf TRUE stehen.

Auf diese Weise ist es möglich, dass nur ein Teil des Nc-Programms für die Fortschrittsanzeige berücksichtigt wird indem entsprechende M-Funktionen mit den FB-Eingängen verknüpft werden.

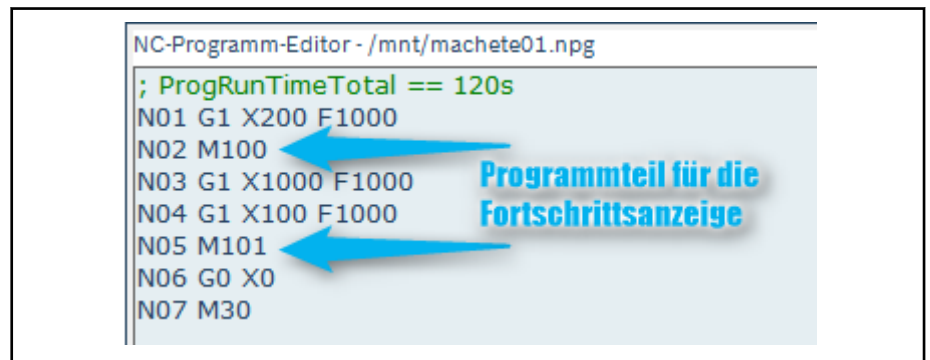


Abb. 14-36: Beispiel: Nc-Programm mit M-Funktion, Berücksichtigung von einem Programmteil zur Fortschrittsanzeige

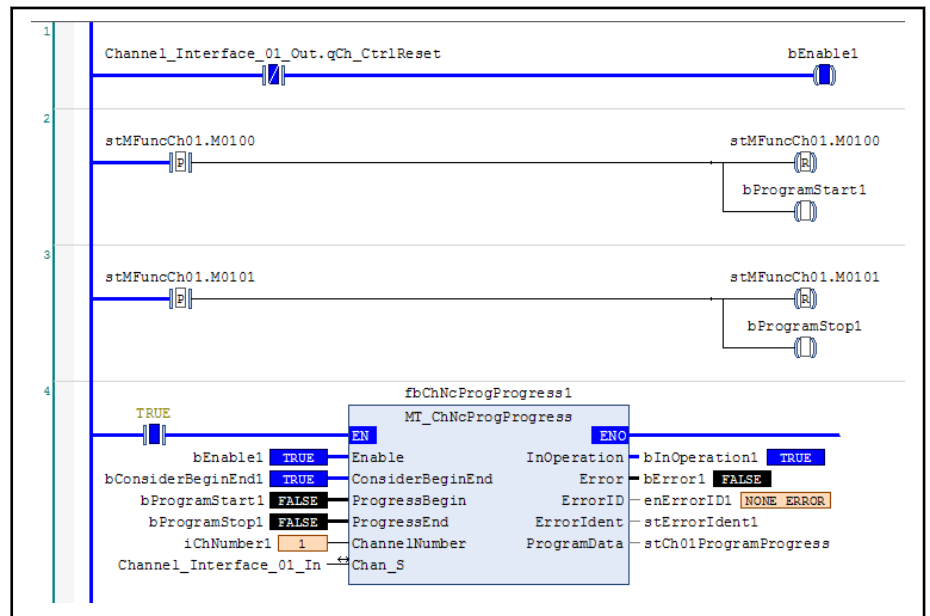


Abb. 14-37: Beispiel: Implementation des FB mit Berücksichtigung der M-Funktionen

HMI, Anzeige im Header des IndraWorksOperation

Im Header des IndraWorksOperation existiert eine Fortschrittsanzeige. Die Daten für diese Anzeige werden von dem FB bereit gestellt. Dazu muss die am Bausteinausgang "ProgramData" angeschlossene Variable folgenden Namen haben "stChXXProgramProgress" (XX muss durch die Kanalnummer ersetzt werden, z.B. stCh01ProgramProgress). Außerdem muss diese Variable mit dem Namespace "HmiHeader" versehen und in die Symbolkonfiguration aufgenommen werden.

```

19 {attribute 'namespace' := 'HmiHeader'}
20 {attribute 'symbol' := 'read'}
21 stCh01ProgramProgress :MT_CHANNEL_PROGRAM_PROGRESS; //result progress calculation
22

```

Abb. 14-38: Deklaration für Kanal 01 mit Anzeige im IndraWorksOperation

Weitere Details zur Anzeige im Header sind in der HMI Dokumentation "Multitouch" erklärt.

Genauigkeit

Die berechneten Werte basieren ausschließlich auf Zeitangaben. Der Override geht zwar in die Berechnungen ein, berücksichtigt aber keine geänderten Beschleunigungen oder ähnlichen. Deshalb werden die vom Baustein berechneten Daten um so ungenauer je mehr von dem Zustand abgewichen wird, bei dem die Programmlaufzeit ursprünglich ermittelt wurde (100% Override, Betriebsart Automatik Folgesatz, keine Unterbrechung). Um erkannte Ungenauigkeiten transparent zu machen liefert der Baustein über den Ausgang "ProgramData.ForecastAccuracy" eine Vorhersage der Genauigkeit der berechneten Werte. Da hauptsächlich Änderungen am Override und der Betriebsart Einfluß auf die Genauigkeit der Fortschrittsanzeige haben wird eine Genauigkeitsvorhersage basierend auf dem Override im LowByte und Änderungen der Betriebsart im HighByte dieses Ausgang abgebildet.

Die Werte in High- und LowByte haben folgende Bedeutung:

- 1 = hohe Genauigkeit der Fortschrittsanzeige
- 2 = mittlere Genauigkeit der Fortschrittsanzeige
- 3 = niedrige Genauigkeit der Fortschrittsanzeige

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen *Referenzen auf die Diagnosetabelle*

- MTX_TABLE

Fehler von "MT_ChNcProgProgress"

Additional1 (Hex)	Additional2 (Hex)	Bedeutung
0x00000000	0x00000000	Kein Fehlercode verfügbar
16#00060401	MaxChannelNumber	Die angegebene Kanalnummer ist ungültig. Additional2 enthält die höchste erlaubte Kanalnummer aus der NC-Konfiguration.

Additional1 (Hex)	Additional2 (Hex)	Bedeutung
0x00090426	ErrorCode MT_MCoDS	Fehler bei interner Verwendung von FB MT_MCoDS, Additional2 enthält Fehlercode dieses FB
0x00091401	ErrorCode SysFileAsync23.SysFileReadAsync	Fehler bei interner Verwendung von "SysFileAsync23.SysFileReadAsync" Additional2 enthält den Fehlercode dieses FB

Tab. 14-23: Fehlercodes von "MT_ChNcProgProgress", nur gültig wenn Ausgang "Error"==TRUE

Implementationsbeispiel

```

6 PROGRAM progTestChnCProgProgress
7 VAR
8   //CHANNEL 01 progress bar FB
9   -----
10  feChnCProgProgress1 : NT_ChnCProgProgress;
11  bEnable1             : BOOL:=FALSE; //enable program progress
12  bConsiderBeginEnd1   : BOOL:=FALSE; //TRUE==fb is waiting for inputs "ProgressBegin/-End", FALSE==fb consider only channel states
13  bProgramStart1       : BOOL:=FALSE; //command that no-program was started
14  bProgramStop1        : BOOL:=FALSE; //command that no-program was stopped
15  iChnNumber1          : UINT:=1; //channel number
16  bInOperation1        : BOOL:=FALSE; //progress calculation in work
17  bError1              : BOOL:=FALSE; //process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid
18  enErrorID1           : ERROR_CODE; //basic error information, only valid when 'bError'==TRUE
19  stErrorIdent1        : ERROR_STRUCT; //detailed error information, only valid when 'bError'==TRUE
20  (attribute 'namespace' := 'HmlHeader')
21  (attribute 'symbol' := 'read')
22  stChn01ProgramProgress : NT_CHANNEL_PROGRAM_PROGRESS; //result progress calculation
23
24  //CHANNEL 02 progress bar FB
25  -----
26  feChnCProgProgress2 : NT_ChnCProgProgress;

```

Abb. 14-39: Deklaration

IndraMotion_MTX.Application.prgTestChnCProgProgress				
Ausdruck	Datentyp	Wert	Kommentar	
* fbChnCProgProgress1	MT_ChnCProgProgress		CHANNEL 01 progress bar FB	
bEnable1	BOOL	TRUE	enable program progress	
bConsiderBeginEnd1	BOOL	FALSE		
bProgramStart1	BOOL	FALSE	command that nc-program was started	
bProgramStop1	BOOL	FALSE	command that nc-program was stopped	
iChnNumber1	UDINT	1	channel number	
bInOperation1	BOOL	TRUE	progress calculation in work	
bError1	BOOL	FALSE	process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid	
enErrorID1	ERROR_CODE	NONE_ERROR	basic error information, only valid when 'bError'==TRUE	
stErrorIdent1	ERROR_STRUCT		detailed error information, only valid when 'bError'==TRUE	
stCh01ProgramProgress	MT_CHANNEL_PROGRAM_P...		result progress calculation	
FileName	STRING(30)	'machete10.ngp'	nc-program file name	
FilePath	STRING(225)	'/mnt/	nc-program path to the file	
* FileTime	SysFileAsync23.FILETIMEAS...		nc-program timestamp (creation, last access, last modificati	
ProcessingTime	UDINT	10	actual program processing time in s	
RemainingTime	UDINT	49	actual program remaining time in s	
TotalTimeAct	UDINT	59	actual program total time in s	
TotalTimeFile	UDINT	65	active program total time, read from file	
TotalTimeFileValid	BOOL	TRUE	active program total time, read from file are valid	
TotalTimeMeasured	UDINT	59	active program total time, measured from plc	
TotalTimeMeasuredValid	BOOL	TRUE	active program total time, measured from plc are valid	
Progress	REAL	16.94915	actual program progress	
ForecastAccuracy	UDINT	257	LowByte Override == 1=high, 2=..., 3=low / HighByte OpM	
* fbChnCProgProgress2	MT_ChnCProgProgress		CHANNEL 02 progress bar FB	

Abb. 14-40:

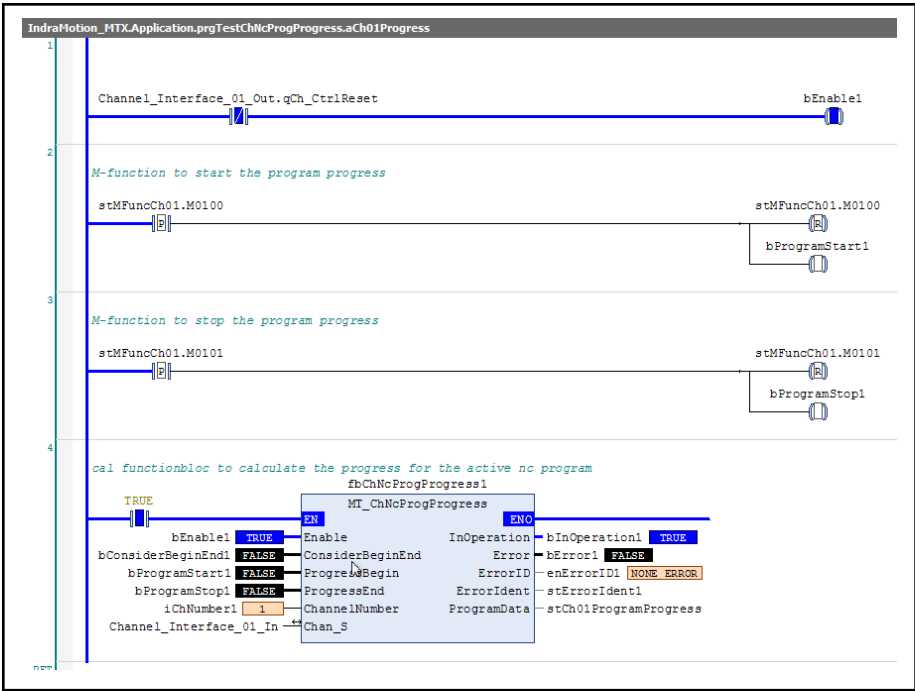


Abb. 14-41: Implementation

14.3.6 MT_ChSetReentryBreakPoint, Zum Wiederanfahren den Unterbrechungspunkt speichern

Kurzbeschreibung Funktionsbaustein aktiviert das speichern des Unterbrechungspunktes in der NC. Dieser kann beim späteren Wiederanfahren verwendet werden.

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-24: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

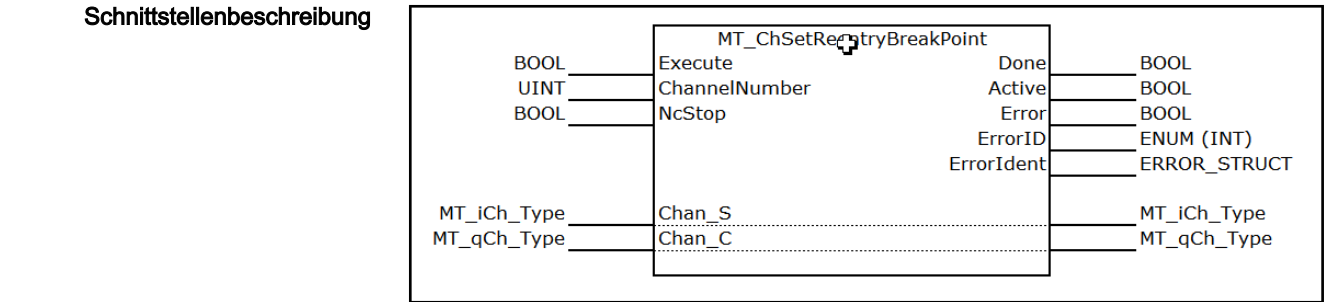


Abb. 14-42: Funktionsbaustein MT_ChSetReentryBreakPoint

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Unterbrechungspunkt speichern starten
	ChannelNumber	INT	Kanalnummer
	NcStop	BOOL	NcStop ausführen bevor Unterbrechungspunkt speichern startet

	Name	Typ	Beschreibung
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "Error" = TRUE)
VAR_IN_OUT	Chan_S	MT_iCh_Type	Kanal-Interface: Kanal-Status-Signale
	Chan_C	MT_qCh_Type	Kanal-Interface: Kanal-Steuer-Signale

Tab. 14-25: Ein-/Ausgänge Funktionsbaustein MT_ChSetReentryBreakPoint

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren die Prüfung genau einmal ausführt.

Das Speichern des Unterbrechungspunktes eines NC-Programm erfolgt automatisch mit jeder Kanalgrundstellung solange die Systemvariable "SD.SysRun[Ch].ReEntryStore" den Wert 1 hat. Die Verwendung dieses Funktionsbaustein bietet sich an wenn der Unterbrechungspunkt nicht mit jeder Kanalgrundstellung sondern nur bei bestimmten Ereignissen (z.B. Stromausfall) gespeichert werden soll.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein und prüft ob das Speichern des Unterbrechungspunktes bereits eingeschaltet ist ("SD.SysRun[Ch].ReEntryStore=1"). Falls Nein wird es Eingeschaltet, eine Kanalgrundstellung durch Setzen des Kanal-Steuer-Signal "qCh_CtrlReset" durchgeführt und an anschließend wieder ausgeschaltet. Ist das Speichern des Unterbrechungspunktes bereits aktiv wird nur eine Kanalgrundstellung ausgelöst.

Ist der Eingang NcStop gesetzt wird der FB vor dem Speichern des Unterbrechungspunktes einen NC-Stop auszuführen. Dazu werden die Kanal-Steuer-Signale Einleesperre (Kanal-Status-Signal: iCh_TransferLockAct) und Vorschub-Halt (Kanal-Status-Signal: iCh_FeedHoldAct) gesetzt.

Sobald der Ausgang "Done" TRUE wird, ist das Speichern des Unterbrechungspunkt erfolgreich abgeschlossen. Andernfalls wird dies durch den Ausgang "Error" angezeigt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einer erneuten Ausführung, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Solange "Active" TRUE ist, haben Signaländerungen an den Eingängen keine Auswirkungen. Ausnahme sind die Eingänge "Chan_S" und "Chan_C"(VAR_IN_OUT), diese werden zyklisch gelesen solange der Ausgang "Active" TRUE ist.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_ChSetReentryBreakPoint"

Additional1 (Hex)	Additional2 (Hex)	Bedeutung
0x0000000 0	0x0000000 0	Kein Fehlercode verfügbar
0x0004020 1	0x0000000 C	Das Ändern de Interface-Signal qCh_TransferLock ist fehlgeschlagen. Eventuell wurden sie außerhalb des FB überschrieben
0x0002020 1	0x0000000 D	Die Rückmeldung das Interface-Signal "iCh_FeedHoldAct" ist innerhalb von 500 ms nicht erfolgt. Eventuell ist die Schnittstelle für einen anderen Kanal am FB angeschlossen.
0x0002020 1	0x0000001 1	Die Rückmeldung über das Interface-Signal "iCh_Reset" ist innerhalb von 500 ms nicht erfolgt. Eventuell ist die Schnittstelle für einen anderen Kanal am FB angeschlossen.
16#000604 01	MaxChannelNumber	Die angegebene Kanalnummer ist ungültig. Additional2 enthält die höchste erlaubte Kanalnummer aus der NC-Konfiguration.
0x0009041 5	ErrorNumber MT_CfgData	Fehler von MT_CfgData, der Wert in Additional2 enthält der Fehlercode für MT_CfgData
0x0009042 2	ErrorNumber MT_SD_RD	Fehler von MT_SD_RD, der Wert in Additional2 enthält der Fehlercode für MT_SD_RD
0x0009042 3	ErrorNumber MT_SD_WR	Fehler von MT_SD_WR, der Wert in Additional2 enthält der Fehlercode für MT_SD_WR

Tab. 14-26: Fehler von "MT_ChSetReentryBreakPoint", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

```

1 PROGRAM prgTestChSetReentryBreakPoint
2 VAR
3   fbChSetReentryBreakPoint: MT_ChSetReentryBreakPoint;
4   bExecute                 :BOOL:=FALSE;      (*execute FB*)
5   bNoStop                  :BOOL:=FALSE;      (*TRUE == the FB triggers a No-Stop before set the reentry breakpoint*)
6   iChNumber                :UINT:=1;          (*Number of the channel to for which you want set the reentry breakpoint*)
7   bDone                    :BOOL:=FALSE;      (*process successful finished*)
8   bActive                  :BOOL:=FALSE;      (*process active*)
9   bError                   :BOOL:=FALSE;      (*process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid*)
10  enErrorID                 :ERROR_CODE;      (*basic error information, only valid when 'bError'==TRUE*)
11  stErrorIdent              :ERROR_STRUCT;     (*detailed error information, only valid when 'bError'==TRUE*)
12 END VAR

```

Abb. 14-43: Deklaration MT_ChSetReentryBreakPoint

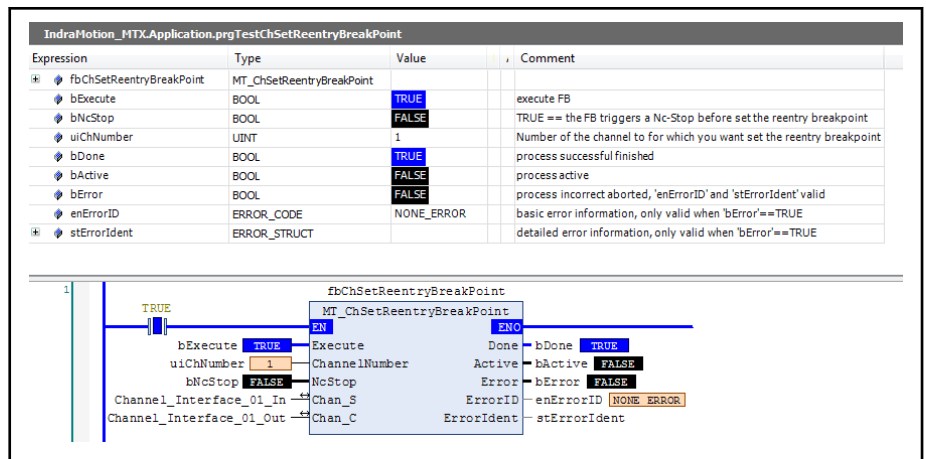


Abb. 14-44: Implementation MT_ChSetReentryBreakPoint

14.3.7 MT_ChSetFeedOverride, Kanalvorschub ändern

Kurzbeschreibung

Funktion zum Setzen des Vorschubwertes für einen Kanal.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-27: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

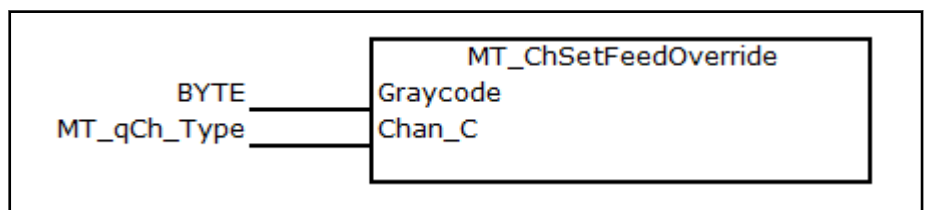


Abb. 14-45: Funktionsbaustein MT_ChSetFeedOverride

	Name	Typ	Beschreibung
VAR_INPUT	Graycode	BYTE	Overridewert als Graycode
VAR_IN_OUT	Chan_C	MT_qCh_Type	Kanal-Interface: Kanal-Steuer-Signale

Tab. 14-28: Ein-/Ausgänge Funktionsbaustein MT_ChSetFeedOverride

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Der Kanalvorschubwert muss für jeden Kanal über dessen Kanal-Interface (MT_qCh_Type) durch das Signal "qCh_Override" von der SPS an die NC übertragen werden. Mit dieser Funktion wird der am Eingang "Graycode" anliegende Wert gewandelt und an das angeschlossene Kanal-Interface übergeben.

Fehlerbehandlung

Für diese Funktion ist keine Fehlerbehandlung notwendig.

Implementationsbeispiel

Beispielprogramm für das Ändern des Override für den ersten Kanal.

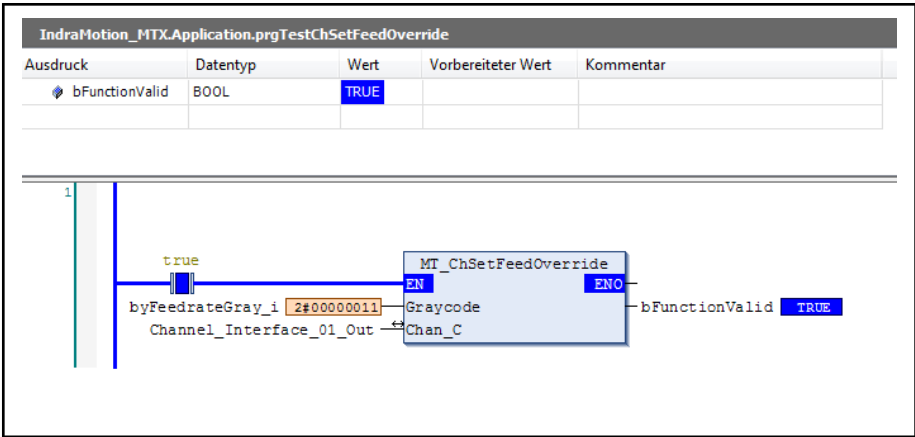


Abb. 14-46: Implementation MT_ChSetFeedOverride

14.3.8 MT_ChGetFeedOverride, Aktiven Vorschub-Override ermitteln

Kurzbeschreibung	Funktion zum Ermitteln des aktiven Vorschub-Override.		
Zuordnung Zielsystem/Bibliothek	Zielsystem	Bibliothek	
	IndraMotion_MTX	MT_BaseAdditional	

Tab. 14-29: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

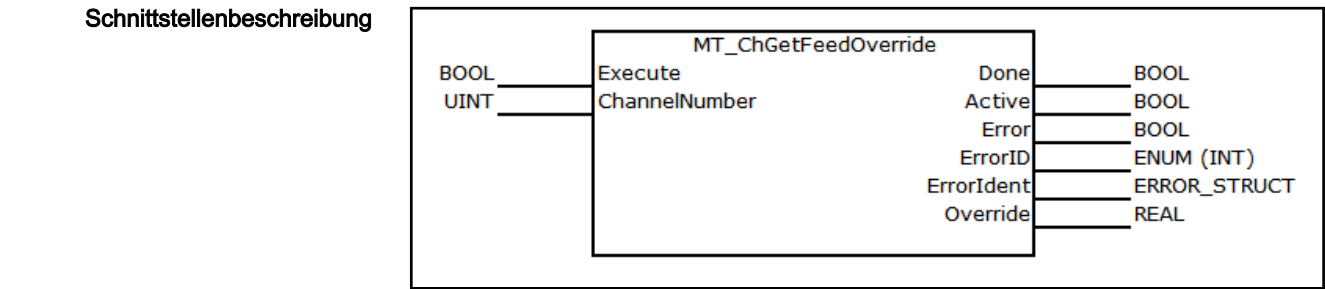


Abb. 14-47: Funktionsbaustein MT_ChGetFeedOverride

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Lesen des Vorschub-Overridee ausführen
	ChannelNumber	UINT	Kanalnummer
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig, wenn "Error" = TRUE)
	Override	REAL	aktueller Override für den vorgegebenen Kanal

Tab. 14-30: Ein-/Ausgänge Funktionsbaustein MT_ChGetFeedOverride

- Signal-Zeit-Diagramm**
- Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.
- Funktionsbeschreibung**
- Durch das Lesen der Funktion MCODS 7 ermittelt der FB den Override für den am Eingang "ChannelNumber" angegebene Kanal.
Sobald Ausgang "Done" TRUE ist ist der ermittelte Wert am Ausgang "Override" gültig.

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen *Referenzen auf die Diagnosetabelle*

- MTX_TABLE

Fehler von "MT_ChGetFeedOverride"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00060401	Ungültige Eingangsbelegung. Es wurde eine ungültige Kanalnummer gewählt
0x00090426	Fehler bei interner Verwendung von FB MT_MCoDS, Additional2 enthält Fehlercode von MT_MCoDS

Tab. 14-31: Fehler von "MT_GetFeedOverride", aus "ErrorIdent.Additional1", gültig wenn "bError" == TRUE

Implementationsbeispiel Im Beispielprogramm wird die Betriebsart des ersten Kanals ermittelt.

```

1 PROGRAM prgTestChGetActiveOverride
2   VAR
3     fbChGetActiveProgram : MT_ChGetFeedOverride;
4     bExecute             : BOOL:=FALSE; (*execute setting operation mode*)
5     iChNumber            : UINT:=1;    (*channel number that should be validated*)
6     bDone                : BOOL:=FALSE; (*process successful finished*)
7     bActive              : BOOL:=FALSE; (*process active*)
8     bError               : BOOL:=FALSE; (*process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid*)
9     enErrorID            : ERROR_CODE;  (*basic error information, only valid when 'bError'==TRUE*)
10    stErrorIdent         : ERROR_STRUCT; (*detailed error information, only valid when 'bError'==TRUE*)
11    rOverride            : REAL:=0.0;
12  END_VAR

```

Abb. 14-48: Deklaration

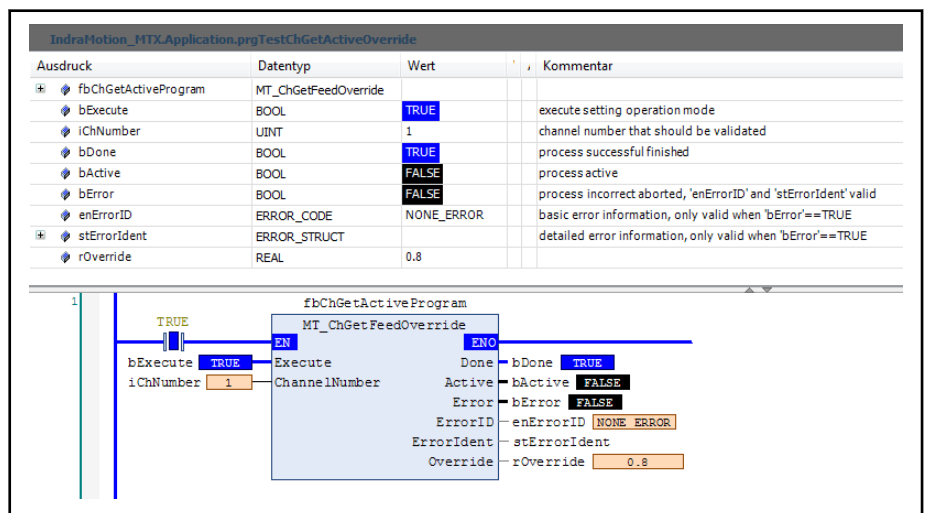


Abb. 14-49: Implementation

14.3.9 MT_ChGetOPMode, Aktive Betriebsart ermitteln

Kurzbeschreibung

Funktion zum Ermitteln der aktiven Haupt- und Unterbetriebsarten.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-32: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

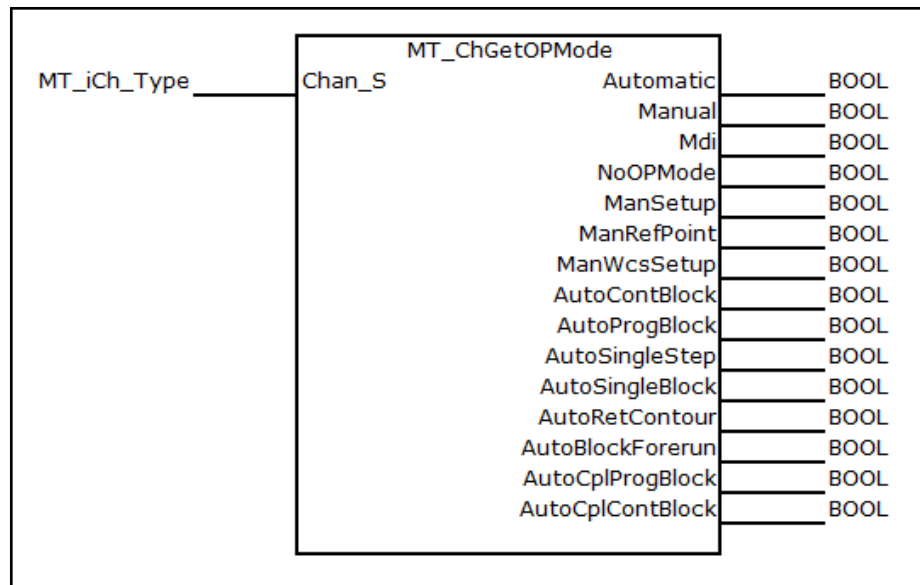


Abb. 14-50: Funktionsbaustein MT_ChGetOPMode

	Name	Typ	Beschreibung
VAR_INPUT	Chan_S	MT_iCH_TYPE	Kanal-Interface: Kanal-Status-Signale
VAR_OUTPUT	Automatic	BOOL	Hauptbetriebsart: Automatikbetrieb
	Manual	BOOL	Hauptbetriebsart: Handbetrieb
	Mdi	BOOL	Hauptbetriebsart: MDI - Handeingabe NC-Satz
	NoOPMode	BOOL	Keine Betriebsart
	ManSetup	BOOL	Unterbetriebsart: Einrichten manuell Tippbetrieb
	ManRefPoint	BOOL	Unterbetriebsart: Einrichten Referenzpunkt anfahren
	ManWcsSetup	BOOL	Unterbetriebsart: Einrichten manuell Werkstückkoordinaten
	AutoContBlock	BOOL	Unterbetriebsart: Folgesatz
	AutoProgBlock	BOOL	Unterbetriebsart: Programmsatz
	AutoSingleStep	BOOL	Unterbetriebsart: Einzelschritt
	AutoSingleBlock	BOOL	Unterbetriebsart: Einzelsatz
	AutoRetContour	BOOL	Unterbetriebsart: Wiederanfahren
	AutoBlockForerun	BOOL	Unterbetriebsart: Satzvorlauf
	AutoCplProgBlock	BOOL	Unterbetriebsart: CPL Debugger Programmsatz
	AutoCplContBlock	BOOL	Unterbetriebsart: CPL Debugger Folgesatz

Tab. 14-33: Ein-/Ausgänge Funktionsbaustein MT_ChGetOPMode

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Die MTX verwendet für jeden Kanal drei Hauptbetriebsarten und elf Unterbetriebsarten. Alle Zustände, in welchen sich ein Kanal im Hinblick auf die Betriebsart befinden kann, werden bitcodiert über das Kanal-Interface (MT_iCh_Type) durch die Signale "iCh_OpMode_00 .. 03" von der NC an die SPS übertragen. Es kann immer nur eine Hauptbetriebsart mit einer zugehö-

rigen Unterbetriebsart (wenn vorhanden) aktiv sein. Zudem gibt es den gültigen Zustand, dass keine Betriebsart gewählt ist. Die folgende Tabelle zeigt alle Hauptbetriebsarten mit der Zuordnung der Unterbetriebsarten und den entsprechenden Signalausgängen des FB, die TRUE sind, wenn der Kanal im entsprechenden Zustand ist.

Haupt- und Unterbetriebsarten

Hauptbetriebsart	Unterbetriebsart	FB Ausgangssignal
Automatikbetrieb	Folgesatz	AutoContBlock, Automatic
Automatikbetrieb	Programmsatz	AutoProgBlock, Automatic
Automatikbetrieb	Einzelsschritt	AutoSingleStep, Automatic
Automatikbetrieb	Einzelssatz	AutoSingleBlock, Automatic
Automatikbetrieb	Wiederanfahren	AutoContBlock, Automatic
Automatikbetrieb	Satzvorlauf	AutoBlockForerun, Automatic
Automatikbetrieb	CPL Debugger Programmsatz	AutoCplProgBlock, Automatic
Automatikbetrieb	CPL Debugger Folgesatz	AutoCplContBlock, Automatic
Handbetrieb	Einrichten manuell Tipbetrieb	ManSetup, Manual
Handbetrieb	Einrichten Referenzpunkt anfahren	ManRefPoint, Manual
Handbetrieb	Einrichten manuell Werkstückkoordinaten	ManWcsSetup, Manual
MDI	Keine	Mdi
Keine	Keine	NoOPMode

Tab. 14-34: Übersicht der Haupt- und Unterbetriebsarten mit den zugehörigen Interface-Signalen

Fehlerbehandlung Implementationsbeispiel

Für diese Funktion ist keine Fehlerbehandlung notwendig.
Im Beispielprogramm wird die Betriebsart des ersten Kanals ermittelt.

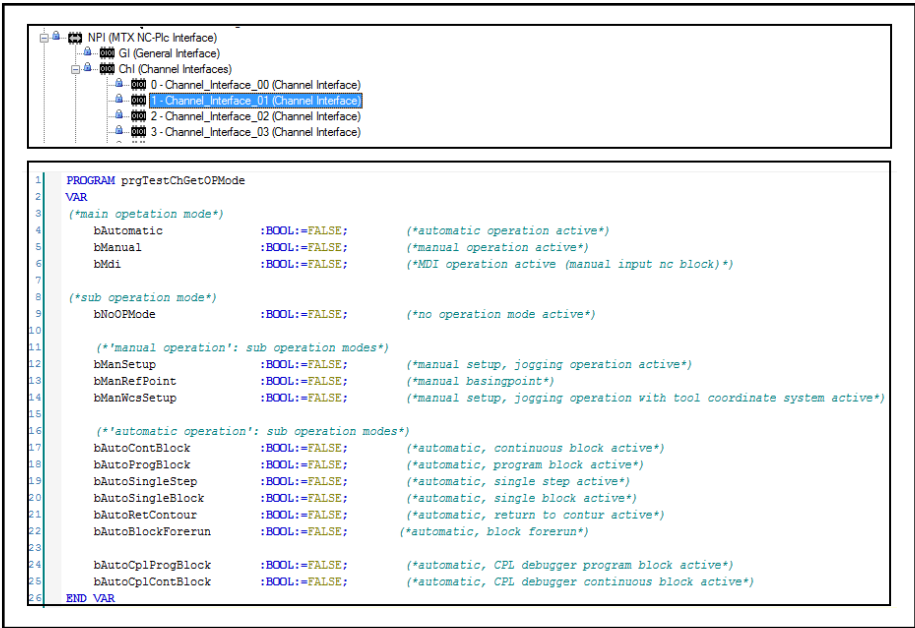


Abb. 14-51: Deklaration MT_ChGetOPMode

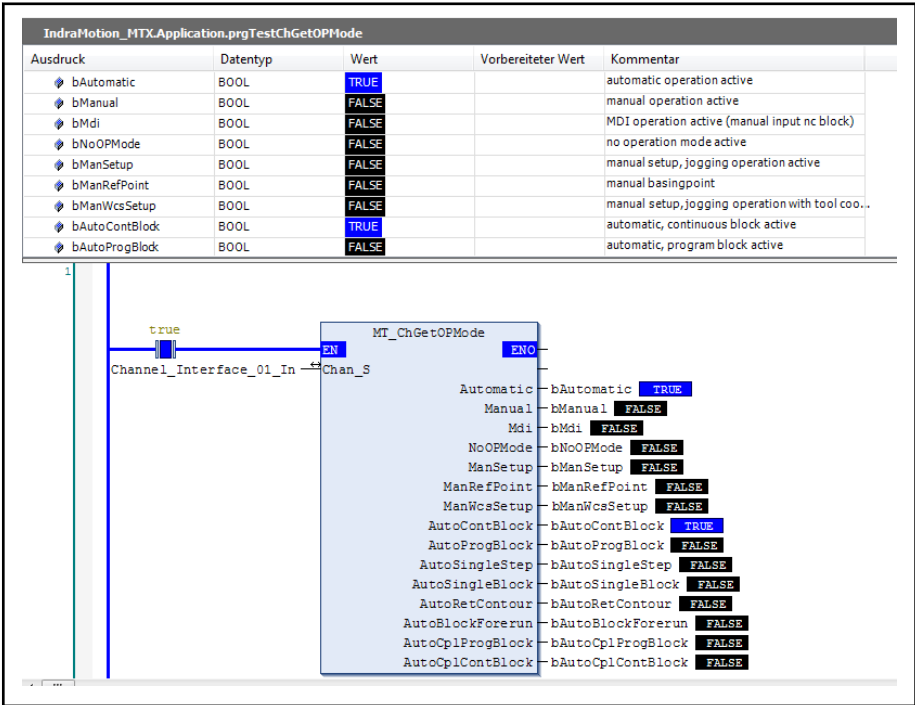


Abb. 14-52: Implementation MT_ChGetOPMode

14.3.10 MT_ChSetMainOPMode, Ändern der NC-Hauptbetriebsarten

Kurzbeschreibung

Zuordnung Zielsystem/Bibliothek

Funktionsbaustein zum Ändern der Hauptbetriebsarten.

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-35: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

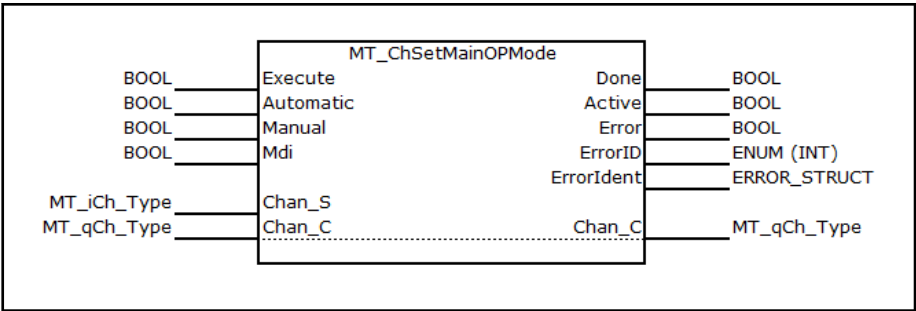


Abb. 14-53: Funktionsbaustein MT_ChSetMainOPMode

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Setzen der Betriebsart ausführen
	Automatic	BOOL	Hauptbetriebsart: Automatikbetrieb
	Manual	BOOL	Hauptbetriebsart: Handbetrieb
	Mdi	BOOL	Hauptbetriebsart: MDI - Handeingabe NC-Satz
	Chan_S	MT_iCh_Type	Kanal-Interface: Kanal-Status-Signale
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "Error"=TRUE)
VAR_IN_OUT	Chan_C	MT_qCh_Type	Kanal-Interface: Kanal-Steuer-Signale

Tab. 14-36: Ein-/Ausgänge Funktionsbaustein MT_ChSetMainOPMode

Signal-Zeit-Diagramm

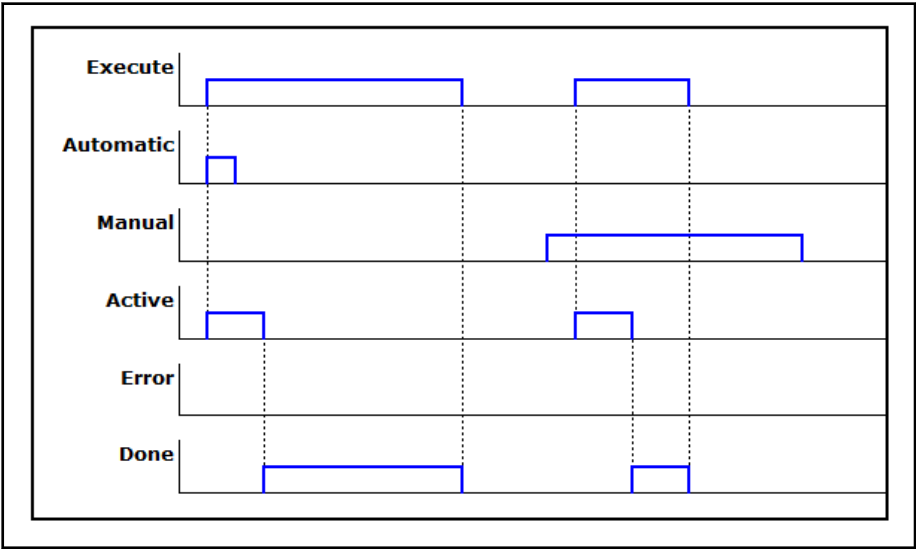


Abb. 14-54: Signal-Zeit-Diagramm MT_ChSetMainOPMode

Links: Setzen Betriebsart "Automatik" erfolgreich

Rechts: Setzen Betriebsart "Hand" erfolgreich

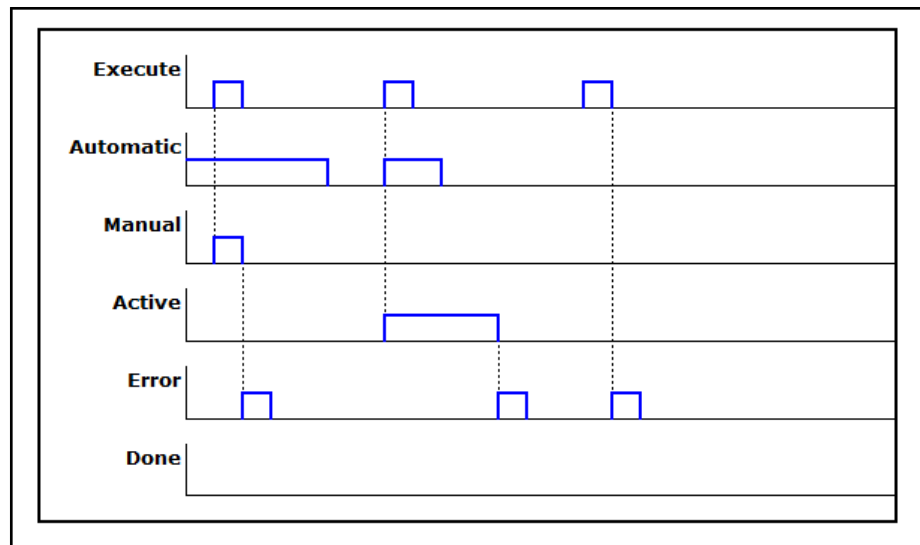


Abb. 14-55: Signal-Zeit-Diagramm MT_ChSetMainOPMode

Fehler links: Mehr als eine Betriebsart vorgewählt

Fehler Mitte: Timeout (500ms) keine Rückmeldung, dass Betriebsart gesetzt wurde

Fehler rechts: Es wurde keine Betriebsart vorgewählt

Funktionsbeschreibung

Die MTX verwendet für jeden Kanal drei Hauptbetriebsarten und elf Unterbetriebsarten. Alle Zustände, in welche ein Kanal im Hinblick auf die Betriebsart gebracht werden kann, werden bitcodiert über das Kanal-Interface (MT_qCh_Type) durch die Signale "qCh_OpModeSel_00 .. 03" von der SPS an die NC übertragen. Es kann immer nur eine Hauptbetriebsart mit einer zugehörigen Unterbetriebsart (wenn vorhanden) aktiv sein. Zudem gibt es den gültigen Zustand, dass keine Betriebsart gewählt ist. Mit dem Funktionsbaustein "MT_ChSetMainOpMode" kann man nur Hauptbetriebsarten schalten. Bei einem Wechsel der Hauptbetriebsart (z. B. Automatik nach Handbetrieb) wird die gerade aktive Unterbetriebsart gesichert (z. B. "Folgesatz"). Schaltet man später zurück in diese Hauptbetriebsart, wird die zuletzt aktive Unterbetriebsart wieder gesetzt.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren das Ändern der Betriebsart genau einmal ausführt.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein und sichert den Zustand der Eingänge zu diesem Zeitpunkt. Lediglich die Eingänge "Chan_S" und "Chan_C" werden zyklisch übernommen, solange der Baustein aktiv ist ("Active"==TRUE). Mit dem Signalwechsel muss die gewünschte Hauptbetriebsart vorgewählt sein (z. B. "Manual"). Sobald der Ausgang "Done" TRUE wird, ist das Setzen der gewünschten Betriebsart erfolgreich abgeschlossen. Konnte die Betriebsart nicht geändert werden, wird dies durch den Ausgang "Error" angezeigt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdnt". Vor einem erneuten Wechsel der Betriebsart, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Solange "Active" TRUE ist, haben Signaländerungen an den Eingängen zur Vorwahl der Betriebsart (z. B. "Manual") keine Auswirkungen. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.



Beim Wechsel der Hauptbetriebsart wird immer Grundstellung des Kanals ausgeführt.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_ChSetMainOPMode"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010001	Ungültige Eingangsbelegung. Es wurde entweder keine oder mehr als eine Betriebsart vorgewählt
0x00040201	Das Ändern der Interface-Signale "Chan_C.qCh_OpModeSel_00..03" ist fehlgeschlagen. Eventuell wurden sie außerhalb des FB überschrieben
0x00020201	Die Rückmeldung über die Änderung der Betriebsart über die Interface-Signale "Chan_S.iCh_OpMode_00..03" ist innerhalb von 500 ms nicht erfolgt. Eventuell ist die Schnittstelle für einen anderen Kanal am FB angeschlossen.
0x 00050001	Interner Fehler
0x 00050002	Interner Fehler

Tab. 14-37: Fehler von "MT_SetMainOPMode", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

In diesem Beispiel wird die Betriebsart auf die Unterbetriebsart Automatik-Folgesatz geändert.

```

1  PROGRAM prgTestChSetMainOPMode
2  VAR
3      fbSetMainOPMode      :MT_ChSetMainOPMode;
4      bExecute             :BOOL:=FALSE;      (*execute setting operation mode*)
5
6      (*main operation mode*)
7      bAutomatic           :BOOL:=FALSE;      (*activate automatic operation *)
8      bManual              :BOOL:=FALSE;      (*activate manual operation *)
9      bMdi                 :BOOL:=FALSE;      (*activate MDI operation (manual input no block)*)
10
11      bDone                :BOOL:=FALSE;      (*process successful finished*)
12      bActive              :BOOL:=FALSE;      (*process active*)
13      bError               :BOOL:=FALSE;      (*process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid*)
14      enErrorID            :ERROR_CODE;      (*basic error information, only valid when 'bError'==TRUE*)
15      stErrorIdent         :ERROR_STRUCT;     (*detailed error information, only valid when 'bError'==TRUE*)
16  END_VAR

```

Abb. 14-56: Deklaration MT_ChSetMainOPMode

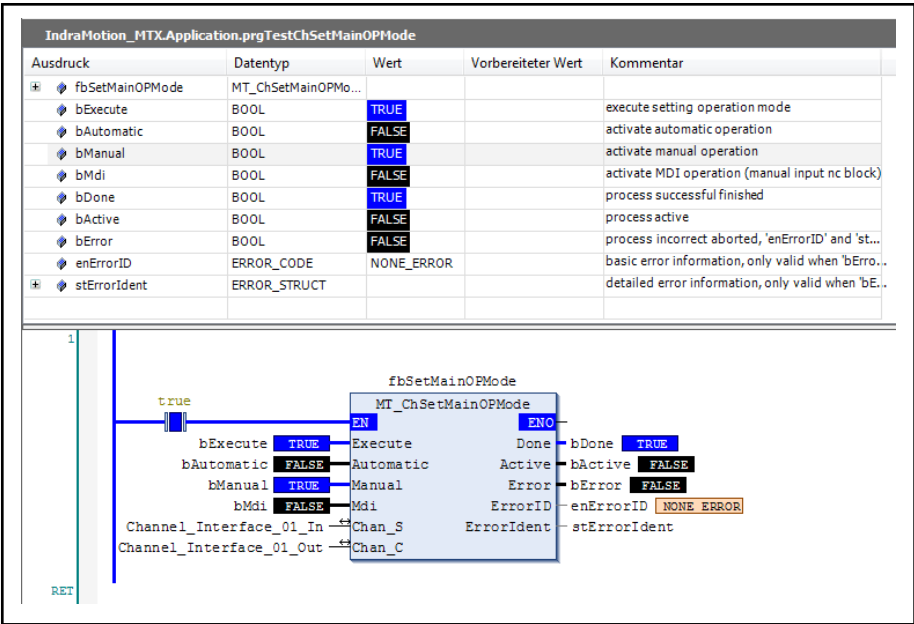


Abb. 14-57: Implementation MT_ChSetMainOPMode, Betriebsart ändern erfolgreich

14.3.11 MT_ChSetOPMode, Ändern der NC-Betriebsarten

Kurzbeschreibung	Funktionsbaustein zum Ändern der NC-Haupt- und Unterbetriebsarten.
Zuordnung Zielsystem/Bibliothek	
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-38: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

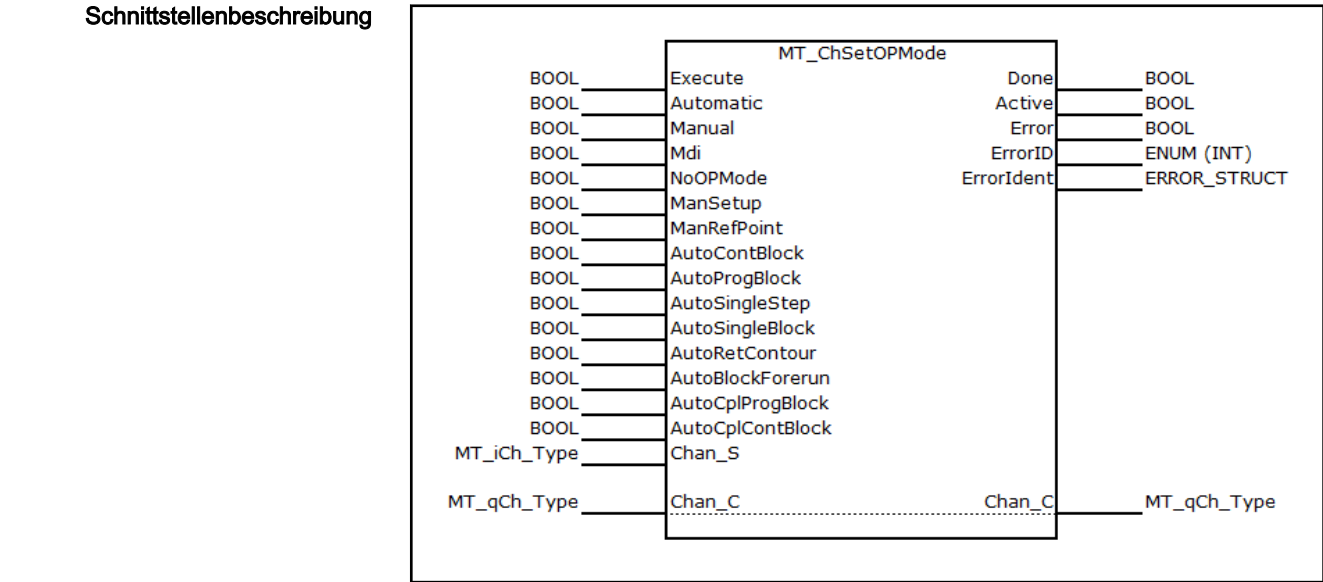


Abb. 14-58: Funktionsbaustein MT_ChSetOPMode

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Setzen der Betriebsart ausführen
	Automatic	BOOL	Hauptbetriebsart: Automatikbetrieb
	Manual	BOOL	Hauptbetriebsart: Handbetrieb
	Mdi	BOOL	Hauptbetriebsart: MDI - Handeingabe NC-Satz
	NoOPMode	BOOL	Keine Betriebsart
	ManSetup	BOOL	Unterbetriebsart: Einrichten manuell Tippbetrieb
	ManRefPoint	BOOL	Unterbetriebsart: Einrichten Referenzpunkt anfahren
	ManWcsSetup	BOOL	Unterbetriebsart: Einrichten manuell Werkstückkoordinaten
	AutoContBlock	BOOL	Unterbetriebsart: Folgesatz
	AutoProgBlock	BOOL	Unterbetriebsart: Programmsatz
	AutoSingleStep	BOOL	Unterbetriebsart: Einzelschritt
	AutoSingleBlock	BOOL	Unterbetriebsart: Einzelsatz
	AutoRetContour	BOOL	Unterbetriebsart: Wiederanfahren
	AutoCplProgBlock	BOOL	Unterbetriebsart: CPL Debugger (NC-Analyser) Programmsatz
	AutoCplContBlock	BOOL	Unterbetriebsart: CPL Debugger Folgesatz
	Chan_S	MT_iCh_Type	Kanal-Interface: Kanal-Status-Signale
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig, wenn "Error" = TRUE)
VAR_IN_OUT	Chan_C	MT_qCh_Type	Kanal-Interface: Kanal-Steuer-Signale

Tab. 14-39: Ein-/Ausgänge Funktionsbaustein MT_ChSetOPMode

Signal-Zeit-Diagramm

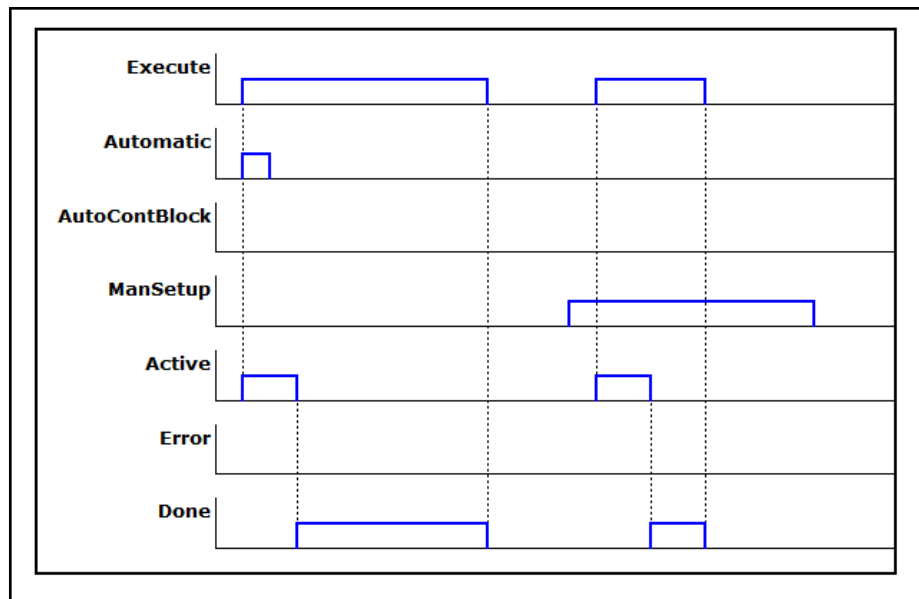


Abb. 14-59: Signal-Zeit-Diagramm MT_ChSetOPMode

Links: Setzen Hauptbetriebsart Automatikbetrieb erfolgreich

Rechts: Setzen Unterbetriebsart Einrichten manuell Tippbetrieb erfolgreich

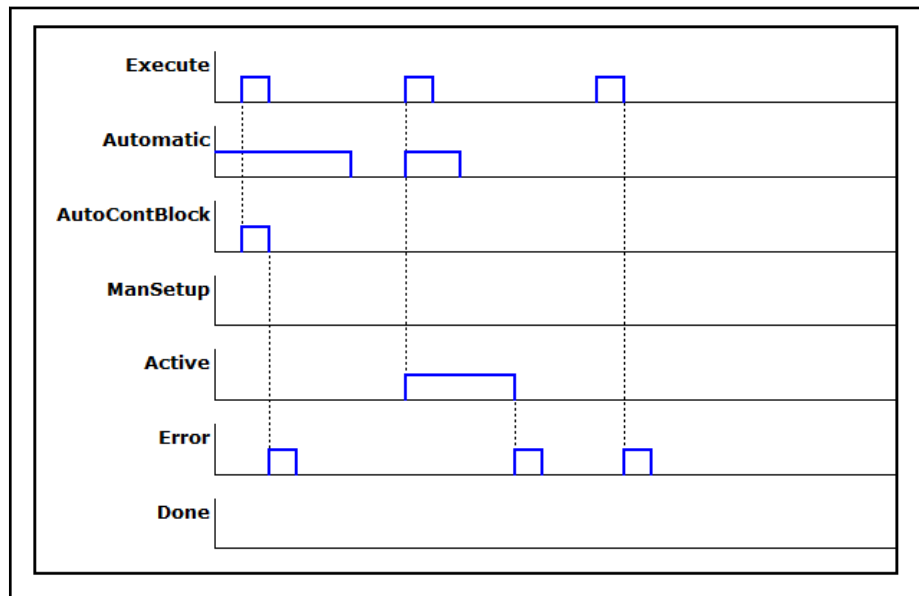


Abb. 14-60: Signal-Zeit-Diagramm MT_ChSetOPMode

Fehler links: Mehr als eine Betriebsart vorgewählt

Fehler Mitte: Timeout (500 ms), keine Rückmeldung, dass Betriebsart gesetzt wurde

Fehler rechts: Es wurde keine Betriebsart vorgewählt

Funktionsbeschreibung

Die MTX verwendet für jeden Kanal drei Hauptbetriebsarten und elf Unterbetriebsarten. Alle Zustände, in welche ein Kanal im Hinblick auf die Betriebsart gebracht werden kann, werden bitcodiert über das Kanal-Interface (MT_qCh_Type) durch die Signale "qCh_OpModeSel_00 .. 03" von der SPS an die NC übertragen. Es kann immer nur eine Hauptbetriebsart mit einer zugehörigen Unterbetriebsart (wenn vorhanden) aktiv sein. Zudem gibt es den gültigen Zustand, dass keine Betriebsart gewählt ist. Die folgende Tabelle

zeigt alle Hauptbetriebsarten mit der Zuordnung der Unterbetriebsarten und den entsprechenden FB-Eingangssignalen.

Haupt- und Unterbetriebsarten

Hauptbetriebsart	Unterbetriebsart	FB Eingangssignal
Automatikbetrieb	Wie vorher	Automatic
Handbetrieb	Wie vorher	Manual
MDI	Keine	Mdi
Keine	Keine	NoOPMode
Automatikbetrieb	Folgesatz	AutoContBlock
Automatikbetrieb	Programmsatz	AutoProgBlock
Automatikbetrieb	Einzelsschritt	AutoSingleStep
Automatikbetrieb	Einzelssatz	AutoSingleBlock
Automatikbetrieb	Wiederanfahren	AutoContBlock
Automatikbetrieb	Satzvorlauf	AutoBlockForerun
Automatikbetrieb	CPL Debugger Programmsatz	AutoCplProgBlock
Automatikbetrieb	CPL Debugger Folgesatz	AutoCplContBlock
Handbetrieb	Einrichten manuell Tippbetrieb	ManSetup
Handbetrieb	Einrichten Referenzpunkt anfahren	ManRefPoint
Handbetrieb	Einrichten manuell Werkstückkoordinaten	ManWcsSetup

Tab. 14-40: Übersicht der Haupt- und Unterbetriebsarten mit den zugehörigen Interface-Signalen

Mit dem Funktionsbaustein "MT_ChSetOpMode" kann man Haupt- und Unterbetriebsarten schalten. Bei einem Wechsel der Hauptbetriebsart (z. B. Automatik nach Handbetrieb) wird die gerade aktive Unterbetriebsart gesichert (z. B. "Folgesatz"). Schaltet man später zurück in diese Hauptbetriebsart wird die zuletzt aktive Unterbetriebsart wieder gesetzt. Wird nur die Unterbetriebsart innerhalb einer Hauptbetriebsart gewechselt (z. B. "Folgesatz" nach "Einzelssatz") bleibt die Hauptbetriebsart gleich. Wechselt man dagegen die Unterbetriebsart außerhalb der Hauptbetriebsart (z. B. "Folgesatz" nach "Einrichten manuell Tippen"), so wird automatisch die Hauptbetriebsart mit gewechselt.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren die Betriebsart genau einmal ändert.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein und sichert den Zustand der Eingänge zu diesem Zeitpunkt. Lediglich die Eingänge "Chan_S" und "Chan_C" werden zyklisch übernommen, solange der Baustein aktiv ist ("Active"==TRUE). Mit dem Signalwechsel muss die gewünschte Haupt- oder Unterbetriebsart vorgewählt sein

(z. B. "AutoContBlock"). Sind beim Setzen von "Execute" keine oder mehrere Betriebsarten vorgewählt, so geht der Baustein in Fehler ("Error" wird TRUE). Sobald der Ausgang "Done" TRUE wird, ist das Setzen der gewünschten Betriebsart erfolgreich abgeschlossen. Konnte die Betriebsart nicht geändert werden, wird dies durch den Ausgang "Error" angezeigt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einem erneuten Wechsel der Betriebsart, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Solange einer der Ausgänge "Done", "Active" oder "Error" TRUE ist, haben Signaländerungen an den Eingängen zur Vorwahl der Betriebsart (z. B. "AutoContBlock") keine Auswirkungen. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.



Beim Wechsel der Hauptbetriebsart wird immer Grundstellung des Kanals ausgeführt.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_ChSetOPMode"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010001	Ungültige Eingangsbelegung. Es wurde entweder keine oder mehr als eine Betriebsart vorgewählt
0x00040201	Das Ändern der Interface-Signale "Chan_C.qCh_OpModeSel_00..03" ist fehlgeschlagen. Eventuell wurden sie außerhalb des FB überschrieben
0x00020201	Die Rückmeldung über die Änderung der Betriebsart über die Interface-Signale "Chan_S.iCh_OpMode_00..03" ist innerhalb von 500 ms nicht erfolgt. Eventuell ist die Schnittstelle für einen anderen Kanal am FB angeschlossen.
0x 00050001	Interner Fehler
0x 00050002	Interner Fehler

Tab. 14-41: Fehler von "MT_SetOPMode", aus "ErrorIdent.Additional1", gültig wenn "bError"==TRUE

Implementationsbeispiel

In diesem Beispiel wird die Betriebsart auf die Unterbetriebsart Automatik-Folgesatz geändert.


```

1 PROGRAM prgTestChSetOPMode
2 VAR
3   fbSetPMode      :MT_ChSetOPMode;
4   bExecute        :BOOL:=FALSE;      (*execute setting operation mode*)
5
6   (*main operation mode*)
7   bAutomatic       :BOOL:=FALSE;      (*activate automatic operation *)
8   bManual          :BOOL:=FALSE;      (*activate manual operation *)
9   bMdi             :BOOL:=FALSE;      (*activate MDI operation (manual input nc block)*)
10
11   (*sub operation mode*)
12   bNoOPMode        :BOOL:=FALSE;      (*no operation mode active*)
13
14   (*manual operation: sub operation modes*)
15   bManSetup        :BOOL:=FALSE;      (*manual setup, jogging operation active*)
16   bManRefPoint     :BOOL:=FALSE;      (*manual basingpoint*)
17   bManWcsSetup     :BOOL:=FALSE;      (*manual setup, jogging operation with tool coordinate system active*)
18
19   (*automatic operation: sub operation modes*)
20   bAutoContBlock   :BOOL:=FALSE;      (*automatic, continuous block active*)
21   bAutoProgBlock   :BOOL:=FALSE;      (*automatic, program block active*)
22   bAutoSingleStep  :BOOL:=FALSE;      (*automatic, single step active*)
23   bAutoSingleBlock :BOOL:=FALSE;      (*automatic, single block active*)
24   bAutoRetContour  :BOOL:=FALSE;      (*automatic, return to contour active*)
25   bAutoBlockForerun :BOOL:=FALSE;      (*automatic, block forerun*)
26   bAutoCplProgBlock :BOOL:=FALSE;      (*automatic, CPL debugger program block active*)
27   bAutoCplContBlock :BOOL:=FALSE;      (*automatic, CPL debugger continuous block active*)
28
29   bDone            :BOOL:=FALSE;      (*process successful finished*)
30   bActive          :BOOL:=FALSE;      (*process active*)
31   bError           :BOOL:=FALSE;      (*process incorrect aborted, 'enErrorID' and 'stErrorIdent' valid*)
32   enErrorID        :ERROR_CODE;      (*basic error information, only valid when 'bError'==TRUE*)
33   stErrorIdent     :ERROR_STRUCT;     (*detailed error information, only valid when 'bError'==TRUE*)
34 END_VAR

```

Abb. 14-61: Deklaration MT_ChSetOPMode

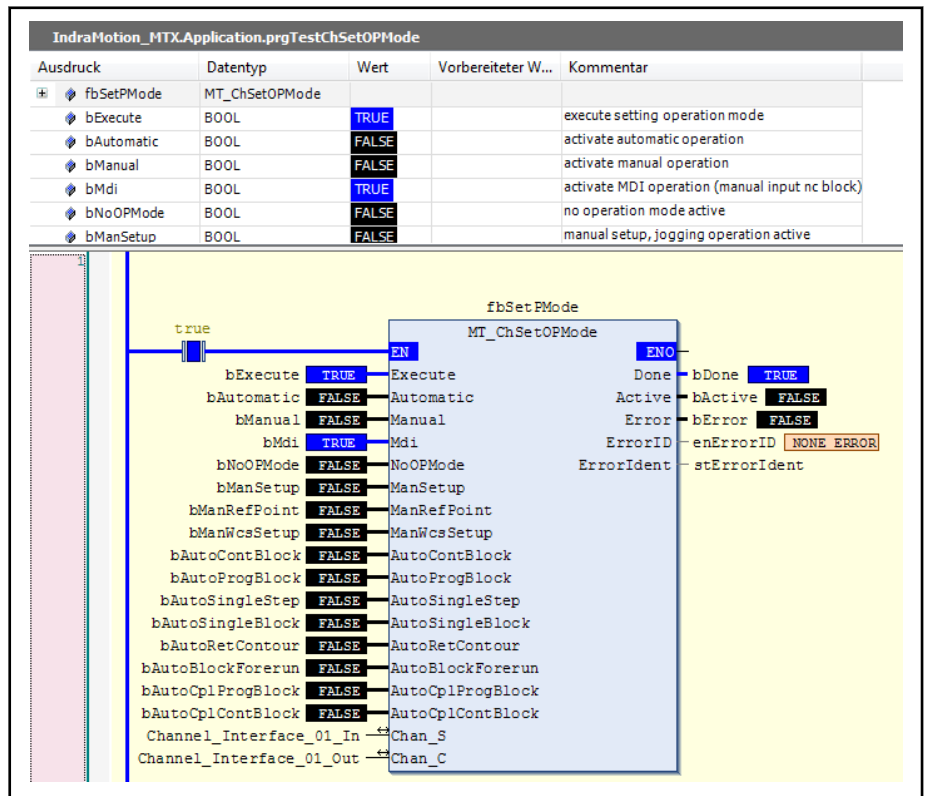


Abb. 14-62: Implementation MT_ChSetOPMode, Betriebsart ändern erfolgreich

14.3.12 MT_ChOPMode, Ändern der NC-Hauptbetriebsarten

Kurzbeschreibung

Mit diesem Funktionsbaustein kann die Haupt- und Unterbetriebsart eines Kanals geändert werden. Außerdem wird zyklisch die aktive Haupt- und Unterbetriebsart eines Kanals ermittelt.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-42: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

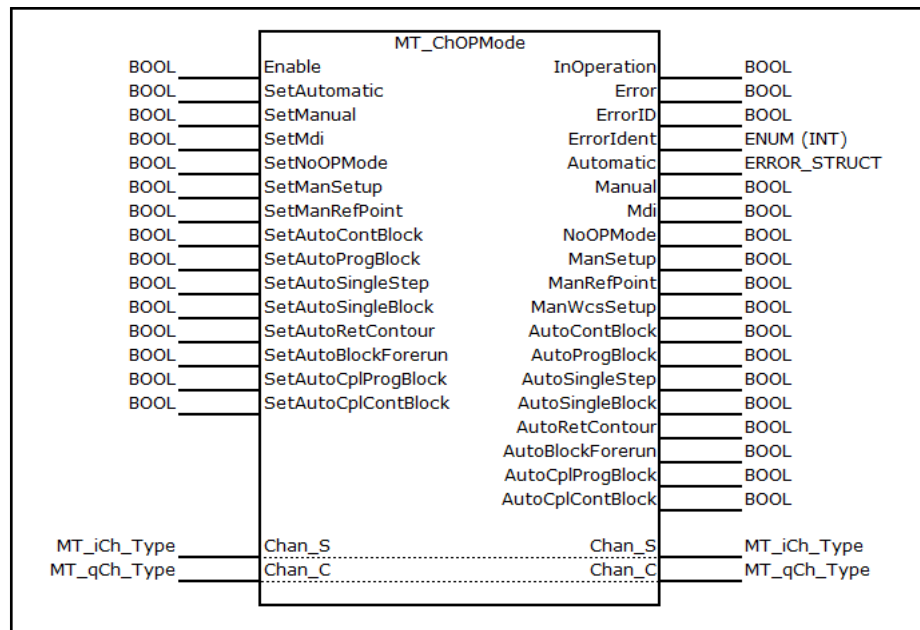


Abb. 14-63: Funktionsbaustein MT_ChOPMode

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Aktiviert den Funktionsbaustein
	SetAutomatic	BOOL	Setze Hauptbetriebsart: Automatikbetrieb
	SetManual	BOOL	Setze Hauptbetriebsart: Handbetrieb
	SetMdi	BOOL	Setze Hauptbetriebsart: MDI - Handeingabe NC-Satz
	SetNoOPMode	BOOL	Setze keine Betriebsart
	SetManSetup	BOOL	Setze Unterbetriebsart: Einrichten manuell Tippbetrieb
	SetManRefPoint	BOOL	Setze Unterbetriebsart: Einrichten Referenzpunkt anfahren
	SetManWcsSetup	BOOL	Setze Unterbetriebsart: Einrichten manuell Werkstückkoordinaten
	SetAutoContBlock	BOOL	Setze Unterbetriebsart: Folgesatz
	SetAutoProgBlock	BOOL	Setze Unterbetriebsart: Programmsatz
	SetAutoSingleStep	BOOL	Setze Unterbetriebsart: Einzelschritt
	SetAutoSingleBlock	BOOL	Setze Unterbetriebsart: Einzelsatz
	SetAutoRetContour	BOOL	Setze Unterbetriebsart: Wiederanfahren
	SetAutoCplProgBlock	BOOL	Setze Unterbetriebsart: CPL Debugger (NC-Analyser) Programmsatz
	SetAutoCplContBlock	BOOL	Setze Unterbetriebsart: CPL Debugger Folgesatz

	Name	Typ	Beschreibung
VAR_OUTPUT	InOperation	BOOL	Bearbeitung erfolgreich abgeschlossen
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "Error" = TRUE)
	Automatic	BOOL	Hauptbetriebsart: Automatikbetrieb aktiv
	Manual	BOOL	Hauptbetriebsart: Handbetrieb aktiv
	Mdi	BOOL	Hauptbetriebsart: MDI - Handeingabe NC-Satz aktiv
	NoOPMode	BOOL	Keine Betriebsart aktiv
	ManSetup	BOOL	Unterbetriebsart: Einrichten manuell Tippbetrieb aktiv
	ManRefPoint	BOOL	Unterbetriebsart: Einrichten Referenzpunkt anfahren aktiv
	ManWcsSetup	BOOL	Unterbetriebsart: Einrichten manuell Werkstückkoordinaten aktiv
	AutoContBlock	BOOL	Unterbetriebsart: Folgesatz aktiv
	AutoProgBlock	BOOL	Unterbetriebsart: Programmsatz aktiv
	AutoSingleStep	BOOL	Unterbetriebsart: Einzelschritt aktiv
	AutoSingleBlock	BOOL	Unterbetriebsart: Einzelsatz aktiv
	AutoRetContour	BOOL	Unterbetriebsart: Wiederanfahren aktiv
	AutoBlockForerun	BOOL	Unterbetriebsart: Satzvorlauf aktiv
	AutoCplProgBlock	BOOL	Unterbetriebsart: CPL Debugger Programmsatz aktiv
	AutoCplContBlock	BOOL	Unterbetriebsart: CPL Debugger Folgesatz aktiv
VAR_IN_OUT	Chan_S	MT_iCh_Type	Kanal-Interface: Kanal-Status-Signale
	Chan_C	MT_qCh_Type	Kanal-Interface: Kanal-Steuer-Signale

Tab. 14-43: Ein-/Ausgänge Funktionsbaustein MT_ChOPMode

Signal-Zeit-Diagramm

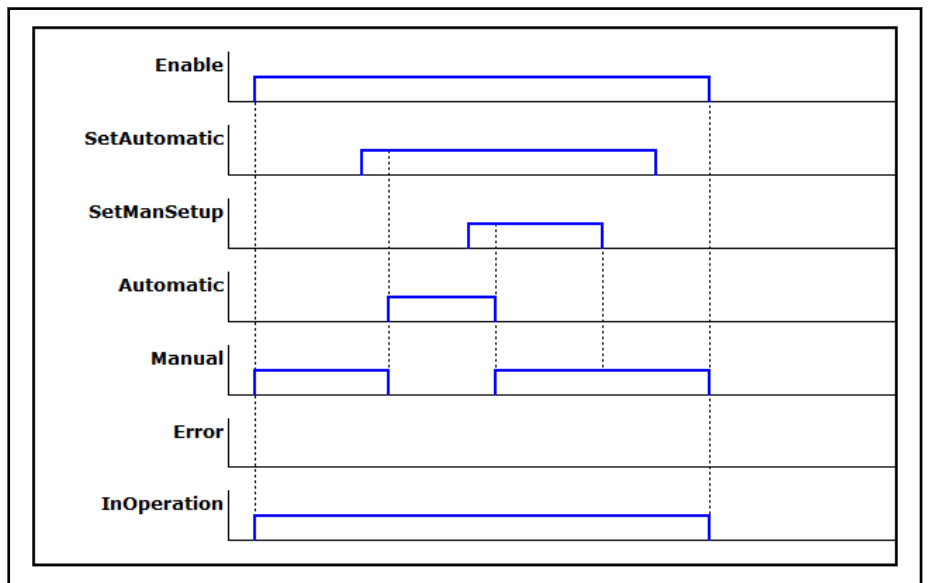


Abb. 14-64: Manuel aktiv, Ändern in Automatic, Ändern in Manual

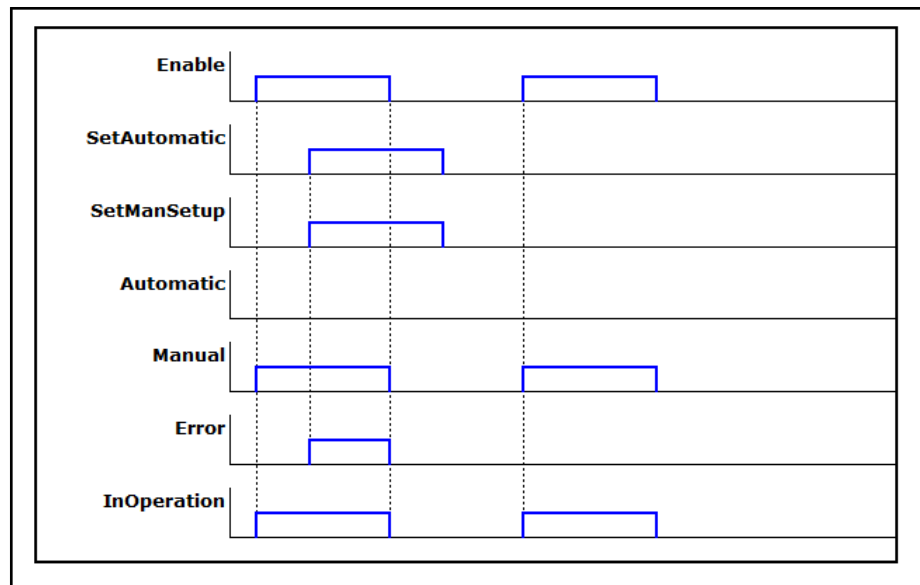


Abb. 14-65: Fehler: Ändern auf 2 Betriebsarten gleichzeitig zeitgleich, Rücksetzen durch Enable=FALSE

Funktionsbeschreibung

Mit diesem Baustein kann die aktive Haupt- und Unterbetriebsart eines Kanals ermittelt und geändert werden.

Die MTX verwendet für jeden Kanal drei Hauptbetriebsarten und elf Unterbetriebsarten. Das Ändern der Kanalbetriebsart erfolgt bitcodiert über das Kanal-Interface (MT_qCh_Type) durch die Signale "qCh_OpModeSel_00 ... 03" von der SPS an die NC. Das Auslesen der aktiven Kanalbetriebsart erfolgt bitcodiert über das Kanal-Interface (MT_iCh_Type) durch die Signale "iCh_OpMode_00 .. 03" von der NC an die SPS. Es kann immer nur eine Hauptbetriebsart mit einer zugehörigen Unterbetriebsart (wenn vorhanden) aktiv sein. Zudem gibt es den gültigen Zustand, dass keine Betriebsart gewählt ist. Bei einem Wechsel der Hauptbetriebsart (z. B. Automatik nach Handbetrieb) wird die gerade aktive Unterbetriebsart gesichert (z. B. "Folgesatz"). Schaltet man später zurück in diese Hauptbetriebsart, wird die zuletzt aktive Unterbetriebsart wieder gesetzt.



Dieser Funktionsbaustein arbeitet zustandsgesteuert. Das bedeutet, dass nach dem Aktivieren das Ändern der Betriebsart immer möglich ist.

Solange der Eingang "Enable" TRUE ist, ist der Funktionsbaustein aktiv. Dies wird über den Ausgang "InOperation" angezeigt. Nur wenn "InOperation" TRUE ist, sind auch die Betriebsartenausgänge gültig und ein Ändern der Betriebsart ist erlaubt. Ein Fehler wird durch den Ausgang "Error" angezeigt. Wenn "Error" TRUE ist, findet man einen gültigen Fehlercode am Ausgang "ErrorIdent". Zum Löschen eines Fehlers muss "Enable" auf FALSE gesetzt werden. Alle Eingänge werden bei jedem Bausteinaufruf übernommen. Somit wird auch die aktuelle Betriebsart mit jedem Bausteinaufruf neu aus den Interface-Signalen "iCh_OpMode_00 .. 03" ermittelt. Das Ändern einer Betriebsart erfolgt durch Signalwechsel (0-->1) am entsprechenden Betriebsarteneingang und wird nach einem SPS-Zyklus wirksam. Sind mehrere Eingänge gesetzt, wird für das Ändern der Betriebsart der Eingang verwendet, an dem zuletzt eine Signaländerung erkannt wurde. Die Ausgänge "InOperation" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.



Beim Wechsel der Hauptbetriebsart wird immer Grundstellung des Kanals ausgeführt.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_ChOPMode"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010001	Ungültige Eingangsbelegung. Es wurde entweder keine oder mehr als eine Betriebsart vorgewählt
0x00040201	Das Ändern der Interface-Signale "Chan_C.qCh_OpModeSel_00..03" ist fehlgeschlagen. Eventuell wurden sie außerhalb des FB überschrieben
0x00020201	Die Rückmeldung über die Änderung der Betriebsart über die Interface-Signale "Chan_S.iCh_OpMode_00..03" ist innerhalb von 500 ms nicht erfolgt. Eventuell ist die Schnittstelle für einen anderen Kanal am FB angeschlossen.
0x 00050001	Interner Fehler
0x 00050002	Interner Fehler

Tab. 14-44: Fehler von **MT_ChOPMode**, aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

In diesem Beispiel wird die Betriebsart auf die Unterbetriebsart Automatik-Folgesatz geändert.

```

1  PROGRAM prgTestChOPMode
2  VAR
3      bEnable                :BOOL:=TRUE;
4      fbOPMode               :MT_ChOPMode;
5  //Eingänge
6      bSetAutomatic          :BOOL:=FALSE;    (*activate automatic operation *)
7      bSetManual             :BOOL:=FALSE;    (*activate manual operation *)
8      bSetMdi                :BOOL:=FALSE;    (*activate MDI operation (manual input nc block)*)
9  (*sub operation mode*)
10     bSetNoOPMode           :BOOL:=FALSE;    (*no operation mode active*)
11     (*'manual operation': sub operation modes*)
12     bSetManSetup           :BOOL:=FALSE;    (*manual setup, jogging operation active*)
13     bSetManRefPoint        :BOOL:=FALSE;    (*manual basingpoint*)
14     bSetManWcsSetup        :BOOL:=FALSE;    (*manual setup, jogging operation with tool coordinate
15     (*'automatic operation': sub operation modes*)
16     bSetAutoContBlock      :BOOL:=FALSE;    (*automatic, continuous block active*)
17     bSetAutoProgBlock      :BOOL:=FALSE;    (*automatic, program block active*)
18     bSetAutoSingleStep     :BOOL:=FALSE;    (*automatic, single step active*)
19     bSetAutoSingleBlock    :BOOL:=FALSE;    (*automatic, single block active*)
20     bSetAutoRetContour     :BOOL:=FALSE;    (*automatic, return to contour active*)
21     bSetAutoBlockForerun   :BOOL:=FALSE;    (*automatic, block forerun*)
22     bSetAutoCplProgBlock   :BOOL:=FALSE;    (*automatic, CPL debugger program block active*)
23     bSetAutoCplContBlock   :BOOL:=FALSE;    (*automatic, CPL debugger continuous block active*)
24
25 //Ausgänge
26 (*main operation mode*)
27     bAutomatic             :BOOL:=FALSE;    (*automatic operation active*)
28     bManual                :BOOL:=FALSE;    (*manual operation active*)
29     bMdi                   :BOOL:=FALSE;    (*MDI operation active (manual input nc block)*)
30     (*sub operation mode*)
31     bNoOPMode              :BOOL:=FALSE;    (*no operation mode active*)
32     (*'manual operation': sub operation modes*)

```

Abb. 14-66: Deklaration MT_ChOPMode

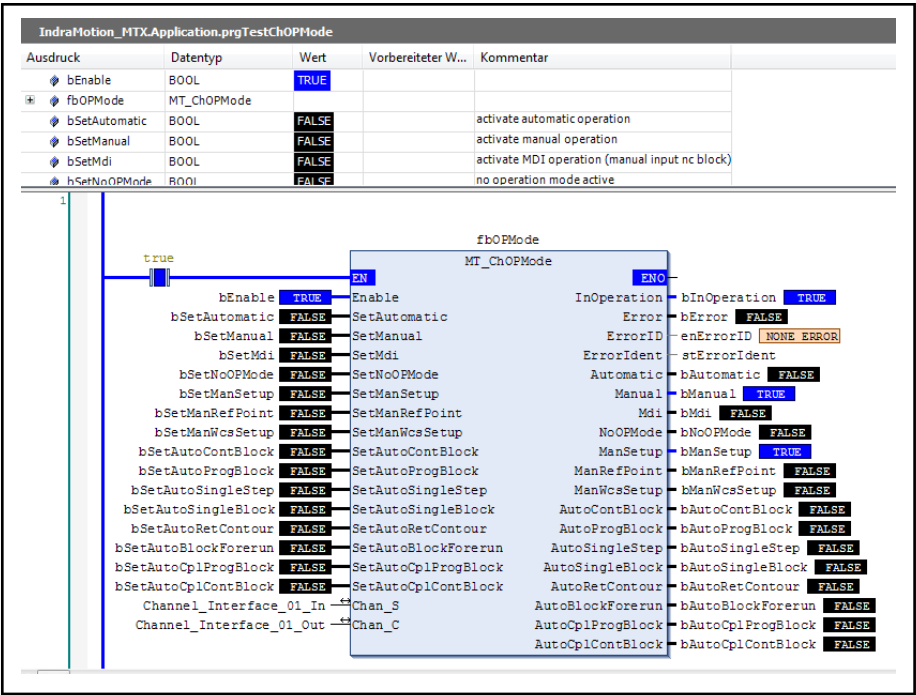


Abb. 14-67: Implementation MT_ChOPMode

14.4 Achsen

14.4.1 Übersicht

Übersicht Achsfunktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung
Allgemein	MT_DrvGetSafeState, Seite 281	Aktiven Sicherheitstechnik-Betriebszustand eines Antriebs ermitteln (auch für Spindel)
	MT_AxGetChIndex, Seite 285	Kanalzuordnung einer Achse prüfen
	MT_AxHandwheelACS, Seite 286	Handrad betreiben für eine Achse im Achs-koordinatensystem
	MT_AxProbeMonitor, Seite 288	Überwachung einer Messtasterachse
Asynchrone Achsen	MT_AxAsyncSetOverride, Seite 308	Vorschub für asynchrone Achse ändern
	MT_AxAsyncOPMode, Seite 305	Ermittelt und Ändert die Betriebsart einer asynchronen Achse (kombiniert MT_AxAsyncGetOpMode und MT_AxAsyncSetOpMode)
	MT_AxAsyncGetOPMode, Seite 301	Aktive Betriebsart einer asynchronen Achse ermitteln
	MT_AxAsyncSetOPMode, Seite 302	Betriebsart einer asynchronen Achse ändern

Tippdis- tanz	MT_AxJogMode, Seite 296	Ermitteln und Ändern der aktiven Tippdis- tanz einer Achse (kombiniert MT_AxGet- JogMode und MT_AxSetJogMode)
	MT_AxGetJogMode, Seite 291	Ermitteln der aktiven Tippdistanz für eine Achse
	MT_AxSetJogMode, Seite 292	Ändern der aktiven Tippdistanz für eine Achse
Leistung & Diagno- se	MT_AxPower, Seite 316	Ermitteln des Antriebsstatus und Setzen/ Rücksetzen der Reglerfreigabe (kombiniert MT_AxSetPowerOn, MT_AxSetPowerOff und MT_AxGetPowerStateExt)
	MT_AxSetPowerOn, Seite 312	Setzt die Reglerfreigabe für eine Achse
	MT_AxSetPowerOff, Seite 314	Rücksetzen der Reglerfreigabe für eine Achse
	MT_AxGetPowerState, Seite 309	Ermittelt den Antriebsstatus für eine Achse
	MT_AxGetPowerStateExt, Seite 311	Ermittelt den Antriebsstatus einer Achse in- klusive Detailinformationen zu Fehler und Warnungen

Tab. 14-45: Übersicht Achsfunktionen

14.4.2 MT_DrvGetSafeState, Aktiven Sicherheitstechnik-Betriebszustand ermitteln

Kurzbeschreibung

Der Funktionsbaustein ermittelt den aktiven Sicherheitstechnik-Betriebszustand eines Antriebes.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-46: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

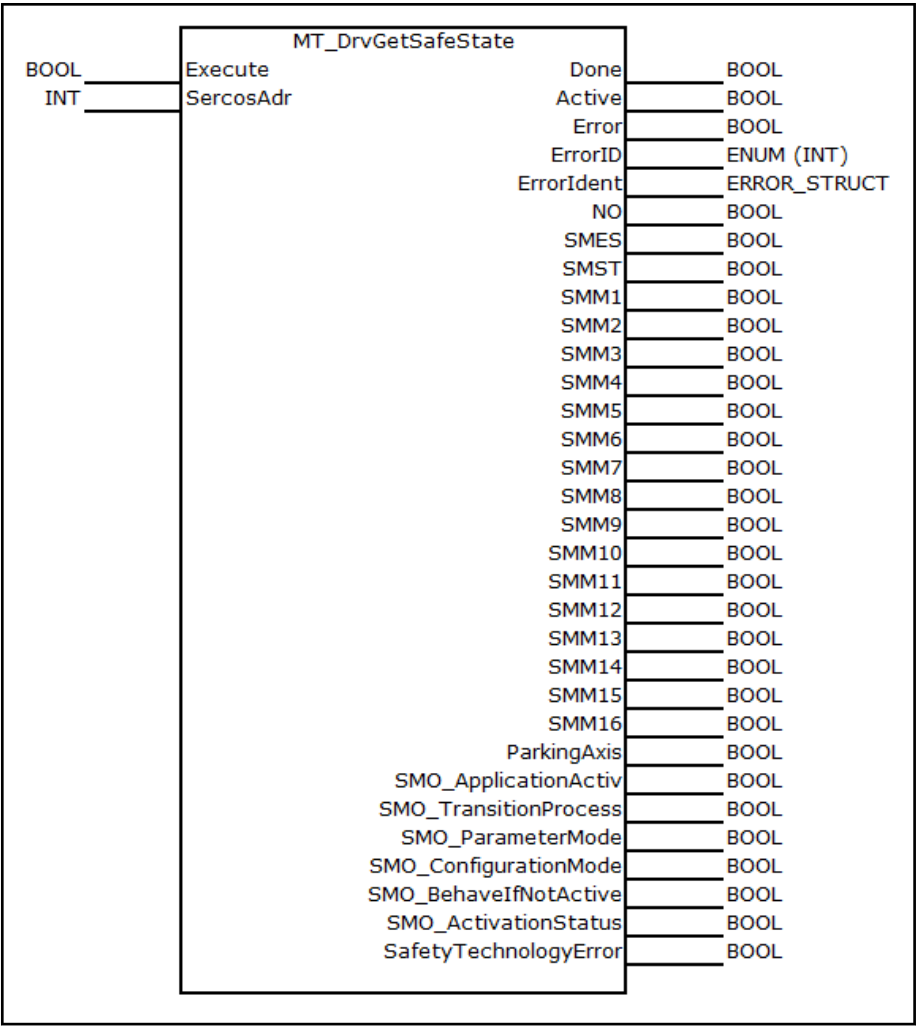


Abb. 14-68: Funktionsbaustein MT_DrvGetSafeState

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Ermitteln des Sicherheitstechnik-Betriebszustand ausführen
	SercosAdr	INT	Sercos-Adresse des Antriebs

	Name	Typ	Beschreibung
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
	NO	BOOL	Normalbetrieb, 1: aktiv, 0: nicht aktiv
	SMES	BOOL	NOT-HALT, 1: aktiv, 0: nicht aktiv
	SMST	BOOL	Sicherer Stillstand, 1: aktiv, 0: nicht aktiv
	SMM1	BOOL	Sichere Bewegung 1, 1: aktiv, 0: nicht aktiv
	SMM2	BOOL	Sichere Bewegung 2, 1: aktiv, 0: nicht aktiv
	SMM3	BOOL	Sichere Bewegung 3, 1: aktiv, 0: nicht aktiv
	SMM4	BOOL	Sichere Bewegung 4, 1: aktiv, 0: nicht aktiv
	SMM5	BOOL	Sichere Bewegung 5, 1: aktiv, 0: nicht aktiv
	SMM6	BOOL	Sichere Bewegung 6, 1: aktiv, 0: nicht aktiv
	SMM7	BOOL	Sichere Bewegung 7, 1: aktiv, 0: nicht aktiv
	SMM8	BOOL	Sichere Bewegung 8, 1: aktiv, 0: nicht aktiv
	SMM9	BOOL	Sichere Bewegung 9, 1: aktiv, 0: nicht aktiv
	SMM10	BOOL	Sichere Bewegung 10, 1: aktiv, 0: nicht aktiv
	SMM11	BOOL	Sichere Bewegung 11, 1: aktiv, 0: nicht aktiv
	SMM12	BOOL	Sichere Bewegung 12, 1: aktiv, 0: nicht aktiv
	SMM13	BOOL	Sichere Bewegung 13, 1: aktiv, 0: nicht aktiv
	SMM14	BOOL	Sichere Bewegung 14, 1: aktiv, 0: nicht aktiv
	SMM15	BOOL	Sichere Bewegung 15, 1: aktiv, 0: nicht aktiv
	SMM16	BOOL	Sichere Bewegung 16, 1: aktiv, 0: nicht aktiv
	ParkingAxis	BOOL	Parkende Achse, 1: aktiv, 0: nicht aktiv
	SMO_ApplicationActiv	BOOL	Applikation aktiv, 1: aktiv, 0: nicht aktiv
	SMO_TransitionProcess	BOOL	SMO Überföhrvorgang, 1: aktiv, 0: nicht aktiv
	SMO_ParameterMode	BOOL	SMO Parametriermodus, 1: aktiv, 0: nicht aktiv
	SMO_Configuration-Mode	BOOL	SMO Konfigurationsmodus, 1: aktiv, 0: nicht aktiv
	SMO_BehaveIfNotActive	BOOL	Verhalten bei nicht aktiver "Safe Motion" (nur relevant wenn Bit 31 =1) 1: funktionale Inbetriebnahme, Endstufe und Bremse(n) werden von der Standardfirmware angesteuert 0: Auslieferungszustand
	SMO_ActivationStatus	BOOL	SMO Aktivierungsstatus 1: "Safe Motion" Funktionalität ist nicht aktiv 0: "Safe Motion" Funktionalität ist aktiv
	SafetyTechnologyError	BOOL	Sicherheitstechnik-Fehler, 1: keine Fehlermeldung 0: Fehlermeldung

Signal-Zeit-Diagramm Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung Der Sercos-Parameter P-0-3231.0.0 stellt binäre Statussignale zur Online-Überwachung des Sicherheitstechnik-Betriebszustands der Safe Motion zur Verfügung. Dieser Funktionsbaustein liest den Parameter und stellt das Ergebnis an den Ausgängen bereit.

Das Lesen des Parameter erfolgt über den Sercos-Servicekanal. Bei mehrmaligem Bausteinaufruf wird der Servicekanal belastet. Empfohlener Abstand zwischen zwei Bausteinaufrufen ist mindestens 500 ms.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren das Auslesen des Sicherheitstechnik-Betriebszustandes genau einmal ausführt.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein und sichert den Zustand der Eingänge zu diesem Zeitpunkt. Sobald der Ausgang "Done" TRUE wird, ist das Auslesen des aktiven Sicherheitstechnik-Betriebszustands erfolgreich abgeschlossen. Andernfalls wird der Ausgang "Error" TRUE. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einem erneuten Ermitteln des Sicherheitstechnik-Betriebszustands, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Solange einer der Ausgänge "Done", "Active" oder "Error" TRUE ist, haben Signaländerungen am Eingang "SercosAdr" keine Auswirkungen. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Der Funktionsbaustein zeigt ausschließlich Fehler des intern genutzten Funktionsbausteines "IL_SIIISvcRead" aus der Bibliothek "RIL_SercosIII", an.

Implementationsbeispiel

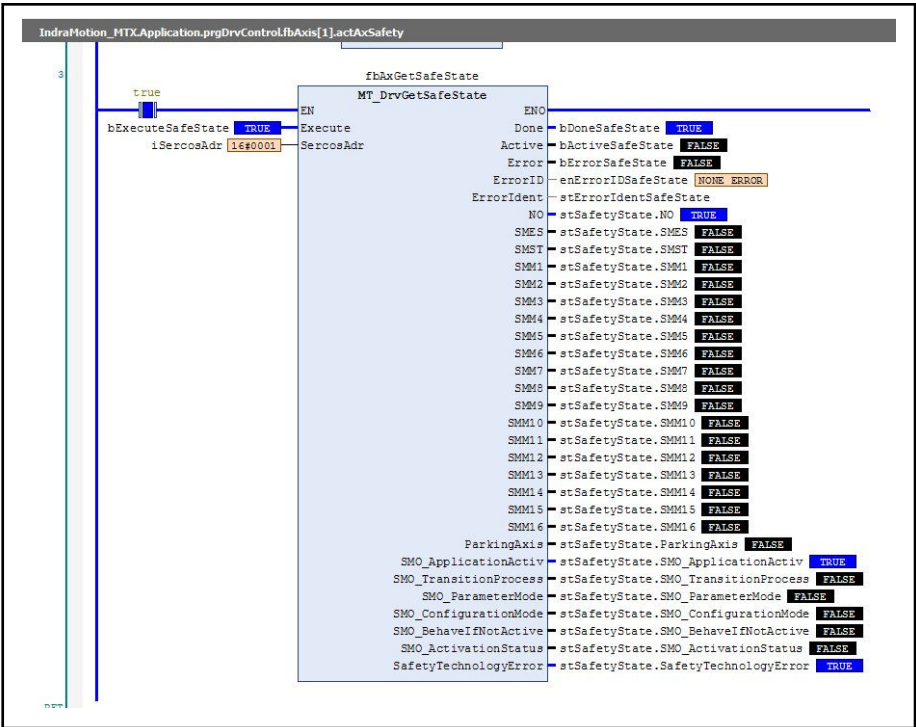


Abb. 14-69: Implementation MT_DrvGetSafeState

14.4.3 MT_AxGetChIndex, Kanalzuordnung der Achse ermitteln

Kurzbeschreibung
Zuordnung Zielsystem/Bibliothek

Funktion zum Ermitteln der Kanalzuordnung einer Achse.

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-48: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

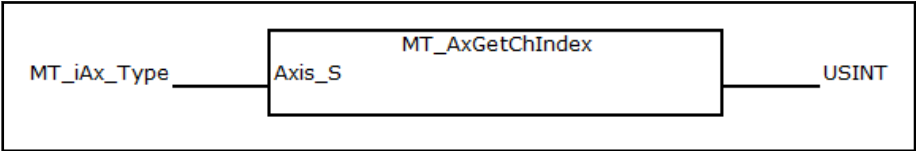


Abb. 14-70: Funktionsbaustein MT_AxGetChIndex

- Signal-Zeit-Diagramm
- Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.
- Funktionsbeschreibung
- Die aktuelle Kanalzugehörigkeit einer Achse wird über das Kanalinterface (MT_iAx_Type) durch die Signale "iAx_ChIndex_00 .. 03" von der NC an die SPS übertragen. Diese Funktion dekodiert diese Information und zeigt die Nummer des Kanals am Ausgang der Funktion an.
- Fehlerbehandlung
- Für diese Funktion ist keine Fehlerbehandlung notwendig.
- Implementationsbeispiel
- Im Beispielprogramm wird die Betriebsart des ersten Kanals ermittelt.

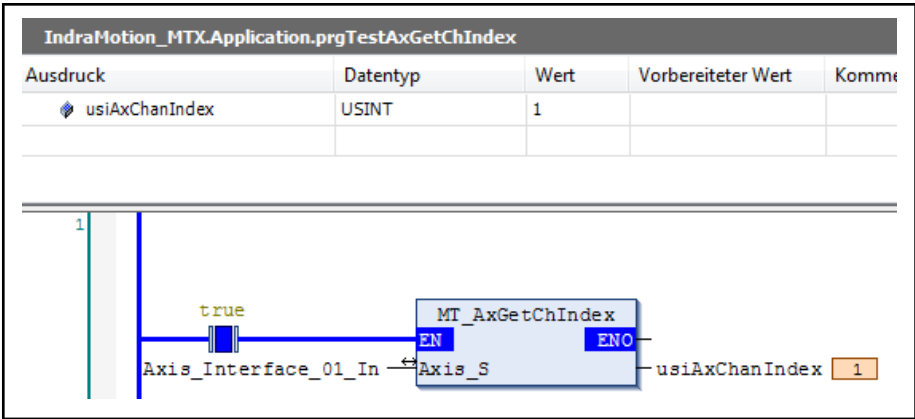


Abb. 14-71: Implementation MT_AxGetChIndex

14.4.4 MT_AxHandwheelAcs, Handrad ändern

Kurzbeschreibung Der Funktionsbaustein vereinfacht das Handling beim Einsatz eines Handrades im Achskoordinatensystem.

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-49: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

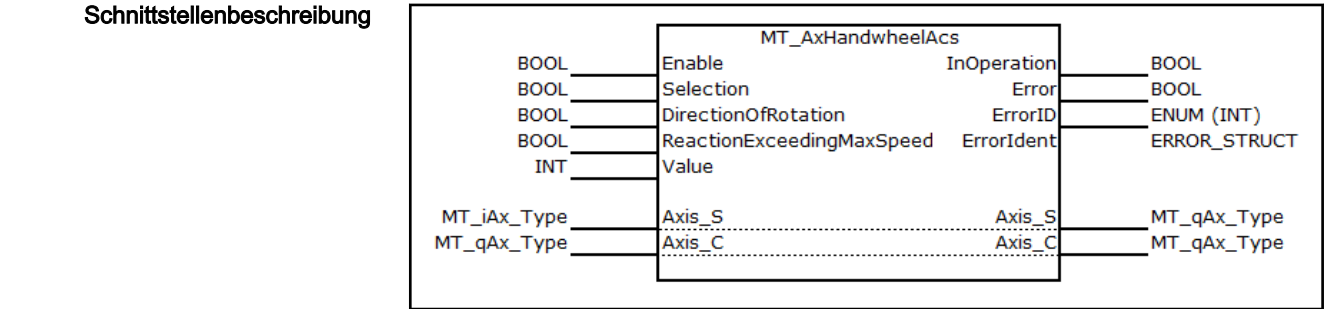


Abb. 14-72: Funktionsbaustein MT_AxHandwheelAcs

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Baustein einschalten
	Selection	BOOL	Auswahl Handrad, FALSE==Handrad1, TRUE==Handrad2
	DirectionOfRotation	BOOL	FALSE==Drehrichtung Handrad und Achse gleich TRUE==Drehrichtung Handrad und Achse entgegengesetzt
	ReactionExceeding-MaxSpeed	BOOL	FALSE==Bei Überschreitung der Maximalgeschwindigkeit wird der Fahrweg entsprechend reduziert. TRUE==Bei Überschreitung der Maximalgeschwindigkeit wird der durch Handradbewegungen vorgegebene Weg auch abgearbeitet. Achtung: Auch nach Beenden der Handradbewegung können noch Achsbewegungen stattfinden
	Value	INT	Handradpulse

	Name	Typ	Beschreibung
VAR_OUTPUT	InOperation	BOOL	Bearbeitung läuft
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "Error"=TRUE)
VAR_IN_OUT	Axis_S	MT_iAx_Type	Achs-Interface: Achs-Status-Signale
	Axis_C	MT_qAx_Type	Achs-Interface: Achs-Steuer-Signale

Tab. 14-50: Ein-/Ausgänge Funktionsbaustein MT_AxHandwheelAcs

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Dieser Funktionsbaustein vereinfacht die Handhabung bei der Inbetriebnahme eines Handrades. Er übernimmt die Beschaltung der notwendigen Interface-Signale "qAx_HandwPosMode", "qAx_HandwSel_00..01" und verarbeitet den Baustein "MT_Handw", so dass keine weiteren Aktionen notwendig sind.

Mit Aktivierung des Bausteins durch das Setzen von "Enable" wird der Status der Eingänge "Selection", "DirectionOfRotation" und "ReactionExceeding-MaxSpeed" einmalig übernommen.

Die Eingänge "Value" sowie die Achs-Interfaces "Axis_S" und "Axis_C" werden bei jedem Aufruf des Bausteins eingelesen und an die NC übergeben.



Einstellung in Maschinenparameter "OPF/Handw/Wh[1..2]/Incr-PerRev" beachten.

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_AxHandwheelAcs"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00060101	Asynchrone Achse ist nicht in Betriebsart Hand
0x00060402	Das gewählte Handrad ist ungültig
0x00030401	NC-interne Kommunikation überlastet. Keine Betriebsmittel zum Versenden des Auftrages vorhanden
0x00030403	Allgemeiner Parameterfehler
0x00040101	Das Ändern mindestens eines der Interface-Signale "qAx_HandwPosMode", "qAx_HandwSel_00..01" ist fehlgeschlagen. Eventuell wurden die Signale außerhalb des Bausteins überschrieben

Tab. 14-51: Fehler von "MT_AxHandwheelAcs", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

Beispielprogramm

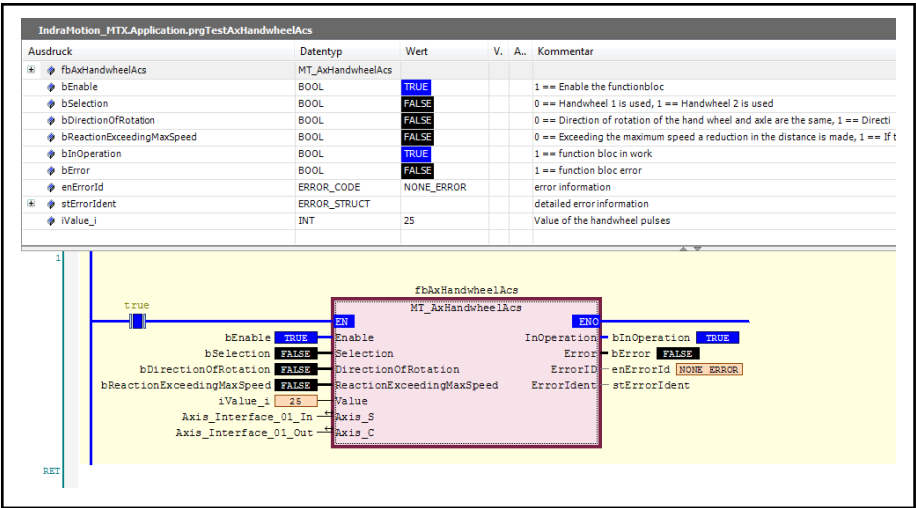


Abb. 14-73: Implementation MT_AxHandwheelAcs

14.4.5 MT_AxProbeMonitor, Messtasterachse überwachen

Kurzbeschreibung

Funktionsbaustein zur Überwachung einer Messtasterachse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-52: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

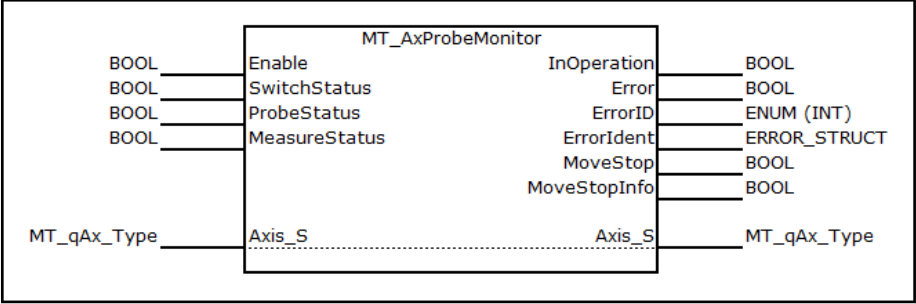


Abb. 14-74: Funktionsbaustein MT_AxProbeMonitor

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Baustein einschalten
	SwitchStatus	BOOL	Einschaltzustand des Messtaster (1==Ein, 0==Aus)
	ProbeStatus	BOOL	Auslenkstatus des Messtaster (1==ausgelenkt, 0==nicht ausgelenkt)
	MeasureStatus	BOOL	Messstatus (1==Messung aktiv (G75), 0==Messung nicht aktiv)

	Name	Typ	Beschreibung
VAR_OUTPUT	InOperation	BOOL	Bearbeitung läuft
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "Error"=TRUE)
	MoveStop	BOOL	Bewegungssperre für diese Achse sollte gesetzt werden
	MoveStopInfo	BOOL	Grund für die Empfehlung der Bewegungssperre
VAR_IN_OUT	Axis_S	MT_iAx_Type	Achs-Interface: Achs-Status-Signale

Tab. 14-53: Ein-/Ausgänge Funktionsbaustein MT_AxProbeMonitor

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Mit diesem Funktionsbaustein kann eine am Messprozess beteiligte Achse überwacht werden. Damit können Achsbewegungen, welche zur Zerstörung des Messtasters führen, minimiert werden.



Das Auslenksignal des Messtasters ist direkt auf den Antrieb und parallel zur SPS verdrahtet. Beim Messen mit G75 bewirkt das Auslenksignal am Antrieb einen sofortigen Stop der Achse. Lenkt der Messtaster ohne G75 aus, reagiert der Antrieb nicht. Das Signal läuft vom Messtaster in die SPS. Von dort wird DriveLock an die NC gesendet und diese leitet das Signal über Sercos an den Antrieb. Diese lange Signallaufzeit hat eine entsprechend lange Reaktionszeit zur Folge. Daher kann ein Stoppen der Achse durch die SPS nicht die Messtasterfunktion im Antrieb ersetzen!

Prinzipiell wird über den Funktionsbausteinausgang "MoveStop" nur eine Empfehlung gegeben, die Achsbewegung über das Interface-Signal "qAx_DrvLock" der entsprechenden Achse zu verhindern. Eine Verknüpfung muss durch den SPS-Programmierer erfolgen. Sobald der FB durch Setzen von "Enable" eingeschaltet ist, werden die Zustände aller Eingänge überwacht. Wenn eine richtungsbezogene Bewegungssperre aktiv ist, wird der Ausgang "InOperation" TRUE. Das Rücksetzen von "Enable" löscht alle im FB gespeicherten Zustände sowie alle Ausgänge.



Bei ausgeschaltetem Messtaster kann der FB dessen tatsächlichen Auslenkstatus nicht ermitteln. Eine Überwachung des Auslenkstatus "ProbeStatus" erfolgt somit nur, wenn der Messtaster eingeschaltet ist "SwitchStatus".

Im Moment des Auslenkens merkt sich der Baustein die zu diesem Zeitpunkt aktive Bewegungsrichtung der Achse und empfiehlt Bewegungssperren in folgenden Situationen:

Bewegungssperre richtungsbezogen:

Bei eingeschaltetem und ausgelenktem Messtaster wird eine Bewegungssperre empfohlen ("MoveStop"), wenn zusätzlich eine Bewegungsankündigung in Auslenkrichtung erfolgt. Bei einer der Auslenkrichtung entgegengesetzten Bewegungsankündigung wird die Bewegungssperre ("MoveStop") gelöscht.

"MoveStopInfo"=2001 od. 2002

Bewegungssperre bei vermutetem Messtasterabbruch:

Bricht ein eingeschalteter Messtaster beim Fahren ohne G75 ab, entspricht dies einem kurzem Auslenken. Dann wird eine Bewegungssperre empfohlen ("MoveStop"), auch wenn der Taster nicht mehr ausgelenkt ist. Dies kann nur durch Löschen des Fahrbefehls wieder aufgehoben werden.

"MoveStopInfo"=1000

Bewegungssperre bei Messen mit ausgeschaltetem Messtaster:

Beim Fahren mit G75 erwartet der NC-Programmierer, dass der Antrieb die Achse stoppt, sobald der Messtaster auslöst. Dies geschieht aber nur, wenn dieser Messtaster auch eingeschaltet ist. Deshalb wird bei nicht eingeschaltetem Messtaster und Fahrbefehl zum Messen (G75) die Bewegungssperre empfohlen. Diese Bewegungssperre kann durch Aufheben des Fahrbefehl oder Einschalten des Messtasters aufgehoben werden.

"iMoveStopInfo"=1003

Bewegungssperre, wenn Messtaster ausgelenkt und abgeschaltet:

Schaltet ein zuvor ausgelenkter Messtaster ab, so wird die zuvor empfohlene Bewegungssperre beibehalten. Diese kann entweder durch Wiedereinschalten des Messtasters oder durch Aufheben des Fahrbefehls zurückgesetzt werden.

"iMoveStopInfo"=1001

MoveStopInfo	Ursache	Behebung
1000	Messtaster ist z. Z. nicht ausgelenkt, wurde aber zuvor beim Fahren ohne G75 ausgelenkt, Abbruch wird vermutet	Fahrbefehl für die Achse aufheben
1001	Messtaster im ausgelenkten Zustand abgeschaltet	Messtaster wieder einschalten oder Fahrbefehl für die Achse aufheben
1003	Fahrbefehl mit Messung (G75) gestartet, aber Messtaster ist nicht eingeschaltet	Messtaster einschalten oder Fahrbefehl für die Achse aufheben
2001	Bewegungsankündigung der Achse in positive Richtung, obwohl der Messtaster in positiver Richtung ausgelenkt ist	Achse in negativer Richtung bewegen
2002	Bewegungsankündigung der Achse in negative Richtung, obwohl der Messtaster in negativer Richtung ausgelenkt ist	Achse in positiver Richtung bewegen

Tab. 14-54: Übersicht der Gründe für die Empfehlung der Bewegungssperre

Fehlerbehandlung

Zur Zeit gibt es für diesen Baustein keine Fehlercodes.

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_AxProbeMonitor"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar

Tab. 14-55: Fehler von "MT_AxProbeMonitor", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

Beispielprogramm

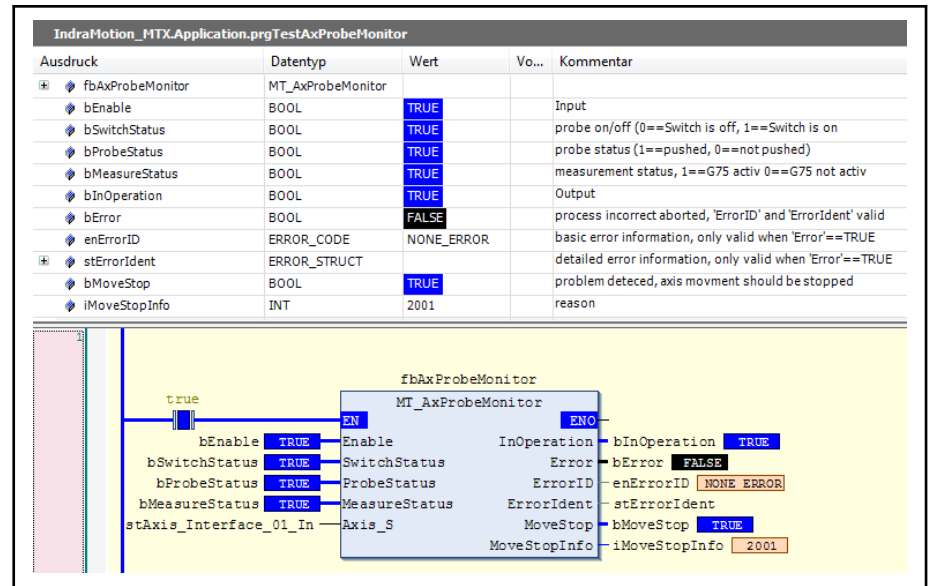


Abb. 14-75: Implementation MT_AxProbeMonitor

14.4.6 MT_AxGetJogMode, Aktiven JogMode ermitteln

Kurzbeschreibung

Funktion zum Ermitteln des aktiven JogMode einer Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-56: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

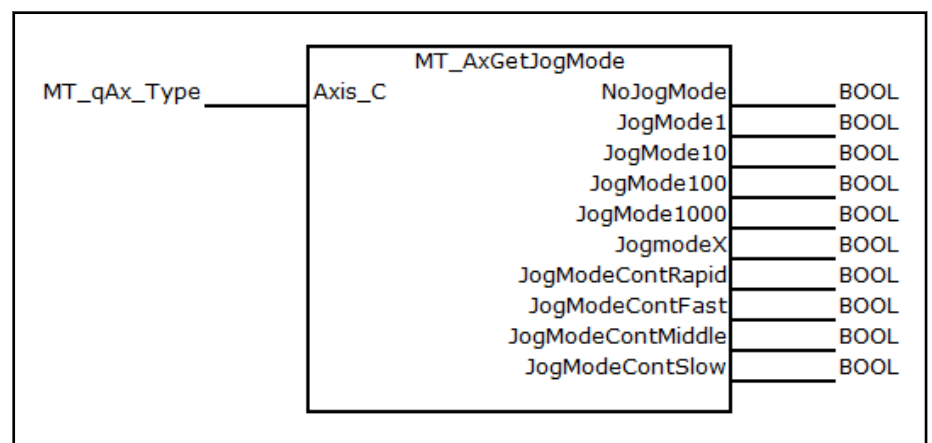


Abb. 14-76: Funktionsbaustein MT_AxGetJogMode

	Name	Typ	Beschreibung
VAR_INPUT	Axis_C	MT_qAX_TYPE	Achs-Interface: Achs-Steuer-Signale
VAR_OUTPUT	NoJogMode	BOOL	Kein JogMode aktiv
	JogMode1	BOOL	1 Inkrement Tippen aktiv
	JogMode10	BOOL	10 Inkremente Tippen aktiv
	JogMode100	BOOL	100 Inkremente Tippen aktiv
	JogMode1000	BOOL	1000 Inkremente Tippen aktiv
	JogModeX	BOOL	x Inkremente Tippen aktiv
	JogModeContRapid	BOOL	Kontinuierliches Tippen im Eilgang
	JogModeContFast	BOOL	Kontinuierliches Tippen schnelle Geschwindigkeit
	JogModeContMiddle	BOOL	Kontinuierliches Tippen mittlere Geschwindigkeit
	JogModeContSlow	BOOL	Kontinuierliches Tippen langsame Geschwindigkeit

Tab. 14-57: Ein-/Ausgänge Funktionsbaustein MT_AxGetJogMode

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Der JogMode wird ausschließlich von der SPS an die NC vorgegeben. Daher ermittelt diese Funktion den zuletzt von der SPS vorgegebenen JogMode aus dem Kanalinterface (MT_qAx_Type) von den Signalen "qAx_Man-Feed_00..03".

Fehlerbehandlung

Für diese Funktion ist keine Fehlerbehandlung notwendig.

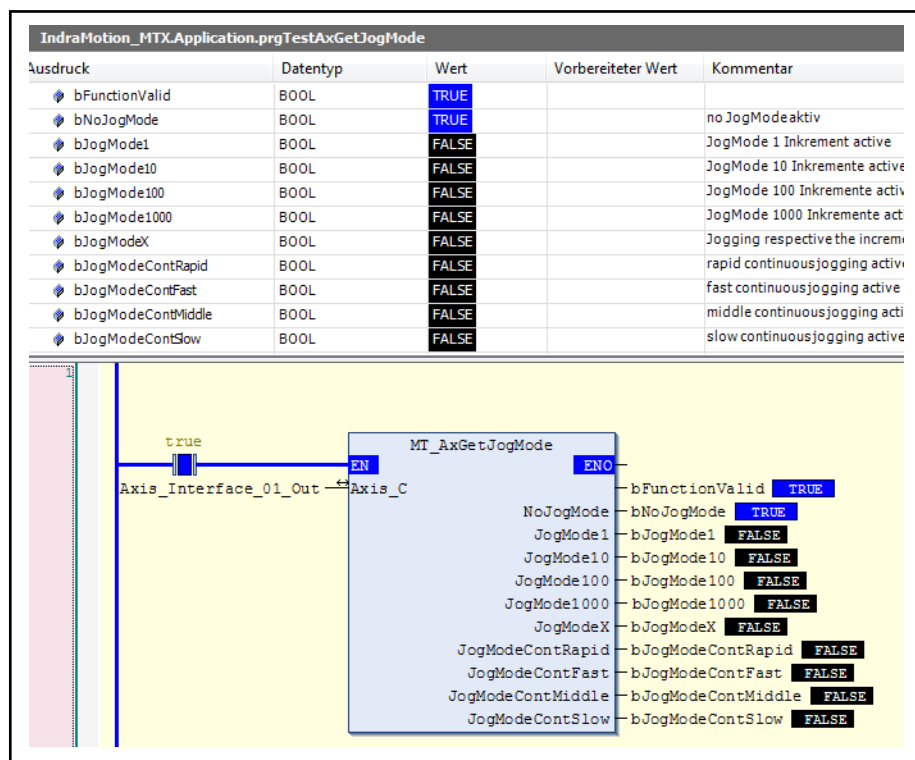
Implementationsbeispiel

Abb. 14-77: Implementation MT_AxGetJogMode

14.4.7 MT_AxSetJogMode, Ändern des JogMode

Kurzbeschreibung Funktionsbaustein zum Ändern des JogMode (Tippdistanz).

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-58: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

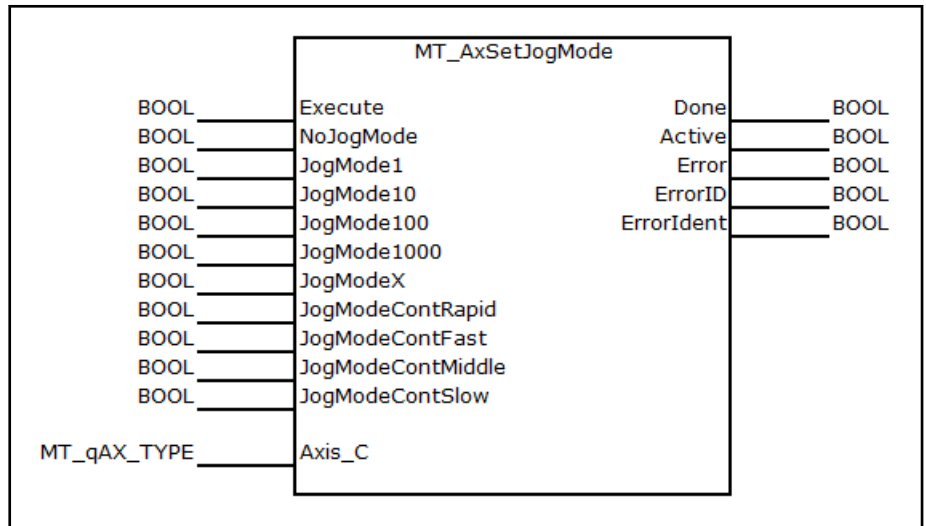


Abb. 14-78: Funktionsbaustein MT_AxSetJogMode

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Setzen des vorgewählten JogMode ausführen
	NoJogMode	BOOL	Kein JogMode
	JogMode1	BOOL	JogMode 1 Inkrement
	JogMode10	BOOL	JogMode 10 Inkremente
	JogMode100	BOOL	JogMode 100 Inkremente
	JogMode1000	BOOL	JogMode 1000 Inkremente
	JogModeX	BOOL	JogMode für Joggen entsprechend der Inkrementvorgaben in der NC
	JogModeContRapid	BOOL	Kontinuierliches Joggen im Eilgang
	JogModeContFast	BOOL	Kontinuierliches Joggen Schnell
	JogModeContMiddle	BOOL	Kontinuierliches Joggen Mittel
	JogModeContSlow	BOOL	Kontinuierliches Joggen Langsam
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
VAR_IN_OUT	Axis_C	MT_qAx_Type	Achs-Interface: Achs-Steuer-Signale

Tab. 14-59: Ein-/Ausgänge Funktionsbaustein MT_AxSetJogMode

Signal-Zeit-Diagramm

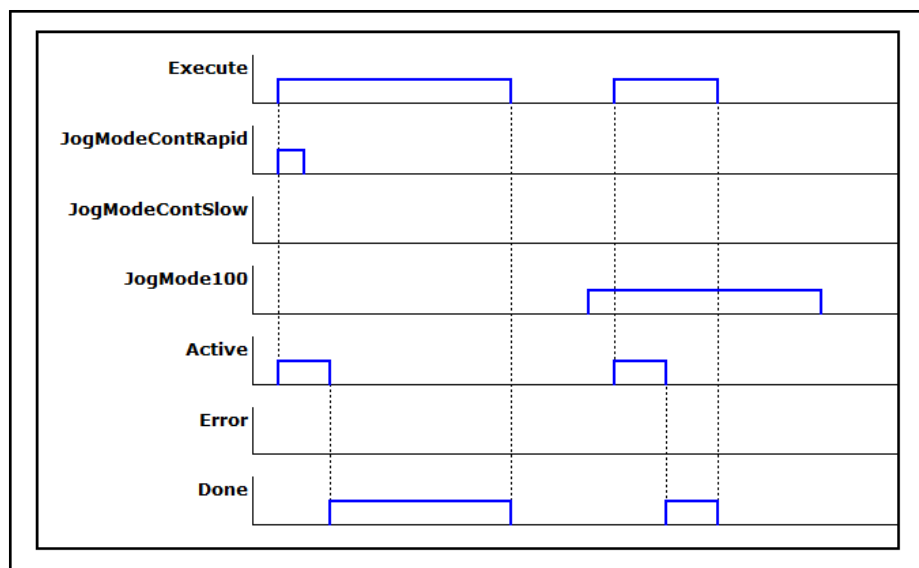


Abb. 14-79: MT_AxSetJogMode Signal-Zeit-Diagramm

Links: Setzen "kontinuierliches Joggen im Eilgang" erfolgreich

Rechts: Setzen "JogMode 100 Inkremente" erfolgreich

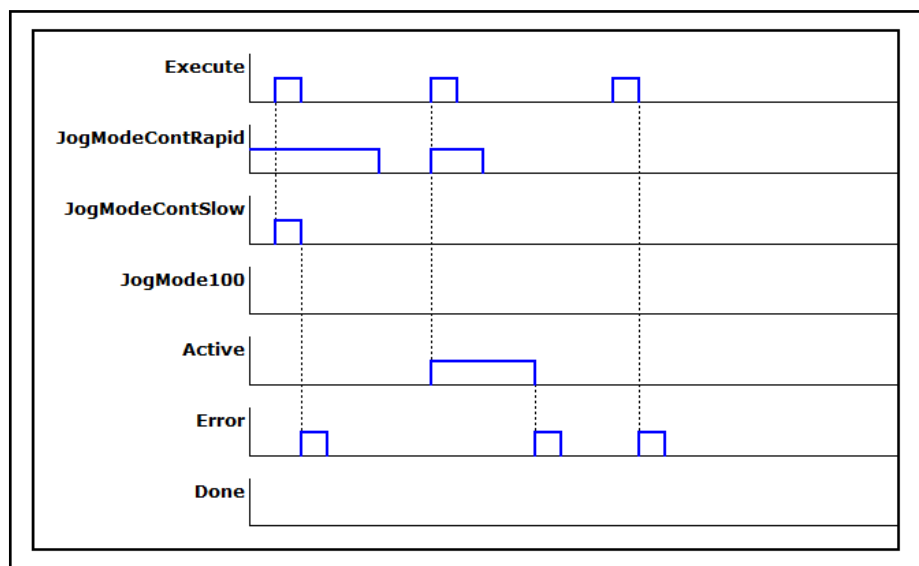


Abb. 14-80: MT_AxSetJogMode Signal-Zeit-Diagramm

Fehler links: Mehr als ein JogMode vorgewählt

Fehler Mitte: Timeout (500ms) keine Rückmeldung, dass JogMode gesetzt wurde

Fehler rechts: Es wurde kein JogMode vorgewählt

Funktionsbeschreibung

Mit der MTX können Achsen kontinuierlich in 4 Geschwindigkeiten oder inkrementell mit 5 verschiedenen Schrittweiten getippt werden (Maschinenparameter "AXSP\Dr[x]\AxFun\Jog\Dist" - "MaCoDa-Nr.: 101500002"). Diese 9 JogModes werden ausschließlich von der SPS über das SPS-Achs-Interface (MT_qAx_Type) durch die Signale "qAx_ManFeed_00..03" codiert vorgewählt. Mit dem Funktionsbaustein "MT_SetJogMode" können diese JogModes für eine Achse aktiviert werden.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren das Ändern des JogMode genau einmal ausführt.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "bExecute" aktiviert den Funktionsbaustein und sichert den Zustand der Eingänge zu diesem Zeitpunkt. Lediglich der Eingang "Axis_C" wird zyklisch übernommen solange der Baustein aktiv ist ("Active"==TRUE). Mit dem Signalwechsel muss der gewünschte JogMode vorgewählt sein (z. B. "JogModeContRapid"). Sind beim Setzen von "Execute" keine oder mehrere JogModes vorgewählt, so geht der Baustein in Fehler ("Error" wird TRUE). Sobald der Ausgang "Done" TRUE wird, ist das Setzen des gewünschten JogMode erfolgreich abgeschlossen. Konnte der JogMode nicht geändert werden, wird dies durch den Ausgang „Error“ angezeigt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einem erneuten Wechsel des JogMode, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Solange einer der Ausgänge "Done", "Active" oder "Error" TRUE ist, haben Signaländerungen an den Eingängen zur Vorwahl des JogMode (z. B. "JogModeContRapid") keine Auswirkungen. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_SetJogMode"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010001	Ungültige Eingangsbelegung. Es wurde entweder kein oder mehr als ein JogMode vorgewählt
0x00040201	Das Ändern der Interface-Signale "qAx_ManFeed_00..03" ist fehlgeschlagen. Eventuell wurden sie außerhalb des FB überschrieben
0x 00050001	Interner Fehler

Tab. 14-60: Fehler von "MT_AxSetJogMode", aus "ErrorIdent.Additional1", wenn "Error"==TRUE

Implementationsbeispiel

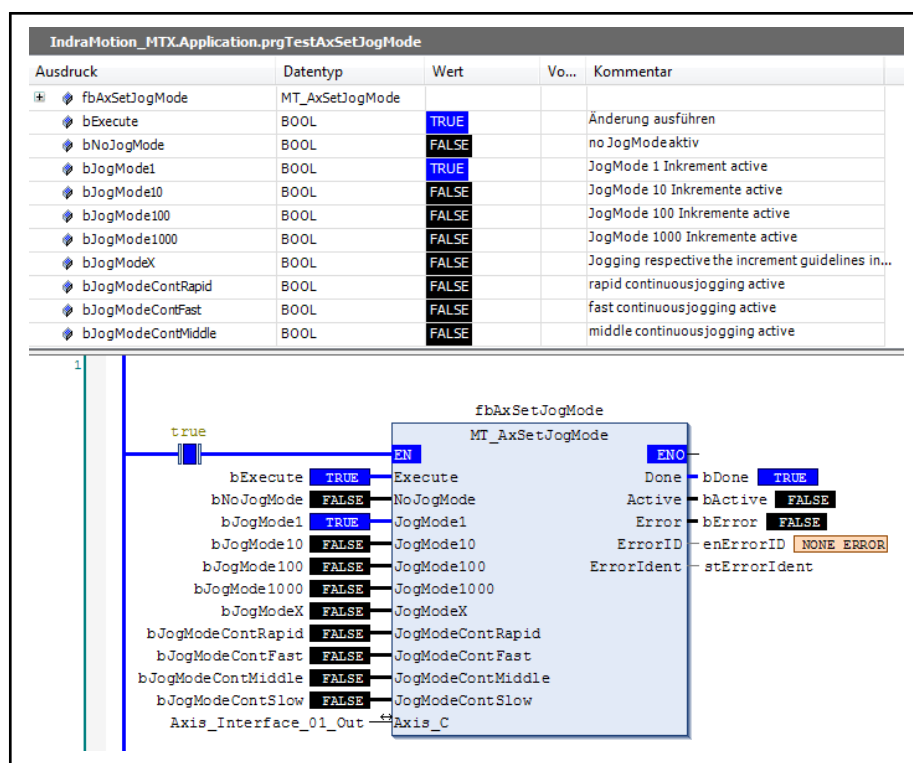


Abb. 14-81: Implementation MT_AxSetJogMode, JogMode ändern erfolgreich

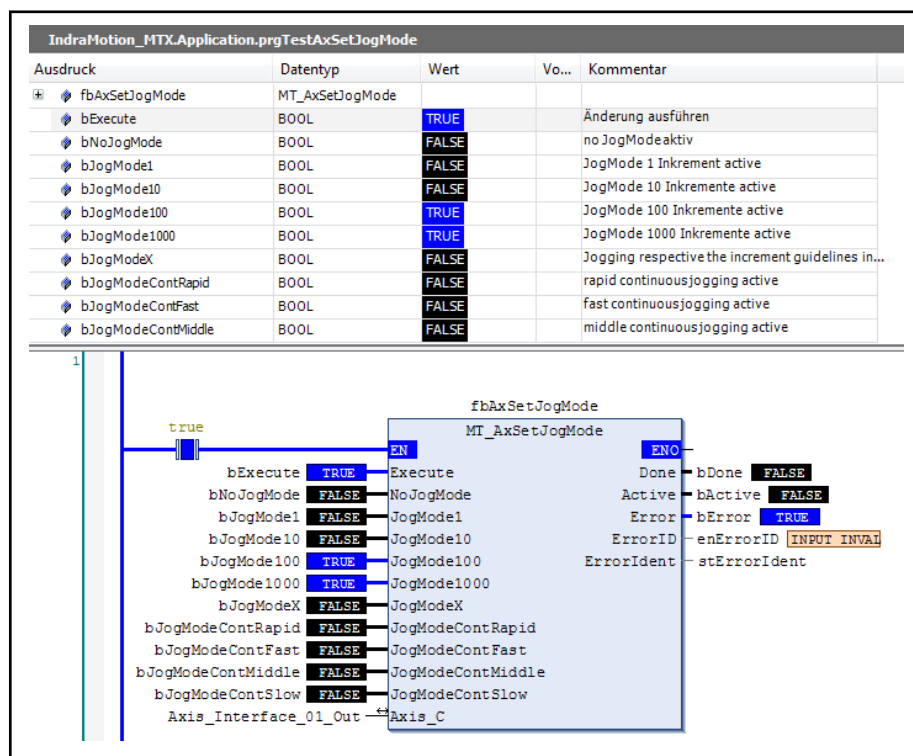


Abb. 14-82: Implementation MT_AxSetJogMode, JogMode ändern fehlgeschlagen (kein JogMode vorgewählt)

14.4.8 MT_AxJogMode, Ändern und Ermitteln der aktiven Tippdistanz

Kurzbeschreibung Funktionsbaustein zum Ermitteln des aktuell aktiven JogMode (Tippdistanz), sowie zum Setzen eines neuen JogMode.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-61: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

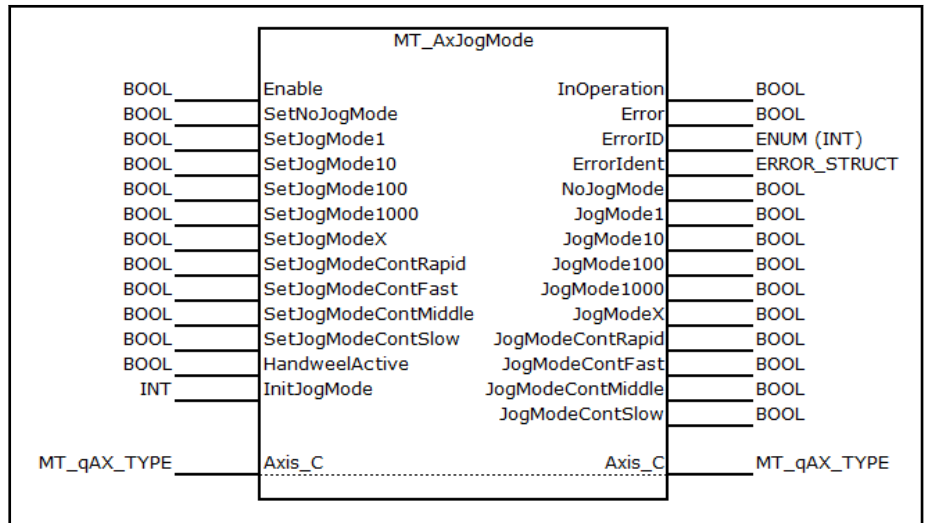


Abb. 14-83: Funktionsbaustein MT_AxJogMode

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Baustein aktivieren
	SetNoJogMode	BOOL	Kein JogMode
	SetJogMode1	BOOL	Jogmode 1 Inkrement
	SetJogMode10	BOOL	Jogmode 10 Inkremente
	SetJogMode100	BOOL	Jogmode 100 Inkremente
	SetJogMode1000	BOOL	Jogmode 1000 Inkremente
	SetJogModeX	BOOL	Jogmode für Joggen entsprechend der Inkrementvorgaben in der NC
	SetJogModeContRapid	BOOL	Kontinuierliches Joggen im Eilgang
	SetJogModeContFast	BOOL	Kontinuierliches Joggen Schnell
	SetJogModeContMiddle	BOOL	Kontinuierliches Joggen Mittel
	SetJogModeContSlow	BOOL	Kontinuierliches Joggen Langsam
	HandweelActive	BOOL	Handrad aktiv, wenn TRUE nur inkrementelle JogModes möglich
	InitJogMode	BOOL	Vorwahl des JogMode bei Einschalten des Bausteins
VAR_OUTPUT	InOperation	BOOL	Bearbeitung läuft
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "Error"=TRUE)

	Name	Typ	Beschreibung
	NoJogMode	BOOL	Kein JogMode aktiv
	JogMode1	BOOL	1 Inkrement Tippen aktiv
	JogMode10	BOOL	10 Inkremente Tippen aktiv
	JogMode100	BOOL	100 Inkremente Tippen aktiv
	JogMode1000	BOOL	1000 Inkremente Tippen aktiv
	JogModeX	BOOL	x Inkremente Tippen aktiv
	JogModeContRapid	BOOL	Kontinuierliches Tippen im Eilgang
	JogModeContFast	BOOL	Kontinuierliches Tippen schnelle Geschwindigkeit
	JogModeContMiddle	BOOL	Kontinuierliches Tippen mittlere Geschwindigkeit
	JogModeContSlow	BOOL	Kontinuierliches Tippen langsame Geschwindigkeit
VAR_IN_OUT	Axis_C	MT_qAx_Type	Achs-Interface: Achs-Steuer-Signale

Tab. 14-62: Ein-/Ausgänge Funktionsbaustein MT_AxJogMode

Signal-Zeit-Diagramm

Grundsätzlich zeigt der Funktionsbaustein das definierte Signalverhalten eines permanent zustandsgesteuerten Bausteins, siehe [Kapitel 14.2 Standard-schnittstellen an Funktionsbausteinen](#), Seite 231.

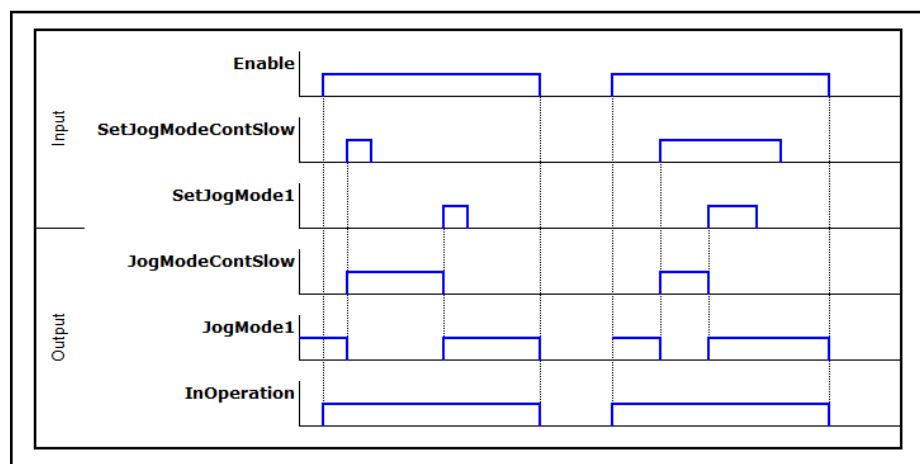


Abb. 14-84: Normales Verhalten bei Änderung der Tippdistanz

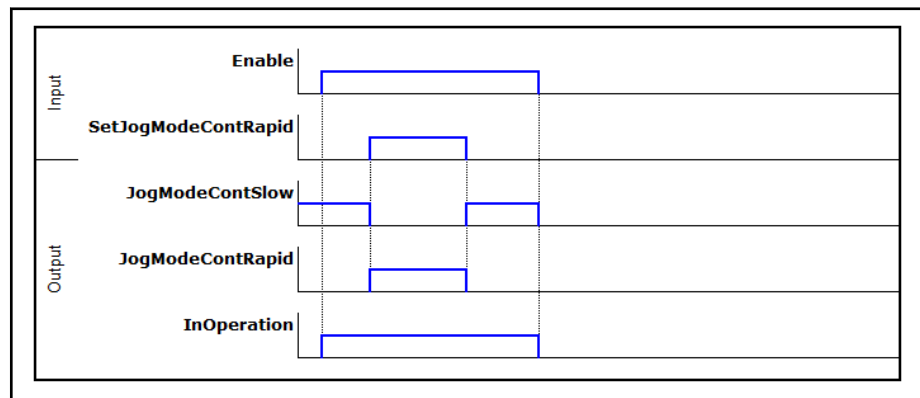


Abb. 14-85: Verhalten des FB bei Setzen von "SetJogModeContRapid"

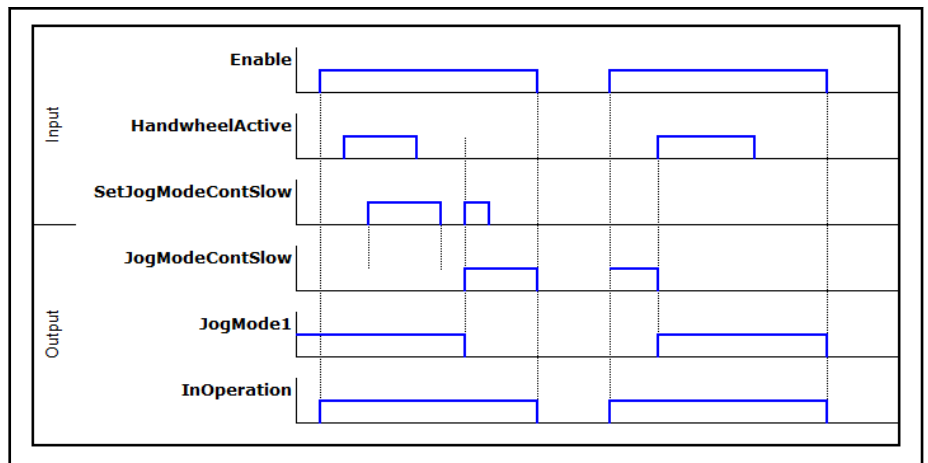


Abb. 14-86: Verhalten des FB bei Setzen von "HandwheelActive"

Funktionsbeschreibung

Dieser Baustein vereint und erweitert die Funktionalitäten von MT_AxGetJogMode und MT_AxSetJogMode.

Mit der MTX können Achsen kontinuierlich in 4 Geschwindigkeiten oder inkrementell mit 5 verschiedenen Schrittweiten getippt werden (Maschinenparameter "AXSP\Dr[x]\AxFun\Jog\Dist" - "MaCoDa-Nr.: 101500002"). Diese 9 JogModes werden ausschließlich von der SPS über das SPS-Achs-Interface (MT_qAx_Type) durch die Signale "qAx_ManFeed_00..03" codiert ausgewählt. Mit diesem Funktionsbaustein können diese JogModes für eine Achse aktiviert und gleichzeitig der aktuelle Status ausgelesen werden.

Sobald "Enable" TRUE ist, wird mit jedem Aufruf des FB der aktuelle JogMode aus dem angeschlossenen Achs-Interface ermittelt. Mit einem Signalwechsel (0-->1) an einem der "SetJog... Eingänge" wird der gewünschte JogMode geändert. Sind mehrere "SetJog... Eingänge" TRUE so wird der zuletzt gesetzte Eingang wirksam. Konnte der JogMode nicht geändert werden, wird dies durch den Ausgang „Error“ angezeigt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Zum Fehler löschen muss "Enable" FALSE werden.

Besonderheiten:

Der Eingang "SetJogModeContRapid" wird gesondert behandelt. Wird dieser JogMode gewählt, so bleibt er nur solange aktiv, wie der Eingang TRUE ist. Sobald der Eingang wieder FALSE ist, wird automatisch der zuvor eingestellte JogMode gesetzt.

Der Eingang "InitJogMode" erlaubt es, mit dem Einschalten des FB einmalig einen JogMode zu setzen ohne das dafür einer der "SetJog... Eingänge" TRUE sein muss.

Folgende Vorgaben sind gültig:

InitJogMode	JogMode
0	Keine Initialisierung
1	NoJogMode
2	JogMode1
3	JogMode10
4	JogMode100

InitJogMode	JogMode
5	JogMode1000
6	JogModeX
7	JogModeContRapid
8	JogModeContFast
9	JogModeContMiddle
10	JogModeContSlow

Tab. 14-63: Initialisierung JogMode

Da Achsbewegungen mit dem Handrad nur mit inkrementellen JogModes möglich ist, können, solange der Eingang "HandweelActive" TRUE ist, nur noch inkrementelle JogModes gesetzt werden. Ist zu dem Zeitpunkt, wenn "HandweelActive" TRUE wird (0-->1) einer der kontinuierlichen JogModes aktiv, so wird automatisch auf "JogMode1" gewechselt.

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_SetJogMode"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010001	Ungültige Eingangsbelegung. Es wurde entweder kein oder mehr als ein JogMode vorgewählt
0x00040201	Das Ändern der Interface-Signale "qAx_ManFeed_00..03" ist fehlgeschlagen. Eventuell wurden sie außerhalb des FB überschrieben
0x 00050001	Interner Fehler

Tab. 14-64: Fehler von "MT_AxSetJogMode", aus "ErrorIdent.Additional1" wenn "Error"==TRUE

Implementationsbeispiel

```

2  VAR
3  fbAxJogMode          :MT_AxJogMode;
4  //Input
5  bEnable               :BOOL:=FALSE; (*FS enable*)
6  SetNoJogMode          :BOOL:=FALSE; (*target Jogmode: no JogMode*)
7  (*incremental JogModi*)
8  SetJogMode1           :BOOL:=FALSE; (*target Jogmode: 1 increment*)
9  SetJogMode10          :BOOL:=FALSE; (*target Jogmode: 10 increment*)
10 SetJogMode100         :BOOL:=FALSE; (*target Jogmode: 100 increment*)
11 SetJogMode1000        :BOOL:=FALSE; (*target Jogmode: 1000 increment*)
12 SetJogModeX           :BOOL:=FALSE; (*target JogMode: Jogging respective the increment guidelines in the machine parameter*)
13 (*continuous JogModi*)
14 SetJogModeContRapid   :BOOL:=FALSE; (*target JogMode: rapid continuous jogging*)
15 SetJogModeContFast    :BOOL:=FALSE; (*target JogMode: fast continuous jogging*)
16 SetJogModeContMiddle  :BOOL:=FALSE; (*target JogMode: middle continuous jogging*)
17 SetJogModeContSlow    :BOOL:=FALSE; (*target JogMode: slow continuous jogging*)
18
19 HandwheelActive       :BOOL:=FALSE; (*Hand wheel is active*)
20 InitJogMode           :DINT:=5;   (*Set JogMode in the first PLC cycle*)
21 //Output
22 InOperation           :BOOL:=FALSE; (*process *)
23 bError                :BOOL:=FALSE; (*process incorrect aborted, 'ErrorID' and 'ErrorIdent' valid*)
24 enErrorID             :ERROR_CODE; (*basic error information, only valid when 'Error'==TRUE*)
25 stErrorIdent          :ERROR_STRUCT; (*detailed error information, only valid when 'Error'==TRUE*)
26 bNoJogMode            :BOOL:=FALSE; (*no JogMode aktiv*)
27 (*Inkrementelle JogModi*)
28 bJogMode1             :BOOL:=FALSE; (*JogMode 1 Inkrement aktiv*)
29 bJogMode10            :BOOL:=FALSE; (*JogMode 10 Inkremente aktiv*)
30 bJogMode100           :BOOL:=FALSE; (*JogMode 100 Inkremente aktiv*)
31 bJogMode1000          :BOOL:=FALSE; (*JogMode 1000 Inkremente aktiv*)
32 bJogModeX             :BOOL:=FALSE; (*Jogging respective the increment guidelines in the machine parameter active*)
33 (*Kontinuierliche JogModi*)
34 bJogModeContRapid     :BOOL:=FALSE; (*rapid continuous jogging active*)
35 bJogModeContFast      :BOOL:=FALSE; (*fast continuous jogging active*)
36 bJogModeContMiddle    :BOOL:=FALSE; (*middle continuous jogging active*)
37 bJogModeContSlow      :BOOL:=FALSE; (*slow continuous jogging active*)
38 END_VAR

```

Abb. 14-87: Deklaration MT_AxJogMode

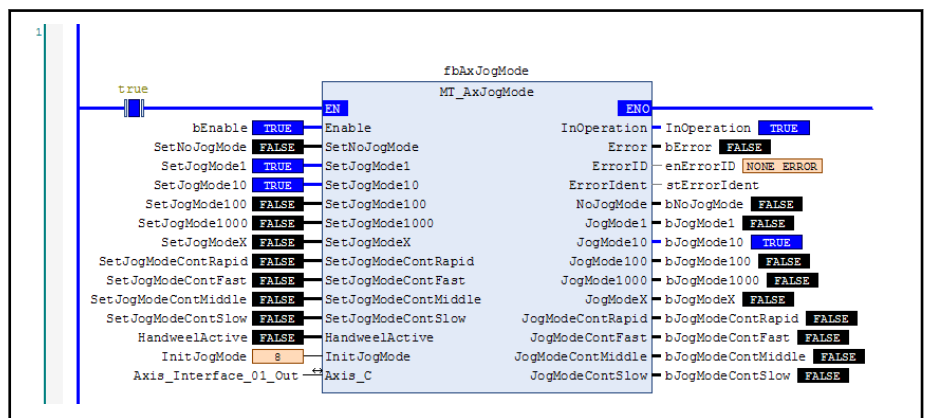


Abb. 14-88: Implementation MT_AxJogMode

14.4.9 MT_AxAsyncGetOPMode, Betriebsart einer asynchronen Achse ermitteln

Kurzbeschreibung

Funktion zum Ermitteln der aktiven Betriebsart einer asynchronen Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-65: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

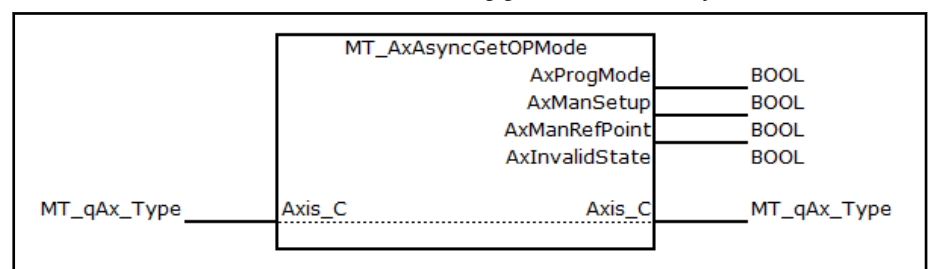


Abb. 14-89: Funktionsbaustein MT_AxAsyncGetOPMode

	Name	Typ	Beschreibung
VAR_INPUT	Axis_C	MT_qAX_TYPE	Achs-Interface: Achs-Steuer-Signale
VAR_OUTPUT	AxProgMode	BOOL	Programmierbetrieb
	AxManSetup	BOOL	Einrichten manuell (Joggen)
	AxManRefPoint	BOOL	Einrichten, Referenzpunkt anfahren
	AxInvalidState	BOOL	Unbekannte Betriebsart

Tab. 14-66: Ein-/Ausgänge Funktionsbaustein MT_AxAsyncGetOPMode

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeitdiagramm notwendig macht.

Funktionsbeschreibung

Die MTX erlaubt für jede asynchrone Achse drei Betriebsarten. Diese Zustände werden bitcodiert über das Achs-Interface (MT_qAx_Type) durch die Signale "qAx_OpModeSel_00 .. 01" repräsentiert. Es kann immer nur eine Betriebsart aktiv sein. Die Funktion dekodiert die Interface-Signale und zeigt die aktive Betriebsart über die entsprechenden Ausgänge an.



Die durch die Funktion angezeigten aktiven Betriebsarten sind für die Achse nur relevant, wenn diese Achse asynchron ist und somit keinem Kanal zugeordnet ist. Liefert die Funktion "MT_Ax-GetChIndex" den Wert NULL, ist die Achse keinem Kanal zugeordnet und somit asynchron.

Fehlerbehandlung

Für diese Funktion ist keine Fehlerbehandlung notwendig.

Implementationsbeispiel

Im Beispielprogramm wird die Betriebsart einer Achse ermittelt.

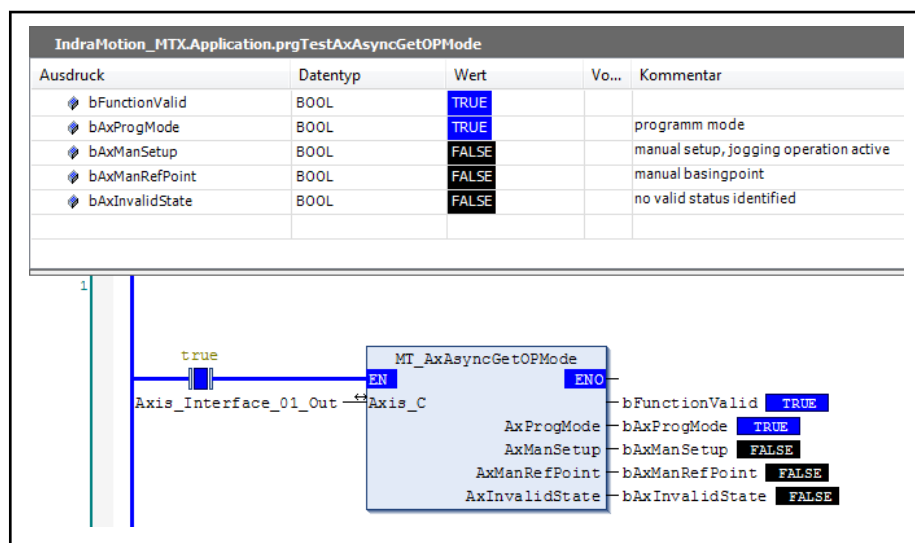


Abb. 14-90: Implementation MT_AxAsyncGetOPMode

14.4.10 MT_AxAsyncSetOPMode, Betriebsart einer asynchronen Achse ändern

Kurzbeschreibung

Funktion zum Ermitteln der aktiven Betriebsart einer asynchronen Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-67: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

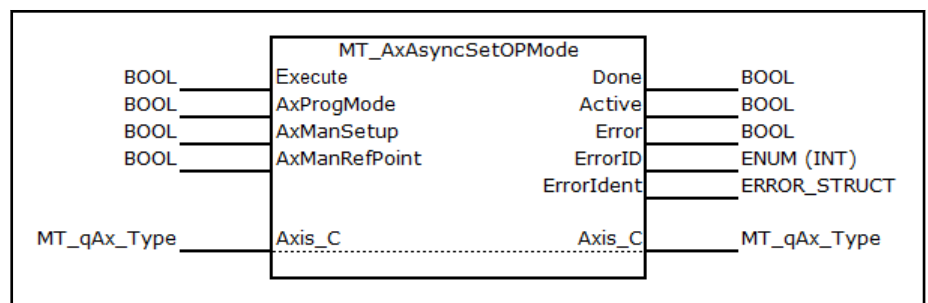


Abb. 14-91: Funktionsbaustein MT_AxAsyncSetOPMode

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Betriebsart setzen ausführen
	AxProgMode	BOOL	Programmierbetrieb
	AxManSetup	BOOL	Einrichten manuell (Joggen)
	AxManRefPoint	BOOL	Einrichten, Referenzpunkt anfahren
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
VAR_IN_OUT	Axis_C	MT_qAX_TYPE	Achs-Interface: Achs-Steuer-Signale

Tab. 14-68: Ein-/Ausgänge Funktionsbaustein MT_AxAsyncSetOPMode

Signal-Zeit-Diagramm

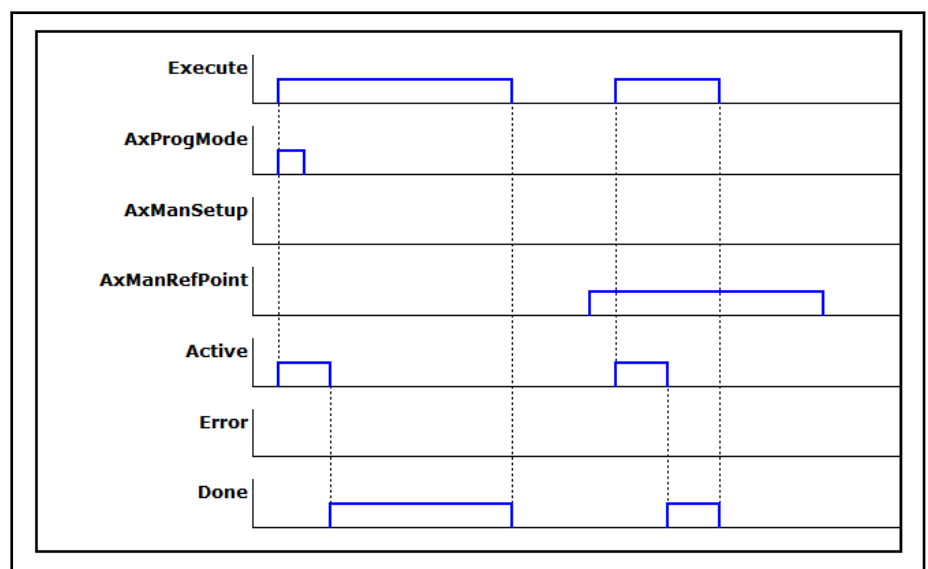


Abb. 14-92: MT_AxAsyncSetOPMode Signal-Zeit-Diagramm

Links: Setzen "Programmierbetrieb" erfolgreich

Rechts: Setzen "Einrichten, Referenzpunkt anfahren" erfolgreich

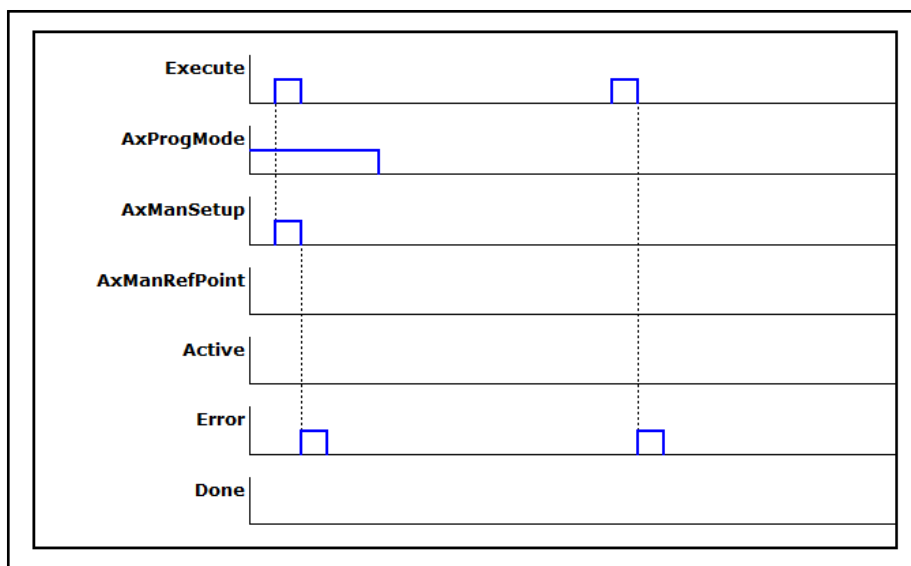


Abb. 14-93: MT_AxAsyncSetOPMode Signal-Zeit-Diagramm

Fehler links: Mehr als eine Betriebsart vorgewählt

Fehler rechts: Es wurde keine Betriebsart vorgewählt

Funktionsbeschreibung

Die MTX erlaubt für jede asynchrone Achse drei Betriebsarten. Diese Zustände werden bitcodiert über das Achsinterface (MT_qAx_Type) durch die Signale "qAx_OpModeSel_00 .. 01" repräsentiert. Mit der steigenden Flanke an Eingang Execute werden die Eingänge einmal übernommen und das entsprechende Achs-Interface-Signal gesetzt. Ist der Ausgang "Done" TRUE ist das Setzen der Betriebsart erfolgreich abgeschlossen. Im Fehlerfall ist stattdessen "Error" TRUE. Vor einem erneuten Ändern der Betriebsart sowie zum Fehler löschen muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur ein Ausgang TRUE.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren das Setzen der Betriebsart "qAx_OpModeSel_00 .. 01" genau einmal ausführt.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_AxAsyncSetOPMode"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010402	Ungültige Eingangsbelegung. Es wurde entweder kein oder mehr als eine Betriebsart vorgewählt
0x00040101	Das Setzen des Achs-Interface-Signals (qAx_OpModeSel) ist fehlgeschlagen, das Signal wurde außerhalb des FB zurückgesetzt

Tab. 14-69: Fehler von "MT_AxAsyncSetOPMode", aus "ErrorIdent.Additional1", wenn "Error"==TRUE

Implementationsbeispiel

Im Beispielprogramm wurde die Betriebsart einer asynchronen Achse auf "Einrichten manuell" geändert.

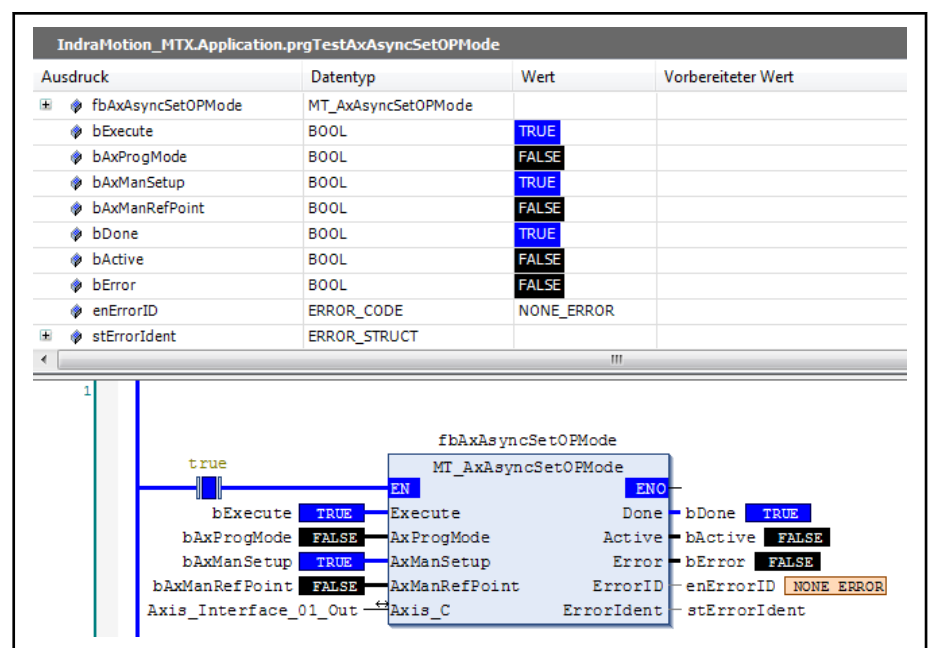


Abb. 14-94: Implementation MT_AxAsyncSetOPMode

14.4.11 MT_AxAsyncOPMode, Betriebsart einer asynchronen Achse

Kurzbeschreibung

Funktion zum Ermitteln der aktiven Betriebsart einer asynchronen Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-70: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

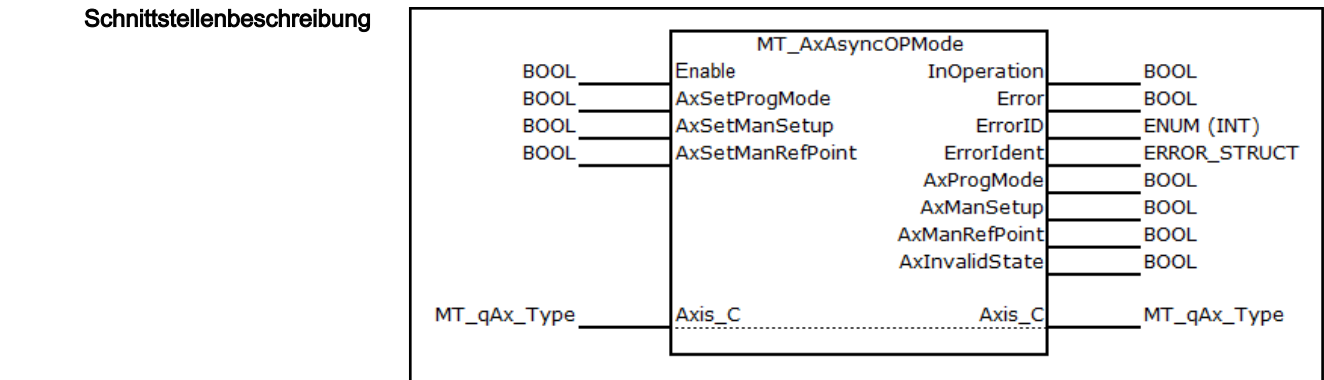


Abb. 14-95: Funktionsbaustein MT_AxAsyncOPMode

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Baustein aktivieren
	AxSetProgMode	BOOL	Programmierbetrieb
	AxSetManSetup	BOOL	Einrichten manuell (Joggen)
	AxSetManRefPoint	BOOL	Einrichten, Referenzpunkt anfahren
VAR_OUTPUT	AxProgMode	BOOL	Programmierbetrieb
	AxManSetup	BOOL	Einrichten manuell (Joggen)
	AxManRefPoint	BOOL	Einrichten, Referenzpunkt anfahren
	AxInvalidState	BOOL	Unbekannte Betriebsart
VAR_IN_OUT	Axis_C	MT_qAX_TYPE	Achs-Interface: Achs-Steuer-Signale

Tab. 14-71: Ein-/Ausgänge Funktionsbaustein MT_AxAsyncOPMode

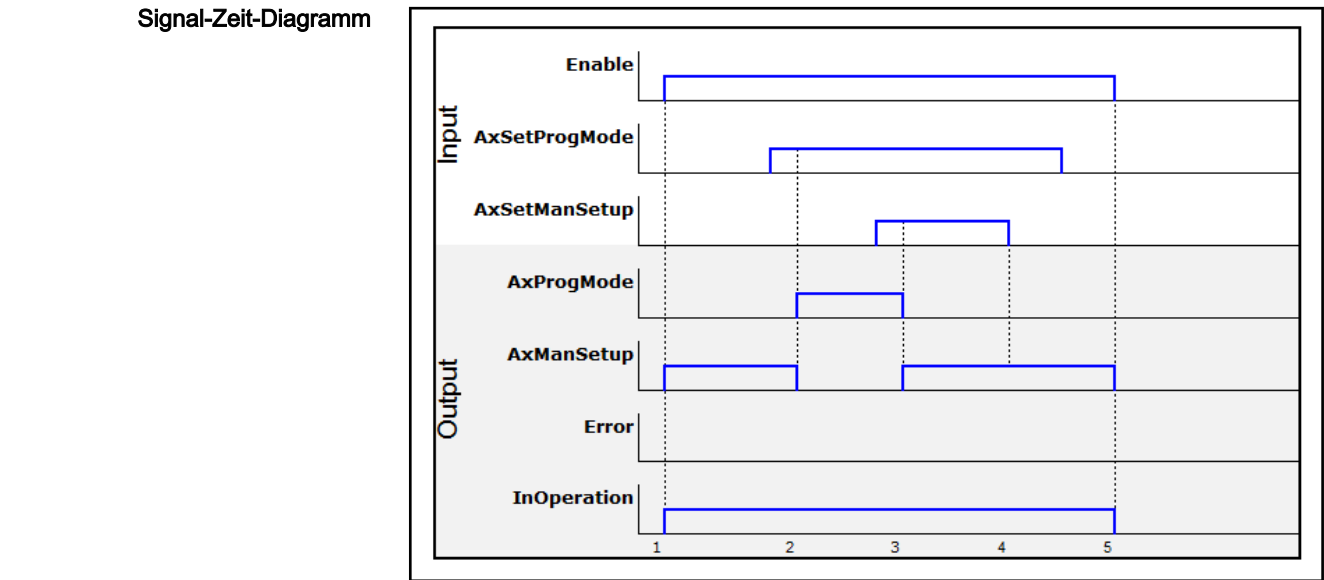


Abb. 14-96: MT_AxAsyncOPMode Signal-Zeit-Diagramm

1. AxManSetup ist aktiv
2. Ein Zyklus nach Schalten ist AxProgMode aktiv
3. Letzter Signalwechsel an AxSetManSetup, daher AxManSetup aktiv
4. AxManSetup bleibt aktiv, auch wenn Eingang FALSE
5. Alle Ausgänge FALSE, wenn Enable FALSE

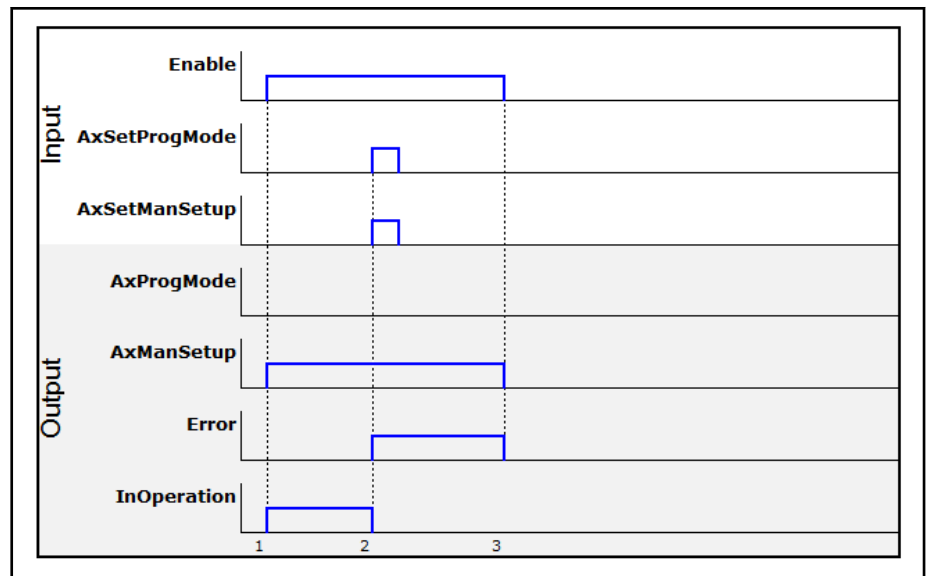


Abb. 14-97: MT_AxAsyncOPMode Signal-Zeit-Diagramm

1. AxManSetup ist aktiv
2. Fehler - Gleichzeitiger Signalwechsel (0-->1) an zwei AxSet...-Eingängen
3. Fehler löschen durch Rücksetzen von Enable

Funktionsbeschreibung

Die MTX erlaubt für jede asynchrone Achse drei Betriebsarten. Diese Zustände werden bitcodiert über das Achs-Interface (MT_qAx_Type) durch die Signale "qAx_OpModeSel_00 .. 01" repräsentiert. Es kann immer nur eine Betriebsart aktiv sein. Der Baustein dekodiert die Interface-Signale, zeigt die aktive Betriebsart und ermöglicht deren Änderung.



Die durch die Funktion angezeigte aktive Betriebsart ist für die Achse nur relevant, wenn diese Achse asynchron ist und somit keinem Kanal zugeordnet ist. Liefert die Funktion "MT_AxGetChIndex" den Wert NULL ist die Achse keinem Kanal zugeordnet und somit asynchron.

Solange der Eingang "Enable" TRUE ist, ist der Funktionsbaustein aktiv. Das wird über den Ausgang "InOperation" angezeigt. Ein Fehler wird durch den Ausgang "Error" angezeigt. Wenn "Error" TRUE ist, findet man einen gültigen Fehlercode am Ausgang "ErrorIdent". Zum Löschen eines Fehlers muss "Enable" auf FALSE gesetzt werden. Alle Eingänge werden bei jedem Bausteinaufruf übernommen. Somit wird auch die aktuelle Betriebsart mit jedem Bausteinaufruf neu aus dem Interface-Signal "qAx_OpModeSel_00 .. 01" ermittelt. Das Ändern einer Betriebsart erfolgt durch Signalwechsel (0-->1) am entsprechenden Betriebsarteneingang und wird nach einem SPS-Zyklus wirksam. Sind mehrere Eingänge TRUE wird für das Ändern der Betriebsart der Eingang verwendet, an dem zuletzt eine Signaländerung (0-->1) erkannt wurde.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von „MT_SetPowerOn“

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010402	Ungültige Eingangsbelegung. Es wurde entweder kein oder mehr als eine Betriebsart vorgewählt
0x00040101	Das Setzen des Achs-Interface-Signals (qAx_OpModeSel) ist fehlgeschlagen, das Signal wurde außerhalb des FB zurückgesetzt

Tab. 14-72: Fehler von "MT_SetPowerOn", aus "ErrorIdent.Additional1", wenn „Error“==TRUE

Implementationsbeispiel

Im Beispielprogramm wurde die Betriebsart auf AxManRefPoint geändert.

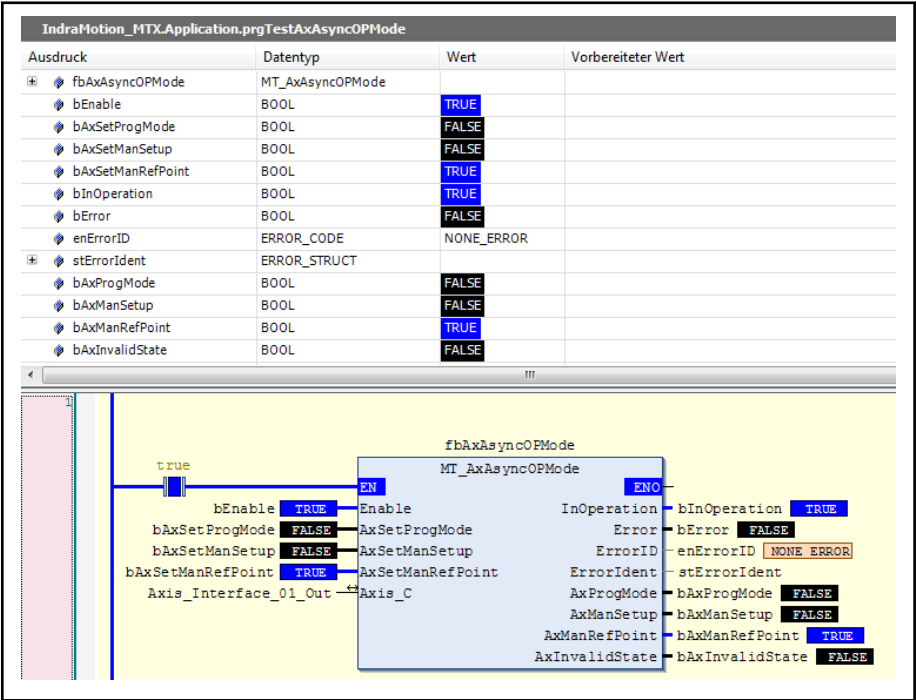


Abb. 14-98: Implementation MT_AxAsyncOPMode

14.4.12 MT_AxAsyncSetOverride, Vorschub für asynchrone Achse ändern

Kurzbeschreibung

Funktion zum Setzen des Vorschubwertes für eine asynchrone Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-73: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

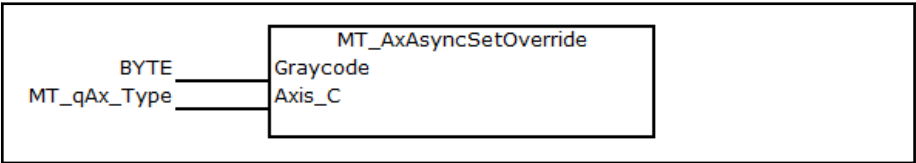


Abb. 14-99: Funktionsbaustein MT_AxAsyncSetOverride

	Name	Typ	Beschreibung
VAR_INPUT	Graycode	BYTE	Overridewert als Graycode
VAR_IN_OUT	Axis_C	MT_qAx_Type	Achs-Interface: Achs-Steuer-Signale

Tab. 14-74: Ein-/Ausgänge Funktionsbaustein MT_AxAsyncSetOverride

Signal-Zeit-Diagramm	Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.
Funktionsbeschreibung	Der Achsvorschubwert muss für jede asynchrone Achse über dessen Achs-Interface (MT_qAx_Type) durch das Signal "qAx_Override" von der SPS an die NC übertragen werden. Mit dieser Funktion wird der am Eingang "Graycode" anliegende Wert gewandelt und an das angeschlossene Achs-Interface übergeben.
Fehlerbehandlung	Für diese Funktion ist keine Fehlerbehandlung notwendig.
Implementationsbeispiel	Beispielprogramm für das Ändern des Override für die erste Achse.

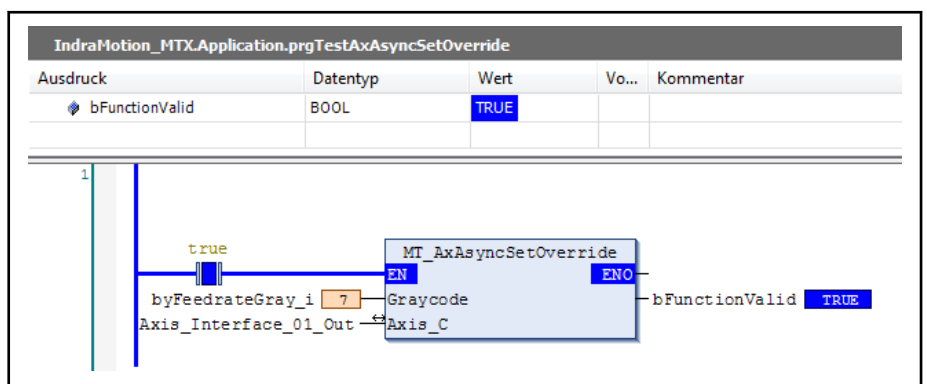


Abb. 14-100: Implementation MT_AxAsyncSetOverride

14.4.13 MT_AxGetPowerState, Antriebsstatus ermitteln

Kurzbeschreibung	Funktion zum Ermitteln des aktiven Antriebsstatus einer Achse.
Zuordnung Zielsystem/Bibliothek	

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-75: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

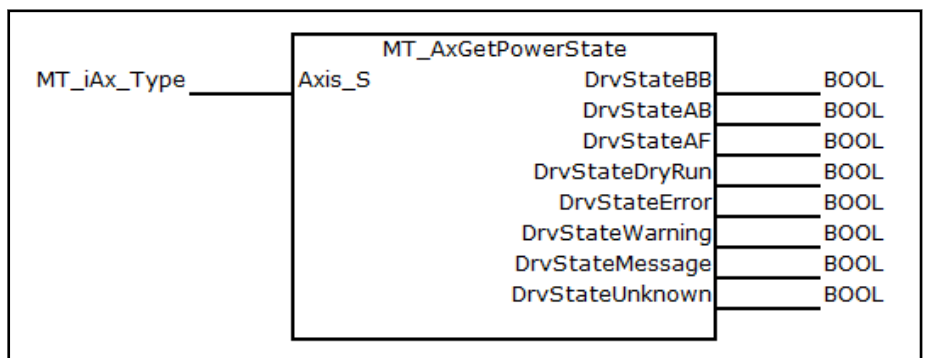


Abb. 14-101: Funktionsbaustein MT_AxGetPowerState

	Name	Typ	Beschreibung
VAR_INPUT	Axis_S	MT_iAX_TYPE	Achs-Interface: Achs-Status-Signale
VAR_OUTPUT	DrvStateBB	BOOL	Antrieb Betriebsbereit
	DrvStateAB	BOOL	Antrieb bereit zur Reglerfreigabe
	DrvStateAF	BOOL	Antrieb in Regelung
	DrvStateDryRun	BOOL	Antrieb im virtuellen Betrieb
	DrvStateError	BOOL	Antrieb meldet einen Fehler
	DrvStateWarning	BOOL	Antrieb meldet eine Warnung
	DrvStateMessage	BOOL	Antrieb meldet einen Hinweis

Tab. 14-76: Ein-/Ausgänge Funktionsbaustein MT_AxGetPowerState

Signal-Zeit-Diagramm Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung Diese Funktion ermittelt aus dem Achs-Interface (MT_iAx_Type) den Leistungszustand und zeigt an, ob Fehler, Warnungen oder Hinweise am Antrieb anstehen.

Dazu werden folgende Interface-Signale interpretiert: , , ,

- iAx_DrvPower
- iAx_DrvReady
- iAx_DrvAct
- iAx_DryRun
- iAx_DrvErrClass1
- iAx_DrvChangeClass2
- iAx_DrvChangeClass3

Fehlerbehandlung Für diese Funktion ist keine Fehlerbehandlung notwendig.

Implementationsbeispiel Im Beispielpogramm wird der Status einer Achse ermittelt.

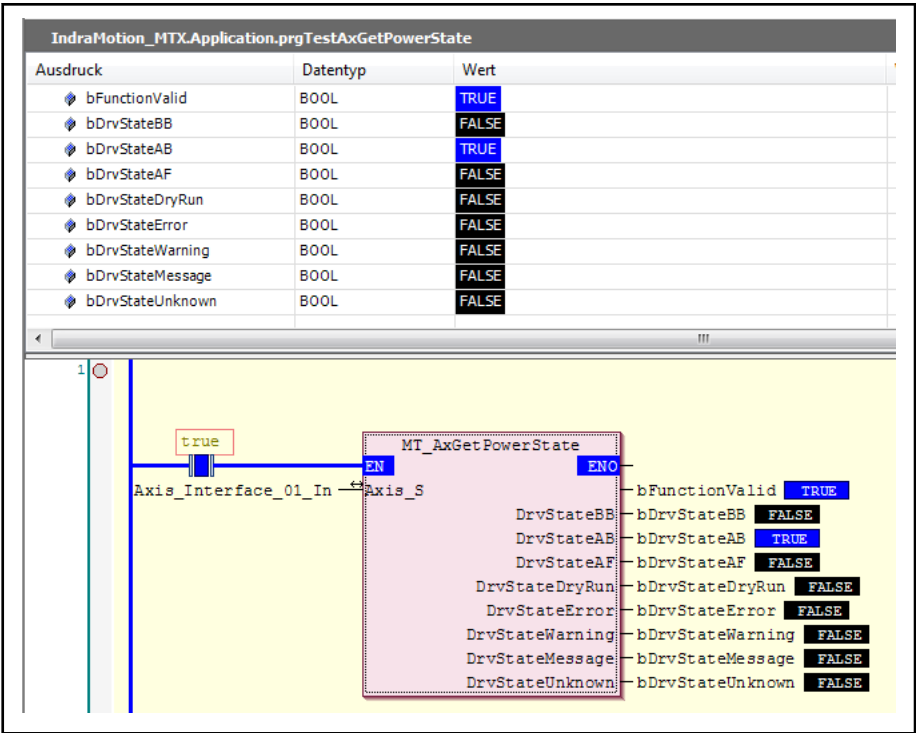


Abb. 14-102: Implementation MT_AxGetPowerState

14.4.14 MT_AxGetPowerStateExt, Antriebsstatus incl. Fehler- und Warnungs- details ermitteln

Kurzbeschreibung Funktion zum Ermitteln des aktiven Antriebsstatus einer Achse mit zusätzli- chen Informationen zu Fehler und Warnungen.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-77: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

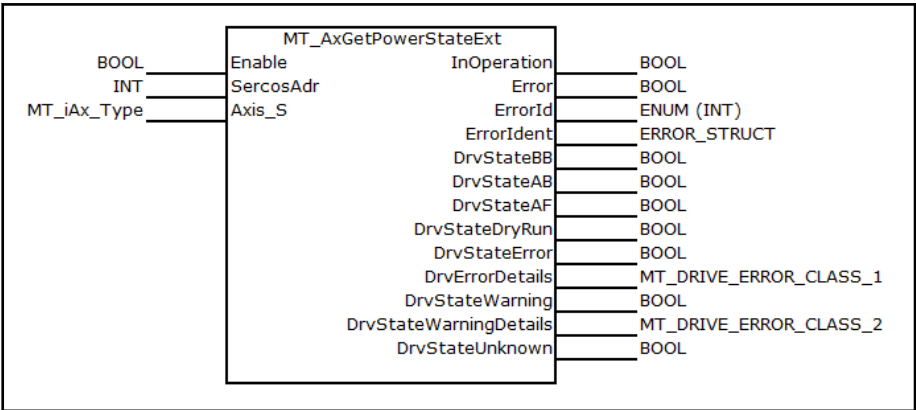


Abb. 14-103: Funktionsbaustein MT_AxGetPowerStateExt

	Name	Typ	Beschreibung
VAR_INPUT	Axis_S	MT_iAX_TYPE	Achs-Interface: Achs-Status-Signale
VAR_OUTPUT	DrvStateBB	BOOL	Antrieb betriebsbereit
	DrvStateAB	BOOL	Antrieb bereit zur Reglerfreigabe
	DrvStateAF	BOOL	Antrieb in Regelung
	DrvStateDryRun	BOOL	Antrieb im virtuellen Betrieb
	DrvStateError	BOOL	Antrieb meldet einen Fehler
	DrvStateErrorDetails	MT_DRIVE_ERROR_CLASS_1	Detailinformationen zum Fehler (S-0-0011)
	DrvStateWarning	BOOL	Antrieb meldet eine Warnung
	DrvStateWarningDetails	MT_DRIVE_ERROR_CLASS_2	Detailinformationen zur Warnung (S-0-0012)

Tab. 14-78: Ein-/Ausgänge Funktionsbaustein MT_AxGetPowerStateExt

Signal-Zeit-Diagramm Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung Dieser Funktionsbaustein ermittelt aus dem Achs-Interface (MT_iAx_Type) den Leistungszustand und zeigt an, ob ein Fehler oder eine Warnung am Antrieb ansteht. Zudem werden bei anstehendem Fehler oder Warnung jede Sekunde Detailinformationen aus den Antriebsparametern S-0-0011 bzw. S-0-0012 ausgegeben.

Dazu werden folgende Interfacesignale interpretiert:

- iAx_DrvPower
- iAx_DrvReady
- iAx_DrvAct
- iAx_DryRun
- iAx_DrvErrClass1
- iAx_DrvChangeClass2
- iAx_DrvChangeClass3

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table).

Der Funktionsbaustein zeigt Fehler des intern genutzten Funktionsbausteines "IL_SIIISvcRead" aus der Bibliothek "RIL_SercosIII" an.

Implementationsbeispiel Im Beispielprogramm wird der Status einer Achse im Fehlerfall ermittelt.

14.4.15 MT_AxSetPowerOn, Setzt die Reglerfreigabe für eine Achse

Kurzbeschreibung Funktionsbaustein zum Aktivieren der Reglerfreigabe für eine Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-79: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

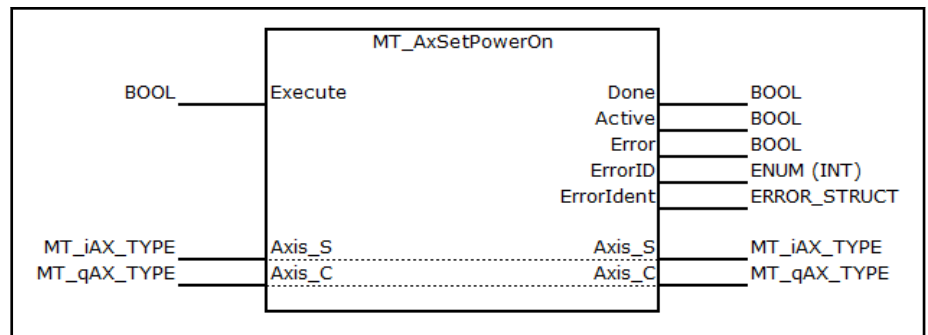


Abb. 14-104: Funktionsbaustein MT_AxSetPowerOn

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Reglerfreigabe für Achse setzen
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
VAR_IN_OUT	Axis_S	MT_iAx_Type	Achs-Interface: Achs-Status-Signale
	Axis_C	MT_qAx_Type	Achs-Interface: Achs-Steuer-Signale

Tab. 14-80: Ein-/Ausgänge Funktionsbaustein MT_AxSetPowerOn

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Das Setzen der Reglerfreigabe (AF) für eine Achse erfolgt bei der MTX im NC-PLC-Achs-Interface (MT_qAx_Type) über das Achssteuersignal "qAx_DrvOn". Der Baustein prüft erst, ob die Achse zur Reglerfreigabe bereit ist, setzt dann dieses Signal und überwacht, ob innerhalb von 5 Sekunden die Rückmeldung über Achsstatussignale (MT_iAx_Type) kommt, dass die Achse in Regelung ist.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass er nach dem Aktivieren das Setzen der Reglerfreigabe "qAx_DrvOn" genau einmal ausführt.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein. Die Eingänge "Axis_S" und "Axis_C" werden zyklisch übernommen, solange der Baustein aktiv ist ("Active"==TRUE). Wenn der Ausgang "Done" TRUE wird, ist das Setzen der Reglerfreigabe erfolgreich abgeschlossen. Konnte für den Antrieb die Reglerfreigabe nicht gesetzt werden, wird dies durch den Ausgang "Error" angezeigt. Im Fehlerfall wird das Interface-Signal "qAx_DrvOn" zurückgesetzt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einem erneuten Setzen der Reglerfreigabe, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Ad-

ditional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_SetPowerOn"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00060102	Antrieb nicht bereit zur Reglerfreigabe, Signalstatus (iAx_DrvReady), der Antrieb muss im Status AB sein
0x00040101	Das Setzen des Achs-Interface-Signals (qAx_DrvOn) ist fehlgeschlagen, das Signal wurde außerhalb des FB zurückgesetzt
0x00020101	Reglerfreigabe wurde angefordert aber es erfolgte keine Rückmeldung Antrieb (Achs-Interface-Signal: iAx_DrvAct), eventuell wurde das falsche Achs-Interface am FB angeschlossen

Tab. 14-81: Fehler von "MT_SetPowerOn", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

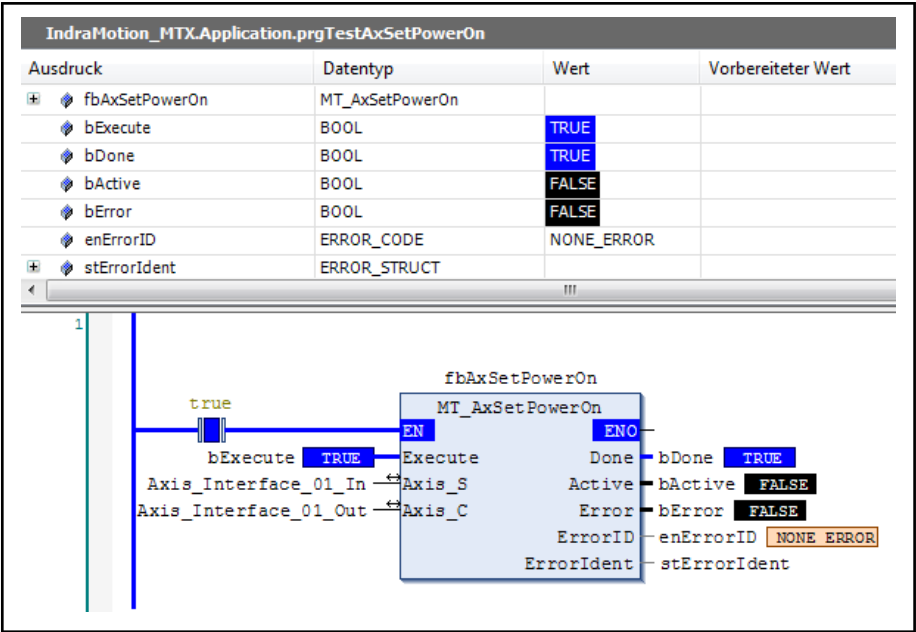


Abb. 14-105: Implementation MT_AxSetPowerOn

14.4.16 MT_AxSetPowerOff, Setzt die Reglerfreigabe für eine Achse

Kurzbeschreibung Funktionsbaustein zum Deaktivieren der Reglerfreigabe für eine Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-82: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

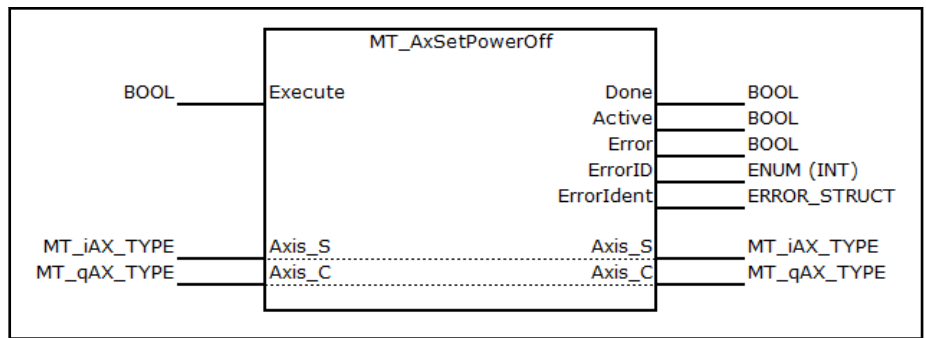


Abb. 14-106: Funktionsbaustein MT_AxSetPowerOff

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Reglerfreigabe für Achse zurücksetzen
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
VAR_IN_OUT	Axis_S	MT_iAx_Type	Achs-Interface: Achs-Status-Signale
	Axis_C	MT_qAx_Type	Achs-Interface: Achs-Steuer-Signale

Tab. 14-83: Ein-/Ausgänge Funktionsbaustein MT_AxSetPowerOff

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Das Zurücksetzen der Reglerfreigabe (AF) für eine Achse erfolgt bei der MTX im NC-PLC-Achs-Interface (MT_qAx_Type) über das Achssteuersignal "qAx_DrvOn". Der Baustein setzt dieses Signal auf FALSE und überwacht, ob innerhalb von 5 Sekunden die Rückmeldung über Achsstatussignale (MT_iAx_Type) kommt, dass die Achse nicht mehr in Regelung ist.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren das Zurücksetzen der Reglerfreigabe "qAx_DrvOn" genau einmal ausführt.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein. Die Eingänge "Axis_S" und "Axis_C" werden zyklisch übernommen, solange der Baustein aktiv ist ("Active"==TRUE). Wenn der Ausgang "Done" TRUE wird, ist das Zurücksetzen der Reglerfreigabe erfolgreich abgeschlossen. Konnte für den Antrieb die Reglerfreigabe nicht zurückgesetzt werden, wird dies durch den Ausgang "Error" angezeigt. Im Fehlerfall bleibt das Interface-Signal "qAx_DrvOn" FALSE. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einem erneuten Zurücksetzen der Reglerfreigabe, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Zurücksetzen von "Execute" werden "Error" und "Done" gelöscht. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Ad-

ditional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_SetPowerOn"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00040101	Das Rücksetzen des Achs-Interface-Signals (qAx_DrvOn) ist fehlgeschlagen, das Signal wurde außerhalb des FB wieder gesetzt
0x00020101	Das Rücksetzen der Reglerfreigabe wurde angefordert, aber es erfolgte keine Rückmeldung vom Antrieb (Achs-Interface-Signal: iAx_DrvAct), eventuell wurde das falsche Achs-Interface am FB angeschlossen

Tab. 14-84: Fehler von "MT_SetPowerOff", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

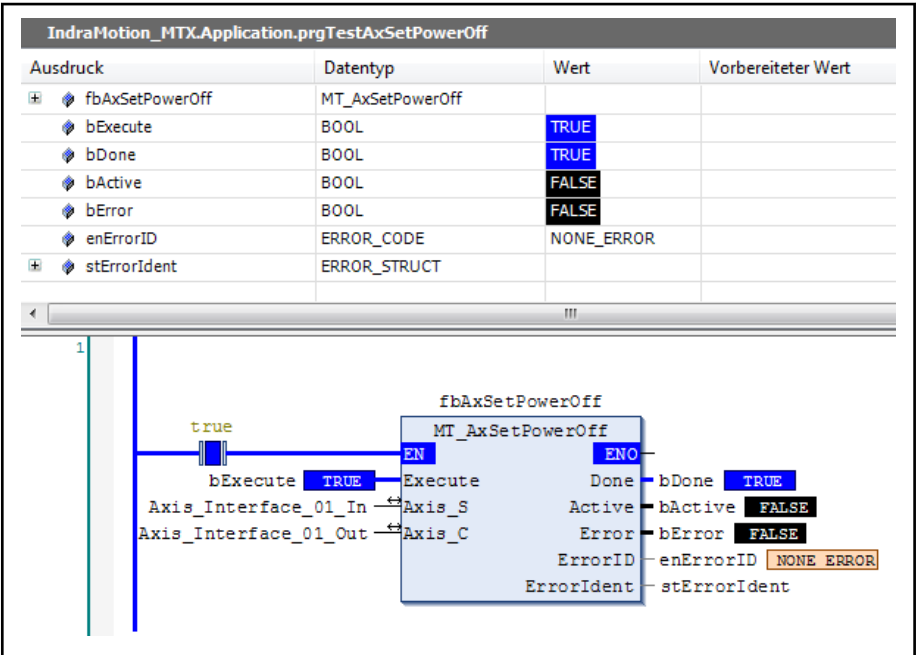


Abb. 14-107: Implementation MT_AxSetPowerOff

14.4.17 MT_AxPower, Reglerfreigabe und Status einer Achse

Kurzbeschreibung Funktionsbaustein zum Aktivieren/Deaktivieren der Reglerfreigabe, sowie zum Ermitteln des aktuellen Status einer Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-85: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

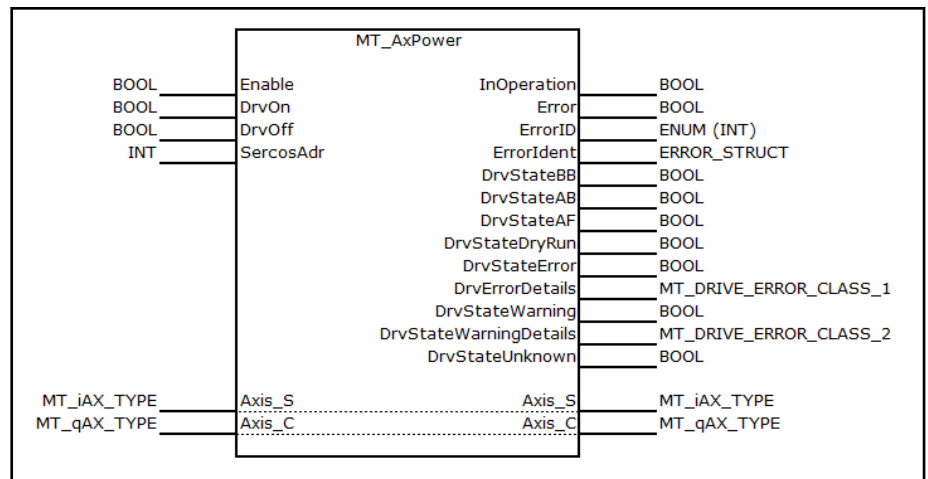


Abb. 14-108: Funktionsbaustein MT_AxPower

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Baustein aktivieren
	DrvOn	BOOL	Setzt Reglerfreigabe (DrvOff muss ebenfalls TRUE sein)
	DrvOff	BOOL	Fallende Flanke setzt Reglerfreigabe zurück
	SercosAdr	INT	Sercos-Adresse des Antriebs
VAR_OUTPUT	InOperation	BOOL	Bearbeitung erfolgreich abgeschlossen
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig, wenn "bError"=TRUE)
	DrvStateBB	BOOL	Antrieb Betriebsbereit
	DrvStateAB	BOOL	Antrieb bereit zur Reglerfreigabe
	DrvStateAF	BOOL	Antrieb in Regelung
	DrvStateDryRun	BOOL	Antrieb im virtuellen Betrieb
	DrvStateError	BOOL	Antrieb meldet einen Fehler
	DrvStateErrorDetails	MT_DRIVE_ERROR_CLASS_1	Detailinformationen zum Fehler (S-0-0011)
	DrvStateWarning	BOOL	Antrieb meldet eine Warnung
	DrvStateWarningDetails	MT_DRIVE_ERROR_CLASS_2	Detailinformationen zur Warnung (S-0-0012)
VAR_IN_OUT	Axis_S	MT_iAx_Type	Achs-Interface: Achs-Status-Signale
	Axis_C	MT_qAx_Type	Achs-Interface: Achs-Steuer-Signale

Tab. 14-86: Ein-/Ausgänge Funktionsbaustein MT_AxPower

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Dieser Baustein vereint die Funktionalität von "MT_AxSetPowerOn", "MT_AxSetPowerOff" und "MT_AxGetPowerStateExt". Er kann die Reglerfreigabe für eine Achse steuern, sowie deren Status zyklisch ermitteln und bei Warnungen oder Fehlern zusätzliche Detailinformationen liefern.

Mit dem Setzen von "Enable" werden alle Eingänge zyklisch gelesen, laufend der Achsstatus aus dem Achs-Interface ermittelt und über die Ausgänge "DrvState..." angezeigt. Solange "Enable" und "DrvOff" TRUE sind, kann mit einer steigenden Flanke an "DrvOn" das Setzen der Reglerfreigabe für die Achse angefordert werden. Das Rücksetzen der Reglerfreigabe erfolgt über eine fallende Flanke an DrvOff. Solange sich eine Achse im Zustand "DrvStateDryRun" befindet, sind "DrvOn" und "DrvOff" wirkungslos.

Das Setzen und Rücksetzen der Reglerfreigabe (AF) erfolgt im NC-PLC-Achs-Interface (MT_qAx_Type) über das Achssteuersignal "qAx_DrvOn". Vor dem Setzen der Reglerfreigabe wird geprüft, ob die Achse bereit zur Reglerfreigabe ist (AB). Erst dann wird dieses Signal gesetzt. Beim Setzen, wie beim Rücksetzen der Reglerfreigabe, wird überwacht, ob innerhalb von 5 Sekunden die Rückmeldung über Achsstatussignale (MT_iAx_Type) erfolgt.

Solange der Ausgang "InOperation" TRUE ist arbeitet der FB fehlerfrei. Kommt es zu einem Fehler, wird dies durch den Ausgang "Error" angezeigt und "InOperation" wird FALSE. Im Fehlerfall wird das Interface-Signal "qAx_DrvOn" zurückgesetzt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent".

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen *Referenzen auf die Diagnosetabelle*

- MTX_TABLE

Zusätzlich zu den unten aufgezählten Fehlern zeigt der Baustein Fehler des intern genutzten Funktionsbausteines "IL_SIIISvcRead" aus der Bibliothek "RIL_SercosIII" (F_RELATED_TABLE) an, sowie Fehler des Sercos-Service-Kanals (Fehlertabelle SERCOS_TABLE).

Fehler von "MT_AxPower", MTX_TABLE

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00060102	Antrieb nicht bereit zur Reglerfreigabe, Signalstatus (iAx_DrvReady), der Antrieb muss im Status AB sein
0x00040101	Das Setzen oder Rücksetzen des Achs-Interface-Signals (qAx_DrvOn) ist fehlgeschlagen, das Signal wurde außerhalb des FB geändert
0x00020101	Setzen/Rücksetzen der Reglerfreigabe wurde angefordert, aber es erfolgte keine Rückmeldung vom Antrieb (Achs-Interface-Signal: iAx_DrvAct), eventuell wurde das falsche Achs-Interface am FB angeschlossen

Tab. 14-87: Fehler von "MT_AxPower" aus "ErrorIdent.Additional1", wenn "Error"==TRUE

Implementationsbeispiel

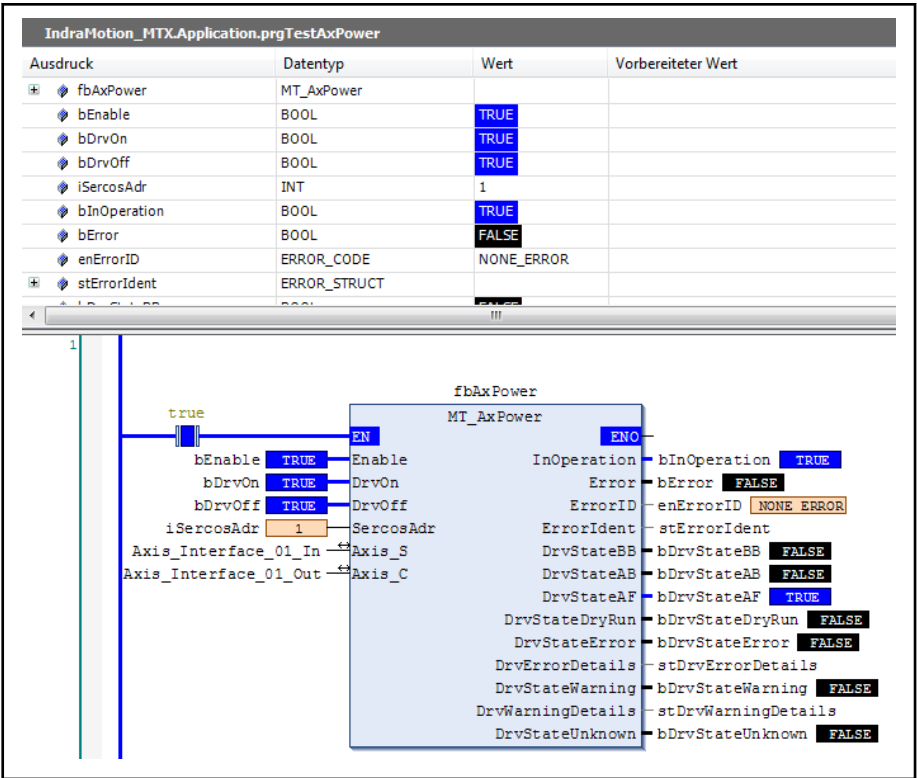


Abb. 14-109: Implementation MT_AxPower

14.5 Spindeln

14.5.1 Übersicht

Übersicht Spindelfunktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung
Allgemein	MT_SpSetOverride, Seite 320	Vorschub für eine Spindel ändern
Tippgeschwindigkeit	MT_SpJogSpeed, Seite 325	Ermitteln und Ändern der aktiven Tippgeschwindigkeit einer Spindel (kombiniert (MT_SpGetJogSpeed und MT_SpSetJogSpeed))
	MT_SpGetJogSpeed, Seite 321	Ermitteln der aktuellen Tippgeschwindigkeit für eine Spindel
	MT_SpSetJogSpeed, Seite 322	Ändern der aktiven Tippgeschwindigkeit für eine Spindel

Leistung & Diagno- se	MT_SpPower, Seite 336	Ermitteln des Antriebsstatus und Setzen/ Rücksetzen der Reglerfreigabe (kombiniert MT_SpSetPowerOn, MT_SpSetPowerOff und MT_SpGetPowerStateExt)
	MT_SpSetPowerOn, Seite 332	Setzt die Reglerfreigabe für eine Spindel
	MT_SpSetPowerOff, Seite 334	Rücksetzen der Reglerfreigabe für eine Spindel
	MT_SpGetPowerState, Sei- te 328	Ermittelt den Antriebsstatus für eine Spin- del
	MT_SpGetPowerStateExt, Seite 330	Ermittelt den Antriebsstatus einer Spindel inklusive Detailinformationen zu Fehler und Warnungen

Tab. 14-88: Übersicht Spindelfunktionen

14.5.2 MT_SpSetOverride, Vorschub für eine Spindel ändern

Kurzbeschreibung

Funktion zum Setzen des Vorschubwertes für eine Spindel.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-89: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

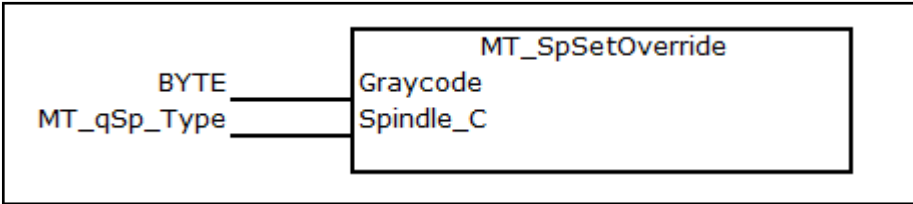


Abb. 14-110: Funktionsbaustein MT_SpSetOverride

	Name	Typ	Beschreibung
VAR_INPUT	Graycode	BYTE	Overridewert als Graycode
VAR_IN_OUT	Spindle_C	MT_qSp_Type	Achs-Interface: Spindel-Steuer-Signale

Tab. 14-90: Ein-/Ausgänge Funktionsbaustein MT_SpSetOverride

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Der Achsvorschubwert muss für jede Spindel über deren Spindel-Interface (MT_qSp_Type) durch das Signal "qSp_Override" von der SPS an die NC übertragen werden. Mit dieser Funktion wird der am Eingang "Graycode" anliegende Wert gewandelt und an das angeschlossene Spindel-Interface übergeben.

Fehlerbehandlung

Für diese Funktion ist keine Fehlerbehandlung notwendig.

Implementationsbeispiel

Beispielprogramm für das Ändern des Override für die erste Achse.

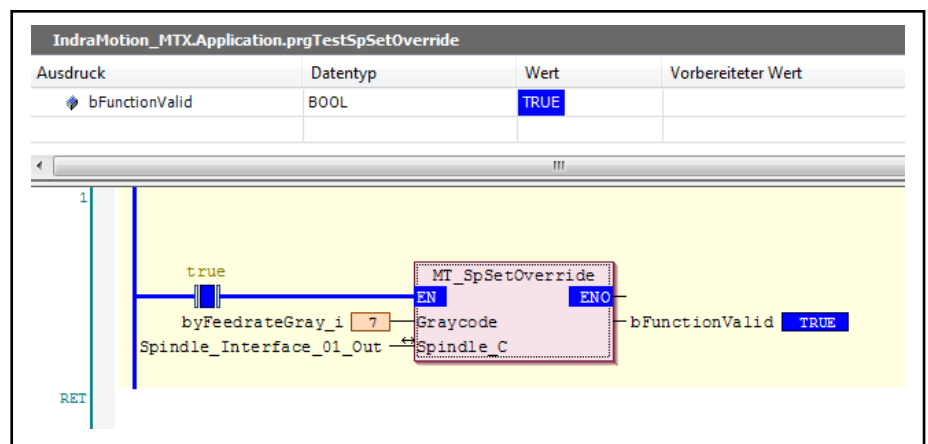


Abb. 14-111: Implementation MT_SpSetOverride

14.5.3 MT_SpGetJogSpeed, Aktive Tippgeschwindigkeit ermitteln

Kurzbeschreibung

Funktion zum Ermitteln der aktiven Tippgeschwindigkeit einer Spindel.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional.lib

Tab. 14-91: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

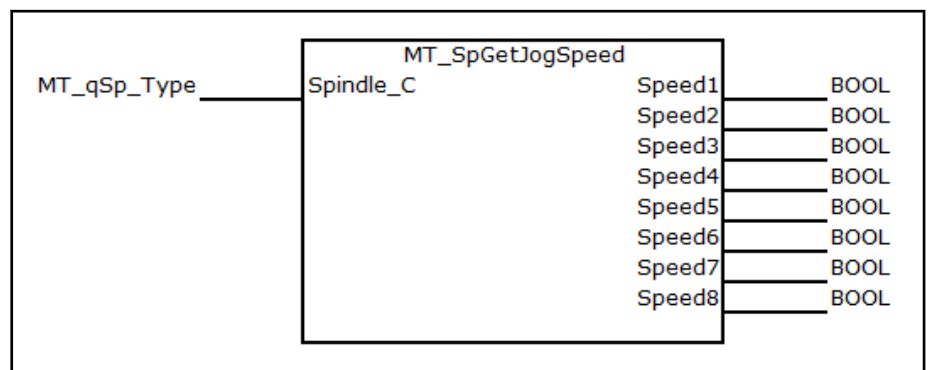


Abb. 14-112: Funktionsbaustein MT_SpGetJogSpeed

	Name	Typ	Beschreibung
VAR_INPUT	Spindle_C	MT_qSP_TYPE	Spindel-Interface: Spindel-Steuer-Signale
VAR_OUTPUT	Speed1	BOOL	Spindeltippgeschwindigkeit 1
	Speed2	BOOL	Spindeltippgeschwindigkeit 2
	Speed3	BOOL	Spindeltippgeschwindigkeit 3
	Speed4	BOOL	Spindeltippgeschwindigkeit 4
	Speed5	BOOL	Spindeltippgeschwindigkeit 5
	Speed6	BOOL	Spindeltippgeschwindigkeit 6
	Speed7	BOOL	Spindeltippgeschwindigkeit 7
	Speed8	BOOL	Spindeltippgeschwindigkeit 8

Tab. 14-92: Ein-/Ausgänge Funktionsbaustein MT_SpGetJogSpeed

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

- Funktionsbeschreibung

Mit der MTX kann die Spindel in 8 verschiedenen Geschwindigkeiten getippt werden (Maschinenparameter: "SP\SPFUNC\Sp[x]\Jog\Speed" - "MaCoDa-Nr.: 104000015"). Die Tippgeschwindigkeit für die Spindel wird ausschließlich von der SPS an die NC vorgegeben. Mit jedem Aufruf der Funktion ermittelt diese Funktion die zuletzt von der SPS vorgegebene Tippgeschwindigkeit aus dem Spindel-Interface (MT_qSp_Type) von den Signalen "qSp_Man-Speed_00..03".
- Fehlerbehandlung

Für diese Funktion ist keine Fehlerbehandlung notwendig.
- Implementationsbeispiel

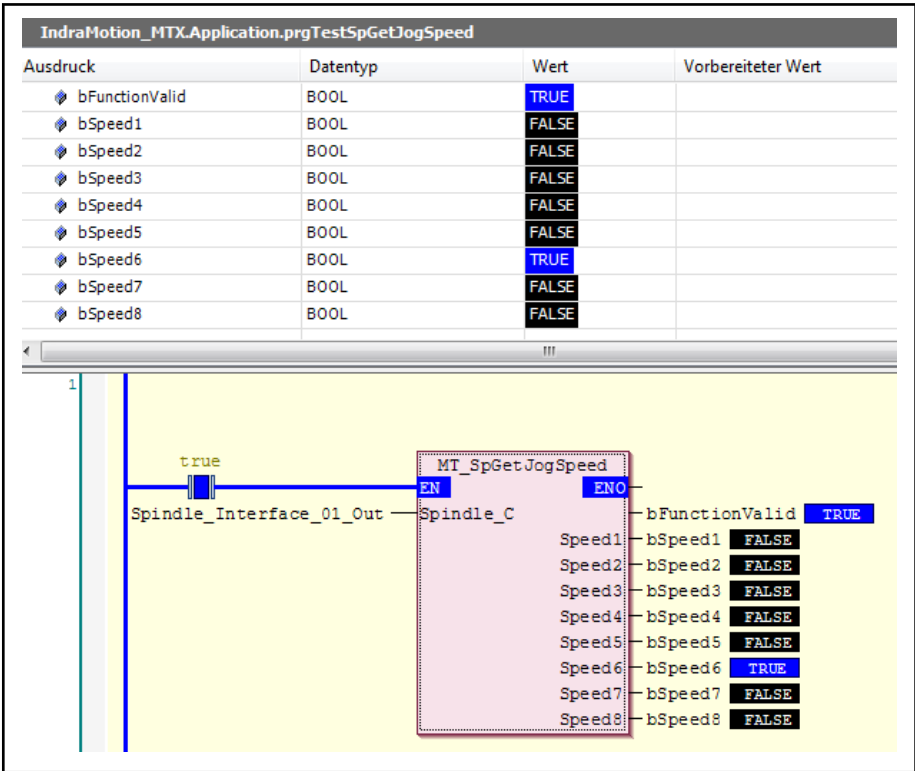


Abb. 14-113: Implementation MT_SpGetJogSpeed

14.5.4 MT_SpSetJogSpeed, Ändern der Spindel-Tippgeschwindigkeit

- Kurzbeschreibung

Funktionsbaustein zum Ändern der Tippgeschwindigkeit einer Spindel.
- Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-93: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

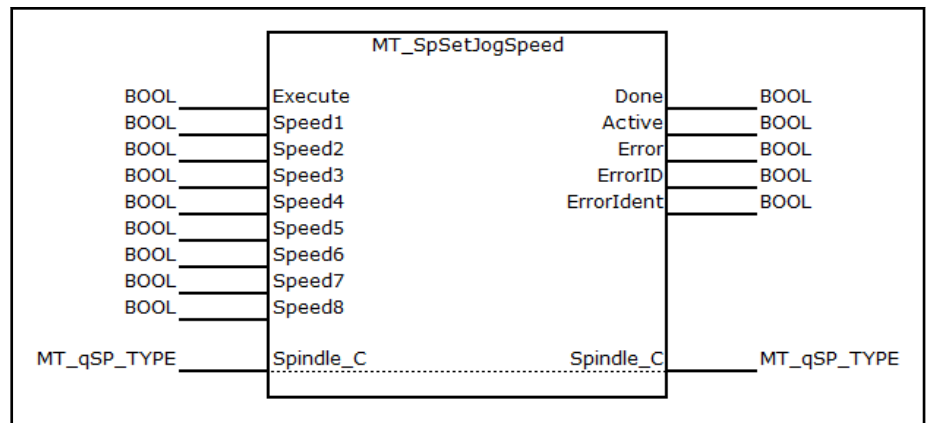


Abb. 14-114: Funktionsbaustein MT_SpSetJogSpeed

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Setzen der vorgewählten Tippgeschwindigkeit ausführen
	Speed1	BOOL	Spindeltippgeschwindigkeit 1
	Speed2	BOOL	Spindeltippgeschwindigkeit 2
	Speed3	BOOL	Spindeltippgeschwindigkeit 3
	Speed4	BOOL	Spindeltippgeschwindigkeit 4
	Speed5	BOOL	Spindeltippgeschwindigkeit 5
	Speed6	BOOL	Spindeltippgeschwindigkeit 6
	Speed7	BOOL	Spindeltippgeschwindigkeit 7
	Speed8	BOOL	Spindeltippgeschwindigkeit 8
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
VAR_IN_OUT	Spindle_C	MT_qSp_Type	Spindel-Interface: Spindel-Steuer-Signale

Tab. 14-94: Ein-/Ausgänge Funktionsbaustein MT_SpSetJogSpeed

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Mit der MTX kann die Spindel in 8 verschiedenen Geschwindigkeiten getippt werden (Maschinenparameter: "SP\SPFUNC\Sp[x]\Jog\Speed" - "MaCoDa-Nr.: 104000015"). Diese 8 Tippgeschwindigkeiten werden ausschließlich von der SPS über das SPS-Spindel-Interface (MT_qSp_Type) durch die Signale "qSp_ManSpeed_00..03" codiert vorgewählt. Mit dem Funktionsbaustein "MT_SpSetJogSpeed" können diese Geschwindigkeiten für eine Spindel aktiviert werden.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren das Ändern der Tippgeschwindigkeit genau einmal ausführt.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein und sichert den Zustand der Eingänge zu diesem Zeitpunkt. Lediglich der Eingang "Spindle_C" wird zyklisch übernommen, solange der Baustein aktiv ist ("Active"==TRUE). Mit dem Signalwechsel muss die gewünschte Tippgeschwindigkeit "Speed1...8" vorgewählt sein. Sind beim Setzen von "Execute" keine oder mehrere Tippgeschwindigkeiten vorgewählt, so geht der Baustein in Fehler ("Error" wird TRUE). Sobald der Ausgang "Done" TRUE wird, ist das Setzen des gewünschten "Speed" erfolgreich abgeschlossen. Konnte der "Speed" nicht geändert werden, wird dies durch den Ausgang "Error" angezeigt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einem erneuten Wechsel des "Speed", sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Solange einer der Ausgänge "Done", "Active" oder "Error" TRUE ist, haben Signaländerungen an den "Speed"-Eingängen keine Auswirkungen. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_SpSetJogSpeed"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010404	Ungültige Eingangsbelegung. Es wurde entweder kein oder mehr als ein "Speed" vorgewählt
0x00040301	Das Ändern der Interface-Signale "qSp_ManSpeed_00..03" ist fehlgeschlagen. Eventuell wurden sie außerhalb des FB überschrieben.

Tab. 14-95: Fehler von "MT_SpSetJogSpeed", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

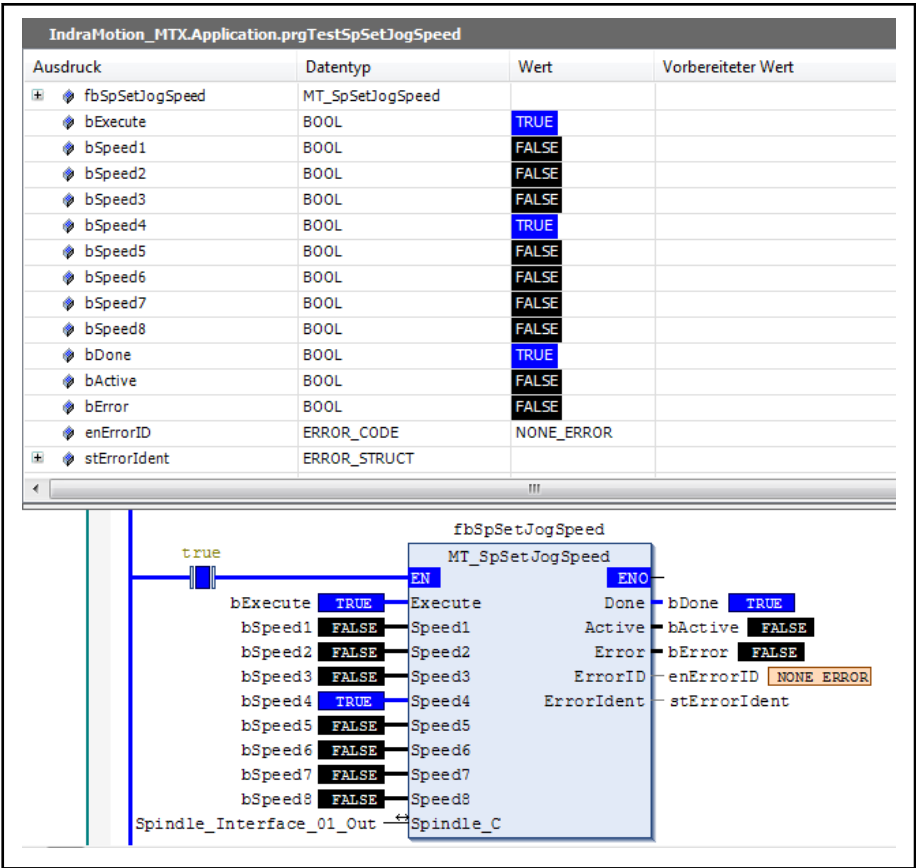


Abb. 14-115: Implementation MT_SpSetJogSpeed

14.5.5 MT_SpJogSpeed, Ändern und Ermitteln der aktiven Tippgeschwindigkeit

Kurzbeschreibung Funktionsbaustein zum Ermitteln der aktuell aktiven Tippgeschwindigkeit, sowie zu deren Änderung.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-96: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

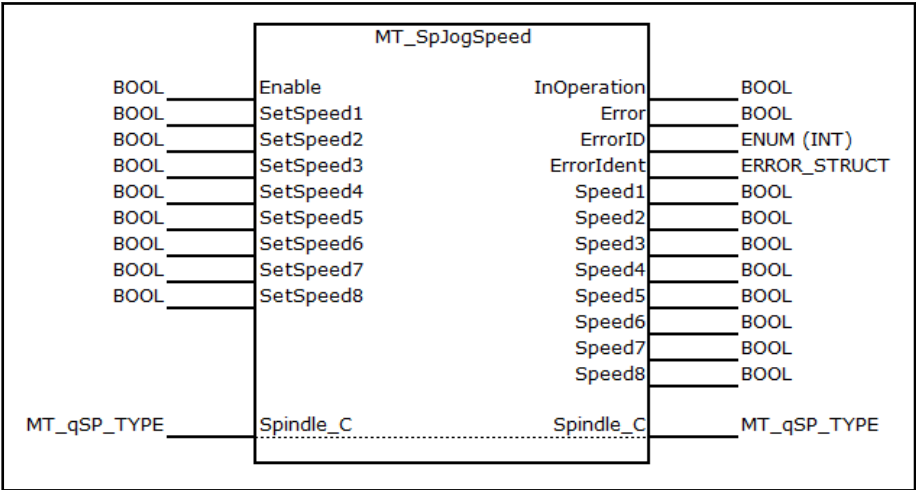


Abb. 14-116: Funktionsbaustein MT_SpJogSpeed

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Baustein aktivieren
	SetSpeed1	BOOL	Spindeltippgeschwindigkeit 1 anfordern
	SetSpeed2	BOOL	Spindeltippgeschwindigkeit 2 anfordern
	SetSpeed3	BOOL	Spindeltippgeschwindigkeit 3 anfordern
	SetSpeed4	BOOL	Spindeltippgeschwindigkeit 4 anfordern
	SetSpeed5	BOOL	Spindeltippgeschwindigkeit 5 anfordern
	SetSpeed6	BOOL	Spindeltippgeschwindigkeit 6 anfordern
	SetSpeed7	BOOL	Spindeltippgeschwindigkeit 7 anfordern
	SetSpeed8	BOOL	Spindeltippgeschwindigkeit 8 anfordern
	InitJogSpeed	BOOL	Vorwahl des JogSpeed bei Einschalten des Bausteins
VAR_OUTPUT	InOperation	BOOL	Bearbeitung läuft
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "Error"=TRUE)
	Speed1	BOOL	Spindeltippgeschwindigkeit 1 ist aktiv
	Speed2	BOOL	Spindeltippgeschwindigkeit 2 ist aktiv
	Speed3	BOOL	Spindeltippgeschwindigkeit 3 ist aktiv
	Speed4	BOOL	Spindeltippgeschwindigkeit 4 ist aktiv
	Speed5	BOOL	Spindeltippgeschwindigkeit 5 ist aktiv
	Speed6	BOOL	Spindeltippgeschwindigkeit 6 ist aktiv
	Speed7	BOOL	Spindeltippgeschwindigkeit 7 ist aktiv
	Speed8	BOOL	Spindeltippgeschwindigkeit 8 ist aktiv
VAR_IN_OUT	Spindle_C	MT_qSp_Type	Spindle-Interface: Spindel-Steuer-Signale

Tab. 14-97: Ein-/Ausgänge Funktionsbaustein MT_SpJogSpeed

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Mit der MTX kann die Spindel in 8 verschiedenen Geschwindigkeiten getippt werden (Maschinenparameter: "SP\SPFUNC\Sp[x]\Jog\Speed" - "MaCoDa-Nr.: 104000015"). Diese 8 Tippgeschwindigkeiten werden ausschließlich von der SPS über das SPS-Spindel-Interface (MT_qSp_Type) durch die Signale "qSp_ManSpeed_00..03" codiert vorgewählt. Mit diesem Funktionsbaustein können diese Tippgeschwindigkeiten für eine Spindel geändert und der aktuelle Status ausgelesen werden.

Sobald "Enable" TRUE ist, wird mit jedem Aufruf des FB die aktuelle Tippgeschwindigkeit aus dem angeschlossenen Spindel-Interface ermittelt. Mit einem Signalwechsel (0-->1) an einem der "SetSpeed... Eingänge" wird die gewünschte Tippgeschwindigkeit geändert. Konnte die Tippgeschwindigkeit nicht geändert werden, wird dies durch den Ausgang "Error" angezeigt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Zum Fehler löschen muss "Enable" FALSE werden.

Der Eingang "InitJogSpeed" erlaubt es, mit dem Einschalten des FB einmalig einen JogSpeed zu setzen, ohne dass dafür einer der "SetJog... Eingänge" TRUE sein muss.

Folgende Vorgaben sind gültig:

InitJogSpeed	JogSpeed
0	Keine Initialisierung
1	JogSpeed1
2	JogSpeed2
3	JogSpeed3
4	JogSpeed4
5	JogSpeed5
6	JogSpeed6
7	JogSpeed7
8	JogSpeed8

Tab. 14-98: Initialisierung JogSpeed

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_SpJogSpeed"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00010404	Ungültige Eingangsbelegung. Es wurde entweder kein oder mehr als ein "Speed" vorgewählt
0x00040301	Das Ändern der Interface-Signale "qSp_ManSpeed_00..03" ist fehlgeschlagen. Eventuell wurden sie außerhalb des FB überschrieben

Tab. 14-99: Fehler von "MT_SpSetJogSpeed", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

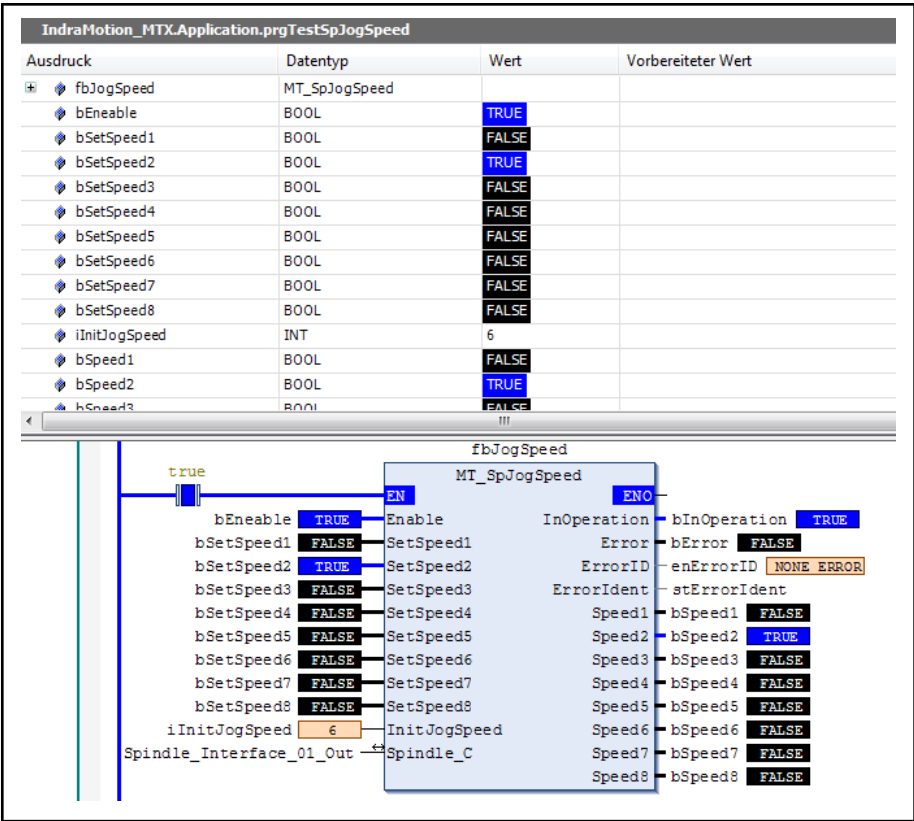


Abb. 14-117: Implementation MT_SpJogSpeed

14.5.6 MT_SpGetPowerState, Antriebsstatus ermitteln

Kurzbeschreibung

Funktion zum Ermitteln des aktiven Antriebsstatus einer Spindel.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-100: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

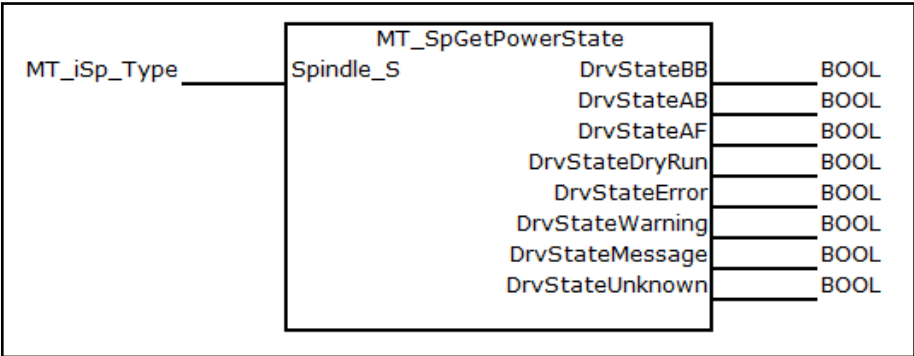


Abb. 14-118: Funktionsbaustein MT_SpGetPowerState

	Name	Typ	Beschreibung
VAR_INPUT	Spindle_S	MT_iSP_TYPE	Achs-Interface: Spindle-Status-Signale
VAR_OUTPUT	DrvStateBB	BOOL	Antrieb Betriebsbereit
	DrvStateAB	BOOL	Antrieb bereit zur Reglerfreigabe
	DrvStateAF	BOOL	Antrieb in Regelung
	DrvStateDryRun	BOOL	Antrieb im virtuellen Betrieb
	DrvStateError	BOOL	Antrieb meldet einen Fehler
	DrvStateWarning	BOOL	Antrieb meldet eine Warnung
	DrvStateMessage	BOOL	Antrieb meldet einen Hinweis

Tab. 14-101: Ein-/Ausgänge Funktionsbaustein MT_SpGetPowerState

Signal-Zeit-Diagramm	Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.
Funktionsbeschreibung	Diese Funktion ermittelt aus dem Achs-Interface (MT_iSp_Type) den Leistungszustand und zeigt an, ob Fehler, Warnungen oder Hinweise am Antrieb anstehen. Dazu werden folgende Interface-Signale interpretiert: • iSp_DrvPower • iSp_DrvReady • iSp_DrvAct • iSp_DryRun • iSp_DrvErrClass1 • iSp_DrvChangeClass2 • iSp_DrvChangeClass3
Fehlerbehandlung	Für diese Funktion ist keine Fehlerbehandlung notwendig.
Implementationsbeispiel	Im Beispielprogramm wird der Status einer Spindel ermittelt.

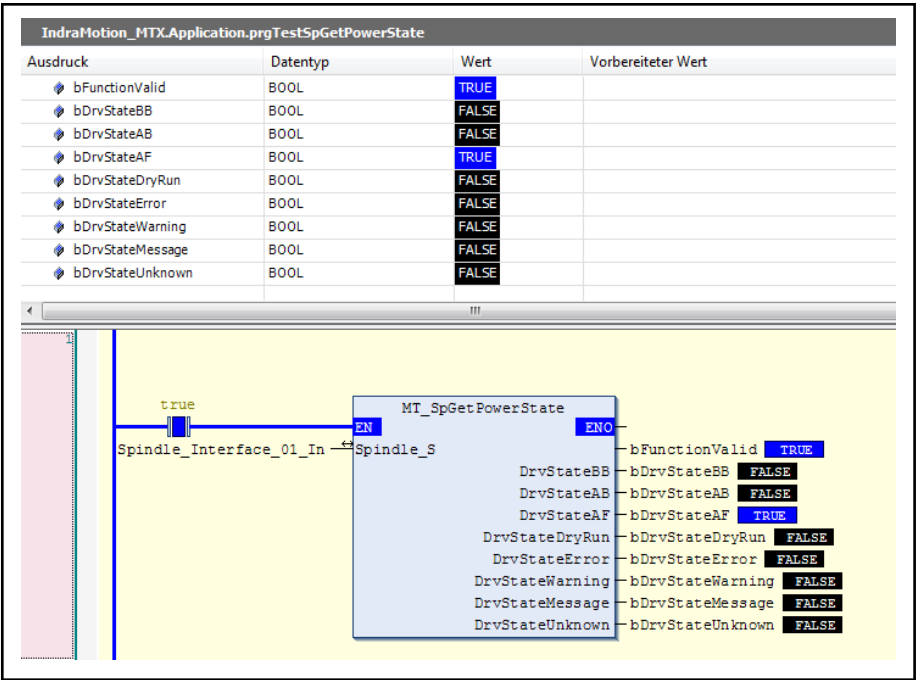


Abb. 14-119: Implementation MT_SpGetPowerState

14.5.7 MT_SpGetPowerStateExt, Antriebsstatus incl. Fehler- und Warnungs-
details ermitteln

Kurzbeschreibung Funktion zum Ermitteln des aktiven Antriebsstatus einer Spindel mit zusätzli-
chen Informationen zu Fehler und Warnungen.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-102: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

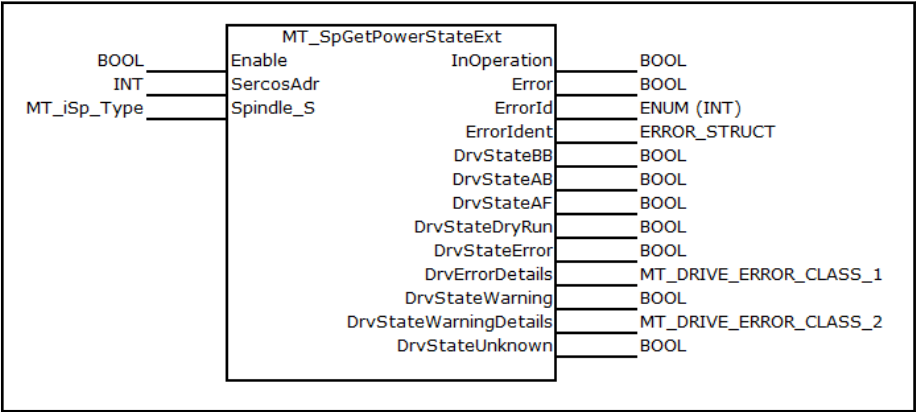


Abb. 14-120: Funktionsbaustein MT_SpGetPowerStateExt

	Name	Typ	Beschreibung
VAR_INPUT	Spindle_S	MT_iSP_TYPE	Achs-Interface: Spindel-Status-Signale
VAR_OUTPUT	DrvStateBB	BOOL	Antrieb Betriebsbereit
	DrvStateAB	BOOL	Antrieb bereit zur Reglerfreigabe
	DrvStateAF	BOOL	Antrieb in Regelung
	DrvStateDryRun	BOOL	Antrieb im virtuellen Betrieb
	DrvStateError	BOOL	Antrieb meldet einen Fehler
	DrvStateErrorDetails	MT_DRIVE_ERROR_CLASS_1	Detailinformationen zum Fehler (S-0-0011)
	DrvStateWarning	BOOL	Antrieb meldet eine Warnung
	DrvStateWarningDetails	MT_DRIVE_ERROR_CLASS_2	Detailinformationen zur Warnung (S-0-0012)

Tab. 14-103: Ein-/Ausgänge Funktionsbaustein MT_SpGetPowerStateExt

Signal-Zeit-Diagramm	Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.
Funktionsbeschreibung	Dieser Funktionsbaustein ermittelt aus dem Achs-Interface (MT_iSp_Type) den Leistungszustand und zeigt an, ob ein Fehler oder eine Warnung am Antrieb ansteht. Zudem werden bei anstehendem Fehler oder Warnung jede Sekunde Detailinformationen aus den Antriebsparametern S-0-0011 bzw. S-0-0012 ausgegeben. Dazu werden folgende Interfacesignale interpretiert: • iSp_DrvPower • iSp_DrvReady • iSp_DrvAct • iSp_DryRun • iSp_DrvErrClass1 • iSp_DrvChangeClass2 • iSp_DrvChangeClass3
Fehlerbehandlung	Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Der Funktionsbaustein zeigt Fehler des intern genutzten Funktionsbausteines "IL_SIIISvcRead" aus der Bibliothek "RIL_SercosIII" an.
Implementationsbeispiel	Im Beispielprogramm wird der Status einer Spindel ermittelt.

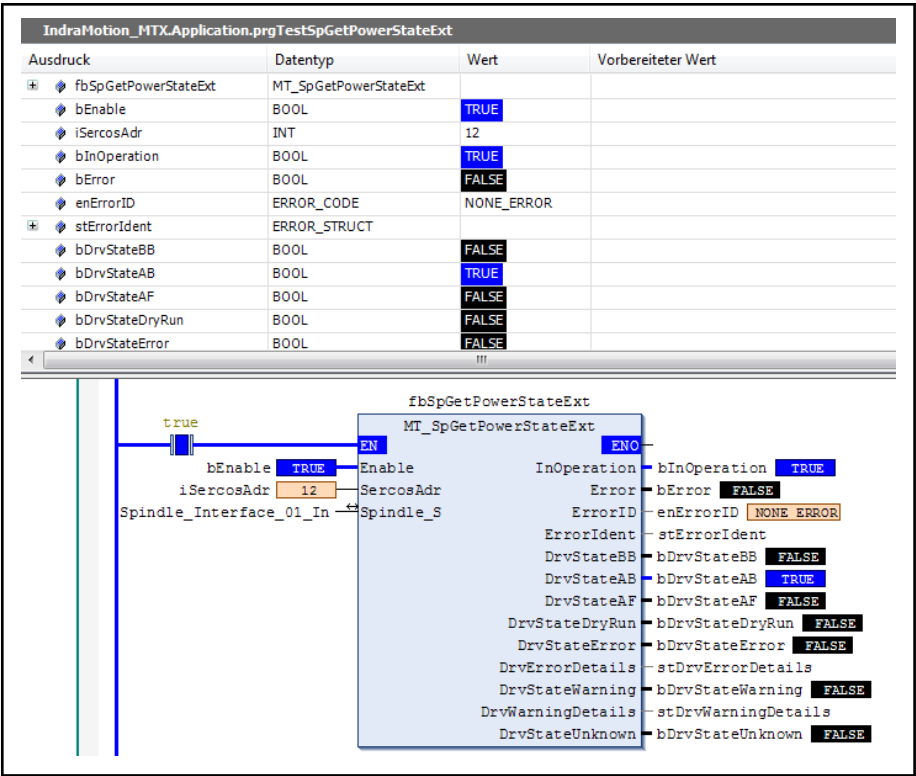


Abb. 14-121: Implementation MT_SpGetPowerStateExt

14.5.8 MT_SpSetPowerOn, Setzt die Reglerfreigabe für eine Spindel

Kurzbeschreibung

Zuordnung Zielsystem/Bibliothek

Funktionsbaustein zum Aktivieren der Reglerfreigabe für eine Spindel.

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-104: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

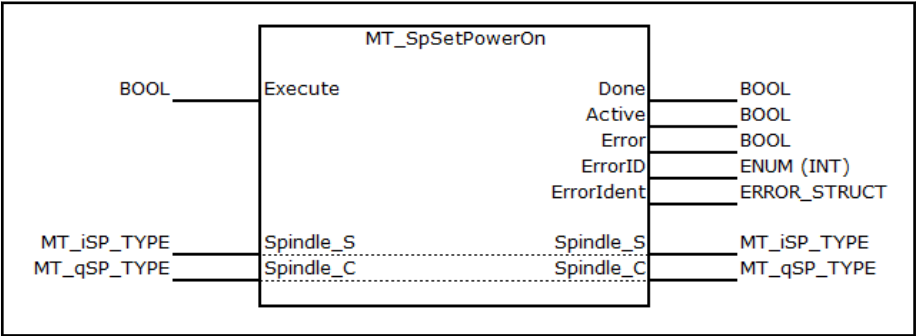


Abb. 14-122: Funktionsbaustein MT_SpSetPowerOn

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Reglerfreigabe für Spindel setzen
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
VAR_IN_OUT	Spindle_S	MT_iSp_Type	Achs-Interface: Spindel-Status-Signale
	Spindle_C	MT_qSp_Type	Achs-Interface: Spindel-Steuer-Signale

Tab. 14-105: Ein-/Ausgänge Funktionsbaustein MT_SpSetPowerOn

Signal-Zeit-Diagramm Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung Das Setzen der Reglerfreigabe (AF) für eine Spindel erfolgt bei der MTX im NC-PLC-Achs-Interface (MT_qSp_Type) über das Achssteuersignal "qSp_DrvOn". Der Baustein prüft erst, ob die Spindel bereit zur Reglerfreigabe ist, setzt dann dieses Signal und überwacht, ob innerhalb von 5 Sekunden die Rückmeldung über Achsstatussignale (MT_iSp_Type) kommt, dass die Spindel in Regelung ist.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren das Setzen der Reglerfreigabe "qSp_DrvOn" genau einmal ausführt.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein. Die Eingänge "Spindle_S" und "Spindle_C" werden zyklisch übernommen, solange der Baustein aktiv ist ("Active"==TRUE). Wenn der Ausgang "Done" TRUE wird, ist das Setzen der Reglerfreigabe erfolgreich abgeschlossen. Konnte für den Antrieb die Reglerfreigabe nicht gesetzt werden, wird dies durch den Ausgang "Error" angezeigt. Im Fehlerfall wird das Interface-Signal "qSp_DrvOn" zurückgesetzt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einem erneuten Setzen der Reglerfreigabe, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d. h. es ist immer nur einer der Ausgänge TRUE.

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_SpSetPowerOn"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00060302	Antrieb nicht bereit zur Reglerfreigabe, Signalstatus (iSp_DrvReady), der Antrieb muss im Status AB sein

Fehlercode (Hex)	Bedeutung
0x00040301	Das Setzen des Achs-Interface-Signals (qSp_DrvOn) ist fehlgeschlagen, das Signal wurde außerhalb des FB zurückgesetzt
0x00020301	Reglerfreigabe wurde angefordert aber es erfolgte keine Rückmeldung vom Antrieb (Achs-Interface-Signal: iSp_DrvAct), eventuell wurde das falsche Achs-Interface am FB angeschlossen

Tab. 14-106: Fehler von "MT_SpSetPowerOn", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

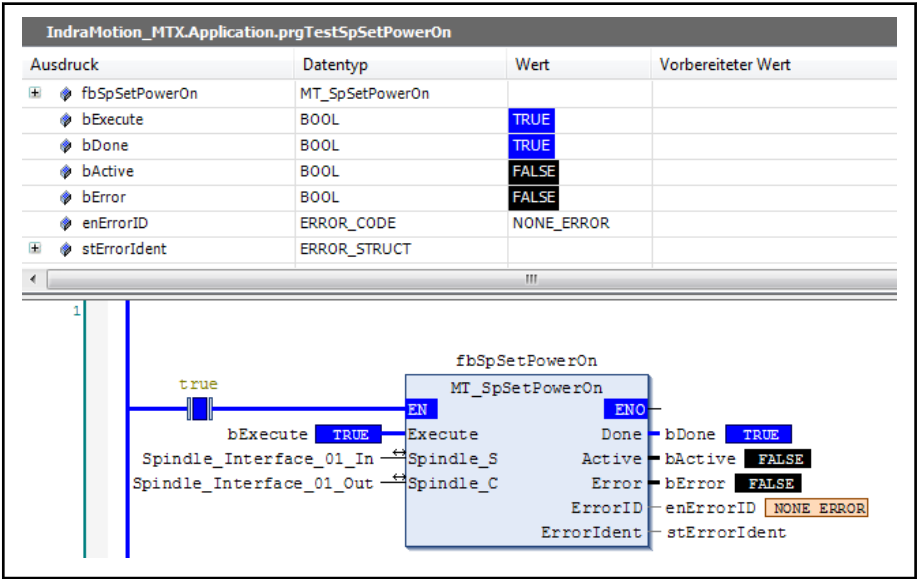


Abb. 14-123: Implementation MT_SpSetPowerOn

14.5.9 MT_SpSetPowerOff, Setzt die Reglerfreigabe für eine Spindel

Kurzbeschreibung

Funktionsbaustein zum Deaktivieren der Reglerfreigabe für eine Spindel.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-107: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

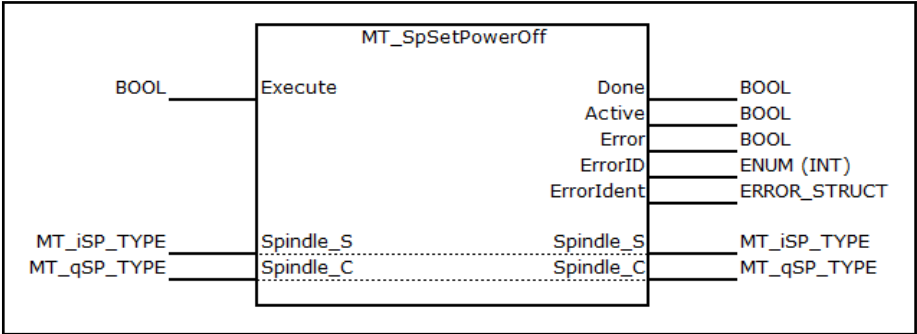


Abb. 14-124: Funktionsbaustein MT_SpSetPowerOff

	Name	Typ	Beschreibung
VAR_INPUT	Execute	BOOL	Reglerfreigabe für Spindel zurücksetzen
VAR_OUTPUT	Done	BOOL	Bearbeitung erfolgreich abgeschlossen
	Active	BOOL	Bearbeitung aktiv
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
VAR_IN_OUT	Spindle_S	MT_iSp_Type	Achs-Interface: Spindel-Status-Signale
	Spindle_C	MT_qSp_Type	Achs-Interface: Spindel-Steuer-Signale

Tab. 14-108: Ein-/Ausgänge Funktionsbaustein MT_SpSetPowerOff

Signal-Zeit-Diagramm Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung Das Rücksetzen der Reglerfreigabe (AF) für eine Spindel erfolgt bei der MTX im NC-PLC-Achs-Interface (MT_qSp_Type) über das Achssteuersignal "qSp_DrvOn". Der Baustein setzt dieses Signal auf FALSE und überwacht, ob innerhalb von 5 Sekunden die Rückmeldung über Achsstatussignale (MT_iSp_Type) kommt, dass die Spindel nicht mehr in Regelung ist.



Dieser Funktionsbaustein arbeitet flankengesteuert. Das bedeutet, dass der Funktionsbaustein nach dem Aktivieren das Rücksetzen der Reglerfreigabe "qSp_DrvOn" genau einmal ausführt.

Ein Signalwechsel von Null nach Eins (0 -> 1) an "Execute" aktiviert den Funktionsbaustein. Die Eingänge "Spindle_S" und "Spindle_C" werden zyklisch übernommen, solange der Baustein aktiv ist ("Active"==TRUE). Wenn der Ausgang "Done" TRUE wird, ist das Rücksetzen der Reglerfreigabe erfolgreich abgeschlossen. Konnte für den Antrieb die Reglerfreigabe nicht zurückgesetzt werden, wird dies durch den Ausgang "Error" angezeigt. Im Fehlerfall bleibt das Interface-Signal "qSp_DrvOn" FALSE. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent". Vor einem erneuten Rücksetzen der Reglerfreigabe, sowie zum Fehler löschen, muss "Execute" FALSE werden. Mit dem Rücksetzen von "Execute" werden "Error" und "Done" gelöscht. Die Ausgänge "Done", "Active" und "Error" schließen einander aus, d.h. es ist immer nur einer der Ausgänge TRUE.

Fehlerbehandlung Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_SetPowerOn"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00040301	Das Rücksetzen des Achs-Interface-Signals (qSp_DrvOn) ist fehlgeschlagen, das Signal wurde außerhalb des FB wieder gesetzt

Fehlercode (Hex)	Bedeutung
0x00020301	Das Rücksetzen der Reglerfreigabe wurde angefordert, aber es erfolgte keine Rückmeldung vom Antrieb (Achs-Interface-Signal: iSp_DrvAct), eventuell wurde das falsche Achs-Interface am FB angeschlossen)

Tab. 14-109: Fehler von "MT_SpSetPowerOff", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

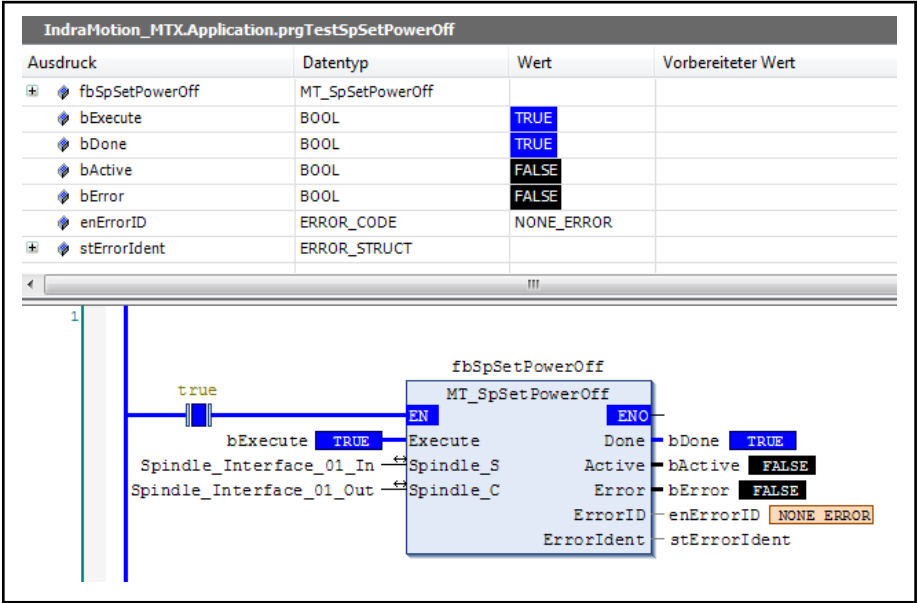


Abb. 14-125: Implementation MT_SpSetPowerOff

14.5.10 MT_SpPower, Reglerfreigabe und Status einer Spindel

Kurzbeschreibung Funktionsbaustein zum Aktivieren/Deaktivieren der Reglerfreigabe, sowie zum Ermitteln des aktuellen Status einer Spindel.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-110: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

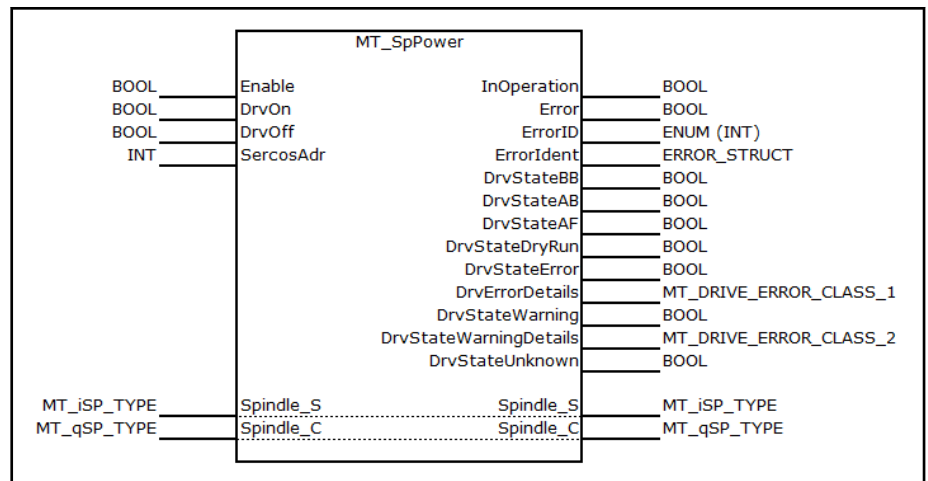


Abb. 14-126: Funktionsbaustein MT_SpPower

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Baustein aktivieren
	DrvOn	BOOL	Setzt Reglerfreigabe (DrvOff muss ebenfalls TRUE sein)
	DrvOff	BOOL	Fallende Flanke setzt Reglerfreigabe zurück
	SercosAdr	INT	Sercos-Adresse des Antriebs
VAR_OUTPUT	InOperation	BOOL	Bearbeitung erfolgreich abgeschlossen
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig, wenn "bError"=TRUE)
	DrvStateBB	BOOL	Antrieb Betriebsbereit
	DrvStateAB	BOOL	Antrieb bereit zur Reglerfreigabe
	DrvStateAF	BOOL	Antrieb in Regelung
	DrvStateDryRun	BOOL	Antrieb im virtuellen Betrieb
	DrvStateError	BOOL	Antrieb meldet einen Fehler
	DrvStateErrorDetails	MT_DRIVE_ERROR_CLASS_1	Detailinformationen zum Fehler (S-0-0011)
	DrvStateWarning	BOOL	Antrieb meldet eine Warnung
	DrvStateWarningDetails	MT_DRIVE_ERROR_CLASS_2	Detailinformationen zur Warnung (S-0-0012)
VAR_IN_OUT	Spindle_S	MT_iSp_Type	Achs-Interface: Spindel-Status-Signale
	Spindle_C	MT_qSp_Type	Achs-Interface: Spindel-Steuer-Signale

Tab. 14-111: Ein-/Ausgänge Funktionsbaustein MT_SpPower

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Dieser Baustein vereint die Funktionalität von "MT_SpSetPowerOn", "MT_SpSetPowerOff", "MT_SpGetPowerStateExt". Der Baustein kann die

Reglerfreigabe für eine Spindel steuern, sowie deren Status zyklisch ermitteln und bei Warnungen oder Fehlern zusätzliche Detailinformationen liefern.

Mit dem Setzen von "Enable" werden alle Eingänge zyklisch gelesen, laufend der Antriebsstatus aus dem Spindel-Interface ermittelt und über die Ausgänge "DrvState..." angezeigt. Solange "Enable" und "DrvOff" TRUE sind, kann mit einer steigenden Flanke an "DrvOn" das Setzen der Reglerfreigabe für die Spindel angefordert werden. Das Rücksetzen der Reglerfreigabe erfolgt über eine fallende Flanke an DrvOff. Solange sich die Spindel im Zustand "DrvStateDryRun" befindet, sind "DrvOn" und "DrvOff" wirkungslos.

Das Setzen und Rücksetzen der Reglerfreigabe (AF) erfolgt im NC-PLC-Achs-Interface (MT_qSp_Type) über das Achssteuersignal "qSp_DrvOn". Vor dem Setzen der Reglerfreigabe wird geprüft, ob die Achse bereit zur Reglerfreigabe ist (AB). Erst dann wird dieses Signal gesetzt. Beim Setzen wie beim Rücksetzen der Reglerfreigabe wird überwacht, ob innerhalb von 5 Sekunden die Rückmeldung über Achsstatussignale (MT_iSp_Type) erfolgt.

Solange der Ausgang "InOperation" TRUE ist, arbeitet der FB fehlerfrei. Kommt es zu einem Fehler, wird dies durch den Ausgang "Error" angezeigt und "InOperation" wird FALSE. Im Fehlerfall wird das Interface-Signal "qSp_DrvOn" zurückgesetzt. Solange "Error" TRUE ist, findet man einen gültigen Fehlercode unter "ErrorIdent".

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Zusätzlich zu den unten aufgezählten Fehlern zeigt der Baustein Fehler des intern genutzten Funktionsbausteines "IL_SIIISvcRead" aus der Bibliothek "RIL_SercosIII" (F_RELATED_TABLE) an, sowie Fehler des Sercos-Servicekanals Fehlertabelle (SERCOS_TABLE).

Fehler von "MT_SpPower", MTX_TABLE

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar
0x00060302	Antrieb nicht bereit zur Reglerfreigabe, Signalstatus (iSp_DrvReady), der Antrieb muss im Status AB sein
0x00040301	Das Setzen oder Rücksetzen des Achs-Interface-Signals (qSp_DrvOn) ist fehlgeschlagen, das Signal wurde außerhalb des FB geändert
0x00020301	Setzen/Rücksetzen der Reglerfreigabe wurde angefordert, aber es erfolgte keine Rückmeldung vom Antrieb (Achs-Interface-Signal: iSp_DrvAct), eventuell wurde das falsche Achs-Interface am FB angeschlossen

Tab. 14-112: Fehler von "MT_SpPower", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

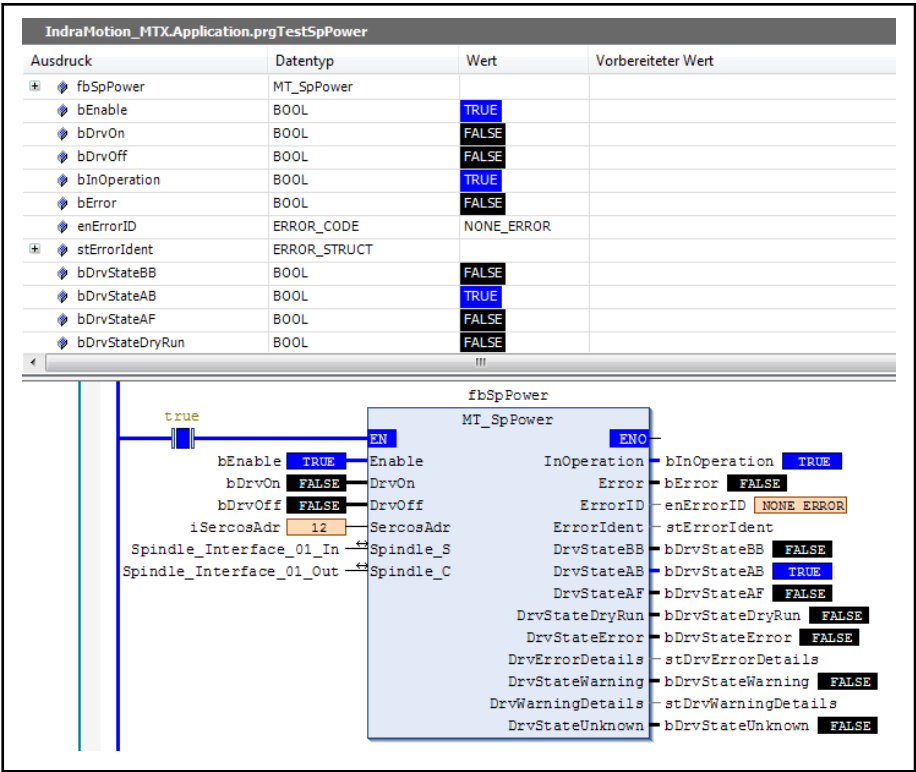


Abb. 14-127: Implementation MT_SpPower

14.6 Werkzeugmanagement

14.6.1 Übersicht

Übersicht Werkzeug-Management-Funktionen		
Kategorie	Funktion/Funktionsbaustein	Kurzbeschreibung
Standzeit	MT_TmToolLife, Seite 339	Ermitteln der Standzeit für Werkzeuge

Tab. 14-113: Übersicht Werkzeug-Management-Funktionen

14.6.2 MT_TmToolLife, Werkzeug Standzeitzählung

Kurzbeschreibung Funktionsbaustein berechnet die Standzeit der aktiven Schneide des aktiven Werkzeugs und zeigt den Verschleißstatus an.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-114: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

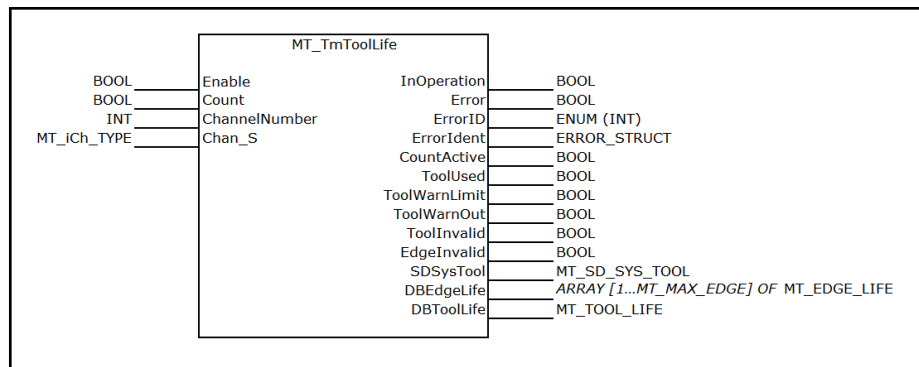


Abb. 14-128: Funktionsbaustein MT_TmToolLife

	Name	Typ	Init	Beschreibung
VAR_INPUT	Enable	BOOL	FALSE	Baustein aktivieren
	Count	BOOL	FALSE	Standzeit zählen/Quittierung durch FB wenn fertig
	ChannelNumber	INT	1	Kanal, auf welchen die Standzeit­zählung wirken soll
VAR_IN_OUT	Chan_S	MT_ICH_TYPE		Kanalinterface Eingänge, für den Kanal auf welchen die Standzeit­zählung wirken soll
VAR_OUT-PUT	InOperation	BOOL	FALSE	Bearbeitung läuft
	Error	BOOL	FALSE	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	NO-NE_ERROR	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	--	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
	CountActive	BOOL	FALSE	
	ToolUsed	BOOL	FALSE	Das aktive Werkzeug wurde bereits verwendet
	ToolWarnLimit	BOOL	FALSE	Das aktive Werkzeug hat die Warn­grenze erreicht
	ToolWornOut	BOOL	FALSE	Das aktive Werkzeug ist verschlissen
	ToolInvalid	BOOL	FALSE	Aktives Werkzeug ungültig, "SD.SysTool.ActToolValid=FALSE"
	EdgeInvalid	BOOL	FALSE	Aktive Schneide ungültig, "SD.SysTool.ActEd<=0 OR >MT_MAX_ED"
	SDSysTool	MT_SD_SYS_TOOL	Kap. 14.8.3 "MT_SD_SYS_TOOL, Struktur von Systemdatum SD.SysTool" auf Seite 353	Daten des aktiven Werkzeug aus dem Systemdatum SD.SysTool

	Name	Typ	Init	Beschreibung
	DBEdgeLife	ARRAY [1..MT_MAX_EDGE] OF MT_EDGE_LIFE		Verschleißdaten aller Schneiden des aktiven Werkzeugen
	DBToolLife	MT_TOOL_LIFE		Verschleiß des aktiven Werkzeug sowie die Information ob die Zählung nach Zeit oder in Zyklen erfolgt

Tab. 14-115: Ein-/Ausgänge Funktionsbaustein MT_TmToolLife

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung**Voraussetzungen:**

Die Grundvoraussetzung zur Verwendung diese Bausteins ist, dass im Werkzeugwechselprogramm das Systemdatum "SD.SysTool" korrekt beschrieben wird. Bei Verwendung des von Bosch Rexroth bereitgestellten Standardwerkzeugwechsel "TCH" ist dies bereits sichergestellt.

Das Systemdatum "SD.SysTool" ist in der Dokumentation der "MTX 14VRS Maschinenparameter" genauer beschrieben.

Ist durch das Werkzeugwechselprogramm ein gültiger Werkzeugdatensatz in "SD.SysTool.ActTool" hinterlegt, muss dies durch Setzen von "SD.SysTool.ActToolValid" bestätigt werden. Andernfalls findet keine Standzeitählung statt und der Ausgang "ToolInvalid" ist TRUE.

Werkzeuge können unterschiedlich viele Schneiden besitzen. Das Werkzeug, welches die höchste Anzahl an Schneiden hat, bestimmt die maximale Schneidenzahl. Dieser Wert wird für die NC in der Werkzeug-Userdatenstruktur "dbt1ud.xsd" definiert und für die SPS durch die globale Konstante "MT_MAX_EDGE". Die maximale Schneidenzahl für NC und SPS sollten übereinstimmen! Ist die in "SD.SysTool.ActTool" eingetragene Schneidenzahl Null oder größer als die in "MT_MAX_EDGE" angegebene Schneidenzahl, dann wird der Ausgang "EdgeInvalid" TRUE und es findet keine Standzeitählung statt!



Die änderbare SPS-Konstante "MT_MAX_EDGE" ist Bestandteil der Bibliothek "MT_Base" und befindet sich im Ordner "Constants/MT_ToolManagement_Const".

Funktionsweise:

Dieser Funktionsbaustein zählt die Standzeit für die aktive Schneide des aktiven Werkzeuges.

Die aktive Schneide wird aus dem Kanalinterfacesignal MT_iCh_Type.iCh_ActFunc09..13 ermittelt und das aktive Werkzeug aus dem Systemdatum "SD.SysTool".

Ob die Standzeit auf der Basis von Zyklen oder der Zeit berechnet werden soll, liest der Baustein aus der Datenbank (/DBT1/Rec/UD/TI/ Life) und zeigt dies am Ausgang "DBToolLife" an. Wird in Zyklen gezählt, so reduziert sich mit jeder steigenden Flanke am Eingang "Count" die Standzeit um eins. Wird jedoch als Berechnungsbasis die Zeit verwendet, so reduziert sich die Standzeit alle 10 Millisekunden um 0,0001, solange der Eingang "Count" TRUE ist.

Die neu berechnete Standzeit für die Schneiden wird bei Änderung jede Sekunde in der Datenbank aktualisiert und am FB Ausgang "DBEdgeLife" ausgegeben.

Es gibt für jede Schneide eine Standzeit (/DBT1/Rec/UD/Ed[x]/Life/RemLife), sowie eine Standzeit für das gesamte Werkzeug (DBT1/Rec/UD/TI/ Life/ RemLife). Die kleinste Standzeit einer Schneide entspricht der Standzeit des Werkzeugs. Der Baustein prüft dies und aktualisiert die Standzeit für das Werkzeug entsprechend. Bei Änderung wird das Ergebnis jede Sekunde in die Datenbank übertragen und am FB Ausgang "DBToolLife" angezeigt.

Entsprechend der ermittelten Standzeit und der in der Datenbank hinterlegten Grenzwerte (/DBT1/Rec/UD/Ed[x]/Life/WarnLife & ... MaxLife) prüft der Baustein, ob das Werkzeug bereits verwendet wurde, die Warngrenze erreicht oder das Werkzeug bereits verschlissen ist. Dabei gilt, ist die aktuelle Standzeit kleiner als die maximale Standzeit (MaxLife), wird der Ausgang "ToolUsed" TRUE. Ist die aktuelle Standzeit kleiner als die vorgegebene Warngrenze (WarnLife), wird der Ausgang "ToolWarnLimit" TRUE. Ist die aktuelle Standzeit Null, dann ist das Werkzeug verschlissen und der Ausgang "ToolWornOut" ist TRUE.

Zudem werden die berechneten Zustände "Werkzeug verwendet", "Warngrenze erreicht" und "Werkzeug verschlissen" bei Änderung jede Sekunde in die Datenbank geschrieben (/DBT1/Rec/Hd/BQ2).



Wurde in der Datenbank die maximale Standzeit mit Null angegeben (/DBT1/Rec/UD/Ed[x]/Life/MaxLife=0), dann wird für diese Schneide die Standzeit nicht gezählt! Damit ist es möglich, die Standzeitermittlung für einzelne Schneiden bzw. Werkzeuge aus der HMI heraus zu deaktivieren ohne die Reststandzeit anpassen zu müssen.

Zusätzlich werden alle Daten des aktiven Werkzeuges aus dem SystemDatum "SD.SysTool" am FB Ausgang "SDSysTool" ausgegeben,

Fehlerbehandlung

Dieser Funktionsbaustein generiert im Fehlerfall eine detaillierte Diagnose. Eine genaue Beschreibung des ausgegebenen Fehlercodes (Additional1, Additional2) finden Sie in der zugehörigen Tabelle (Table). Dieser Funktionsbaustein kann Fehlercodes für folgende Tabellen ausgeben:

Diagnosen

Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von „MT_TmToolLifeControl“

Fehlercode (Add1)	Fehlercode (Add2)	Bedeutung
0x00000000	0x00000000	Kein Fehlercode verfügbar
16#0009041B	Statusmeldung MT_DbData	Fehler von FB MT_DbData, Code in Additional 2 ist eine Statusmeldung des FB MT_DbData, siehe Beschreibung MT_DbData

MT_BaseAdditional.library

```

1 PROGRAM pr7000ToolManagement
2 (*example for tool count
3 M0296 == enable FB
4 M0297 == disable FB
5 *)
6 VAR
7 //function block for ToolLife control
8 -----
9 fbToolCount : MT_TmToolLife; //functionblock
10 //input
11 bEnableFB : BOOL:=FALSE; //enable FB
12 bCount : BOOL:=FALSE; //start count
13 iChanNo : UINT:=1; //channel number
14 //output
15 bError : BOOL:=FALSE; //functionblock error occur, for detailed infomation view ErrorIdent
16 ErrorID : ERROR_CODE; //basic error information
17 ErrorIdent : ERROR_STRUCT; //detailed error information
18 CountActive : BOOL := FALSE; //Life count active
19 ToolUsed : BOOL := FALSE; //Active tool has already been used
20 ToolWarnLimit : BOOL := FALSE; //Active tool has reached the warning level
21 ToolWarnOut : BOOL := FALSE; //Active Tool is worn out
22 ToolInvalid : BOOL := FALSE; //no count, Act.Tool is invalid "SD.SysTool.ActToolValid=FALSE"
23 EdgeInvalid : BOOL := FALSE; //no count, Act.Tool has no valid edge "SD.SysTool.ActEdc=0 or >MT_MAX_EDGE"
24 SDSysTool : MT_SD_SYS_TOOL; //Value SD.SysTool, contains active tool and active edge (basis for database access)
25 DBEdgeLife : ARRAY [0..MT_MAX_EDGE] OF MT_EDGE_LIFE; //new calculated edge life data
26 DBToolLife : MT_TOOL_LIFE; //new calculated tool life data and the cyc/min information
27 END_VAR
28

```

Abb. 14-131: Deklaration MT_TmToolLife

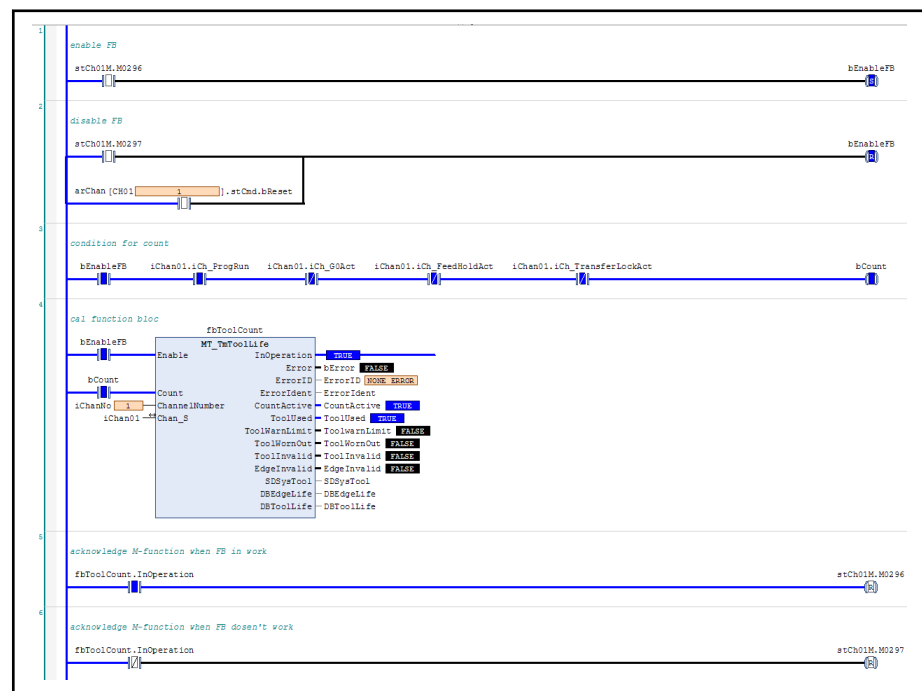


Abb. 14-132: Implementation MT_TmToolLife

fbToolCount	MT_TmToolLife	TRUE	functionblo
bEnableFB	BOOL	TRUE	enable FB
bCount	BOOL	TRUE	start count
iChanNo	UINT	1	channel number
bError	BOOL	FALSE	functionblo error occur, for detailed information view ErrorIdent
ErrorID	ERROR_CODE	NONE_ERROR	basic error information
ErrorIdent	ERROR_STRUCT		detailed error information
CountActive	BOOL	TRUE	Life count active
ToolUsed	BOOL	TRUE	Active tool has already been used
ToolWarnLimit	BOOL	FALSE	Active tool has reached the warning level
ToolWornOut	BOOL	FALSE	Active Tool is worn out
ToolInvalid	BOOL	FALSE	no count, Act.Tool is invalid "SD.SysTool.ActToolValid=FALSE"
EdgeInvalid	BOOL	FALSE	no count, Act.Tool has no valid edge "SD.SysTool.ActEd<=0 or >MT_MAX_EDGE"
SDSysTool	MT_SD_SYS_TOOL		Value SD.SysTool, contains active tool and active edge (basis for database access)
ActToolValid	BOOL	TRUE	
ActEd	DINT	1	
ActTool	MT_RECORD_HEADER_DATA		
K1	INT	1	Key1
K2	INT	1	Key2
SKQ	STRING(31)	"Stufenbohrer 15/5 mm"	String
JKQ1	DINT	0	Integerwert 1 (Schlüssel-Wert)
JKQ2	DINT	1005	Integerwert 1 (Schlüssel-Wert)
JKQ3	DINT	18	Integerwert 1 (Schlüssel-Wert)
IQ1	DINT	0	Integerwert 1
IQ2	DINT	0	Integerwert 1
IQ3	DINT	0	Integerwert 1
BQ1	DWORD	0	Bitfeld 1
BQ2	DWORD	6	Bitfeld 1
BQ3	DWORD	27788289	Bitfeld 1
PreTool	MT_RECORD_HEADER_DATA		
DBEdgeLife	ARRAY [0..MT_MAX_EDGE] OF MT_EDGE_LIFE		new calculated edge life data
DBEdgeLife[0]	MT_EDGE_LIFE		
DBEdgeLife[1]	MT_EDGE_LIFE		
MonOn	BOOL	TRUE	
RemLife	REAL	99.3207	
WarnLife	REAL	20	
MaxLife	REAL	100	
DBEdgeLife[2]	MT_EDGE_LIFE		
MonOn	BOOL	TRUE	
RemLife	REAL	97.57322	
WarnLife	REAL	20	
MaxLife	REAL	100	
DBToolLife	MT_TOOL_LIFE		new calculated tool life data and the cyc/min information
LifeU	STRING(3)	'min'	
RemTLife	REAL	97.57322	

Abb. 14-133: Beispieldaten SDSysTool, DBEdgeLife, DBToolLife wenn MT_Tool-Life aktiv

14.6.3 MT_TmChGetActiveEdge, Aktive Schneide des aktiven Werkzeug ermitteln

Kurzbeschreibung

Funktion um die aktive Schneide des aktiven Werkzeuges zu ermitteln.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-117: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

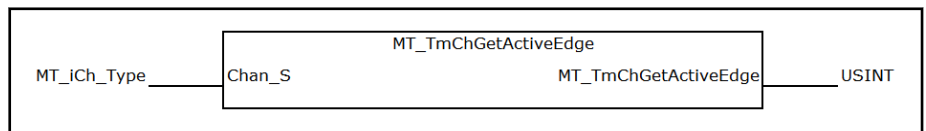


Abb. 14-134: Funktionsbaustein MT_TmChGetActiveEdge

Signal-Zeit-Diagramm

Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung

Die aktive Schneide des aktiven Werkzeuges wird über das Kanalinterface (MT_iCh_Type) durch die Signale "iCh_ActFunc09...13" von der NC an die SPS übertragen. Diese Funktion dekodiert diese Information und zeigt das Ergebnis am Ausgang der Funktion an.

Fehlerbehandlung

Für diese Funktion ist keine Fehlerbehandlung notwendig.

Implementationsbeispiel

Im Beispielpogramm wird die Betriebsart des ersten Kanals ermittelt.

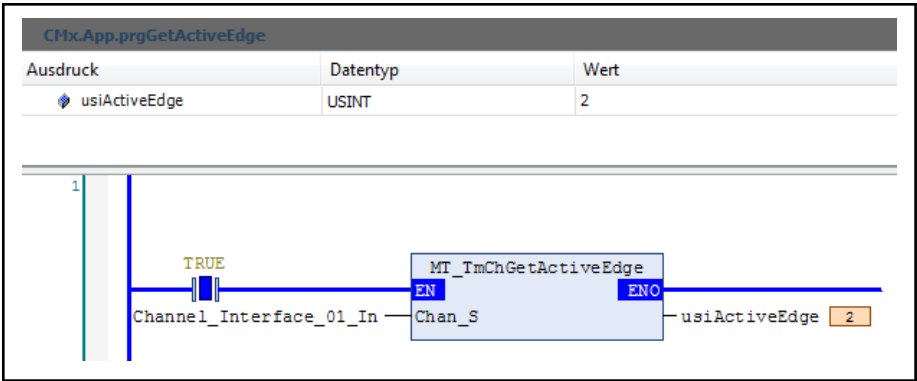


Abb. 14-135: Implementation MT_TmChGetActiveEdge

14.7 Sonstige

14.7.1 MT_DistanceAccumulator, Berechnet die Wegstrecke aus den angegebenen Positionsdaten

Kurzbeschreibung Funktionsbaustein zum Ermitteln der zurückgelegten Strecke anhand von Positionsdaten.

Zuordnung Zielsystem/Bibliothek	Zielsystem	Bibliothek
	IndraMotion_MTX	MT_BaseAdditional

Tab. 14-118: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

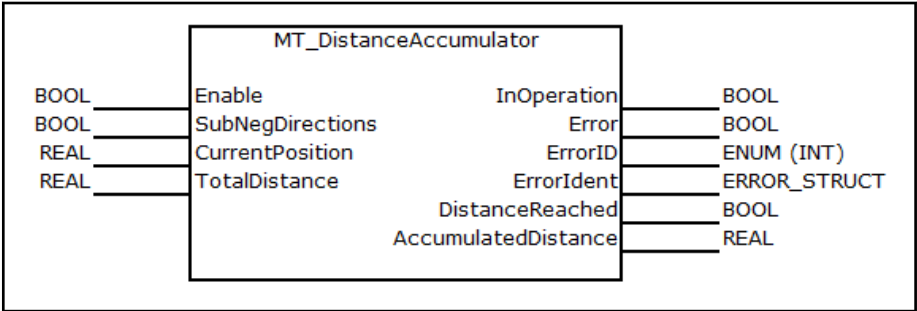


Abb. 14-136: Funktionsbaustein MT_DistanceAccumulator

	Name	Typ	Beschreibung
VAR_INPUT	Enable	BOOL	Baustein aktivieren
	SubNegDirections	BOOL	Bewegungen in negative Richtung "FALSE=addieren", "TRUE=subtrahieren"
	CurrentPosition	REAL	Aktuelle Positionswerte
	TotalDistance	REAL	Wegstrecke, welche erreicht werden darf
VAR_OUTPUT	InOperation	BOOL	Bearbeitung läuft
	Error	BOOL	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	Detaillierte Fehlerinformation (nur gültig, wenn "bError"=TRUE)

	Name	Typ	Beschreibung
	DistanceReached	BOOL	Die unter "TotalDistance" angegebene Wegstrecke wurde erreicht
	AccumulatedDistance	REAL	Aktuell gefahrene Wegstrecke

Tab. 14-119: Ein-/Ausgänge Funktionsbaustein MT_DistanceAccumulator

Signal-Zeit-Diagramm

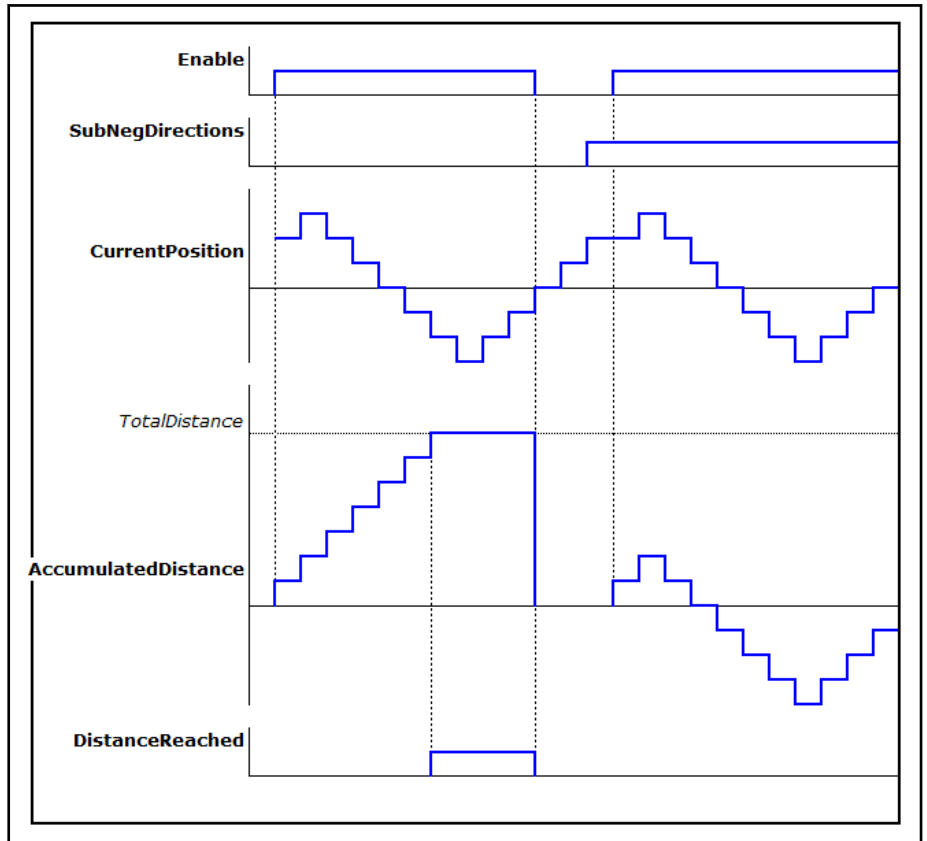


Abb. 14-137: Signal-Zeit-Diagramm MT_DistanceAccumulator

Funktionsbeschreibung

Dieser Funktionsbaustein berechnet aus den Differenzen der am Eingang "CurrentPosition" angegebenen Positionswerte den zurückgelegten Weg und gibt diesen Weg am Ausgang "AccumulatedDistance" aus. Dabei kann über den Eingang "SubNegDirections" festgelegt werden, ob Bewegungen in negative Richtung aufaddiert oder abgezogen werden sollen. Über den Eingang "TotalDistance" wird definiert wie groß die zurückgelegte Strecke ("AccumulatedDistance") werden darf, bevor der Ausgang "DistanceReached" gesetzt wird.

Dabei gilt, ist die vorgegebene Gesamtstrecke ("TotalDistance") größer oder gleich Null, dann wird "DistanceReached" erst TRUE, sobald die ermittelte Wegstrecke ("AccumulatedDistance") größer oder gleich der vorgegebenen Gesamtstrecke ("TotalDistance") ist.

Ist dagegen die vorgegebene Gesamtstrecke ("TotalDistance") kleiner Null, dann wird "DistanceReached" erst TRUE, sobald die ermittelte Wegstrecke ("AccumulatedDistance") kleiner oder gleich der vorgegebenen Gesamtstrecke ("TotalDistance") ist.

Sobald "TotalDistance" TRUE wird, ist die Messung beendet und "AccumulatedDistance" behält den zuletzt ermittelten Wert. Um eine neue Messung zu starten, muss der Eingang "Enable" neu gesetzt werden (0-->1).

Fehlerbehandlung

Zur Zeit gibt es für diesen Baustein keine Fehlercodes.

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_DistanceAccumulator"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar

Tab. 14-120: Fehler von "MT_DistanceAccumulator", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel

Im nachfolgenden Beispiel wird zuerst die Position des Antriebs 1 von der MTX-Steuerung gelesen. Zur Ermittlung der verfahrenen Strecke wird anschließend dieser Positionswert an den Eingang "CurrentPosition" des FB "MT_DistanceAccumulator" angelegt.

```

1 PROGRAM prgTestDistanceAccumulator
2 VAR
3     //Get Drive Position
4     fbGetPosData          : MT_ScsPos;          //FB to get the drive position
5     //Input
6     bGetPosDataStart      : BOOL :=FALSE;      //start
7     iGetPosDataChan       : INT :=1;          //channel
8     //Output
9     stPositionData       : ARRAY [1..4] OF DINT; //position data
10    bGetPosDataActive      : BOOL :=FALSE;      //read position active
11    bGetPosDataError       : BOOL :=FALSE;      //error read position
12    iGetPosDataErrorNo     : INT :=0;           //error detail
13    iGetPosDataNoElem      : INT :=0;           //number received elements
14
15    //calculate how far the drive has moved
16    fbDistanceAccumulator  : MT_DistanceAccumulator; //FB to calculate how far the drive has moved
17    //Input
18    bEnable                : BOOL :=FALSE;      //enable the distance accumulator
19    rTotalDistance         : REAL := 0;         //total distance to travel
20    rCurrentPosition       : REAL := 0;         //current drive position
21    bSubNegDirections      : BOOL := FALSE;     //TRUE=subtract negative difference from accumulated distance,
22                                           //FALSE=add negative difference from accumulated distance
23    //Output
24    bInOperation           : BOOL :=FALSE;      //FB active
25    bError                 : BOOL :=FALSE;      //FB error
26    enErrorID              : ERROR_CODE;        //FB error information
27    stErrorIdent           : ERROR_STRUCT;       //FB detailed error information
28    bDistanceReached       : BOOL := FALSE;     //Calculations are completed
29    rAccumulatedDistance   : REAL := 0.0;       //Distance traveled
30 END_VAR

```

Abb. 14-138: Deklaration MT_DistanceAccumulator

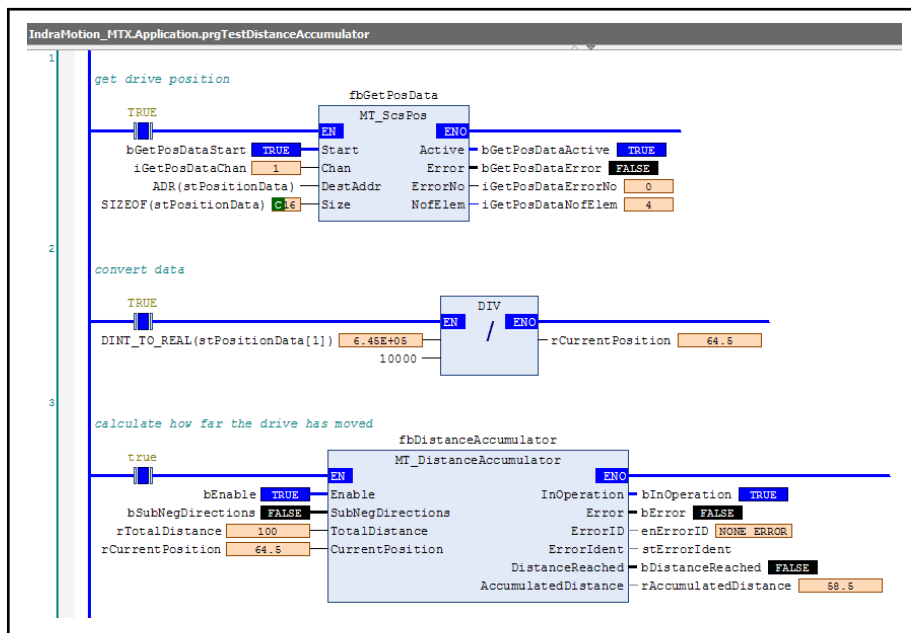


Abb. 14-139: Implementation MT_DistanceAccumulator

14.7.2 MT_TimeAccumulator, Berechnet die verstrichene Zeit

Kurzbeschreibung Funktionsbaustein ermittelt die Zeit, in welcher ein definierter Eingang im Zustand TRUE ist.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-121: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

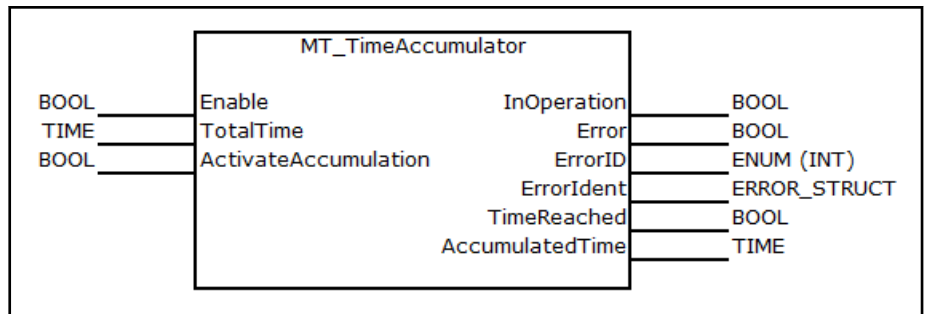


Abb. 14-140: Funktionsbaustein MT_TimeAccumulator

	Name	Typ	Init	Beschreibung
VAR_INPUT	Enable	BOOL	FALSE	Baustein aktivieren
	TotalTime	TIME	T#49d17h2m47s295ms	Zeit, die erreicht werden darf Initialwert entspricht Maximalwert für Datentyp TIME
	ActivateAccumulation	BOOL	FALSE	Zeit wird gezählt
VAR_OUTPUT	InOperation	BOOL	FALSE	Bearbeitung läuft
	Error	BOOL	FALSE	Bearbeitung fehlerhaft
	ErrorID	ENUM (INT)	NO-NE_ERROR	Grobe Fehlerinformation
	ErrorIdent	ERROR_STRUCT	--	Detaillierte Fehlerinformation (nur gültig wenn "bError"=TRUE)
	TimeReached	BOOL	FALSE	die unter "TotalTime" angegebene Zeit wurde erreicht
	AccumulatedTime	TIME	T#0MS	Aktuell verstrichene Zeit

Tab. 14-122: Ein-/Ausgänge Funktionsbaustein MT_TimeAccumulator

Signal-Zeit-Diagramm

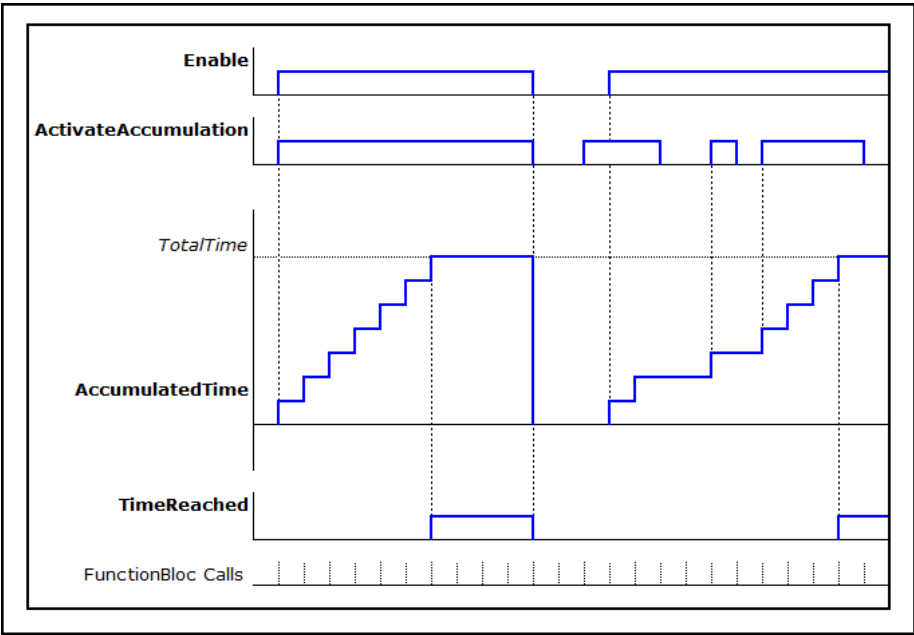


Abb. 14-141: Signal-Zeit-Diagramm MT_TimeAccumulator

Funktionsbeschreibung Dieser Baustein ermittelt die Zeit, in welcher der Eingang "ActivateAccumulation" TRUE ist. Solange der Eingang "ActivateAccumulation" TRUE ist, wird mit jedem Aufruf des FB die Systemzeit gelesen. Die Zeit, welche zwischen zwei Aufrufen vergangen ist, wird zu "AccumulatedTime" addiert. Über den Eingang "TotalTime" wird definiert, wie groß die verstrichene Zeit ("AccumulatedTime") werden darf, bevor der Ausgang "TimeReached" gesetzt wird.

Sobald "TimeReached" TRUE wird, ist die Messung beendet und "AccumulatedTime" behält den zuletzt ermittelten Wert. Um eine neue Messung zu starten, muss der Eingang "Enable" neu gesetzt werden (0-->1).

Fehlerbehandlung Zur Zeit gibt es für diesen Baustein keine Fehlercodes.

Diagnosen Referenzen auf die Diagnosetabelle

- MTX_TABLE

Fehler von "MT_DistanceAccumulator"

Fehlercode (Hex)	Bedeutung
0x00000000	Kein Fehlercode verfügbar

Tab. 14-123: Fehler von "MT_TimeAccumulator", aus "ErrorIdent.Additional1", gültig wenn "Error"==TRUE

Implementationsbeispiel Im nachfolgenden Beispiel wird die Bewegungszeit einer Achse überwacht. Nach 10 s wird die Schmierung der Achse ausgeführt und die Überwachung anschließend neu gestartet.

```
1 PROGRAM prgTestTimeAccumulator
2 VAR
3   //calculate
4   fbTimeAccumulator      : MT_TimeAccumulator ; //FB to calculate
5   //Input
6   bEnable                : BOOL := FALSE;      //enable the distance accumulator
7   tTotalTime              : TIME := T#10S;      //total time
8   bActivateAccumulation   : BOOL := FALSE;      //accumulation runs,
9   //Output
10  bInOperation            : BOOL := FALSE;      //FB active
11  bError                  : BOOL := FALSE;      //FB error
12  enErrorID               : ERROR_CODE;         //FB error information
13  stErrorIdent            : ERROR_STRUCT;        //FB detailed error information
14  bTimeReached            : BOOL := FALSE;      //Calculations are completed
15  tAccumulatedTime        : TIME := T#0H0M0S;   //travel time
16
17  //other
18  bLubricateAxisStart     : BOOL := FALSE;      //start lubrication
19  bLubricateAxisActive    : BOOL := FALSE;      //lubrication runs
20 END_VAR
```

Abb. 14-142: Deklaration MT_TimeAccumulator

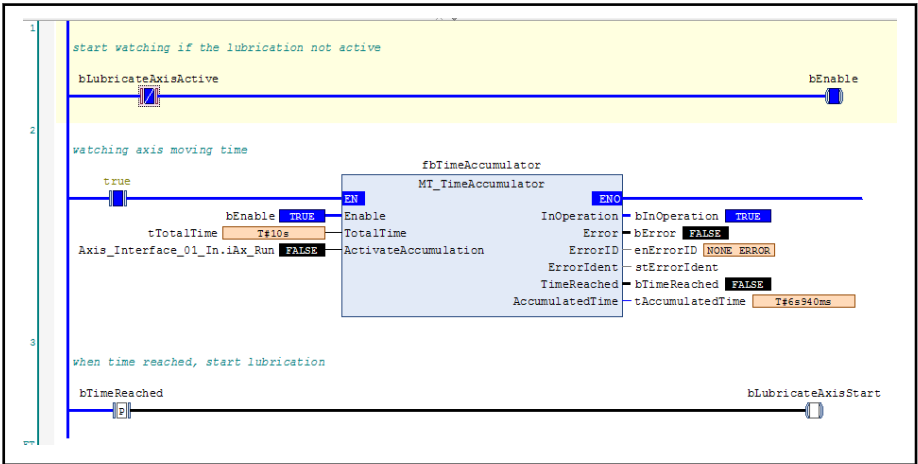


Abb. 14-143: Implementation MT_TimeAccumulator

14.7.3 MT_GetRandomNumber, Generiert eine ganzzahlige Zufallszahl

Kurzbeschreibung

Die Funktion berechnet eine ganzzahlige Zufallszahl innerhalb der vorgegebenen Minimal- und Maximalwerte.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-124: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Schnittstellenbeschreibung

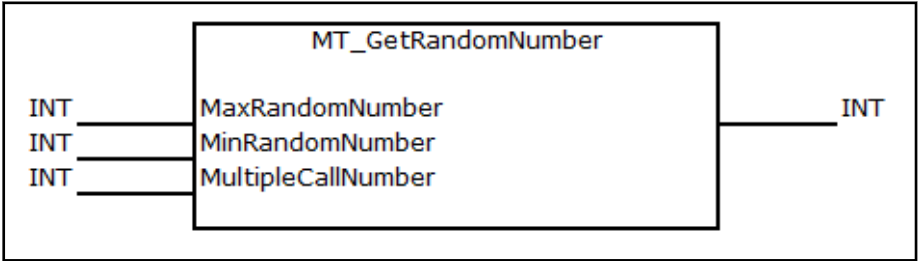


Abb. 14-144: Funktionsbaustein MT_GetRandomNumber

	Name	Typ	Init	Beschreibung
VAR_INPUT	MaxRandomNumber	INT	32767	Höchste erlaubte Zufallszahl
	MinRandomNumber	INT	-32768	Niedrigste erlaubte Zufallszahl
	MultipleCallNumber	INT	0	Verschiedene Werte bei Mehrfachaufruf in einem SPS-Zyklus
VAR_OUTPUT	MT_GetRandomNumber	INT	0	Zufallszahl

Tab. 14-125: Ein-/Ausgänge Funktionsbaustein MT_GetRandomNumber

Signal-Zeit-Diagramm Kein spezielles Verhalten, welches ein Signal-Zeit-Diagramm notwendig macht.

Funktionsbeschreibung Diese Funktion generiert bei jedem Aufruf eine Zufallszahl. Über die Eingänge "MaxRandomNumber" und "MinRandomNumber" wird festgelegt, welchen maximalen und minimalen Wert die Zufallszahl annehmen darf. Für den Fall, dass die Funktion innerhalb eines SPS-Zyklus mehrfach aufgerufen wird, muss bei jedem Aufruf der Eingang "MultipleCallNumber" einen anderen Wert haben.

Implementationsbeispiel Im nachfolgenden Beispiel wird die Funktion mehrfach in einem Zyklus aufgerufen. Beim ersten Aufruf wird eine Zufallszahl zwischen -1000 und +1000 erzeugt. Der zweite Aufruf generiert nur Zufallszahlen zwischen +20 und +90.

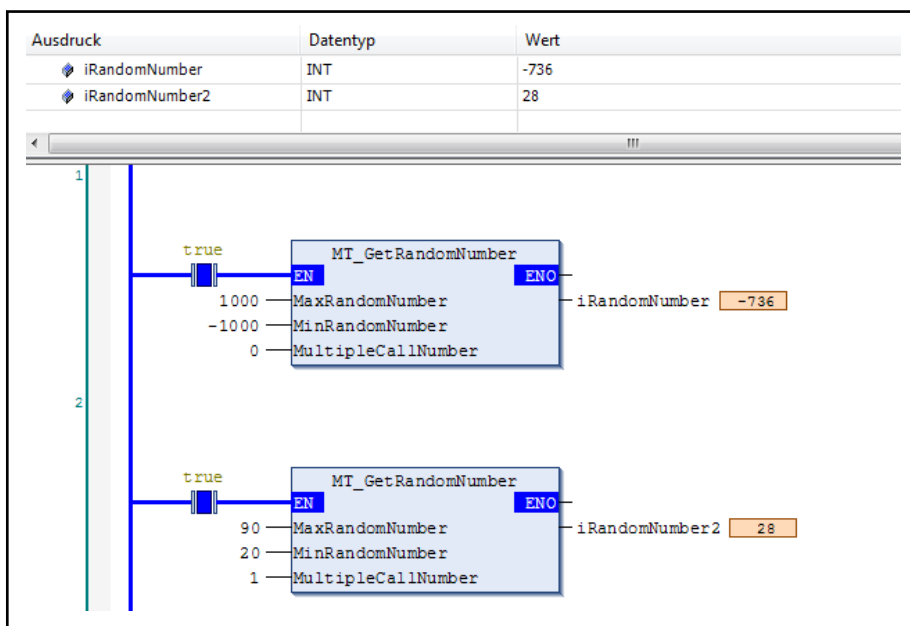


Abb. 14-145: Implementation MT_GetRandomNumber

14.8 Data Types - Datentypen

14.8.1 MT_CHANNEL_OPERATION_MODE, Struktur mit allen Kanalbetriebsarten

Kurzbeschreibung Die Struktur enthält alle verfügbaren Betriebsarten für einen NC-Kanal.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional

Tab. 14-126: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Typdefinition

```

TYPE MT_CHANNEL_OPERATION_MODE :
STRUCT
(*main operation mode*)
  Automatic      :BOOL:=FALSE;  (*automatic operation active*)
  Manual         :BOOL:=FALSE;  (*manual operation active*)
  Mdi            :BOOL:=FALSE;  (*MDI operation active (manual input nc block)*)

(*sub operation mode*)
  NoOPMode       :BOOL:=FALSE; (*no operation mode active*)

  (*'manual operation': sub operation modes*)
  ManSetup       :BOOL:=FALSE; (*manual setup, jogging operation active*)
  ManRefPoint    :BOOL:=FALSE; (*manual basingpoint*)
  ManWcsSetup    :BOOL:=FALSE; (*manual setup, jogging operation with tool coordinate system active*)

  (*'automatic operation': sub operation modes*)
  AutoContBlock  :BOOL:=FALSE;  (*automatic, continuous block active*)
  AutoProgBlock  :BOOL:=FALSE;  (*automatic, program block active*)
  AutoSingleStep :BOOL:=FALSE;  (*automatic, single step active*)
  AutoSingleBlock:BOOL:=FALSE;  (*automatic, single block active*)
  AutoRetContour :BOOL:=FALSE;  (*automatic, return to contour active*)
  AutoBlockForerun:BOOL:=FALSE; (*automatic, block forerun*)

  AutoCplProgBlock:BOOL:=FALSE; (*automatic, CPL debugger program block active*)
  AutoCplContBlock:BOOL:=FALSE; (*automatic, CPL debugger continuous block active*)
END_STRUCT
END_TYPE

```

14.8.2 MT_CHANNEL_STATE, Struktur enthält alle Kanalzustände

Kurzbeschreibung Die Struktur enthält alle verfügbaren Zustände für einen NC-Kanal.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional.lib

Tab. 14-127: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Typdefinition

```

TYPE MT_CHANNEL_STATE :
STRUCT
  Invalid :BOOL;  (*no valid channelstatus identified*)
  Inaktiv :BOOL;  (*no process active, any process can be preselected*)
  Ready   :BOOL;  (*a process is selected an ready to start*)
  Reset   :BOOL;  (*process reset is active, nothing orders accepted as far as the process is 'inactive'*)
  Error   :BOOL;  (*process error occurred which can only repaired with channel reset or program deselect*)

  CancelDistancePrepare:BOOL; (*process canceling distance prepare active*)
  CancelDistanceRunning:BOOL; (*process is active and processed available buffer new*)

  ProgSelActive :BOOL; (*a program is selected and is in preparation (e.g. linking)*)
  ProgNextBlkReady:BOOL; (*programm ready for next nc block, on buffered nc block selecting there are
                           all blocks processed, waiting for the next presetting*)
  ProgStartReady :BOOL; (*a process is selected, the nc program is at the beginning and ready for start*)
  ProgRunning    :BOOL; (*a process is selected, a nc program is running and processed a nc block*)
END_STRUCT
END_TYPE

```

14.8.3 MT_SD_SYS_TOOL, Struktur von Systemdatum SD.SysTool

Kurzbeschreibung Das im Funktionsumfang enthaltene NC-Werkzeugwechselprogramm (TCH) arbeitet mit dem Systemdatum SD.SysTool. Diese Struktur ist das passende Abbild dieses Systemdatums in der SPS.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional\Data Types\Ch

*Tab. 14-128: Bibliotheksnamen in Abhängigkeit von den Zielsystemen**Typdefinition*

```
{attribute 'pack_mode' := '1'}
TYPE MT_SD_SYS_TOOL :
STRUCT
  ActToolValid :BOOL:=FALSE;           //active tool is valid
  ActEd        :DINT:=0;                //active edge
  ActTool      :MT_RECORD_HEADER_DATA; //active tool header data
  PreTool      :MT_RECORD_HEADER_DATA; //
END_STRUCT
END_TYPE
```

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_Base\Data Types\ToolManagement Default Types\IndraWorks 09Vxx

*Tab. 14-129: Bibliotheksnamen in Abhängigkeit von den Zielsystemen**Typdefinition*

```
TYPE MT_RECORD_HEADER_DATA :
STRUCT
  K1 :INT;           //Sector
  K2 :INT;           //Place
  SKQ :STRING(31);  //Tool-ID
  IKQ1 :DINT;        //Duplo Number
  IKQ2 :DINT;        //Tool Type
  IKQ3 :DINT;        //Tool Number
  IQ1 :DINT;         //reserve
  IQ2 :DINT;         //reserve
  IQ3 :DINT;         //reserve
  BQ1 :DINT;         //Place Status (Bit Array)
  BQ2 :DINT;         //Tool Status (Bit Array)
  BQ3 :DINT;         //Tooltype description (Bit Array)
END_STRUCT
END_TYPE
```

14.8.4 MT_AXIS_ASYNC_OPERATION_MODE, Struktur mit allen Achsbetriebsarten

Kurzbeschreibung Die Struktur enthält alle verfügbaren Betriebsarten einer asynchronen Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional\Data Types\Drive

*Tab. 14-130: Bibliotheksnamen in Abhängigkeit von den Zielsystemen**Typdefinition*

```
TYPE MT_AXIS_ASYNC_OPERATION_MODE :
STRUCT
  ProgMode :BOOL; (*axis can be programed by a channel*)
  ManSetup :BOOL; (*manual setup, jogging operation active*)
  ManRefPoint :BOOL; (*manual basingpoint*)
  InvalidState :BOOL; (*no valid status identified*)
END_STRUCT
END_TYPE
```


14.8.5 MT_AXIS_JOG_MODE, Struktur mit Tippdistanzen einer Achse

Kurzbeschreibung Die Struktur enthält alle verfügbaren Tippdistanzen einer Achse.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional\Data Types\Drive

Tab. 14-131: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Typdefinition

```

TYPE MT_AXIS_JOG_MODE :
STRUCT
  NoJogMode          :BOOL:=FALSE;      (*no JogMode aktiv*)
  (*incremental JogModi*)
  JogMode1           :BOOL:=FALSE;      (*JogMode 1 increment active*)
  JogMode10          :BOOL:=FALSE;      (*JogMode 10 increment active*)
  JogMode100         :BOOL:=FALSE;      (*JogMode 100 increment active*)
  JogMode1000        :BOOL:=FALSE;      (*JogMode 1000 increment active*)
  JogModeX           :BOOL:=FALSE;      (*Jogging respective the increment guidelines in machine parameter active*)
  (*continuous JogModi*)
  JogModeContRapid   :BOOL:=FALSE;      (*rapid continuous jogging active*)
  JogModeContFast    :BOOL:=FALSE;      (*fast continuous jogging active*)
  JogModeContMiddle  :BOOL:=FALSE;      (*middle continuous jogging active*)
  JogModeContSlow    :BOOL:=FALSE;      (*slow continuous jogging active*)
END_STRUCT
END_TYPE

```

14.8.6 MT_DRIVE_ERROR_CLASS_1, Struktur Fehlerdetails aus S-0-0011 einer Achse

Kurzbeschreibung Die Struktur enthält die in S-0-0011 bitcodierten Fehlerinformationen.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional\Data Types\Drive

Tab. 14-132: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Typdefinition

```

TYPE MT_DRIVE_ERROR_CLASS_1 :
STRUCT
  //S-0-0011
  OverloadShutdown           :BIT:=FALSE; //Bit0
  AmplifierOvertemperatureShutdown :BIT:=FALSE; //Bit1
  MotorShutdownTemperature   :BIT:=FALSE; //Bit2
  ReservedBit03              :BIT:=FALSE; //Bit3
  ControlVoltageError        :BIT:=FALSE; //Bit4
  EncoderError               :BIT:=FALSE; //Bit5
  ElectronicCommunicationError :BIT:=FALSE; //Bit6
  OverCurrentError           :BIT:=FALSE; //Bit7
  OverVoltageError           :BIT:=FALSE; //Bit8
  UnderVoltageError          :BIT:=FALSE; //Bit9
  PhaseErrorInPowerSupply    :BIT:=FALSE; //Bit10
  ExcessiveControllerDeviation :BIT:=FALSE; //Bit11
  CommunicationError         :BIT:=FALSE; //Bit12
  PositionLimitValueExceeded  :BIT:=FALSE; //Bit13
  ReservedBit14              :BIT:=FALSE; //Bit14
  ManufacturerSpecificError   :BIT:=FALSE; //Bit15
END_STRUCT
END_TYPE

```

14.8.7 MT_DRIVE_ERROR_CLASS_2, Struktur Warnungsdetails aus S-0-0012 einer Achse

Kurzbeschreibung Die Struktur enthält die in S-0-0012 bitcodierten Detailinformationen zu einer Warnung.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional\Data Types\Drive

Tab. 14-133: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Typdefinition

```

TYPE MT_DRIVE_ERROR_CLASS_2 :
STRUCT
//S-0-0012
OverloadWarning           :BIT:=FALSE; //Bit0
AmplifierOvertemperatureWarning :BIT:=FALSE; //Bit1
MotorOvertemperatureWarning :BIT:=FALSE; //Bit2
ReservedBit03             :BIT:=FALSE; //Bit3
ReservedBit04             :BIT:=FALSE; //Bit4
PositioningVelocityGreaterLimit :BIT:=FALSE; //Bit5
ReservedBit06             :BIT:=FALSE; //Bit6
ReservedBit07             :BIT:=FALSE; //Bit7
ReservedBit08             :BIT:=FALSE; //Bit8
UndervoltageInDCBus       :BIT:=FALSE; //Bit9
ReservedBit10             :BIT:=FALSE; //Bit10
ReservedBit11             :BIT:=FALSE; //Bit11
ReservedBit12             :BIT:=FALSE; //Bit12
TargetPositionOutsideOfPositionLimitValues :BIT:=FALSE; //Bit13
ReservedBit14             :BIT:=FALSE; //Bit14
ManufacturerSpecificWarning :BIT:=FALSE; //Bit15
END_STRUCT
END_TYPE

```

14.8.8 MT_DRIVE_POWER_STATE, Antriebsstatus

Kurzbeschreibung Die Struktur enthält die Antriebszustände, welche von FBs MT_AxPower, MT_SpPower bereitgestellt werden.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional\Data Types\Drive

Tab. 14-134: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Typdefinition

```

TYPE MT_DRIVE_POWER_STATE :
STRUCT
BB           :BOOL:=FALSE; //Status BB - drive ready
AB           :BOOL:=FALSE; //Status AB - drive enabled for power activation
AF           :BOOL:=FALSE; //Status AF - drive in operation
DryRun       :BOOL:=FALSE; //Status DryRun - drive in dryrun mode
Error        :BOOL:=FALSE; //Status Error (class1) active - for details read S-0-0011
ErrorDetails :MT_DRIVE_ERROR_CLASS_1;//Error detail information
Warning      :BOOL:=FALSE; //Status Warning (class2) active - for details read S-0-0012
WarningDetails :MT_DRIVE_ERROR_CLASS_2;//Warning detail information
Unknown      :BOOL:=FALSE; //Status unknown - drive unknown status
END_STRUCT
END_TYPE

```

14.8.9 MT_DRIVE_SAFE_OP_MODE, Struktur enthält Sicherheitsbetriebszustand eines Antriebs

Kurzbeschreibung Die Struktur bildet den sicherheitstechnischen Betriebszustand eines Antriebs ab, wie er vom FB "MT_DrvGetSafeState" ermittelt wird.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional\Data Types\Drive

Tab. 14-135: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Typdefinition

```

TYPE MT_DRIVE_SAFE_OP_MODE :
STRUCT
NO      : BOOL:=FALSE; //0: not active, 1: active / Normal operation
SMES    : BOOL:=FALSE; //0: not active, 1: active / EMERGENCY STOP
SMST    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe standstill

SMM1    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 1
SMM2    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 2
SMM3    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 3
SMM4    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 4
SMM5    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 5
SMM6    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 6
SMM7    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 7
SMM8    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 8
SMM9    : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 9
SMM10   : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 10
SMM11   : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 11
SMM12   : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 12
SMM13   : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 13
SMM14   : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 14
SMM15   : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 15
SMM16   : BOOL:=FALSE; //0: not active, 1: active / Special mode safe motion 16

ParkingAxis      : BOOL:=FALSE; //0: not active, 1: active / parking axis

SMO_ApplicationActiv : BOOL:=FALSE; //0: not active, 1: active / safe motion application
SMO_TransitionProcess : BOOL:=FALSE; //0: not active, 1: active / safe motion transition process
SMO_ParameterMode    : BOOL:=FALSE; //0: not active, 1: active / safe motion parameter mode
SMO_ConfigurationMode : BOOL:=FALSE; //0: not active, 1: active / safe motion configuration mode
SMO_BehaveIfNotActive : BOOL:=FALSE; //0: Condition as supplied,
//1: Functional commissioning; output stage and brake(s) are activated by the standard firmware
//Behavior with non-active "Safe Motion", only relevant if "SMO_ActivationStatus==FALSE"

SMO_ActivationStatus : BOOL:=FALSE; //1: not active, 0: active / safe motion functionality
SafetyTechnologyError : BOOL:=FALSE; //1: no error message, 0: Error message/Safety technology error

END_STRUCT
END_TYPE

```

14.8.10 MT_SPINDLE_JOG_SPEED, Struktur mit Tippgeschwindigkeiten einer Spindel

Kurzbeschreibung Die Struktur enthält alle verfügbaren Tippgeschwindigkeiten einer Spindel.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek
IndraMotion_MTX	MT_BaseAdditional\Data Types\Drive

Tab. 14-136: Bibliotheksnamen in Abhängigkeit von den Zielsystemen

Typdefinition

```

TYPE MT_SPINDLE_JOG_SPEED :
STRUCT
(*Variables for the current SpeedMode*)
Speed1 : BOOL; (*spindle jogging speed 1*)

```

MT_BaseAdditional.library

```
Speed2      :BOOL; (*spindle jogging speed 2*)
Speed3      :BOOL; (*spindle jogging speed 3*)
Speed4      :BOOL; (*spindle jogging speed 4*)
Speed5      :BOOL; (*spindle jogging speed 5*)
Speed6      :BOOL; (*spindle jogging speed 6*)
Speed7      :BOOL; (*spindle jogging speed 7*)
Speed8      :BOOL; (*spindle jogging speed 8*)
END_STRUCT
END_TYPE
```

15 MT_PLCopen.library

15.1 Einführung und Übersicht

Die Bibliothek MT_PLCopen enthält die für die Realisierung des CNC-Systems MTX wesentlichen Funktionsbausteine für Sercos-basierte Achsen.

Die von den Bausteinen an ihrer Schnittstelle (VAR_INPUT, VAR_OUTPUT, VAR_IN_OUT, CONSTANT, VAR_GLOBAL / EXTERNAL) benötigten Datentypen sind elementar in der IEC-Norm EN 61131-3 enthalten oder durch die Firma definiert in nachstehenden Bibliotheken:

- Siehe Kap. RIL_CommonTypes.library in der Dokumentation "IndraLogic XLC, IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490), Datentypen zur allgemeinen Nutzung in allen IndraWorks-Systemen.

Funktionsbausteine, Funktionen und Datentypen von MT_PLCopen

Anwendungsgebiet (Ordner in der Bibliothek)	
POUs	Beschreibung
POUs (Parameter)	
MB_Command, Seite 361	Erlaubt die Ausführung eines Kommandos
MB_ReadListParameter, Seite 366	Gibt den Wert eines anwenderspezifischen Parameters zurück. Dem Baustein wird ein Pointer übergeben, der einen in Anwenderverantwortung festgelegten Speicherraum von maximal 64 KB adressiert
MB_ReadMaxRealValue, Seite 368	Gibt den zulässigen Maximalwert eines durch den Anwender vorgegebenen Parameters zurück (REAL)
MB_ReadMaxValue, Seite 370	Gibt den zulässigen Maximalwert eines durch den Anwender vorgegebenen Parameters zurück (DINT)
MB_ReadMinRealValue, Seite 372	Gibt den zulässigen Minimalwert eines durch den Anwender vorgegebenen Parameters zurück (REAL)
MB_ReadMinValue, Seite 374	Gibt den zulässigen Minimalwert eines durch den Anwender vorgegebenen Parameters zurück (DINT)
MB_ReadName, Seite 376	Gibt den Namen eines durch den Anwender vorgegebenen Parameters zurück
MB_ReadParameter, Seite 377	Gibt den Wert eines vorgegebenen Parameters zurück (DINT)
MB_ReadRealParameter, Seite 379	Gibt den Wert eines vorgegebenen Parameters zurück (REAL)
MB_ReadSercosAttribute, Seite 381	Erlaubt das Lesen der Sercos-Attribute eines Parameters
MB_ReadSercosDataStatus, Seite 383	Gibt den Status eines durch den Anwender vorgegebenen Parameters zurück
MB_ReadStringParameter, Seite 385	Gibt den Wert eines anwenderspezifischen Parameters zurück (STRING, Zeichenfolge)
MB_ReadUnit, Seite 387	Gibt die Maßeinheit eines durch den Anwender vorgegebenen Parameters zurück

MB_WriteListParameter Siehe Kap. ML_PLCopen.library in der Dokumentation "IndraLogic XLC, IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)	Schreibt den Speicherinhalt in definierter Länge, ab Anfangsadresse, in einen vorgegebenen Parameter
MB_WriteParameter, Seite 388	Schreibt einen Wert in einen durch den Anwender vorgegebenen Parameter (DINT)
MB_WriteRealParameter, Seite 390	Schreibt einen Wert in einen durch den Anwender vorgegebenen Parameter (REAL)
MB_WriteStringParameter, Seite 392	Schreibt eine Zeichenfolge mit max. 255 Zeichen in einen vorgegebenen Parameter (STRING, Zeichenfolge)

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, MTX usw. vergleichbare Bausteine existieren.

Beginnt der jeweilige Bausteinname mit "MB_" bedeutet dies, dass für unterschiedliche Systeme wie XLC, MLC, MLD, MTX usw. vergleichbare Bausteine existieren.

Beginnt der jeweilige Bausteinname mit "MT_" bedeutet dies, dass er für das CNC-System MTX entwickelt wurde, um dieses bestmöglich zu unterstützen.

15.2 Globale_Variablen

15.2.1 Data Types - Datentypen

IL_SIII_ATTRIBUTE, Struktur enthält alle Sercos-Attribute

Kurzbeschreibung Die Struktur enthält alle verfügbaren Sercos-Attribute.

Typdefinition

```

TYPE IL_SIII_ATTRIBUTE :
STRUCT
  WrProtectP4 : BOOL;           //Operation date of the parameter in the operation mode
                                //(Sercos phase 4 - P4) write-protected
  WrProtectP3 : BOOL;           //Operating date of the parameter in Sercos phase 3
                                //(P3) write-protected
  WrProtectP2 : BOOL;           //Operation date of the parameter in the parameterization mode
                                //(Sercos phase 2 - P2) write-protected
  DecPoints   : USINT;           //Number of decimal places of the operating date of the parameter
  DataType    : IL_SIII_DATA_TYPE; //Data type and display format of the operating date of the parameter
  Cmd         : BOOL;           //Operating date calls for command function
  DataLength  : IL_SIII_DATA_LENGTH; //Data length of the operating date of the parameter
  Conversion  : UINT;           //Evaluation factor of the operating date of the parameter
END_STRUCT
END_TYPE

```

MB_SERCOS_DATA_STATUS, Struktur enthält alle Sercos-Kommandozustände

Kurzbeschreibung Die Struktur enthält alle verfügbaren Sercos-Kommandozustände.

Typdefinition

```

TYPE MB_SERCOS_DATA_STATUS :
(
  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
```

15.3 POU

15.4 MB_Command

Kurzbeschreibung Der Funktionsbaustein "MB_Command" dient zur Ausführung, zur Überwachung und zum Abbruch von Kommandos.

Schnittstellenbeschreibung

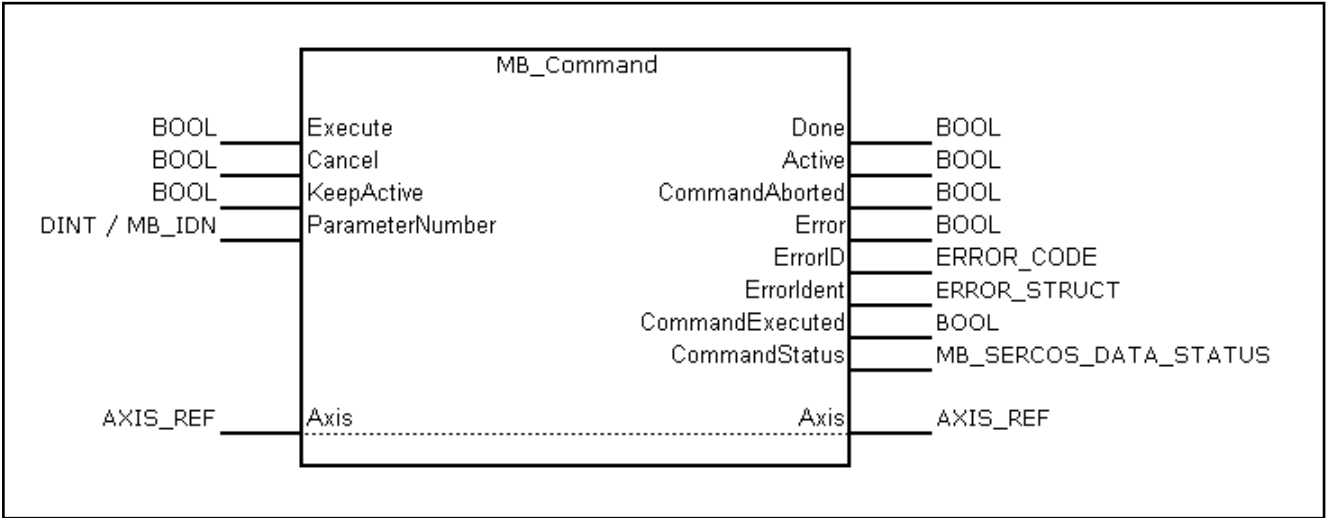


Abb. 15-1: Funktionsbaustein "MB_Command"

Schnittstellenvariablen

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF	Referenz zur Achse
VAR_INPUT	Execute	BOOL	Freigabe des Funktionsbausteins (einmalig, flankengesteuert)
	Cancel	BOOL	Abbruch der Kommandoausführung (einmalig, flankengesteuert)
	KeepActive	BOOL	Aktivierung des Kommandos nach Ausführung aktiv halten (permanent, zustandsgesteuert)
	ParameterNumber	DINT bzw. MB_IDN (ab MPx06)	Kommando

	Name	Typ	Kommentar
VAR_OUTPUT	Done	BOOL	Kommando erfolgreich abgeschlossen und zurückgesetzt
	Active	BOOL	Kommandoausführung aktiv
	Command-Aborted	BOOL	Kommando erfolgreich abgebrochen und zurückgesetzt
	Error	BOOL	Fehler bei der Kommandoausführung
	ErrorID	ERROR_CODE	bei gesetztem "Error"-Ausgang enthält dieser Ausgang eine grobe Klassifizierung des Fehlers
	ErrorIdent	ERROR_STRUCT	bei gesetztem "Error"-Ausgang enthält dieser Ausgang detaillierte Informationen über den Fehler siehe "Fehlerbehandlung" auf Seite 366
	Command-Executed	BOOL	Kommando erfolgreich abgeschlossen
	CommandStatus	MB_SERCOS_DATA_STATUS	Status der Kommandoausführung

Tab. 15-1: Schnittstellenvariablen Funktionsbaustein "MB_Command"

Signal-Zeit-Diagramm

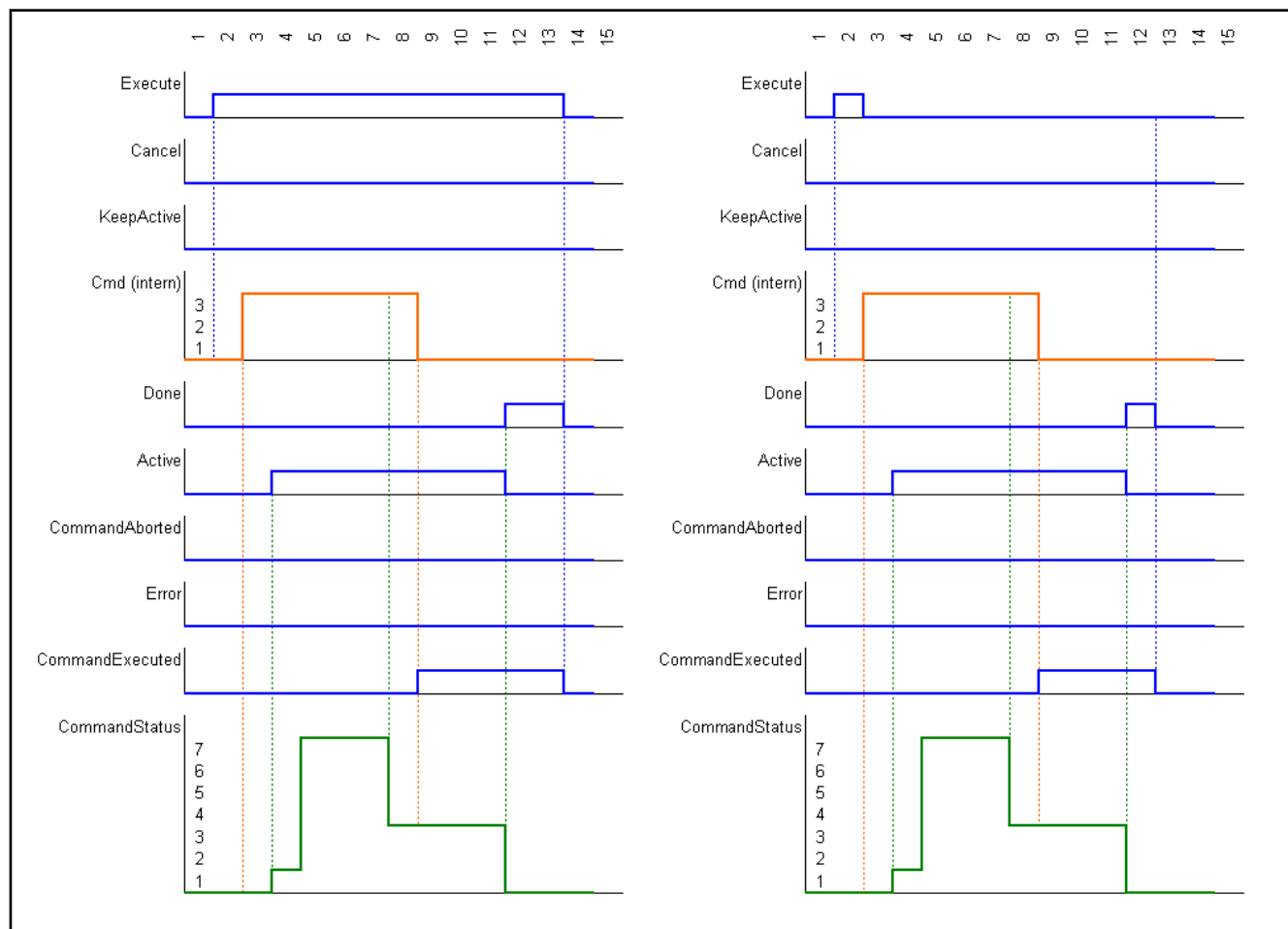


Abb. 15-2: Signal-Zeit-Diagramm für Signalverlauf "Execute-Done"

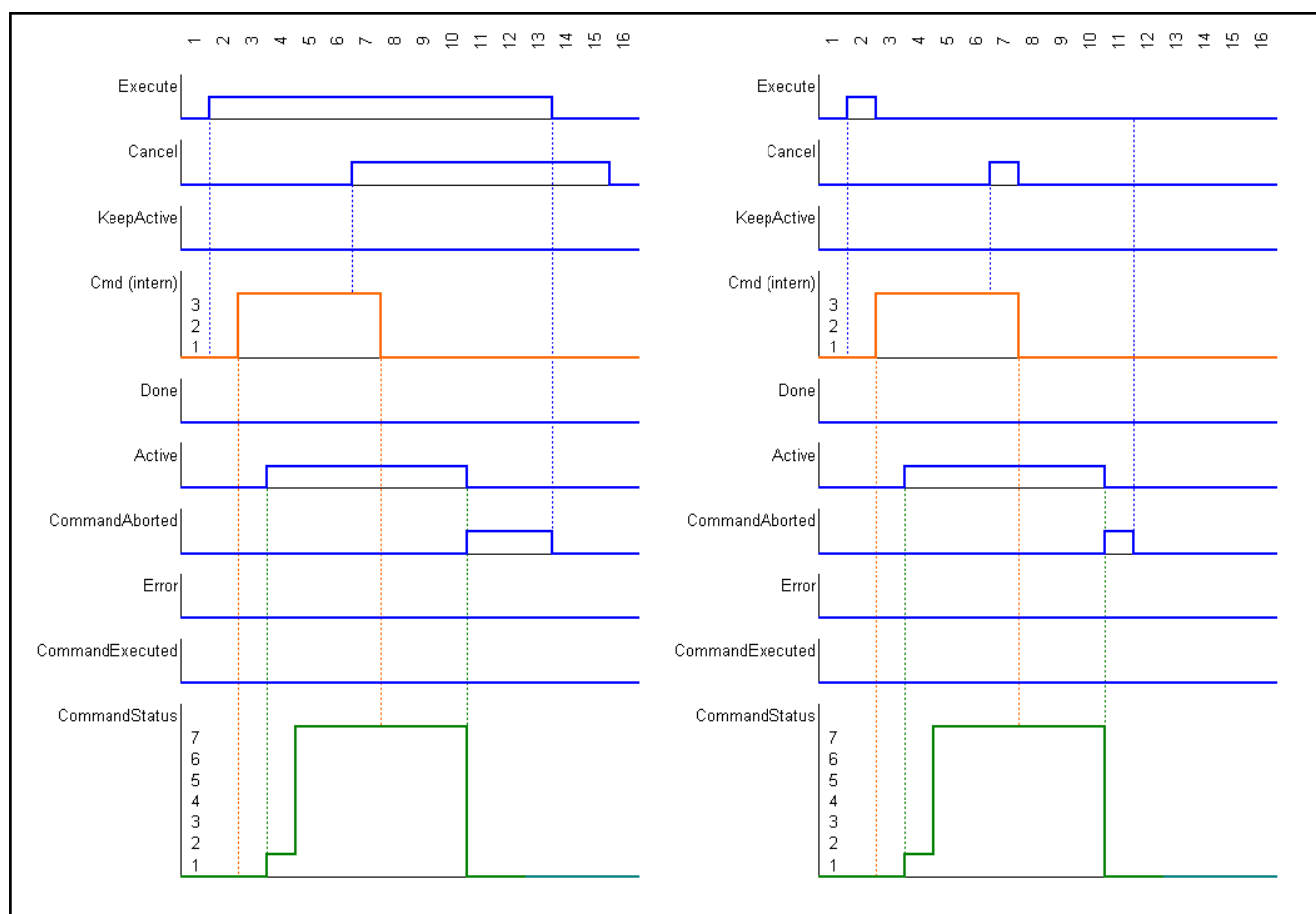


Abb. 15-3: Signal-Zeit-Diagramm für Signalverlauf "Execute-Cancel-CommandAborted"

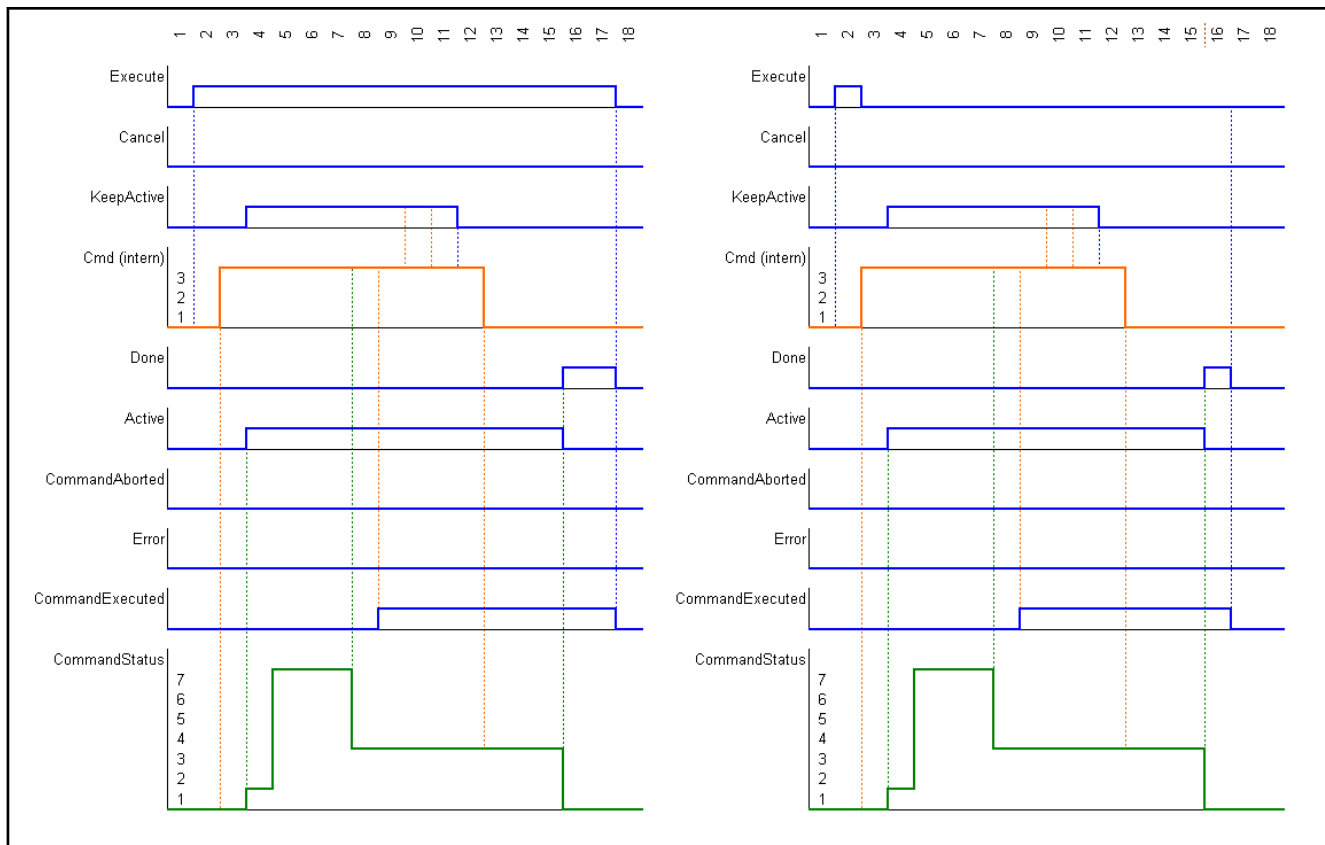


Abb. 15-4: Signal-Zeit-Diagramm für Signalverlauf "Execute-KeepActive-Done "

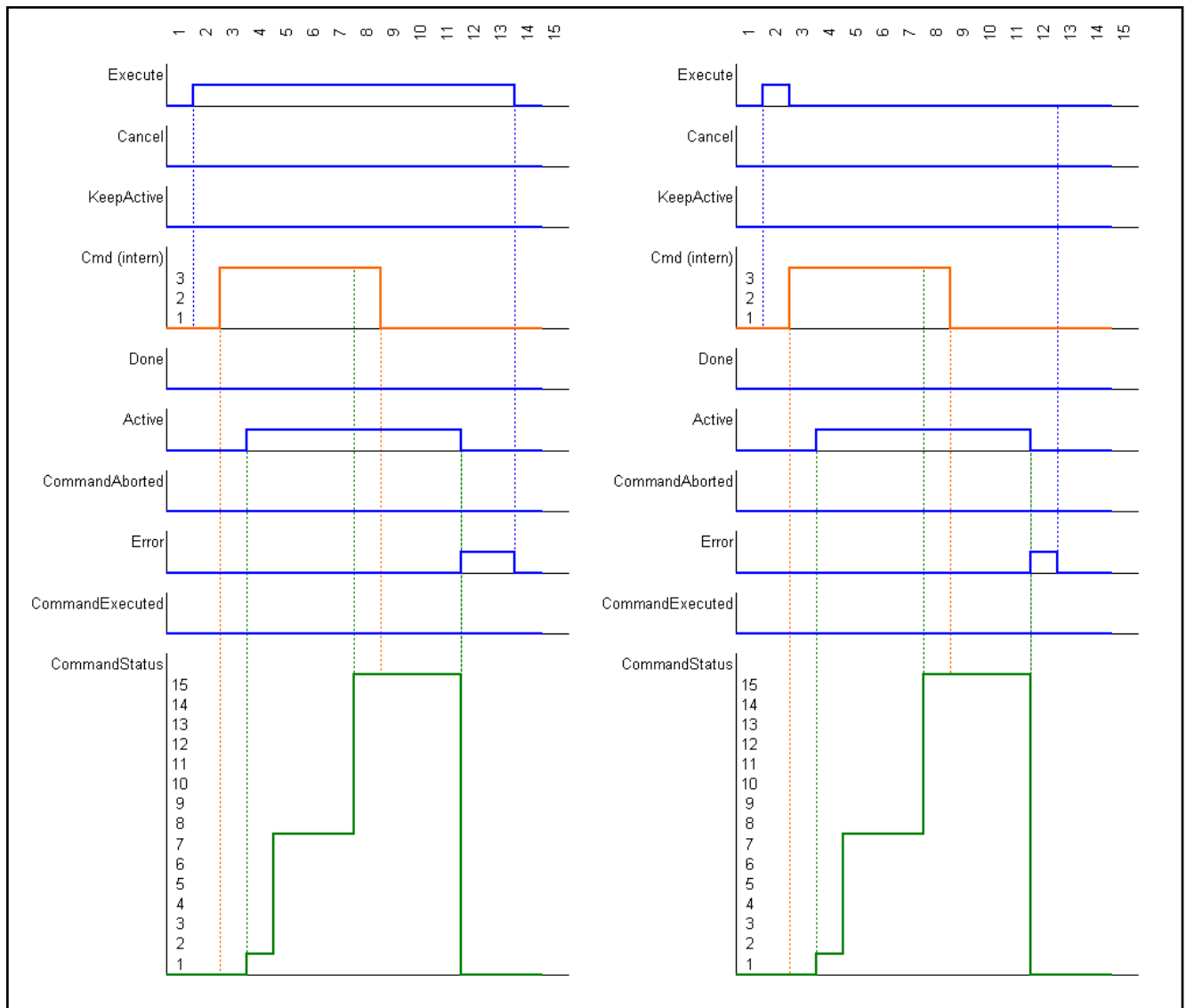


Abb. 15-5: Signal-Zeit-Diagramm für Signalverlauf "Execute-Error "

Funktionsbeschreibung

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. Das Kommando muss im Parameter-Identnummerformat nach SERCOS Norm angegeben werden. Neben dieser direkten Angabe ist es weiterhin möglich, eine entsprechend vordefinierte Konstante aus dem Bereich der globalen Variablen der Basisbibliothek (z. B. "ML_Base.lib") bzw. eine Umwandlungsfunktion aus der Bibliothek "RIL_Uutilities.lib" zu verwenden. 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" sowie mittels des entsprechenden Datums am Ausgang "CommandStatus" dargestellt. Der aktuel-

le 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 die Elemente der Fehlerklassifizierung (ErrorID, ErrorIdent) für die Zeitdauer von mindestens einem Aufruf signalisiert.

Fehlerbehandlung

Der Funktionsbaustein "MB_Command" nutzt die Fehlertabellen **F_RELATED_TABLE**, 16#0170, sowie **INDRV_TABLE**, 16#0070. Er kann in Additional1 und Additional2 die, in der nachfolgenden Tabelle, aufgeführten Fehler erzeugen. Zusätzlich zu diesem Fehler können die eingebundene Funktionsbausteine "MB_WriteParameter", "MB_ReadSercosDataStatus", "MB_ReadParameter" sowie "MB_ReadSercosAttribute" weitere Fehler melden.

ErrorID	Additional1	Additional2	Beschreibung
INPUT_RANGE_ERROR, 16#0006	16#0002	16#0001	ungültiger Eingang - Die Retriggierung der Kommandoausführung wird durch den Baustein MB_Command nicht unterstützt.
INPUT_RANGE_ERROR, 16#0006	16#0002	16#0002	ungültiger Eingang - Die Eingänge Cancel bzw. KeepActive werden durch Steuerungskommandos nicht unterstützt.
INPUT_RANGE_ERROR, 16#0006	16#0002	16#0003	ungültiger Eingang - Der Eingang ParameterNumber enthält kein gültiges Kommando.
DEVICE_ERROR, 16#0008	16#1701	16#0000	Fehler bei der Kommandoausführung (Steuerung)
STATE_MACHINE_ERROR, 16#0005	16#0006	16#????	ungültiger Zustand der Zustandsmaschine 16#???? gibt den ungültigen Zustand an

Tab. 15-2: Fehlernummern der Fehlertabelle "F_RELATED_TABLE", verursacht durch "MB_Command"

ErrorID	Additional1	Additional2	Beschreibung
DEVICE_ERROR, 16#0008	16#?????	16#0000	Fehler bei der Kommandoausführung (Drive) - 16#???? enthält den aktuellen Wert aus S-0-0390.

Tab. 15-3: Fehlernummern der Fehlertabelle "INDRV_TABLE", verursacht durch "MB_Command"

15.5 MB_ReadListParameter

Kurzbeschreibung

Der Funktionsbaustein kehrt mit dem 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.lib	Parameter
15VRS	

Tab. 15-4: Bibliothekszuordnung

Schnittstellenbeschreibung

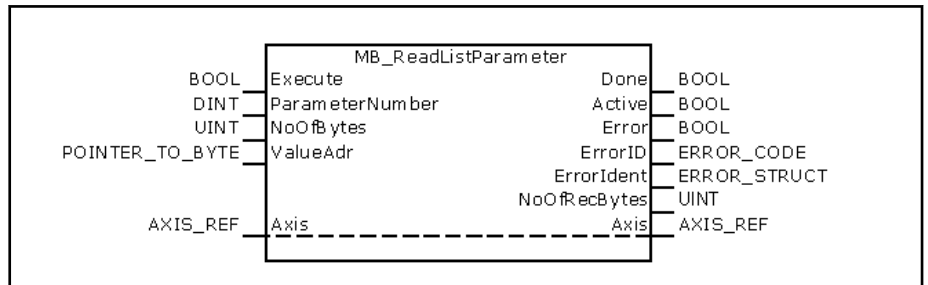


Abb. 15-6: FB MB_ReadListParameter

	Name	Typ	Kommentar
VAR_IN_OUT	Axis	AXIS_REF Siehe Kap. RIL_CommonTypes.library in der Dokumentation "IndraLogic XLC, IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)	Enthält Informationen zur konkreten Achse
VAR_INPUT	Execute	BOOL	Lesen des Parameters, getriggert durch die positive Flanke von "Execute"
	Parameter-Number	DINT	Nummer des gewünschten Parameters, Siehe Kap. PLCopen-Funktionsbausteine für die IndraMotion MLC 02VRS in der Dokumentation "Rexroth IndraMotion MLC02VRS PLCopen-Funktionsbausteine und -Datentypen" Bibliotheksbeschreibung, R911316073
	NoOfBytes	UINT	Maximal zu lesende Datenmenge in Byte
	ValueAdr	POINTER_TO_BYTE	Zeigt auf die 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 Siehe Kap. RIL_CommonTypes.library in der Dokumentation "IndraLogic XLC, IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)	Kurzhinweis auf die Fehlerursache

	Name	Typ	Kommentar
	ErrorIdent	ERROR_STRUCT Siehe Kap. RIL_CommonTypes.library in der Dokumentation "IndraLogic XLC, IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)	Detailinformation über den Fehler
	NoOf Rec-Bytes	UINT	Anzahl der tatsächlich gelesenen Bytes

Tab. 15-5: Schnittstelle von FB MB_ReadListParameter

Signal-Zeit-Diagramm

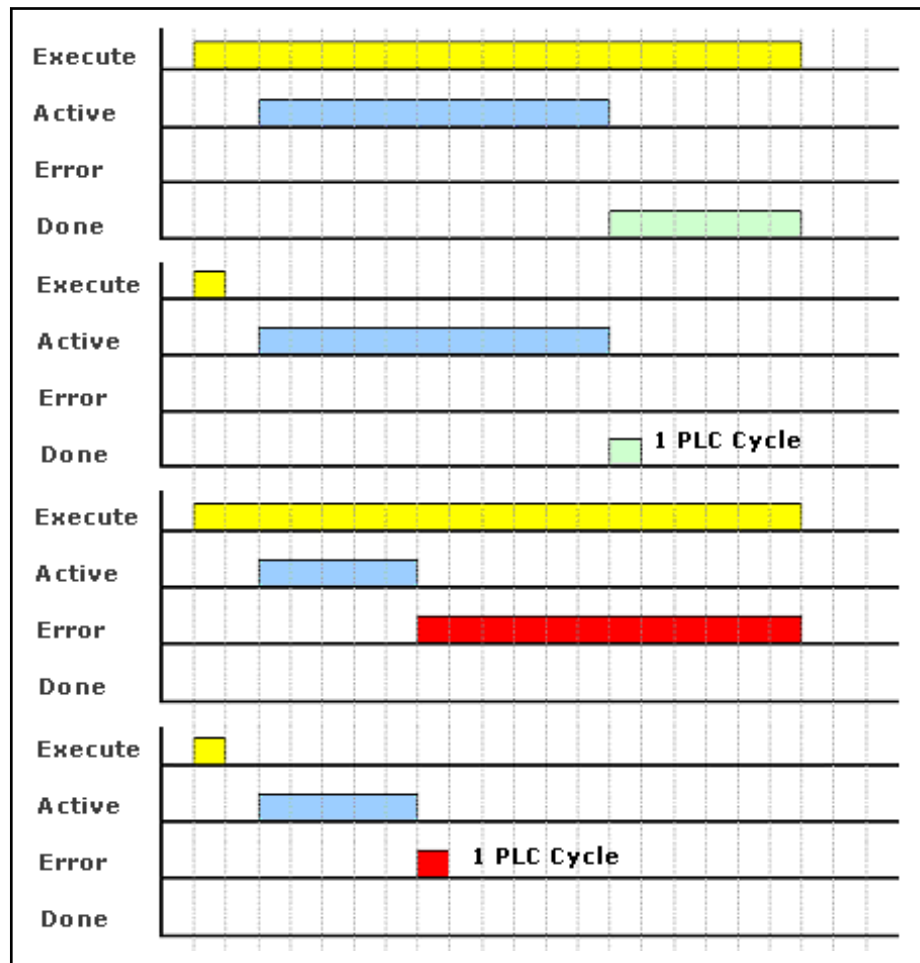


Abb. 15-7: Standard-Signal-Zeit-Diagramm

15.6 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCOpen / Parameter

Tab. 15-6: Bibliothekszuordnung

Schnittstellenbeschreibung

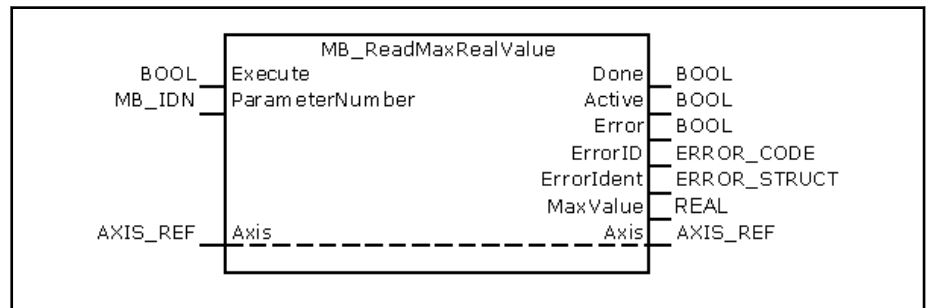


Abb. 15-8: FB MB_ReadMaxRealValue

	Name	Typ	Kommentar
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 Kap. ML_Base.library in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCOpen-Bibliotheken", Bibliothek (R911341490))
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 (" Fehlerbehandlung " auf Seite 370)
	MaxValue	Real	Ist "Done" = TRUE, zeigt "MaxValue" den zulässigen maximalen Eingabewert des ausgewählten Parameters an.
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse

Tab. 15-7: Schnittstelle von FB MB_ReadMaxRealValue

Kurzbeschreibung
Signal-Zeit-Diagramm

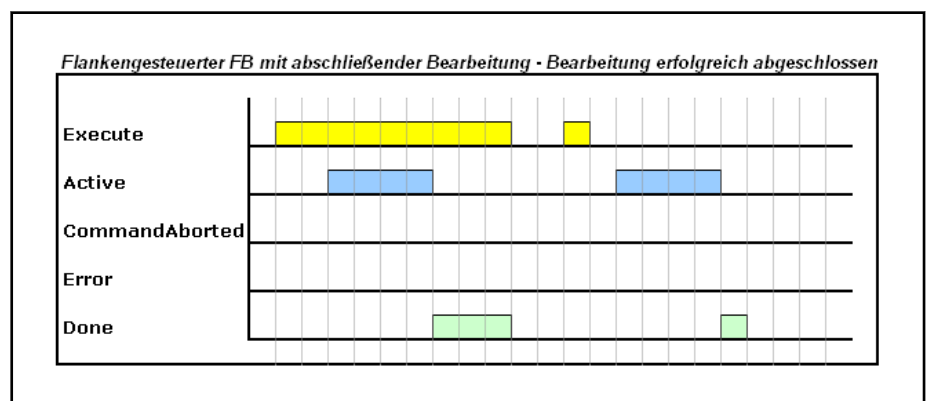


Abb. 15-9: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung Der Funktionsbaustein MB_ReadMaxRealValue erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.7 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-8: Bibliothekszuordnung

Schnittstellenbeschreibung

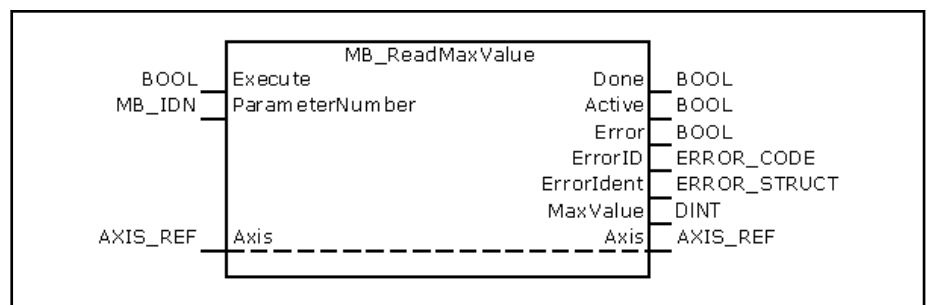


Abb. 15-10: FB MB_ReadMaxValue

	Name	Typ	Kommentar
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 Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
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 (" Fehlerbehandlung " auf Seite 372)
	MaxValue	DINT	Ist "Done" = TRUE, zeigt "MaxValue" den zulässigen maximalen Eingabewert des ausgewählten Parameters an
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse

Tab. 15-9: Schnittstelle von FB MB_ReadMaxValue

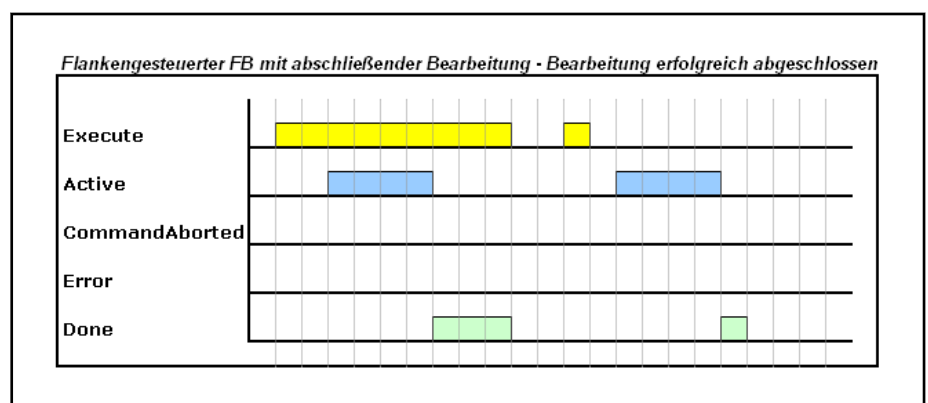
Kurzbeschreibung
Signal-Zeit-Diagramm

Abb. 15-11: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung Der Funktionsbaustein MB_ReadMaxValue erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.8 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-10: Bibliothekszuordnung

Schnittstellenbeschreibung

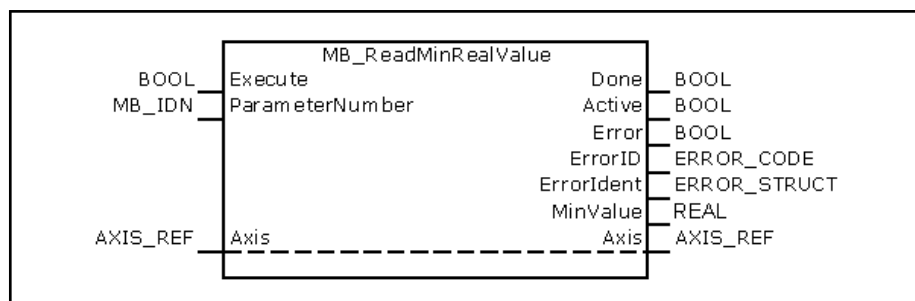


Abb. 15-12: FB MB_ReadMinRealValue

	Name	Typ	Kommentar
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 Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
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 (" Fehlerbehandlung " auf Seite 374)
	MinValue	REAL	Ist "Done" = TRUE, zeigt "MinValue" den zulässigen minimalen Eingabewert des ausgewählten Parameters an
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse

Tab. 15-11: Schnittstelle von FB MB_ReadMinRealValue

Signal-Zeit-Diagramm

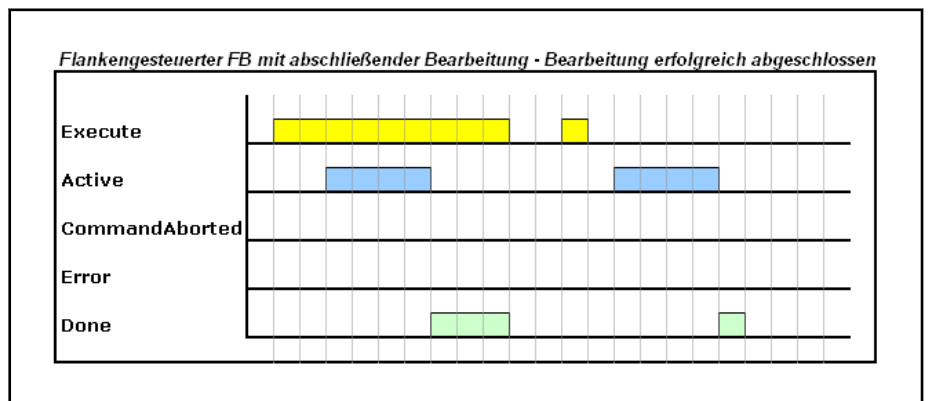


Abb. 15-13: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung Der Funktionsbaustein MB_ReadMinRealValue erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.9 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Execute " voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-12: Bibliothekszuordnung

Schnittstellenbeschreibung

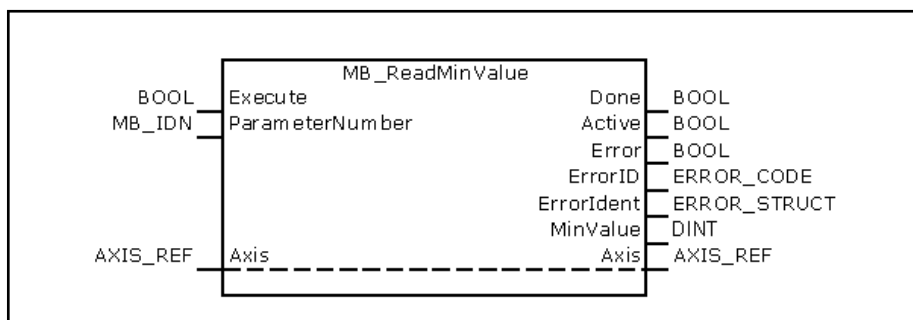


Abb. 15-14: FB MB_ReadMinValue

	Name	Typ	Kommentar
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 Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
VAR_OUTPUT	Done	BOOL	Gültiger Wert ist verfügbar

	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 (" Fehlerbehandlung " auf Seite 375)
	MinValue	DINT	Ist "Done" = TRUE, zeigt "MinValue" den zulässigen minimalen Eingabewert des ausgewählten Parameters an
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse

Tab. 15-13: Schnittstelle von FB MB_ReadMinValue

Signal-Zeit-Diagramm

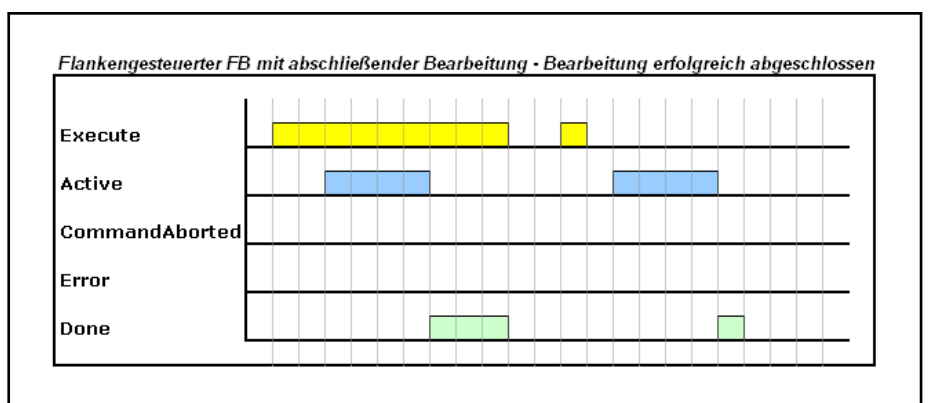


Abb. 15-15: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadMinValue erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek

(R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).

- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.10 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Execute " voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS	ML_PLCopen / Parameter
Basis-Funktionspaket PLC	

Tab. 15-14: Bibliothekszuordnung

Schnittstellenbeschreibung

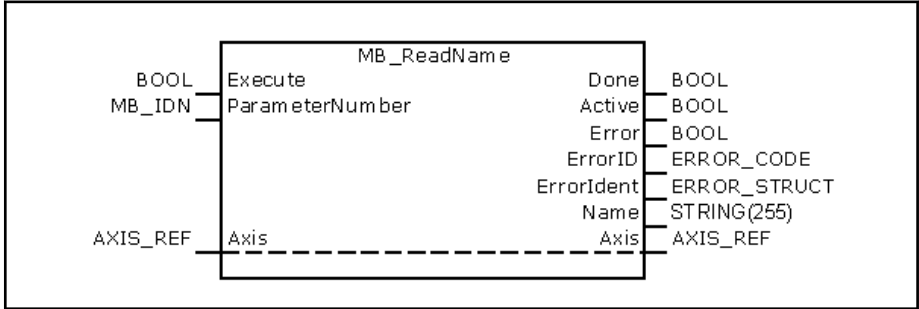


Abb. 15-16: FB MB_ReadName

	Name	Typ	Kommentar
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 Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490
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 (" Fehlerbehandlung " auf Seite 377)

	Name	Typ	Kommentar
	Name	STRING(255)	Ist "Done" = TRUE, zeigt "MinValue" den Namen des ausgewählten Parameters an
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse

Tab. 15-15: Schnittstelle von FB MB_ReadName

Signal-Zeit-Diagramm

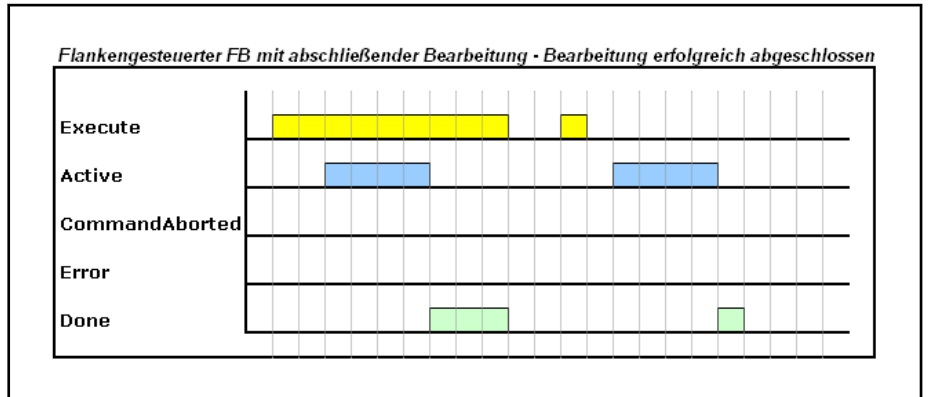


Abb. 15-17: Standard-Signal-Zeit-Diagramm



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadName erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.11 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS	ML_PLCopen / Parameter
Basis-Funktionspaket PLC	

Tab. 15-16: Bibliothekszuordnung

Schnittstellenbeschreibung

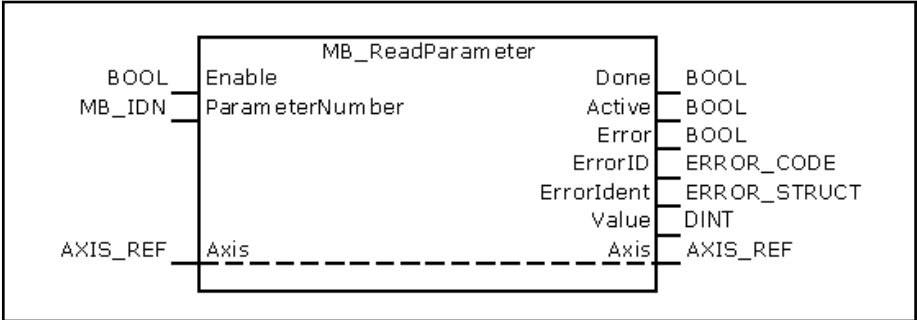


Abb. 15-18: FB MB_ReadParameter

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, siehe Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
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 (" Fehlerbehandlung " auf Seite 379)
	Value	DINT	4-Byte-Wert, als DINT dargestellt
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse, bzw. zum adressierten Objekt.

Tab. 15-17: Schnittstelle von FB MB_ReadParameter

Signal-Zeit-Diagramm

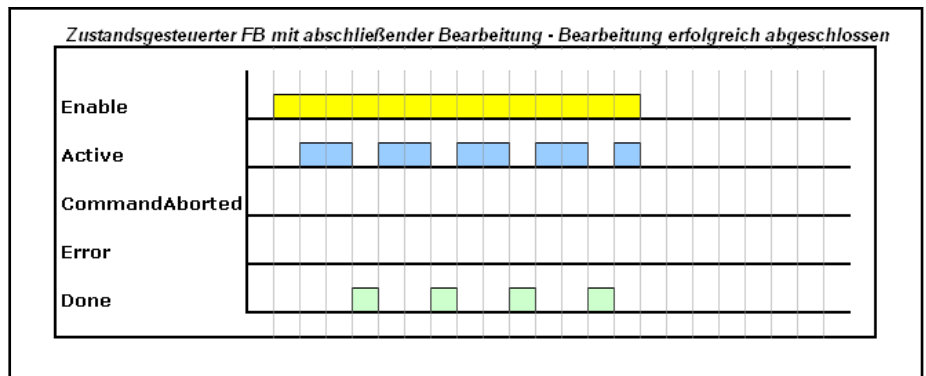


Abb. 15-19: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadParameter erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.12 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCOpen / Parameter

Tab. 15-18: Bibliothekszuordnung

Schnittstellenbeschreibung

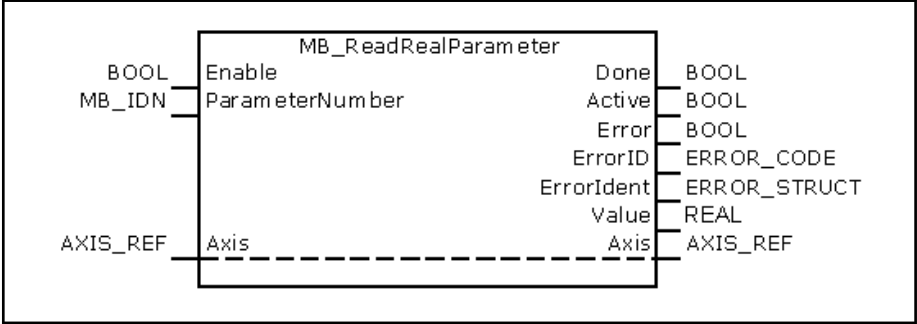


Abb. 15-20: FB MB_ReadRealParameter

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, siehe Kap. ML_PLCOpen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCOpen-Bibliotheken", Bibliothek (R911341490)
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 (" Fehlerbehandlung " auf Seite 381)
	Value	REAL	4-Byte-Wert als REAL-Zahl dargestellt
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt

Tab. 15-19: Schnittstelle von FB MB_ReadRealParameter

Signal-Zeit-Diagramm

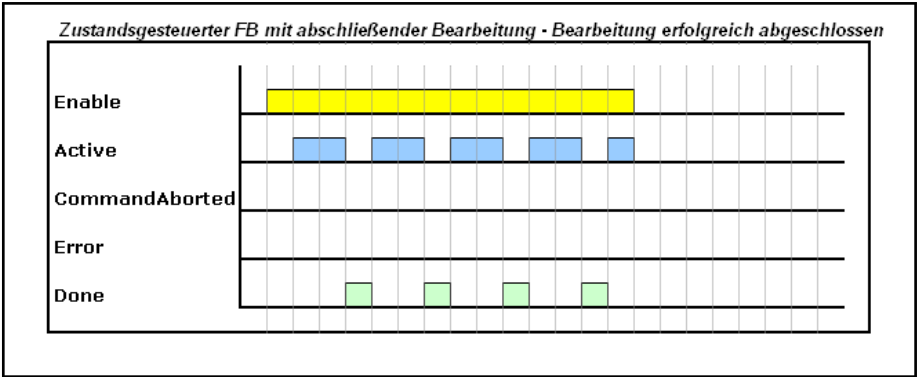


Abb. 15-21: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadRealParameter erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.13 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-20: Bibliothekszuordnung

Schnittstellenbeschreibung

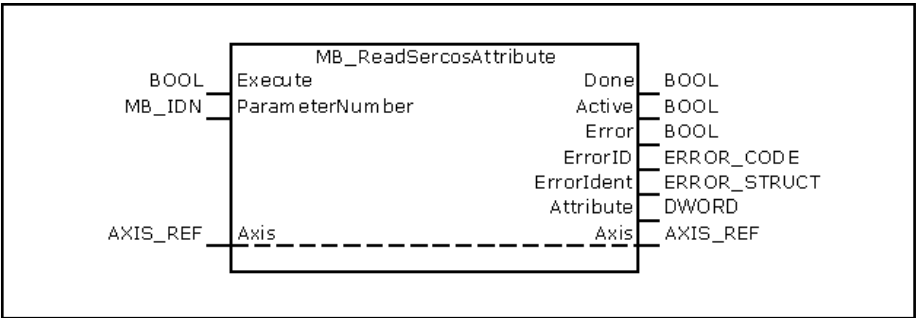


Abb. 15-22: FB MB_ReadSercosAttribute

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Lesen des Parameter-Attributs, bei 0-1-Übergang
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, siehe Kap. ML_PLCCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
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 (" Fehlerbehandlung " auf Seite 383)
	Attribute	DWORD	Ist "Done" = TRUE, zeigt "Attribute" das Attribut des ausgewählten Parameters an
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	enthält Informationen zur konkreten Achse

Tab. 15-21: Schnittstelle von FB MB_ReadSercosAttribute

Dekodierung des Attributs

Zur Dekodierung wird in der Bibliothek "ML_Base" der Funktionsbaustein IL_SercosAttribute bereitgestellt.

Signal-Zeit-Diagramm

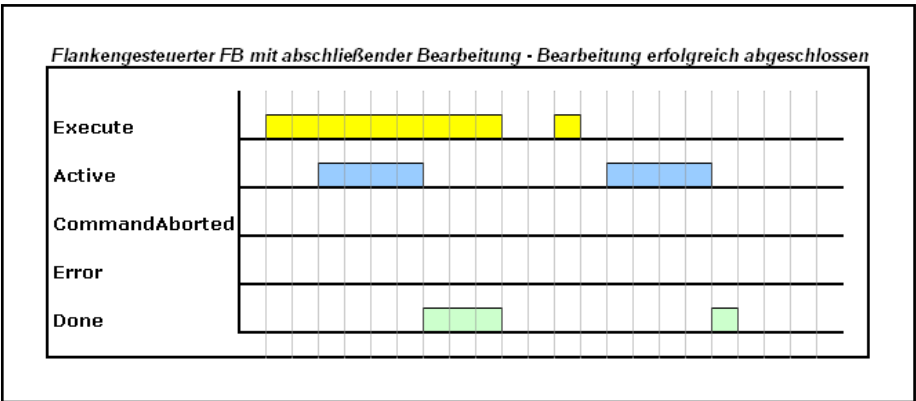


Abb. 15-23: Standard-Signal-Zeit-Diagramm

**Zur Belegung**

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadSercosAttribute erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.14 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Enable" voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-22: Bibliothekszuordnung

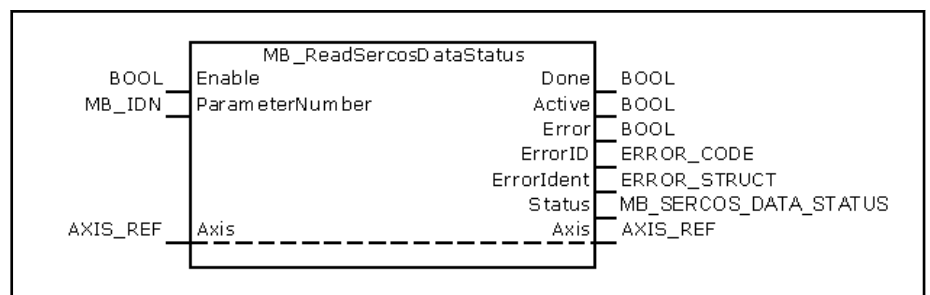
Schnittstellenbeschreibung

Abb. 15-24: FB MB_ReadSercosDataStatus

	Name	Typ	Kommentar
VAR_INPUT	Enable	BOOL	Kontinuierliches Lesen des Parameters, solange "Enable" TRUE ist
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, siehe Kap. ML_PLCOpen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
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 (" Fehlerbehandlung " auf Seite 385)
	Status	MB_SERCOS_DATA_STATUS	Status des gewählten Parameters Kap. "MB_SERCOS_DATA_STATUS, Struktur enthält alle Sercos-Kommandozustände" auf Seite 360
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse

Tab. 15-23: Schnittstelle von FB MB_ReadSercosDataStatus

Kurzbeschreibung Mit Hilfe dieses Funktionsbausteins und eines [MB_WriteParameter](#), Seite 388, ist es möglich, Antriebskommandos auszulösen und in ihrer Abarbeitung zu verfolgen:

- 0 - Kommando nicht gesetzt und nicht freigegeben
- 1 - Kommando gesetzt und freigegeben
- 3 - Kommando korrekt abgearbeitet
- 5 - Kommandoabarbeitung unterbrochen
- 7 - Kommandoabarbeitung läuft
- 15 - Fehler, Kommandoabarbeitung unmöglich
- 256 - Wert des Parameters außerhalb seines Min./Max.-Bereiches

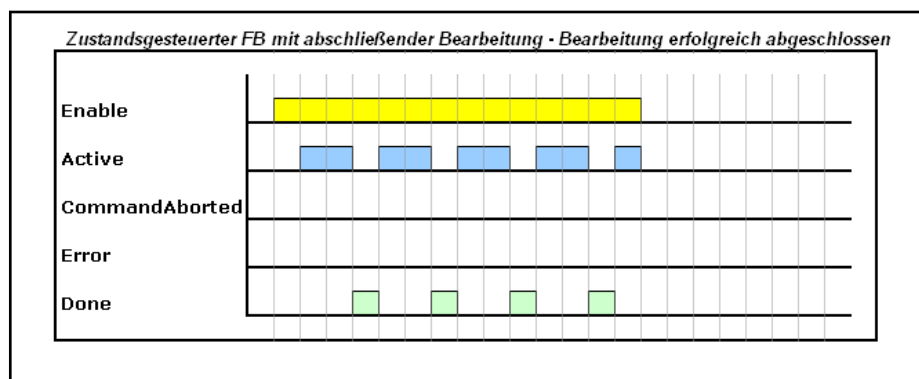
Signal-Zeit-Diagramm

Abb. 15-25: Standard-Signal-Zeit-Diagramm

**Zur Belegung**

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadSercosDataStatus erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.15 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.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek / Ordner
MLC 15VRS	ML_PLCopen / Parameter
Basis-Funktionspaket PLC	

Tab. 15-24: Bibliothekszuordnung

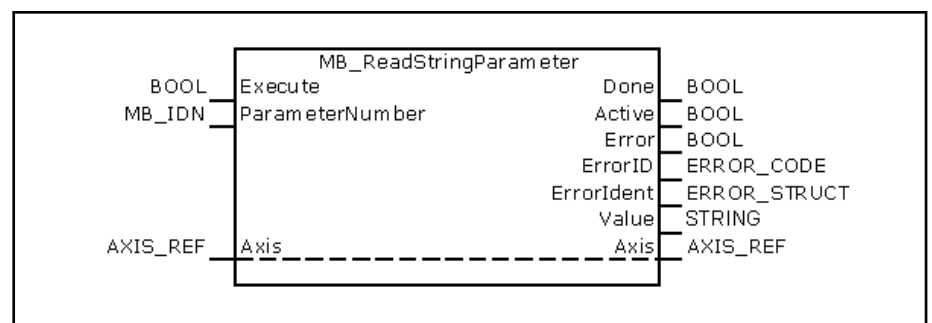
Schnittstellenbeschreibung

Abb. 15-26: FB MB_ReadStringParameter

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Startet das Lesen des Textes vom Parameter mit steigender Flanke
	Parameter- Number	MB_IDN	Nummer des gewünschten Parameters, siehe Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
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 (" Fehlerbehandlung " auf Seite 386)
	Value	STRING	Zeichenkette mit maximal 255 Zeichen
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse

Tab. 15-25: Schnittstelle von FB MB_ReadStringParameter

Signal-Zeit-Diagramm

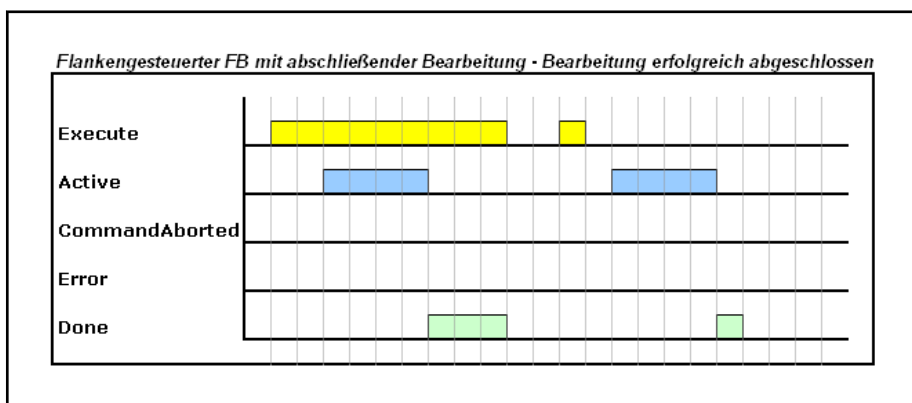


Abb. 15-27: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_ReadStringParameter erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.16 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 Funktionsbausteins, setzt einen erneuten 0/1-Übergang an "Execute" voraus.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-26: Bibliothekszuordnung

Schnittstellenbeschreibung

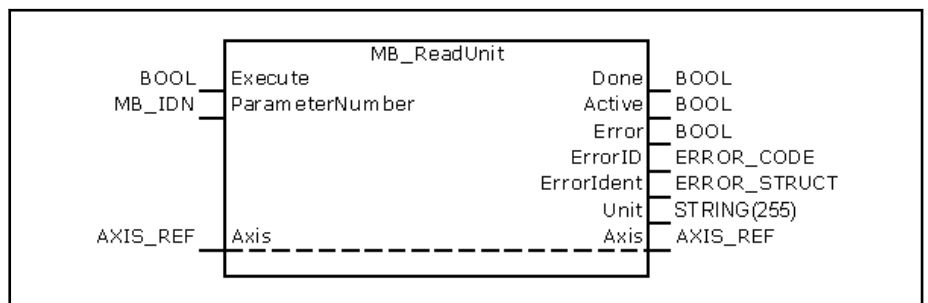


Abb. 15-28: FB MB_ReadUnit

	Name	Typ	Kommentar
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 Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
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.
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur konkreten Achse

Tab. 15-27: Schnittstelle von FB MB_ReadUnit

Signal-Zeit-Diagramm

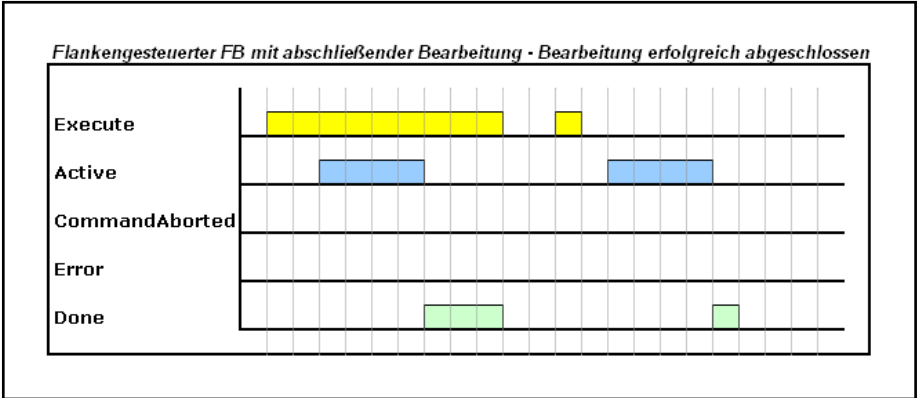


Abb. 15-29: Standard-Signal-Zeit-Diagramm



- Zur Belegung
- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
 - des AxisGroup-Eingangs bei K-Parametern
- Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung Der Funktionsbaustein MB_ReadUnit erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.17 MB_WriteParameter

Kurzbeschreibung Der Funktionsbaustein schreibt einen Wert in einen durch den Anwender vorgegebenen Parameter.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-28: Bibliothekszuordnung

Schnittstellenbeschreibung

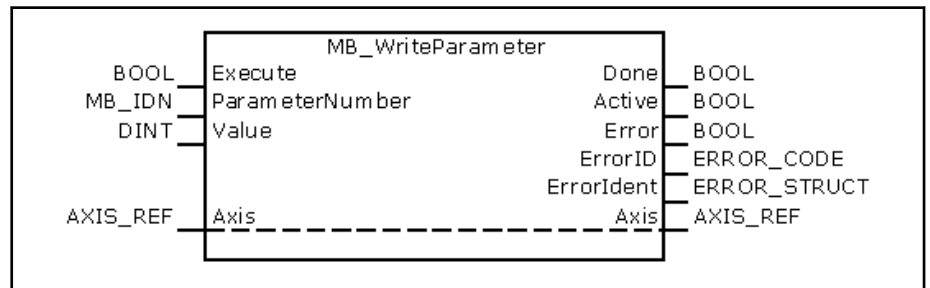


Abb. 15-30: FB MB_WriteParameter

	Name	Typ	Kommentar
VAR_INPUT	Execute	BOOL	Startet das Schreiben des Werts, der bei steigender Flanke ansteht, in den Parameter
	Parameter-Number	MB_IDN	Nummer des gewünschten Parameters, siehe Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
	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 (" Fehlerbehandlung " auf Seite 390)
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt

Tab. 15-29: Schnittstelle von FB MB_WriteParameter

Signal-Zeit-Diagramm

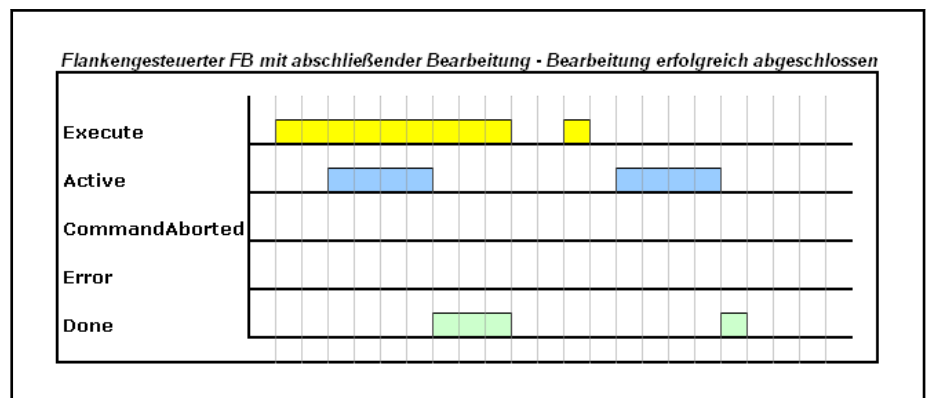


Abb. 15-31: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung Der Funktionsbaustein MB_WriteParameter erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.18 MB_WriteRealParameter

Kurzbeschreibung Der Funktionsbaustein schreibt einen Wert in einen anwenderspezifischen Parameter.

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-30: Bibliothekszuordnung

Schnittstellenbeschreibung

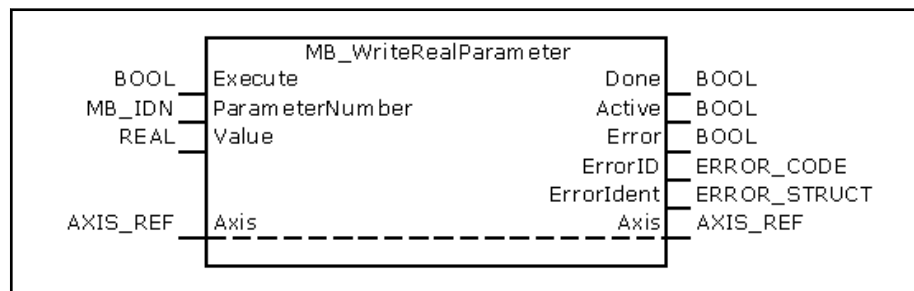


Abb. 15-32: FB MB_WriteRealParameter

	Name	Typ	Kommentar
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 Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
	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 (" Fehlerbehandlung " auf Seite 391)
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt

Tab. 15-31: Schnittstelle von FB MB_WriteRealParameter

Signal-Zeit-Diagramm

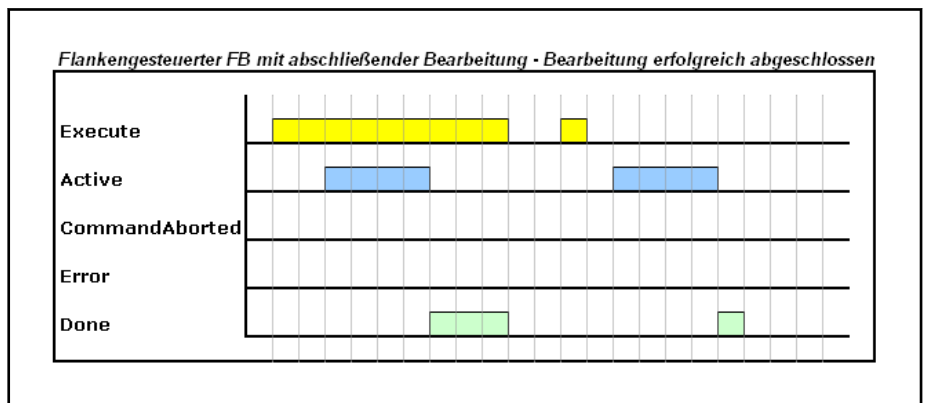


Abb. 15-33: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_WriteRealParameter erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. ML_PLCopen.library; Überblick: Diagnosen; SERCOS_TABLE, ERROR_TABLE 16#0010 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. ML_PLCopen.library; Überblick: Diagnosen; MLC_TABLE, ERROR_TABLE 16#0030 in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

15.19 MB_WriteStringParameter

Kurzbeschreibung

Der Funktionsbaustein schreibt eine Zeichenkette mit max. 255 Zeichen in einen anwenderspezifizierten Parameter.

Zuordnung Zielsystem/Bibliothek

Zielsystem	Bibliothek / Ordner
MLC 15VRS Basis-Funktionspaket PLC	ML_PLCopen / Parameter

Tab. 15-32: Bibliothekszuordnung

Schnittstellenbeschreibung

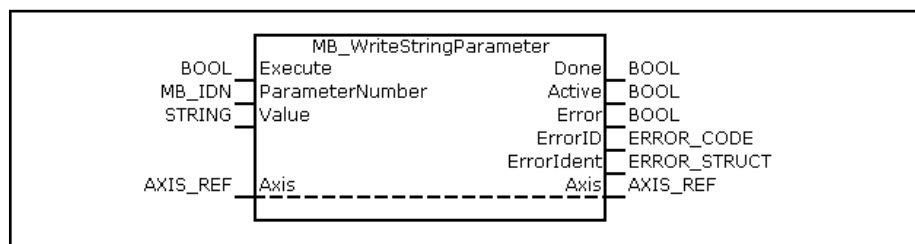


Abb. 15-34: FB MB_WriteStringParameter

	Name	Typ	Kommentar
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 Kap. ML_PLCopen.library; Parameterzugriff; Zugriff auf Daten einer Achse... in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490)
	Value	STRING[255]	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 (" Fehlerbehandlung " auf Seite 393)
VAR_IN_OUT CONSTANT	Axis	AXIS_REF	Enthält Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt

Tab. 15-33: Schnittstelle von FB MB_WriteStringParameter

Signal-Zeit-Diagramm

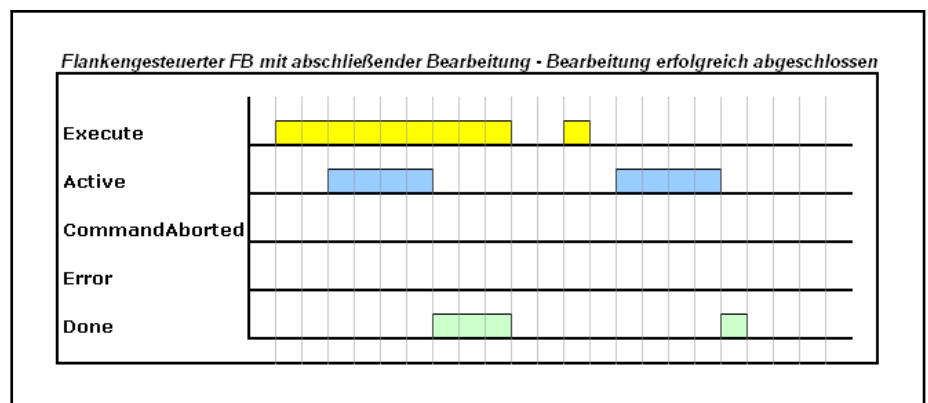


Abb. 15-35: Standard-Signal-Zeit-Diagramm



Zum Zeitbedarf für die Datenübertragung siehe Kap. ML_PLCopen.library, Zeitbedarf für den Datentransport mit Read- und WriteParameter- Funktionsbausteinen in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).



Zur Belegung

- des Axis-Eingangs bei A-, C-, M-, N-, O-Parametern, bzw.
- des AxisGroup-Eingangs bei K-Parametern

Siehe Kap. ML_PLCopen.library, Zugriff auf Daten einer Achse, Steuerung, Kinematik (Achsgruppe) oder eines anderen Objekts in der Dokumentation "IndraLogic XLC IndraMotion MLC 14VRS PLCopen-Bibliotheken", Bibliothek (R911341490).

Fehlerbehandlung

Der Funktionsbaustein MB_WriteStringParameter erzeugt folgende Arten von Fehlermeldungen:

ErrorID	Table	Add1	Add2	Beschreibung
ACCESS_ERROR	MLC_TABLE	16#F0220501	16#00000000	Falscher Datentyp

Weitere Fehler siehe

- Kap. SERCOS_TABLE, ERROR_TABLE 16#0010; IndraLogic XLC IndraMotion MLC 14VRS; PLCopen-Bibliotheken, Bibliothek (R911341490) - Fehler beim Zugriff auf den S- oder P-Parameter (z.B. Parameter wird nicht unterstützt).
- Kap. MLC_TABLE, ERROR_TABLE 16#0030 in der Doku IndraLogic XLC IndraMotion MLC 14VRS; PLCopen-Bibliotheken, Bibliothek (R911341490) - Fehler beim Zugriff auf die Achse/ die Steuerung.

16 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

- Abschaltvorwarnung..... 117
- Achsbezogene Ausgangssignale (SPS --> NC)
 - Achs-Override 100% qAx_Override100..... 79
 - Achsbetriebsart Bit 0 und 1 qAx_OpMo-
deSel_00 ... 01..... 71
 - Achse abgebaut qAx_Discharge..... 81
 - Achsgrundstellung qAx_Reset..... 74
 - Allgemeines..... 71
 - Antrieb Ein qAx_DrvOn..... 81
 - Antriebsperre qAx_SafDrvLock..... 75
 - Betriebsartenwahl (BA) qAx_SafOpMo-
deSwitch..... 75
 - Endschalter ausblenden qAx_SwLimOff..... 74
 - Endschalterbereich 0 ... 1
qAx_TrVLim_00 ... 01..... 74
 - Festanschlag aufheben qAx_FxStopRel..... 75
 - Gantry auf Masterposition qAx_MasterPos.... 80
 - Handrad Drehrichtung qAx_HandwDir..... 77
 - Handradauswahl Bit 0 und Bit 1
qAx_HandwSel_00 ... 01..... 77
 - Handvorschub Bit 0 ... Bit 3 qAx_Man-
Feed_00 ... 03..... 78
 - Istwert-Versatz übernehmen qAx-Ta-
keActOffs..... 80
 - Kopplungsfehler ausblenden qAx_LagEr-
rOff..... 79
 - Kundeneingang 1 ... 8 qAx_Custom1 ... 8..... 79
 - Manuell+ qAx_JogPlus, Manuell-
qAx_JogMinus..... 71
 - Momentenreduktion qAx_TrqLi..... 81
 - Nächste Rasterposition qAx_NextNotch..... 74
 - Override Bit 0 ... Bit 15 qAx_Overri-
de_00 ... 15..... 79
 - Positionshandrad qAx_HandwPosMode..... 77
 - Prüfeingang Si-Status qAx_SafCheckIn-
putState..... 76
 - Referenzschalter qAx_HomeSwitch..... 78
 - Schrittmaß als Durchmesser qAx_JogDia.... 73
 - Si-Schalter 1 (S2) qAx_SafSwitch1..... 75
 - Si-Schalter 2 (S2) qAx_SafSwitch2..... 75
 - Sollposition halten qAx_FrzlpoPos..... 81
 - Status Si-Signale qAx_SafTechState..... 76
 - Stillstandsfehler ausblenden qAx_TrqEr-
rOff..... 79
 - Überführzeit abkürzen qAx_SafRedT-
ransTime..... 76
 - Verzögerte Fehlerreaktion abkürzen
qAx_RedDelayErrClass1..... 80
 - Vorschub Sperre qAx_DrvLock..... 82
 - Zustimmtaste (ZT) qAx_SafEnablCtrl..... 75
- Achsbezogene Ausgangssignale (SPS -->NC)
 - Schrittmaß in Inch qAx_JogInch..... 73
- Achsbezogene Eingangssignale (SPS --> NC)
 - Achs-Override 0% iAx_Override0..... 89
 - Achs-Override 100% iAx_Override100..... 89
 - Achse auf Rasterposition iAx_NotchPos..... 85
 - Achse grundgestellt iAx_Reset..... 86
 - Achse in Position iAx_InPos..... 85
 - Achse läuft iAx_Run..... 85
 - Achse vor Endpunkt iAx_DistCtrl..... 86
 - Achsgeschwindigkeit erreicht iAx_Prog-
SpReach..... 86
 - Änderung Zustandsklasse-2
iAx_DrvChangeClass2..... 91
 - Änderung Zustandsklasse-3
iAx_DrvChangeClass3..... 92
 - Antrieb betriebsbereit iAx_DrvReady..... 93
 - Antrieb in Betrieb iAx_DrvAct..... 94
 - Ausgleichbarer Gantry-Sollwert-Versatz
überschritten iAx_CmdOffsExceed..... 90
 - Fahrbefehl iAx_TrVCmd..... 84
 - Fehler Zustandsklasse-1 iAx_DrvErr-
Class1..... 91
 - Festanschlag aktiv iAx_FxStopAct..... 86
 - Festanschlag erreicht iAx_FxStopRea-
ched..... 86
 - Freigabe zur Leistungszuschaltung
iAx_DrvPower..... 93
 - Gantry-Sollwert-Versatz aktiv
iAx_CmdOffsExst..... 90
 - Index Master-Achse Bit 0 ... Bit 7
iAx_MasterAxindex_00 ... 07..... 90
 - Kanalnummer Bit 0 ... Bit 5 iAx_ChIn-
dex_00 ... 05..... 88
 - Kundenausgang 1 ... 8 iAx_Custom1 ... 8..... 90
 - Moment reduziert iAx_TrqLim..... 93
 - Nachlauffehler iAx_CoupleLag..... 90
 - Negative Verfahrrichtung iAx_TrVDirNeg..... 85
 - Positionsschaltpunkt 1 ... 8 iAx_PosS-
witch1 ... 8..... 87
 - Referenzpunkt bekannt iAx_RefKnown..... 83
 - Referenzpunkt erreicht iAx_RefReached..... 84
 - SCS-Signalstatus 0 ... 15 iAx_ScsSta-
te00 ... 15..... 89
 - Si-Betriebsart Bit 0 ... Bit 3 iAx_SafOp-
Mode_00 ... 03..... 86
 - Status sichere Lage iAx_SafStatePos..... 87
 - Statusausgang Regler iAx_SafCtrlOut-
putState..... 87
 - Stillstandsfehler iAx_TrqExceed..... 90
 - Testbetrieb iAx_DryRun..... 93
 - Verzögerte Fehlerreaktion iAx_DelayErr-
Class1..... 90
- Achsbezogene Interface-Signale..... 67
 - SPS-Ausgangssignale - Übersicht..... 24
 - SPS-Eingangssignale - Übersicht..... 25
 - Übersicht achsbezogene Interface-Signale... 67
- Achsen..... 280
 - Übersicht..... 280

Index

Achsenistwerte lesen (MT_ScsPos)		
Funktion.....	129	
Projektierungsbeispiel.....	130	
Statusmeldungen.....	131	
Active.....	233	
Aktive Betriebsart ermitteln.....	263	
Aktive Schneide des aktiven Werkzeug ermitteln.....	345	
Aktive Tippgeschwindigkeit ermitteln.....	321	
Aktiven JogMode ermitteln.....	291	
Aktiven Sicherheitstechnik-Betriebszustand ermitteln.....	281	
Aktiven Vorschub-Override ermitteln.....	262	
Aktives Nc-Programm ermitteln.....	250	
Aktuellen Kanalstatus ermitteln.....	248	
Aktuellen Programmfortschritt ermitteln.....	252	
Anbindung digitale I/Os.....	127	
Ändern der NC-Betriebsarten.....	270	
Ändern der NC-Hauptbetriebsarten.....	266, 275	
Ändern des JodMode.....	292	
Ändern und Ermitteln der aktiven Tippdistanz.....	296	
Ändern und Ermitteln der aktiven Tippgeschwindigkeit.....	325	
Ändern von Datensätzen.....	167	
Ändern von Werkzeugdaten.....	157	
Änderungsverlauf.....	1	
Anregungen.....	7	
ANSI Z535.6-2006.....	6	
Antriebsabschaltung mit Verriegelung.....	116	
Antriebsstatus.....	356	
Antriebsstatus ermitteln.....	309, 328	
Antriebsstatus incl. Fehler- und Warnungsdetails ermitteln.....	311, 330	
Ausgangssignale (SPS --> NC).....	32	
Automatischer Sercos III I/O-Tausch (MT_S3AutoAdapt)		
Allgemeines.....	208	
Bausteinparameter.....	208	
Projektierungsbeispiel.....	208	
Statusmeldungen.....	209	
AxisNo.....	142	
B		
Bausteintypen.....	231	
Bausteintypen und ihre Schnittstellen.....	232	
BCD-Code.....	123	
Berechnet die verstrichene Zeit.....	349	
Berechnet die Wegstrecke aus den angegebenen Positionsdaten.....	346	
Beschwerde.....	7	
Bestimmungsgemäßer Gebrauch		
Anwendungsbereiche.....	10	
Einleitung.....	9	
Einsatzfälle.....	10	
Betriebsart einer asynchronen Achse.....	285, 305	
Betriebsart einer asynchronen Achse ändern.....	302	
Betriebsart einer asynchronen Achse ermitteln.....	301	
Bibliothek		
MT_BaseAdditional.library.....	229	
MT_PLCCopen.....	359	
Binär codierte aktive Kanalzustände.....	58	
Bitcodierte Hilfsfunktionen.....	122	
C		
Chan.....	144	
CommandAborted.....	233	
CondMask.....	161	
CPL-Variablen.....	185	
Crash-Zustand von NC an SPS.....	36	
Crash-Zustand von SPS an NC (IPO).....	34	
D		
Darstellung von Informationen		
Bezeichnungen und Abkürzungen.....	7	
Data Types.....	352, 360	
Datenbanktabelle laden (MT_DbLoad)		
Funktion.....	178	
Projektierungsbeispiel.....	179	
Statusmeldungen.....	180	
Datenbanktabelle sichern (MT_DbSave)		
Funktion.....	174	
Projektierungsbeispiel.....	176	
Statusmeldungen.....	177	
Datensätze bearbeiten (MT_DbData)		
Funktion.....	167	
Projektierungsbeispiel.....	172	
Statusmeldungen.....	173	
Datensätze bearbeiten (MT_P_DbData)		
Funktion.....	157	
Projektierungsbeispiel.....	158	
Statusmeldungen.....	160	
Datensatzlisten bearbeiten (MT_DbRecList)		
Funktion.....	161	
Projektierungsbeispiel.....	164	
Statusmeldungen.....	166	
Datentypen.....	352, 360	
Diagnosedaten lesen (MT_DiagCode)		
Allgemeines.....	198	
Bausteinparameter.....	199	
Projektierungsbeispiel.....	202	
Statusmeldungen.....	202	
Diagnosetexte lesen (MT_DiagText)		
Allgemeines.....	203	
Bausteinparameter.....	204	
Projektierungsbeispiel.....	206	
Statusmeldungen.....	206	
Digitale I/Os.....	127	
Dokumentation		
Änderungsverlauf.....	1	
Done.....	233	

DP-Master-Schnittstelle.....	127
Durchmesser-Inkremente.....	73

E

Ein- und Ausgänge flankengesteuerter Funktionsbausteine, abschließende Bearbeitung.....	237
Ein- und Ausgänge flankengesteuerter Funktionsbausteine, permanente Bearbeitung.....	243
Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, abschließende Bearbeitung.....	234
Ein- und Ausgänge zustandsgesteuerter Funktionsbausteine, permanente Bearbeitung.....	240
Eingangssignale (NC --> SPS).....	35
Achstrafo 2 inaktiv iCh_ActFunc04.....	63
Aktive Betriebsart iCh_OpMode_00 ... 03.....	53
Asynchrones Unterprogramm 1 ... 8	
iCh_ASub1 ... 8.....	58
Bereit zum Wiederanfahren iCh_Ready-ReEnter.....	57
CPL-Kundenausgang 1 ... 16	
iCh_Cpl01 ... 16.....	61
Eilgang aktiv iCh_G0Act.....	61
Eilgang Override 0% iCh_RapOverride0.....	65
Eilgang Override 100% iCh_RapOverride100.....	65
Einlesesperre aktiv iCh_TransferLockAct.....	55
G41/141 aktiv iCh_G41G141Act,	
G42/142 aktiv iCh_G42G142Act.....	62
G70 aktiv iCh_ActFunc01.....	63
G95 aktiv iCh_ActFunc03.....	63
G96 aktiv iCh_ActFunc18.....	65
Gewindebohren aktiv iCh_ActFunc07.....	64
Gewindeschneiden aktiv iCh_ActFunc08.....	64
Gewindezyklus aktiv iCh_ActFunc06.....	63
iCh_OptJump.....	59
Inpos Bereich 2 aktiv iCh_InPosAct.....	62
Kanal Grundstellung iCh_Reset.....	56
Kanalzustand Bit 0 ... 4 iCh_State_00 .. 04.....	58
Kundenausgang 1 ... 8 iCh_Custom1 ... 8.....	61
NC-Bereit iCh_NCReady.....	54
Override 0% iCh_Override0.....	61
Override 100% iCh_Override100.....	61
Programm Ende M30 iCh_ProgStopM30.....	56
Programm Halt M0 iCh_ProgStopM0.....	56
Programm läuft iCh_ProgRun.....	55
Programmposition aktiv iCh_ActFunc05.....	63
Rückwärtsbetrieb aktiv iCh_RetraceAct.....	62
Satz überlesen aktivieren iCh_Blkslash.....	59
Satzvorlauf aktiv iCh_SRunAct.....	60
Testbetrieb iCh_DryRun.....	54
Übersicht Interface-Signale für Satzvorlauf...	60
Vorschub 100% aktiv iCh_ActFunc02.....	63
Vorschub-Halt aktiv iCh_FeedHoldAct.....	56

Wahlweiser Halt aktivieren iCh_OptStop.....	59
Wegfahren beendet iCh_RemoveFinish.....	57
Werkzeugdrehung (TangTool (TTL))	
iCh_TangTRotCmd.....	62
Werkzeugkorrektur aktiv Bit 0 ... Bit 4	
iCh_ActFunc09 ... 13.....	64
Wiederanfahren aktiv iCh_SRunRepos.....	61
Wiederanfahren auf Wegtipp-Pfad	
iCh_RetContPath.....	57
Wiederanfahren zum Anfahrpunkt aktiv	
iCh_RetpaAct.....	57
Wiedereintritt aktiv iCh_ReEnterAct.....	57
Wiedereintritt aktiv iCh_SRunReEnter.....	61
Zweiter Vorschub iCh_Feedrate2.....	66
ElemPath.....	167
Enable.....	233
Endsatz.....	133
Erweiterte NC-Satzvorgabe (MT_NcBlkExt)	
Funktion.....	137
Projektierungsbeispiel.....	138
Statusmeldungen.....	139
Execute.....	233
ExecuteLock.....	233

F

Fast Action Ausgangsdaten schreiben (MT_FaDataWrite)	
Allgemeines.....	222
Projektierungsbeispiel.....	223
Statusmeldungen.....	222
Fast Action Eingangsdaten lesen (MT_FaDataRead)	
Allgemeines.....	219
Bausteinparameter.....	219
Projektierungsbeispiel.....	220
Statusmeldungen.....	221
Feedback.....	7
Fehler Infos löschen.....	32
Fehlercode.....	129
flankengesteuerte Signale.....	71

G

G0-Eilgang.....	49
Gefahrenhinweise.....	5
General Interface.....	15
Generiert eine ganzzahlige Zufallszahl.....	351
Getriebe in Leerlaufstellung.....	113
Getriebestufenanwahl.....	112
Getriebeumschaltung.....	113
Globale Interface.....	15
Globale Interface-Signale.....	31
Signalbeschreibung.....	32
SPS-Ausgangssignale.....	32
SPS-Ausgangssignale - Übersicht.....	19
SPS-Eingangssignale.....	35
SPS-Eingangssignale - Übersicht.....	19
Übersicht globale Interface-Signale.....	31

Index

H

Handrad ändern.....	286
Handraddaten (MT_Handw)	
Funktion.....	145
Projektierungsbeispiel.....	146
Statusmeldungen.....	146
Hardware Warnung.....	35
Helpdesk.....	395
Hilfsfunktionen.....	121
Hotline.....	395
Hub beabsichtigt.....	35
Hub ein.....	33
Hub läuft nicht.....	35
Hubreservierung.....	33
Hubsperre.....	33

I

iAx_ChIndex_00 ... 05.....	88
iAx_CmdOffsExceed.....	90
iAx_CmdOffsExst.....	90
iAx_CoupleLag.....	90
iAx_Custom1 ... 8.....	90
iAx_DelayErrClass1.....	90
iAx_DistCtrl.....	86
iAx_DrvAct.....	94
iAx_DrvChangeClass2.....	91
iAx_DrvChangeClass3.....	92
iAx_DrvErrClass1.....	91
iAx_DrvPower.....	93
iAx_DrvReady.....	93
iAx_DryRun.....	93
iAx_FxStopAct.....	86
iAx_FxStopReached.....	86
iAx_InPos.....	85
iAx_MasterAxindex_00 ... 07.....	90
iAx_NotchPos.....	85
iAx_Override0.....	89
iAx_Override100.....	89
iAx_PosSwitch1 ... 8.....	87
iAx_ProgSpReach.....	86
iAx_RefKnown.....	83
iAx_RefReached.....	84
iAx_Reset.....	86
iAx_Run.....	85
iAx_SafCtrlOutputState.....	87
iAx_SafOpMode_00 ... 03.....	86
iAx_SafStatePos.....	87
iAx_ScsState00 ... 15.....	89
iAx_TrqExceed.....	90
iAx_TrqLim.....	93
iAx_TrxCmd.....	84
iAx_TrVDirNeg.....	85
iCh_ActFunc01.....	63
iCh_ActFunc02.....	63
iCh_ActFunc03.....	63
iCh_ActFunc04.....	63
iCh_ActFunc05.....	63

iCh_ActFunc06.....	63
iCh_ActFunc07.....	64
iCh_ActFunc08.....	64
iCh_ActFunc09 ... 13.....	64
iCh_ActFunc18.....	65
iCh_ASub1 ... 8.....	58
iCh_Blkslash.....	59
iCh_CoordCoupleAct.....	62
iCh_Cpl01 ... 16.....	61
iCh_Custom1 ... 8.....	61
iCh_DryRun.....	54
iCh_FeedHoldAct.....	56
iCh_Feedrate2.....	66
iCh_G0Act.....	61
iCh_G41G141Act.....	62
iCh_G42G142Act.....	62
iCh_InPosAct.....	62
iCh_NCReady.....	54
iCh_OpMode_00 ... 03.....	53
iCh_OptJump.....	59
iCh_OptStop.....	59
iCh_Override0.....	61
iCh_Override100.....	61
iCh_ProgRun.....	55
iCh_ProgStopM0.....	56
iCh_ProgStopM30.....	56
iCh_RapOverride0.....	65
iCh_RapOverride100.....	65
iCh_ReadyReEnter.....	57
iCh_ReEnterAct.....	57
iCh_RemoveFinish.....	57
iCh_Reset.....	56
iCh_RetContPath.....	57
iCh_RetpaAct.....	57
iCh_RetraceAct.....	62
iCh_SRunAct.....	60
iCh_SRunReEnter.....	61
iCh_SRunRepos.....	61
iCh_State_00 .. 04.....	58
iCh_TangTRotCmd.....	62
iCh_TransferLockAct.....	55
iGen_CrashAckn.....	36
iGen_CrashStoreData.....	36
iGen_HardwareState.....	35
iGen_NoStroke.....	35
iGen_Reset.....	35
iGen_SercosIO_Reset.....	35
iGen_StrokeIntend.....	35
IL_SIII_ATTRIBUTE.....	360
Inch-Inkmente.....	73
InGear.....	232
InOperation.....	232
InSync.....	232
InTorque.....	232
InVelocity.....	232
iSp_CAxAct.....	108
iSp_CAxSwitch.....	109
iSp_CoupleErr.....	115

iSp_CoupleIndex_00 ... 02.....	115
iSp_Custom1 ... 8.....	115
iSp_DelayErrClass1.....	116
iSp_DrvAct.....	119
iSp_DrvChangeClass2.....	117
iSp_DrvChangeClass3.....	117
iSp_DrvErrClass1.....	116
iSp_DrvPower.....	118
iSp_DrvReady.....	119
iSp_DryRun.....	118
iSp_Gear1Sel ... 4Sel.....	112
iSp_GearChange.....	113
iSp_GearIdleSel.....	113
iSp_IdleSpeed.....	113
iSp_InPos.....	110
iSp_Master.....	115
iSp_OrientateAct.....	111
iSp_OrientateFinish.....	111
iSp_Override0.....	114
iSp_Override100.....	114
iSp_PosCtrl.....	110
iSp_ProgSpReach.....	110
iSp_Reset.....	110
iSp_SafCtrlOutputState.....	112
iSp_SafOpMode_00 ... 03.....	111
iSp_SafStatePos.....	112
iSp_ScsState00 ... 15.....	114
iSp_SpLim.....	110
iSp_Stop.....	110
iSp_Synchr1.....	115
iSp_Synchr2.....	116
iSp_ToolLimAct.....	114
iSp_TurnCmd.....	109
iSp_TurnDirM4.....	109

K

Kanal-Neustart (MT_ChانRestart)	
Funktion.....	139
Projektierungsbeispiel.....	140
Statusmeldungen.....	141
Kanalbezogene Ausgangssignale (SPS --> NC)	
Anwahl Betriebsart qCh_OpMode-	
Sel_00 ... 03.....	41
Asynchrones Unterprogramm 1 ... 8	
qCh_ASub1 ... 8.....	47
Automatische Wiederanwahl aus	
qCh_ReSelOff.....	44
Automatischer Restart qCh_Restart.....	43
Bahnbewegung rückwärts qCh_RetraceMov-	
Rev.....	51
Bahnbewegung vorwärts qCh_Retrace-	
MovFwd.....	51
Bedingter Sprung qCh_OptJump.....	48
Betriebsart von SPS qCh_OpModePlc.....	42
Eilgang 100% qCh_Rapid100.....	49
Eilgang Override Bit 0 ... Bit 15 qCh_Ra-	
pOverride_00 ... 15.....	52

Einlesesperre qCh_TransferLock.....	43
Grundstellung qCh_CtrlReset.....	44
Handradauswahl Bit 0 qCh_Hand-	
wSelWcs_00 und Handradauswahl Bit 1	
qCh_HandwSelWcs_01.....	47
Handraddrehrichtung qCh_HandwDirWcs.....	48
Kopplung beenden qCh_CoordCoupleOff.....	51
Kundeneingang 1 ... 8 qCh_Custom1 ... 8.....	50
Maschinenfunktion qCh_MachineFunction.....	44
NC-Start qCh_NCStart.....	43
Online-Korrektur Freigabe qCh_Onl-	
CorrWcs.....	50
Online-Korrektur Richtung qCh_Onl-	
CorrWcsDir.....	50
Override 100% qCh_Override100.....	49
Override Bit 0 ... Bit 15 qCh_Overri-	
de_00 ... 15.....	50
Positionshandrad qCh_HandwPosMode.....	48
Reduzierter Eilgang qCh_RedRap.....	49
Restweg löschen qCh_CancDist.....	45
Rückwärtsbetrieb qCh_Retrace.....	50
Satz überlesen qCh_BlkSlash.....	48
Schneller Rückzug qCh_Retract.....	47
Syntaxfehler löschen qCh_CancSyntaxEr-	
ror.....	45
Testeilgang qCh_TestRap.....	52
Testvorschub qCh_TestFeed.....	52
TTL Freigabe qCh_TangTRotRel.....	51
Umschalten auf nächsten Satz	
qCh_NextBlk.....	46
Vorschub Halt qCh_FeedHold.....	43
Vorschubsperr qCh_FeedStop.....	44
Wahlweiser Halt qCh_OptStop.....	48
WCS Manuell + qCh_JogPlusWcs und	
WCS Manuell - qCh_JogMinusWcs.....	47
Wiederanfahren qCh_RetCont.....	46
Zweiter Vorschub qCh_Feedrate2.....	53
Kanalbezogene Interface-Signale.....	37
Eingangssignale (NC --> SPS).....	53
Signalbeschreibung.....	41
SPS-Ausgangssignale - Übersicht.....	20
SPS-Eingangssignale - Übersicht.....	22, 39
SPS-Kanalbezogene Ausgangssignale.....	41
SPS-kanalbezogene Ausgangssignale -	
Übersicht.....	37
Übersicht kanalbezogene Interface-Sig-	
nale.....	37
Kanalnummer prüfen.....	246
Kanalstruktur.....	37
Kanalvorschub ändern.....	261
Kommunikationsstruktur.....	13
Konfiguration der Maschinenzustandsanzei-	
ge (MZA)	
Aktivierung der Meldungen.....	227
Allgemeines.....	225
Aufbau der MZA-Datei.....	226
Ergänzende Hinweise.....	228
Konfiguration des MZA-Bit-Interface.....	225

Index

Parametrierung der Maschinenzustands- anzeige (MZA).....	225	MB_ReadListParameter.....	366
Konfiguration des SPS-NC-Bit-Interface		MB_ReadMaxRealValue.....	368
Allgemeines.....	15	MB_ReadMaxValue.....	370
Das Achs-Interface (Axes Interface).....	16	MB_ReadMinRealValue.....	372
Das Globale Interface (General Interface).....	15	MB_ReadMinValue.....	374
Das Kanal-Interface (Channel Interface).....	15	MB_ReadName.....	376
Das Spindel-Interface (Spindle Interface).....	17	MB_ReadParameter.....	377
Konfiguration von SPS-spezifischen Daten		MB_ReadRealParameter.....	379
in IndraWorks.....	225	MB_ReadSercosAttribute.....	381
Konfiguration der Maschinenzustandsan- zeige (MZA).....	225	MB_ReadSercosDataStatus.....	383
Konfigurationen.....	15	MB_ReadStringParameter.....	385
Konfiguration des SPS-NC-Bit-Interface.....	15	MB_ReadUnit.....	387
Koordinatenkopplung.....	51	MB_SERCOS_DATA_STATUS.....	360
Koppelabstand.....	90	MB_WriteParameter.....	388
Kritik.....	7	MB_WriteRealParameter.....	390
Kundenfeedback.....	7	MB_WriteStringParameter.....	392
L		Meldungen der Betriebszustände.....	117
Leerlaufstellung des Getriebes.....	113	Messtasterachse überwachen.....	288
Lesen der Motion Control Data-Services		MT_AxAsyncGetOPMode.....	301
(MT_MCoDS)		MT_AxAsyncOPMode.....	305
Allgemeines.....	206	MT_AxAsyncSetOPMode.....	302
Bausteinparameter.....	207	MT_AxAsyncSetOverride.....	308
Projektierungsbeispiel.....	207	MT_AxGetChIndex.....	285
Statusmeldungen.....	208	MT_AxGetJogMode.....	291
Lesen eines Parameterwertes.....	147	MT_AxGetPowerState.....	309
Lesen von Datensätzen.....	167	MT_AxGetPowerStateExt.....	311
Lesen von Werkzeugdaten.....	157	MT_AxHandwheelAcs.....	286
Lesen/Schreiben von permanenten CPL-Va- riablen (MT_CplData)		MT_AXIS_ASYNC_OPERATION_MODE.....	354
Funktion.....	185	MT_AXIS_JOG_MODE.....	355
Projektierungsbeispiel.....	187	MT_AxJogMode.....	296
Statusmeldungen.....	188	MT_AxPower.....	316
Lesen/Schreiben von permanenten CPL-Va- riablen mit Typkonvertierung (MT_CplDataConv)		MT_AxProbeMonitor.....	288
Funktion.....	188	MT_AxSetJogMode.....	292
Projektierungsbeispiel.....	190	MT_AxSetPowerOff.....	314
Statusmeldungen.....	191	MT_AxSetPowerOn.....	312
Log-Eintrag im Aktionsrekorder (MT_LogData)		MT_BaseAdditional.library.....	229
Allgemeines.....	211	Achsen.....	280
Bausteinparameter.....	211	Datentypen.....	352
Projektierungsbeispiel.....	213	Einführung und Übersicht.....	229
Statusmeldungen.....	213	NC-Kanal.....	245
Löschen von Datensätzen.....	161	Sonstige.....	346
Löschen von Werkzeugdatensätzen.....	153	Spindeln.....	319
M		Werkzeugmanagement.....	339
Manuell.....	71	MT_CfgData.....	143
Manuell+.....	71	MT_CHANNEL_OPERATION_MODE.....	352
Maschinenparameter lesen (MT_CfgData)		MT_CHANNEL_STATE.....	353
Funktion.....	143	MT_ChanRestart.....	139
Projektierungsbeispiel.....	144	MT_ChGetActiveProgram.....	250
Statusmeldungen.....	145	MT_ChGetFeedOverride.....	262
Maschinenzustandsanzeige.....	225	MT_ChGetOPMode.....	263
MB_Command.....	361	MT_ChGetState.....	248
		MT_ChNumberValidation.....	246
		MT_ChOPMode.....	275
		MT_ChSetFeedOverride.....	261
		MT_ChSetMainOPMode.....	266
		MT_ChSetOPMode.....	270
		MT_ChSetReentryBreakPoint.....	258
		MT_CplData.....	185

MT_CplDataConv.....	188
MT_DbData.....	167
MT_DbLoad.....	178
MT_DbRecList.....	161
MT_DbSave.....	174
MT_DiagCode.....	198
MT_DiagText.....	203
MT_DistanceAccumulator.....	346
MT_DRIVE_ERROR_CLASS_1.....	355
MT_DRIVE_ERROR_CLASS_2.....	356
MT_DRIVE_POWER_STATE.....	356
MT_DRIVE_SAFE_OP_MODE.....	357
MT_DrvGetSafeState.....	281
MT_FaDataRead.....	219
MT_FaDataWrite.....	222
MT_GetRandomNumber.....	351
MT_Handw.....	145
MT_LogData.....	211
MT_MCoDS.....	206
MT_MT_ChNcProgProgress.....	252
MT_NcBlk.....	135
MT_NcBlkExt.....	137
MT_P_DbData.....	157
MT_P_DbRecList.....	152
MT_PlcOpen.library	
Datentypen.....	360
MT_PLCOpen.library.....	359
MT_ProgSel.....	131
MT_RingTop.....	209
MT_S3AutoAdapt.....	208
MT_ScsData.....	147
MT_ScsPos.....	129
MT_SD_RD.....	192
MT_SD_Rd_Seg.....	214
MT_SD_SYS_TOOL.....	353
MT_SD_WR.....	195
MT_SD_Wr_Seg.....	216
MT_SpGetJogSpeed.....	321
MT_SpGetPowerState.....	328
MT_SpGetPowerStateExt.....	330
MT_SPINDLE_JOG_SPEED.....	357
MT_SpJogSpeed.....	325
MT_SpPower.....	336
MT_SpSetJogSpeed.....	322
MT_SpSetOverride.....	320
MT_SpSetPowerOff.....	334
MT_SpSetPowerOn.....	332
MT_TCorr.....	181
MT_TempComp.....	141
MT_TimeAccumulator.....	349
MT_TmChGetActiveEdge.....	345
MT_TmToolLife.....	339
MZA-Datei.....	226
MZA-Meldungen aktivieren.....	227
MZA-Text.....	226

N

NC-Kanal.....	245
Übersicht.....	245
NC-Satzvorgabe (MT_NcBlk)	
Funktion.....	135
Projektierungsbeispiel.....	136
Nicht bitcodierte kanalabhängige Hilfsfunktio- nen.....	124
Nicht bitcodierte kanalunabhängige Hilfs- funktionen.....	123
Nicht-bestimmungsgemäßer Gebrauch.....	10
Folgen, Haftungsausschluss.....	9
NmbOfValues.....	142
NofElem.....	161

P

Parametrierung der Maschinenzustandsan- zeige.....	225
PROFIBUS-DP	
I/O-Konfiguration.....	127
PROFIBUS-DP-Schnittstelle.....	127
Programmanwahl/-abwahl (MT_ProgSel)	
Funktion.....	131
Projektierungsbeispiel.....	134
Statusmeldungen.....	134
Programmbausteine.....	129
Achsenistwerte lesen (MT_ScsPos).....	129
Automatischer Sercos III I/O-Tausch (MT_S3AutoAdapt).....	208
Datenbanktabelle laden (MT_DbLoad).....	178
Datenbanktabelle sichern (MT_DbSave).....	174
Datensätze bearbeiten (MT_DbData).....	167
Datensätze bearbeiten (MT_P_DbData).....	157
Datensatzlisten bearbeiten (MT_DbRec- List).....	161
Diagnosedaten lesen (MT_DiagCode).....	198
Diagnosetexte lesen (MT_DiagText).....	203
Erweiterte NC-Satzvorgabe (MT_NcBlkExt).....	137
Fast Action Ausgangsdaten schrei- ben(MT_FaDataWrite).....	222
Fast Action Eingangsdaten lesen (MT_FaDataRead).....	219
Genereller Aufbau von Programmbaustei- nen.....	129
Handraddaten (MT_Handw).....	145
Kanal-Neustart (MT_ChanRestart).....	139
Lesen der Motion Control Data-Services (MT_MCoDS).....	206
Lesen/Schreiben von permanenten CPL- Variablen (MT_CplData).....	185
Lesen/Schreiben von permanenten CPL- Variablen mit Typkonvertierung (MT_CplDataConv).....	188
Log-Eintrag im Aktionsrekorder (MT_Log- Data).....	211

Index

Maschinenparameter lesen (MT_CfgDa- ta).....	143	qCh_ASub1 ... 8.....	47
NC-Satzvorgabe (MT_NcBlk).....	135	qCh_BlkSlash.....	48
Programmanwahl/-abwahl (MT_ProgSel)....	131	qCh_CancDist.....	45
Sercos Parameter (MT_ScsData).....	147	qCh_CancSyntaxError.....	45
Sercos Ringtopologie (MT_RingTop).....	209	qCh_CoordCoupleOff.....	51
Systemdaten lesen (MT_SD_RD).....	192	qCh_CtrlReset.....	44
Systemdaten schreiben (MT_SD_WR).....	195	qCh_Custom1 ... 8.....	50
Systemdaten segmentiert lesen (MT_SD_Rd_Seg).....	214	qCh_FeedHold.....	43
Systemdaten segmentiert schreiben (MT_SD_Wr_Seg).....	216	qCh_Feedrate2.....	53
Temperaturkompensation (MT_Temp- Comp).....	141	qCh_FeedStop.....	44
Werkzeugkorrektur (MT_TCorr).....	181	qCh_HandwDirWcs.....	48
Werkzeuglisten bearbeiten (MT_P_DbRecList).....	152	qCh_HandwPosMode.....	48
Programmwechsel.....	131	qCh_HandwSelWcs_00.....	47
ProgramName.....	133	qCh_JogPlusWcs.....	47
Q		qCh_MachineFunction.....	44
qAx_Custom1 ... 8.....	79	qCh_NCStart.....	43
qAx_Discharge.....	81	qCh_NextBlk.....	46
qAx_DrvLock.....	82	qCh_OnlCorrWcs.....	50
qAx_DrvOn.....	81	qCh_OnlCorrWcsDir.....	50
qAx_FrzlpoPos.....	81	qCh_OpModePlc.....	42
qAx_FxStopRel.....	75	qCh_OpModeSel_00 ... 03.....	41
qAx_HandwDir.....	77	qCh_OptJump.....	48
qAx_HandwPosMode.....	77	qCh_OptStop.....	48
qAx_HandwSel_00 ... 01.....	77	qCh_Override_00 ... 15.....	50
qAx_HomeSwitch.....	78	qCh_Override100.....	49
qAx_JogDia.....	73	qCh_Rapid100.....	49
qAx_JogInch.....	73	qCh_RapOverride_00 ... 15.....	52
qAx_JogMinus.....	71	qCh_RedRap.....	49
qAx_JogPlus.....	71	qCh_ReSelOff.....	44
qAx_LagErrOff.....	79	qCh_Restart.....	43
qAx_ManFeed_00 ... 03.....	78	qCh_RetCont.....	46
qAx_MasterPos.....	80	qCh_Retrace.....	50
qAx_NextNotch.....	74	qCh_RetraceMovFwd.....	51
qAx_OpModeSel_00 ... 01.....	71	qCh_RetraceMovRev.....	51
qAx_Override_00 ... 15.....	79	qCh_Retract.....	47
qAx_Override100.....	79	qCh_TangTRotRel.....	51
qAx_RedDelayErrClass1.....	80	qCh_TestFeed.....	52
qAx_Reset.....	74	qCh_TestRap.....	52
qAx_SafCheckInputState.....	76	qCh_TransferLock.....	43
qAx_SafDrvLock.....	75	qGen_Crash.....	34
qAx_SafEnablCtrl.....	75	qGen_CrashStoreDataAckn.....	34
qAx_SafOpModeSwitch.....	75	qGen_DelErrInfo.....	32
qAx_SafRedTransTime.....	76	qGen_Reset.....	32
qAx_SafSwitch1.....	75	qGen_SercosIO_Reset.....	32
qAx_SafSwitch2.....	75	qGen_StrokeInhibit.....	33
qAx_SafTechState.....	76	qGen_StrokeRel.....	33
qAx_SwLimOff.....	74	qGen_StrokeReserv.....	33
qAx_TakeActOffs.....	80	qSp_CAxOff.....	99
qAx_TrqErrOff.....	79	qSp_CAxOn.....	98
qAx_TrqLim.....	81	qSp_Custom1 ... 8.....	106
qAx_TrvLim_00 ... 01.....	74	qSp_DrvLock.....	107
		qSp_DrvOn.....	106
		qSp_Gear1Act ... 4Act.....	104
		qSp_GearIdleAct.....	104
		qSp_HomeSwitch.....	105
		qSp_JogMinus.....	99
		qSp_JogPlus.....	99
		qSp_ManSpeed_00 ... 02.....	104

qSp_Orientate.....	101
qSp_Override_00 ... 15.....	106
qSp_Override100.....	106
qSp_RedDelayErrClass1.....	106
qSp_Reset.....	100
qSp_SafCheckInputState.....	103
qSp_SafDrvLock.....	102
qSp_SafEnablCtrl.....	103
qSp_SafOpModeSwitch.....	102
qSp_SafRedTransTime.....	104
qSp_SafSwitch1.....	103
qSp_SafSwitch2.....	103
qSp_SafTechState.....	103
qSp_Stop.....	100
qSp_SValueSD.....	105
qSp_TurnCCW.....	100
qSp_TurnCW.....	100
Quittierung Crash-Zustand von SPS an NC (IPO).....	36
Quittung Crash-Zustand von NC an SPS.....	34

R

Reglerfreigabe und Status einer Achse.....	316
Reglerfreigabe und Status einer Spindel.....	336

S

Schreiben von Systemdaten.....	195
SD qSp_SpeedLimit.....	105
SelMask.....	174, 178
Sercos Parameter (MT_ScsData)	
Funktion.....	147
Projektierungsbeispiel.....	148
Statusmeldungen.....	151
Sercos Ringtopologie (MT_RingTop)	
Allgemeines.....	209
Bausteinparameter.....	209
Projektierungsbeispiel.....	210
Statusmeldungen.....	210
Sercos-I/O-Geräte-Fehler gelöscht.....	35
Sercos-I/O-Geräte-Fehler löschen.....	32
Service-Hotline.....	395
SetPos-Verschiebung.....	63
Setzt die Reglerfreigabe für eine Achse... 312, 314	
Setzt die Reglerfreigabe für eine Spin- del.....	332, 334
Shutdown.....	233
Sichere Absolutlage.....	87
Sicherheitshinweise.....	5
Signalbeschreibung	
Ausgangssignale (SPS --> NC).....	41
Spindelbezogene Ausgangssignale SPS --> NC.....	98
Spindelbezogene Eingangssignale (NC -- > SPS).....	108
Signalgrafik.....	6
Signalwörter.....	6
Sonstige.....	346

Spindel Linkslauf.....	100
Spindel Rechtslauf.....	100
Spindel richten.....	101
Spindel Stopp.....	100
Spindelbezogene Ausgangssignale (SPS --> NC)	
Allgemeines.....	98
Antrieb Ein qSp_DrvOn.....	106
Antriebsperre qSp_SafDrvLock.....	102
Betriebsartenwahl (BA) qSp_SafOpMo- deSwitch.....	102
C-Achse Aus qSp_CAxOff.....	99
C-Achse Ein qSp_CAxOn.....	98
Drehzahl tippen Bit 0 ... Bit 2 qSp_Man- Speed_00 ... 02.....	104
Drehzahlbegrenzung SD qSp_SpeedLi- mit.....	105
Override Bit 0 ... Bit 15 qSp_Overri- de_00 ... 15.....	106
Prüfeingang Si-Status qSp_SafCheckIn- putState.....	103
Quittung GTS 1...4 qSp_Gear1Act ... 4Act.....	104
Quittung Leerlauf qSp_GearIdleAct.....	104
Referenzschalter qSp_HomeSwitch.....	105
S-Wert-Vorgabe über SD qSp_SValu- eSD.....	105
Si-Schalter 1 (S2) qSp_SafSwitch1.....	103
Si-Schalter 2 (S2) qSp_SafSwitch2.....	103
Spindel M3 manuell qSp_TurnCW.....	100
Spindel M4 manuell qSp_TurnCCW	100
Spindel M5 manuell qSp_Stop.....	100
Spindel M19 manuell qSp_Orientate.....	101
Spindel Sperre qSp_DrvLock.....	107
Spindel tippen M3 qSp_JogPlus.....	99
Spindel tippen M4 qSp_JogMinus.....	99
Spindelgrundstellung qSp_Reset.....	100
Spindeloverride 100% qSp_Override100....	106
Status Si-Signale qSp_SafTechState.....	103
Überführzeit abkürzen qSp_SafRedT- ransTime.....	104
Verzögerte Fehlerreaktion abkürzen qSp_RedDelayErrClass1.....	106
Zustimmaste (ZT) qSp_SafEnablCtrl.....	103
Spindelbezogene Eingangssignale (NC --> SPS)	
Allgemeines.....	108
Änderung Zustandsklasse-2 iSp_DrvChangeClass2.....	117
Änderung Zustandsklasse-3 iSp_DrvChangeClass3.....	117
Antrieb betriebsbereit iSp_DrvReady.....	119
Antrieb in Betrieb iSp_DrvAct.....	119
Anwahl Getriebe-Leerlauf iSp_GearIdle- Sel.....	113
Anwahl GTS 1 ... 4 iSp_Gear1Sel ... 4Sel...	112
C-Achse ist aktiv iSp_CAxAct.....	108
C-Achsumschaltung iSp_CAxSwitch.....	109
Drehbefehl iSp_TurnCmd.....	109
Drehrichtung M4 iSp_TurnDirM4.....	109

Index

Drehzahl begrenzt iSp_SpLim.....	110	Struktur enthält Sicherheitsbetriebszustand eines Antriebs	357
Drehzahl erreicht iSp_ProgSpReach.....	110	Struktur Fehlerdetails aus S-0-0011 einer Achse.....	355
Fehler Zustandsklasse-1 iSp_DrvErr-Class1.....	116	Struktur mit allen Achsbetriebsarten.....	354
Freigabe zur Leistungszuschaltung iSp_DrvPower.....	118	Struktur mit allen Kanalbetriebsarten.....	352
GTS Umschaltung iSp_GearChange.....	113	Struktur mit Tippdistanzen einer Achse.....	355
Kopplungsfehler iSp_CoupleErr.....	115	Struktur mit Tippgeschwindigkeiten einer Spindel.....	357
Kundenausgang 1 ... 8 iSp_Custom1 ... 8.....	115	Struktur von Systemdatum SD.SysTool.....	353
Lageregelung aktiv iSp_PosCtrl.....	110	Struktur Warnungdetails aus S-0-0012 einer Achse.....	356
Nr. der Kopplung Bit 0 ... Bit 2 iSp_CoupleIndex_00 ... 02.....	115	Suche nach Datensätzen.....	161
SCS-Signalstatus 0 ... 15 iSp_ScsState00 ... 15.....	114	Suche nach Werkzeugdatensätzen.....	152
Si-Betriebsart Bit 0 ... Bit 3 iSp_SafOp-Mode_00 ... 03.....	111	Support.....	395
Spindel gerichtet iSp_OrientateFinish.....	111	Symbole.....	7
Spindel grundgestellt iSp_Reset.....	110	Systemdaten lesen (MT_SD_RD)	
Spindel in Position iSp_InPos.....	110	Funktion.....	192
Spindel ist Master iSp_Master.....	115	Projektierungsbeispiel.....	193
Spindel richten aktiv iSp_OrientateAct.....	111	Statusmeldungen.....	194
Spindel steht iSp_Stop.....	110	Systemdaten schreiben (MT_SD_WR)	
Spindeloverride 0% iSp_Override0.....	114	Funktion.....	195
Spindeloverride 100% iSp_Override100.....	114	Projektierungsbeispiel.....	196
Status sichere Lage iSp_SafStatePos.....	112	Statusmeldungen.....	197
Statusausgang Regler iSp_SafCtrlOutputState.....	112	Systemdaten segmentiert lesen (MT_SD_Rd_Seg)	
Synchronlauf 1 iSp_Synchr1.....	115	Allgemeines.....	214
Synchronlauf 2 iSp_Synchr2.....	116	Bausteinparameter.....	214
Testbetrieb iSp_DryRun.....	118	Projektierungsbeispiel.....	215
Trudeldrehzahl erreicht iSp_IdleSpeed.....	113	Statusmeldungen.....	215
Verzögerte Fehlerreaktion iSp_DelayErr-Class1.....	116	Systemdaten segmentiert schreiben (MT_SD_Wr_Seg)	
Werkzeugbegrenzungen aktiv.....	114	Allgemeines.....	216
Spindelbezogene Interface-Signale.....	95	Bausteinparameter.....	217
Signalbeschreibung.....	98	Projektierungsbeispiel.....	217
SPS-Ausgangssignale - Übersicht	27, 95	Statusmeldungen.....	218
SPS-Eingangssignale - Übersicht	29, 97	Systemgrundstellung.....	32
Übersicht spindelbezogene Interface-Signale.....	95	Systemgrundstellung geführt.....	35
Spindeln.....	319		
Übersicht.....	319	T	
SPS-Ausgangssignale		TangTool.....	62
Globale Interfacesignale.....	32	Temperaturkompensation (MT_TempComp)	
SPS-Eingangssignale.....	53	Funktion.....	141
Globale Interfacesignale.....	35	Projektierungsbeispiel.....	142
SPS-Kanalbezogene Ausgangssignale.....	41	Statusmeldungen.....	142
Standardschnittstellen an Funktionsbausteinen.....	231	ThreadSet.....	63
Stanz-HS-Logik.....	33	Typeld.....	167
Startsatz.....	133		
statische Signale.....	71	U	
Struktur enthält alle Kanalzustände.....	353	Über diese Dokumentation.....	1
Struktur enthält alle Sercos-Attribute.....	360	Übersicht achsbezogene Interface-Signale	
Struktur enthält alle Sercos-Kommando-zustände.....	360	Achsbezogene Ausgangssignale (SPS --> NC).....	71
		Achsbezogene Eingangssignale (SPS --> NC).....	83
		Allgemeines.....	67
		Signalbeschreibung.....	71

Übersicht achsbezogene Ausgangssig- nale (SPS --> NC).....	67
Übersicht achsbezogene Eingangssigna- le (NC --> SPS).....	69
Übersicht globale Interface-Signale	
Allgemeines.....	31
SPS-Ausgangssignale - Übersicht.....	31
SPS-Eingangssignale - Übersicht.....	31
Übersicht kanalbezogene Interface-Signale	
Allgemeines.....	37
Übersicht spindelbezogene Interface-Signale	
Allgemeines.....	95
Übersicht über Zielgruppen und Produktpha- sen.....	1
Unterbrechungspunkt speichern.....	258
Ursprungs kanal.....	88
UTF8-Zeichenkodierung.....	227
V	
Verliehene Achse.....	88
Vorgeben eines Parameterwertes.....	147
Vorgeben von Korrekturwerten.....	181
Vorschub für asynchrone Achse ändern.....	308
Vorschub für eine Spindel ändern.....	320
W	
Warnhinweise.....	5
Werkzeug Standzeit zählung	339
Werkzeugkorrektur (MT_TCorr)	
Funktion.....	181
Projektierungsbeispiel.....	183
Statusmeldungen.....	185
Werkzeuglisten bearbeiten (MT_P_DbRecList)	
Funktion.....	152
Projektierungsbeispiel.....	155
Statusmeldungen.....	156
Werkzeugmanagement.....	339
Übersicht.....	339
X	
XPath.....	192, 195
Z	
Zielgruppen.....	1
Zu dieser Dokumentation	
Darstellung von Informationen.....	5
Zweck.....	1

Notizen

Notizen

Bosch Rexroth AG

Postfach 13 57

97803 Lohr a.Main, Deutschland

Bgm.-Dr.-Nebel-Str. 2

97816 Lohr a.Main, Deutschland

Tel. +49 932 18 0

Fax +49 9352 18 8400

www.boschrexroth.com/electrics



R911400171