SIEMENS

SIMATIC

S7
S7-1200 Automatisierungssystem

Systemhandbuch

Vorwort

Produktübersicht 1

Neue Funktionen 2

STEP 7 Programmiersoftware 3

Einbau 4

SPS-Grundlagen 5

Gerätekonfiguration 6

Programmiergrundlagen 7

Anweisungen 8

Erweiterte Anweisungen 9

Technologieanweisungen 10

Kommunikation 11

Webserver 12

Kommunikationsprozessor und Modbus-TCP 13

TeleService-Kommunikation (SMTP-E-Mail) 14

Online- und Diagnose-Tools 15

Technische Daten A

Berechnen der Leistungsbilanz B

Bestellinformationen C

Geräteaustausch und Ersatzteilkompatibilität D

01/2015

A5E02486681-AH

Rechtliche Hinweise

Warnhinweiskonzept

Dieses Handbuch enthält Hinweise, die Sie zu Ihrer persönlichen Sicherheit sowie zur Vermeidung von Sachschäden beachten müssen. Die Hinweise zu Ihrer persönlichen Sicherheit sind durch ein Warndreieck hervorgehoben, Hinweise zu alleinigen Sachschäden stehen ohne Warndreieck. Je nach Gefährdungsstufe werden die Warnhinweise in abnehmender Reihenfolge wie folgt dargestellt.

⚠ GEFAHR
bedeutet, dass Tod oder schwere Körperverletzung eintreten **wird**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

⚠ WARNUNG
bedeutet, dass Tod oder schwere Körperverletzung eintreten **kann**, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

⚠ VORSICHT
bedeutet, dass eine leichte Körperverletzung eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

ACHTUNG
bedeutet, dass Sachschaden eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Beim Auftreten mehrerer Gefährdungsstufen wird immer der Warnhinweis zur jeweils höchsten Stufe verwendet. Wenn in einem Warnhinweis mit dem Warndreieck vor Personenschäden gewarnt wird, dann kann im selben Warnhinweis zusätzlich eine Warnung vor Sachschäden angefügt sein.

Qualifiziertes Personal

Das zu dieser Dokumentation zugehörige Produkt/System darf nur von für die jeweilige Aufgabenstellung **qualifiziertem Personal** gehandhabt werden unter Beachtung der für die jeweilige Aufgabenstellung zugehörigen Dokumentation, insbesondere der darin enthaltenen Sicherheits- und Warnhinweise. Qualifiziertes Personal ist auf Grund seiner Ausbildung und Erfahrung befähigt, im Umgang mit diesen Produkten/Systemen Risiken zu erkennen und mögliche Gefährdungen zu vermeiden.

Bestimmungsgemäßer Gebrauch von Siemens-Produkten

Beachten Sie Folgendes:

⚠ WARNUNG
Siemens-Produkte dürfen nur für die im Katalog und in der zugehörigen technischen Dokumentation vorgesehenen Einsatzfälle verwendet werden. Falls Fremdprodukte und -komponenten zum Einsatz kommen, müssen diese von Siemens empfohlen bzw. zugelassen sein. Der einwandfreie und sichere Betrieb der Produkte setzt sachgemäßen Transport, sachgemäße Lagerung, Aufstellung, Montage, Installation, Inbetriebnahme, Bedienung und Instandhaltung voraus. Die zulässigen Umgebungsbedingungen müssen eingehalten werden. Hinweise in den zugehörigen Dokumentationen müssen beachtet werden.

Marken

Alle mit dem Schutzrechtsvermerk ® gekennzeichneten Bezeichnungen sind eingetragene Marken der Siemens AG. Die übrigen Bezeichnungen in dieser Schrift können Marken sein, deren Benutzung durch Dritte für deren Zwecke die Rechte der Inhaber verletzen kann.

Haftungsausschluss

Wir haben den Inhalt der Druckschrift auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Dennoch können Abweichungen nicht ausgeschlossen werden, so dass wir für die vollständige Übereinstimmung keine Gewähr übernehmen. Die Angaben in dieser Druckschrift werden regelmäßig überprüft, notwendige Korrekturen sind in den nachfolgenden Auflagen enthalten.

Siemens AG
Division Digital Factory
Postfach 48 48
90026 NÜRNBERG
DEUTSCHLAND

A5E02486681-AH
℗ 12/2014 Änderungen vorbehalten

Copyright © Siemens AG 2015.
Alle Rechte vorbehalten

Vorwort

Zweck des Handbuchs

Die Familie S7-1200 umfasst verschiedene speicherprogrammierbare Steuerungen (SPS), mit denen Sie eine breite Palette von Automatisierungsaufgaben lösen können. Durch das kompakte Design, die Möglichkeit der Erweiterung, den günstigen Preis und einen leistungsstarken Befehlssatz eignet sich die S71200 hervorragend für eine Vielzahl von Steuerungsanwendungen. Die Ausführungen der S7-1200 und die vielfältigen STEP 7-Programmiermöglichkeiten (Seite 37) unter Windows bieten Ihnen eine extrem hohe Flexibilität beim Umsetzen Ihrer Automatisierungslösungen.

Dieses Handbuch bietet Informationen zum Installieren und Programmieren von S7-1200 Steuerungen und wendet sich an Ingenieure, Programmierer und Wartungspersonal mit allgemeinen Kenntnissen über Automatisierungssysteme.

Erforderliche Grundkenntnisse

Damit Sie mit diesem Handbuch arbeiten können, benötigen Sie allgemeine Kenntnisse im Bereich der Automatisierung und der speicherprogrammierbaren Steuerungen.

Umfang des Handbuchs

Dieses Handbuch beschreibt die folgenden Produkte:

- STEP 7 V13 SP1 Basic und Professional (Seite 37)
- S7-1200 CPU Firmware-Version V4.1

Eine vollständige Liste der S71200 Produkte finden Sie in den technischen Daten (Seite 1151).

Zertifizierung, CE-Kennzeichen, C-Tick und andere Normen

Ausführliche Informationen finden Sie in den technischen Daten (Seite 1151).

Service & Support

Zusätzlich zu unserem Dokumentations-Angebot bietet Siemens Ihnen im Internet technisches Know-how auf der Kundensupport-Website (http://www.siemens.com/tiaportal) an.

Falls Sie technische Fragen haben, eine Schulung benötigen oder S7-Produkte bestellen wollen, wenden Sie sich bitte an Ihre Siemens-Vertretung. Das technisch geschulte Vertriebspersonal verfügt über sehr spezifische Kenntnisse zu Einsatzmöglichkeiten und Prozessen sowie zu den verschiedenen Siemens-Produkten und kann Ihnen deshalb am schnellsten und besten weiterhelfen, wenn Probleme auftreten.

Dokumentation und Information

S7-1200 und STEP 7 bieten eine Vielzahl von Dokumentationen und anderen Quellen mit technischen Informationen.

- Das Systemhandbuch S7-1200 Automatisierungssystem bietet spezielle Informationen zu Funktionsweise, Programmierung und technischen Daten der gesamten S7-1200-Produktfamilie. Neben dem Systemhandbuch bietet das S7-1200 Easy Book einen eher allgemeinen Überblick über die Fähigkeiten der S7-1200 Familie.

  Sowohl das Systemhandbuch als auch das Easy Book stehen elektronisch (im PDF-Format) zur Verfügung. Die elektronischen Handbücher können von der Kundensupport-Website heruntergeladen werden. Sie finden sie außerdem auf den Datenträgern, die mit jeder S7-1200 CPU ausgeliefert werden.

- Das Online-Informationssystem von STEP 7 bietet unmittelbaren Zugriff auf Konzeptinformationen und spezielle Hinweise zur Funktionsweise und Funktionalität des Programmierpakets sowie zur grundlegenden Funktionsweise der SIMATIC CPUs.

- Mit My Documentation Manager greifen Sie auf die elektronische Version (PDF) des SIMATIC-Dokumentationssatzes zu, der das Systemhandbuch, das Easy Book und Informationssystem von STEP 7 umfasst. Mit My Documentation Manager können Sie Themen aus verschiedenen Dokumenten per Drag & Drop selbst anordnen und so eigene benutzerspezifische Handbücher anlegen.

  Im Kundensupport-Portal (http://support.automation.siemens.com) finden Sie einen Link auf My Documentation Manager unter mySupport.

- Die Kundensupport-Website bietet außerdem Podcasts, FAQs und weitere nützliche Dokumente für S7-1200 und STEP 7. Die Podcasts zeigen kurze Lernvideos zu speziellen Funktionen oder Szenarien, mit denen die Interaktionen, die praktische Bedienung und Effizienz von STEP 7 demonstriert werden. Die Podcast-Sammlung finden Sie auf folgenden Websites:

  – Webseite zu STEP 7 Basic (http://www.automation.siemens.com/mcms/simatic-controller-software/de/step7/step7-basic/Seiten/Default.aspx)

  – Webseite zu STEP 7 Professional (http://www.automation.siemens.com/mcms/simatic-controller-software/de/step7/step7-professional/Seiten/Default.aspx)

- Im technischen Forum unter Service & Support (https://www.automation.siemens.com/WW/forum/guests/Conferences.aspx?Language=de&onlyInternet=False) können Sie außerdem Produktdiskussionen verfolgen oder sich daran beteiligen. Folgende Foren bieten Ihnen die Möglichkeit, mit verschiedenen Produktexperten in Kontakt zu treten.

  – Forum für S7-1200 (https://www.automation.siemens.com/WW/forum/guests/Conference.aspx?SortField=LastPostDate&SortOrder=Descending&ForumID=257&Language=de&onlyInternet=False)

  – Forum für STEP 7 Basic (https://www.automation.siemens.com/WW/forum/guests/Conference.aspx?SortField=LastPostDate&SortOrder=Descending&ForumID=262&Language=de&onlyInternet=False)

Security-Hinweise

Siemens bietet Produkte und Lösungen mit Industrial Security-Funktionen an, die den sicheren Betrieb von Anlagen, Lösungen, Maschinen, Geräten und/oder Netzwerken unterstützen. Sie sind wichtige Komponenten in einem ganzheitlichen Industrial Security-Konzept. Die Produkte und Lösungen von Siemens werden unter diesem Gesichtspunkt ständig weiterentwickelt. Siemens empfiehlt, sich unbedingt regelmäßig über Produkt-Updates zu informieren.

Für den sicheren Betrieb von Produkten und Lösungen von Siemens ist es erforderlich, geeignete Schutzmaßnahmen (z. B. Zellenschutzkonzept) zu ergreifen und jede Komponente in ein ganzheitliches Industrial Security-Konzept zu integrieren, das dem aktuellen Stand der Technik entspricht. Dabei sind auch eingesetzte Produkte von anderen Herstellern zu berücksichtigen. Weitergehende Informationen über Industrial Security finden Sie unter (http://www.siemens.com/industrialsecurity).

Um stets über Produkt-Updates informiert zu sein, melden Sie sich für unseren produktspezifischen Newsletter an. Weitere Informationen hierzu finden Sie unter (http://support.automation.siemens.com).

Inhaltsverzeichnis

Vorwort ... **3**

1 Produktübersicht ... **25**

1.1 Einführung in die S7-1200 SPS ... 25

1.2 Erweiterung der CPU-Funktionen ... 29

1.3 Grundlegende HMI-Panels ... 31

2 Neue Funktionen ... **33**

3 STEP 7 Programmiersoftware ... **37**

3.1 Systemvoraussetzungen ... 38

3.2 Einfaches Arbeiten mit unterschiedlichen Ansichten ... 39

3.3 Bedienerfreundliche Werkzeuge ... 40
3.3.1 Anweisungen in Ihr Anwenderprogramm einfügen ... 40
3.3.2 Zugreifen auf Anweisungen in der Funktionsleiste "Favoriten" ... 40
3.3.3 Erstellen einer komplexen Gleichung mit einer einfachen Anweisung ... 41
3.3.4 Ergänzen von Eingängen oder Ausgängen in einer KOP- oder FUP-Anweisung ... 43
3.3.5 Erweiterbare Anweisungen ... 43
3.3.6 Auswählen einer Version für eine Anweisung ... 44
3.3.7 Ändern des Erscheinungsbilds und der Konfiguration von STEP 7 ... 44
3.3.8 Drag & Drop zwischen Editoren ... 45
3.3.9 Wechseln des Betriebszustands der CPU ... 46
3.3.10 Aufruftyp eines DB ändern ... 47
3.3.11 Geräte vorübergehend vom Netzwerk trennen ... 48
3.3.12 Virtuelles Abziehen von Geräten aus der Konfiguration ... 49

3.4 Rückwärtskompatibilität ... 50

4 Einbau ... **51**

4.1 Richtlinien für den Einbau von S71200 Geräten ... 51

4.2 Leistungsbilanz ... 54

4.3 Ein- und Ausbauanweisungen ... 56
4.3.1 Einbaumaße der S7-1200 Geräte ... 56
4.3.2 Einbau und Ausbau der CPU ... 60
4.3.3 Ein- und Ausbau eines SBs, CBs oder BBs ... 62
4.3.4 Einbau und Ausbau eines SMs ... 64
4.3.5 Einbau und Ausbau eines CMs oder CPs ... 66
4.3.6 Ausbau und Wiedereinbau des S7-1200 Klemmenblocks ... 67
4.3.7 Einbau und Ausbau des Erweiterungskabels ... 68
4.3.8 TS (TeleService)-Adapter ... 70
4.3.8.1 Anschließen des TeleService-Adapters ... 70
4.3.8.2 Einlegen der SIM-Karte ... 72
4.3.8.3 Einbau des TS-Adapters auf einer DIN-Schiene ... 73
4.3.8.4 Schalttafeleinbau des TS-Adapters ... 74

4.4 Verdrahtungsrichtlinien ... 75

5 SPS-Grundlagen ... 81

5.1 Ausführung des Anwenderprogramms ... 81
5.1.1 Betriebszustände der CPU ... 85
5.1.2 Bearbeitung des Zyklus im Betriebszustand RUN ... 89
5.1.3 Organisationsbausteine (OBs) ... 90
5.1.3.1 Programmzyklus-OB ... 90
5.1.3.2 Anlauf-OB ... 91
5.1.3.3 Verzögerungsalarm-OB ... 91
5.1.3.4 Weckalarm-OB ... 92
5.1.3.5 Prozessalarm-OB ... 93
5.1.3.6 Zeitfehler-OB ... 93
5.1.3.7 Diagnosefehler-OB ... 95
5.1.3.8 OB "Ziehen oder Stecken von Modulen" ... 97
5.1.3.9 Baugruppenträger- oder Stationsfehler-OB ("Rack or station failure OB") ... 97
5.1.3.10 Uhrzeit-OB ... 98
5.1.3.11 Zustands-OB ... 99
5.1.3.12 Aktualisierungs-OB ... 99
5.1.3.13 Profil-OB ... 100
5.1.3.14 Prioritäten und Warteschlange für die Ausführung von Ereignissen ... 100
5.1.4 Überwachen und Konfigurieren der Zykluszeit ... 104
5.1.5 CPU-Speicher ... 106
5.1.5.1 System- und Taktmerker ... 108
5.1.6 Diagnosepuffer ... 110
5.1.7 Echtzeituhr ... 110
5.1.8 Konfigurieren der Ausgänge für den Wechsel von RUN in STOP ... 111

5.2 Datenspeicher, Speicherbereiche, E/A und Adressierung ... 112
5.2.1 Zugriff auf Daten der S7-1200 ... 112

5.3 Verarbeitung von Analogwerten ... 118

5.4 Datentypen ... 120
5.4.1 Datentypen Bool, Byte, Word und DWord ... 121
5.4.2 Ganzzahlige Datentypen ... 122
5.4.3 Gleitpunktzahl/Realzahl-Datentypen ... 122
5.4.4 Uhrzeit- und Datums-Datentypen ... 123
5.4.5 Zeichen- und Zeichenfolge-Datentypen ... 125
5.4.6 Datentyp ARRAY ... 127
5.4.7 Datentyp Struktur ... 128
5.4.8 PLC-Datentyp ... 128
5.4.9 Pointer-Datentypen ... 128
5.4.9.1 Pointer-Datentyp "Pointer" ... 129
5.4.9.2 Pointer-Datentyp "Any" ... 130
5.4.9.3 Pointer-Datentyp "Variant" ... 132
5.4.10 Zugriff auf eine "Scheibe" eines Variablendatentyps ... 132
5.4.11 Zugriff auf eine Variable mit einer AT-Überlagerung ... 134

5.5 Memory Card verwenden ... 136
5.5.1 Memory Card in die CPU stecken ... 137
5.5.2 Anlaufparameter der CPU vor dem Kopieren des Projekts auf die Memory Card konfigurieren ... 139
5.5.3 Einsatz der Memory Card als Übertragungskarte ... 139

5.5.4 Einsatz der Memory Card als Programmkarte 142
5.5.5 Firmware-Update 146
5.6 Vorgehensweise bei verlorenem Passwort....149
6 Gerätekonfiguration 151
6.1 CPU einfügen....152
6.2 Konfiguration aus einer angeschlossenen CPU laden 154
6.3 Module zur Konfiguration hinzufügen 156
6.4 Konfigurationssteuerung 157
6.4.1 Vorteile und Nutzung der Konfigurationssteuerung....157
6.4.2 Den zentralen Aufbau und optionale Module konfigurieren....157
6.4.3 Beispiel für die Konfigurationssteuerung 164
6.5 Ändern eines Geräts 168
6.6 Konfigurieren des CPU-Betriebs....168
6.6.1 Übersicht....168
6.6.2 Filterzeiten für Digitaleingänge einrichten 170
6.6.3 Impulsabgriff 171
6.7 Modulparameter konfigurieren....173
6.8 CPU für die Kommunikation konfigurieren....175
7 Programmiergrundlagen.... 177
7.1 Richtlinien für das Entwerfen einer Automatisierungslösung mit einem PLC-Gerät 177
7.2 Strukturieren Ihres Anwenderprogramms....179
7.3 Strukturierung des Programms mit Hilfe von Bausteinen....181
7.3.1 Organisationsbaustein (OB) 182
7.3.2 Funktion (FC) 184
7.3.3 Funktionsbaustein (FB)....184
7.3.4 Datenbaustein (DB) 186
7.3.5 Anlegen wiederverwendbarer Codebausteine....187
7.3.6 Übergabe von Parametern an Bausteine 188
7.4 Datenkonsistenz 191
7.5 Programmiersprache 193
7.5.1 Kontaktplan (KOP) 193
7.5.2 Funktionsplan (FUP) 194
7.5.3 SCL 195
7.5.3.1 SCL-Programmiereditor....195
7.5.3.2 SCL-Ausdrücke und -Operationen....196
7.5.3.3 Indexierte Adressierung mit den Anweisungen PEEK und POKE 200
7.5.4 EN und ENO in KOP, FUP und SCL 202
7.6 Schutz....205
7.6.1 Zugriffsschutz für die CPU....205
7.6.2 Knowhow-Schutz 208
7.6.3 Kopierschutz 209
7.7 Laden der Programmelemente 211
7.8 Laden von der Online-CPU....212

7.8.1 Vergleich der Online-CPU mit der Offline-CPU ... 212

7.9 Debugging und Testen des Programms ... 213
7.9.1 Daten in der CPU beobachten und steuern ... 213
7.9.2 Beobachtungstabellen und Forcetabellen ... 213
7.9.3 Querverweis zum Anzeigen der Verwendung ... 214
7.9.4 Aufrufstruktur zur Prüfung der Aufrufhierarchie ... 215

8 Anweisungen ... 217

8.1 Bitverknüpfungen ... 217
8.1.1 Bitverknüpfungsanweisungen ... 217
8.1.2 Setz- und Rücksetzoperationen ... 220
8.1.3 Operationen Steigende Flanke und Fallende Flanke ... 223

8.2 Funktionsweise der Zeiten ... 227

8.3 Funktionsweise der Zähler ... 235

8.4 Funktionsweise von Vergleichern ... 242
8.4.1 Vergleichsoperationen ... 242
8.4.2 Anweisungen IN_Range (Wert innerhalb Bereich) und OUT_Range (Wert außerhalb Bereich) ... 243
8.4.3 Anweisungen OK (Gültigkeit prüfen) und NOT_OK (Ungültigkeit prüfen) ... 244
8.4.4 Variant- und Array-Vergleichsoperationen ... 245
8.4.4.1 Gleich- und Ungleich-Vergleichsoperationen ... 245
8.4.4.2 Null-Vergleichsoperationen ... 246
8.4.4.3 IS_ARRAY (Auf ARRAY prüfen) ... 246

8.5 Arithmetische Funktionen ... 247
8.5.1 Anweisung CALCULATE (Berechnen) ... 247
8.5.2 Operationen Addieren, Subtrahieren, Multiplizieren und Dividieren ... 248
8.5.3 Anweisung MOD (Divisionsrest einer Division) ... 249
8.5.4 Anweisung NEG (Zweierkomplement erzeugen) ... 250
8.5.5 Anweisungen INC (Inkrementieren) und DEC (Dekrementieren) ... 251
8.5.6 Anweisung ABS (Absolutwert bilden) ... 251
8.5.7 Anweisungen MIN (Minimum ermitteln) und MAX (Maximum ermitteln) ... 252
8.5.8 Anweisung LIMIT (Limitieren) ... 253
8.5.9 Exponential-, Logarithmus- und Trigonometrie-Anweisungen ... 254

8.6 Anweisungen zum Übertragen von Daten ... 257
8.6.1 Anweisungen MOVE (Wert kopieren), MOVE_BLK (Bereich kopieren), UMOVE_BLK (Bereich ununterbrechbar kopieren) und MOVE_BLK_VARIANT (Bereich kopieren) ... 257
8.6.2 Deserialize ... 260
8.6.3 Serialize ... 263
8.6.4 FILL_BLK (Bereich befüllen) und UFILL_BLK (Bereich ununterbrechbar befüllen) ... 266
8.6.5 Anweisung SWAP (Anordnung ändern) ... 268
8.6.6 Anweisungen Speicher lesen / in Speicher schreiben ... 269
8.6.6.1 Anweisungen PEEK und POKE (nur SCL) ... 269
8.6.6.2 Anweisungen Big- und Little-Endian-Format lesen und schreiben (SCL) ... 271
8.6.7 Variant-Anweisungen ... 272
8.6.7.1 VariantGet (VARIANT Variablenwert lesen) ... 272
8.6.7.2 Anweisung VariantPut (VARIANT Variablenwert schreiben) ... 273
8.6.7.3 Anweisung CountOfElements (Anzahl ARRAY-Elemente abfragen) ... 274
8.6.8 Anweisungen in älteren Systemen ... 275
8.6.8.1 Anweisungen FieldRead (Feld lesen) und FieldWrite (Feld schreiben) ... 275

8.7 Umwandlungsoperationen 278
8.7.1 Anweisung CONV (Wert konvertieren) 278
8.7.2 Umwandlungsanweisungen in SCL 279
8.7.3 Anweisungen ROUND (Zahl runden) und TRUNC (Ganzzahl erzeugen) 282
8.7.4 Anweisungen CEIL und FLOOR (Aus Gleitpunktzahl nächsthöhere und nächstniedere Ganzzahl erzeugen) 283
8.7.5 Anweisungen SCALE_X (Skalieren) und NORM_X (Normalisieren) 284
8.7.6 Variant-Umwandlungsanweisungen 287
8.7.6.1 Anweisung VARIANT_TO_DB_ANY (VARIANT und DB_ANY umwandeln) 287
8.7.6.2 Anweisung DB_ANY_TO_VARIANT (DB_ANY in VARIANT umwandeln) 288

8.8 Programmsteuerungsoperationen 290
8.8.1 Anweisungen JMP (Springen bei VKE = 1), JMPN (Springen bei VKE = 0) und Label (Sprungmarke) 290
8.8.2 Anweisung JMP_LIST (Sprungliste definieren) 291
8.8.3 Anweisung SWITCH (Sprungverteiler) 292
8.8.4 Anweisung RET (Zurück springen) 294
8.8.5 Anweisung ENDIS_PW (Passwort-Legitimierung einschränken und freigeben) 295
8.8.6 Anweisung RE_TRIGR (Zyklusüberwachungszeit neu starten) 297
8.8.7 Anweisung STP (Programm beenden) 298
8.8.8 Anweisungen GET_ERROR und GET_ERROR_ID (Fehler lokal abrufen und Fehler-ID lokal abrufen) 299
8.8.9 Anweisung RUNTIME (Programmlaufzeit messen) 303
8.8.10 Programmsteuerungsanweisungen in SCL 305
8.8.10.1 Übersicht über die Programmsteuerungsanweisungen in SCL 305
8.8.10.2 IF-THEN-Anweisung 306
8.8.10.3 CASE-Anweisung 307
8.8.10.4 FOR-Anweisung 308
8.8.10.5 WHILE-DO-Anweisung 310
8.8.10.6 REPEAT-UNTIL-Anweisung 311
8.8.10.7 CONTINUE-Anweisung 312
8.8.10.8 EXIT-Anweisung 313
8.8.10.9 GOTO-Anweisung 314
8.8.10.10 RETURN-Anweisung 314

8.9 Wortverknüpfung 315
8.9.1 Verknüpfungsoperationen AND (UND), OR (ODER) und XOR (EXKLUSIV ODER) 315
8.9.2 Anweisung INV (Einerkomplement erzeugen) 316
8.9.3 Anweisungen DECO (Decodieren) and ENCO (Encodieren) 316
8.9.4 Anweisungen SEL (Selektieren), MUX (Multiplexen) und DEMUX (Demultiplexen) 318

8.10 Schieben und Rotieren 321
8.10.1 Anweisungen SHL (Rechts schieben) und SHL (Links schieben) 321
8.10.2 Anweisungen ROR (Rechts rotieren) und ROL (Links rotieren) 322

9 Erweiterte Anweisungen 323

9.1 Datums-, Uhrzeit- und Uhrfunktionen 323
9.1.1 Datums- und Uhrzeitanweisungen 323
9.1.2 Uhrzeitfunktionen 326
9.1.3 Datenstruktur TimeTransformationRule 329
9.1.4 Anweisung SET_TIMEZONE (Zeitzone setzen) 330
9.1.5 Anweisung RTM (Betriebsstundenzähler) 331

9.2 Zeichenketten- und Zeichenanweisungen 333

9.2.1 Datentyp String ... 333
9.2.2 Anweisung S_MOVE (Zeichenkette verschieben) ... 333
9.2.3 Zeichenkettenkonvertierungsoperationen ... 334
9.2.3.1 Anweisungen S_CONV, STRG_VAL und VAL_STRG (In/von Zeichenkette und Zahlenwert umwandeln) ... 334
9.2.3.2 Anweisungen Strg_TO_Chars und Chars_TO_Strg (In/aus Zeichenkette und Array aus CHAR umwandeln) ... 343
9.2.3.3 ATH und HTA (In/aus ASCII-Zeichenkette und Hexadezimalzahl umwandeln) ... 345
9.2.4 Zeichenkettenanweisungen ... 347
9.2.4.1 Anweisung MAX_LEN (Maximale Länge einer Zeichenkette) ... 348
9.2.4.2 Anweisung LEN (Länge einer Zeichenkette ermitteln) ... 348
9.2.4.3 Anweisung CONCAT (Zeichenketten verketten) ... 349
9.2.4.4 Anweisungen LEFT, RIGHT und MID (Teilzeichenketten in einer Zeichenkette lesen) ... 350
9.2.4.5 Anweisung DELETE (Zeichen in einer Zeichenkette löschen) ... 351
9.2.4.6 Anweisung INSERT (Zeichen in einer Zeichenkette einfügen) ... 352
9.2.4.7 Anweisung REPLACE (Zeichen in einer Zeichenkette ersetzen) ... 354
9.2.4.8 Anweisung FIND (Zeichen in einer Zeichenkette finden) ... 355

9.3 Dezentrale E/A (PROFINET, PROFIBUS oder AS-i) ... 356
9.3.1 Anweisungen für die dezentrale E/A ... 356
9.3.2 Anweisungen RDREC und WRREC (Datensatz lesen/schreiben) ... 357
9.3.3 Anweisung RALRM (Alarm empfangen) ... 360
9.3.4 STATUS-Parameter für RDREC, WRREC und RALRM ... 364
9.3.5 Anweisungen DPRD_DAT und DPWR_DAT (Konsistente Daten für DP-Slaves lesen/schreiben) ... 368
9.3.6 Anweisung DPNRM_DG (Diagnosedaten eines DP-Slaves lesen) ... 371

9.4 Alarme ... 374
9.4.1 Anweisungen ATTACH und DETACH (OB und Alarmereignis einander zuweisen/Zuweisung aufheben) ... 374
9.4.2 Weckalarme ... 378
9.4.2.1 Anweisung SET_CINT (Weckalarm parametrieren) ... 378
9.4.2.2 Anweisung QRY_CINT (Weckalarmparameter abfragen) ... 380
9.4.3 Uhrzeitalarme ... 381
9.4.3.1 SET_TINTL (Uhrzeitalarm festlegen) ... 382
9.4.3.2 CAN_TINT (Uhrzeitalarm löschen) ... 383
9.4.3.3 ACT_TINT (Uhrzeitalarm aktivieren) ... 384
9.4.3.4 QRY_TINT (Status des Uhrzeitalarms abfragen) ... 385
9.4.4 Verzögerungsalarme ... 386
9.4.5 Anweisungen DIS_AIRT und EN_AIRT (Ausführung von Alarmen höherer Priorität und asynchronen Fehlerereignissen verzögern/aktivieren) ... 389

9.5 Diagnose (PROFINET oder PROFIBUS) ... 390
9.5.1 Diagnoseanweisungen ... 390
9.5.2 Diagnoseereignisse für die dezentrale Peripherie ... 390
9.5.3 Anweisung LED (LED-Status lesen) ... 392
9.5.4 Anweisung DeviceStates ... 393
9.5.4.1 Beispiele für die Konfiguration von DeviceStates ... 395
9.5.5 Anweisung ModuleStates ... 399
9.5.5.1 Beispiele für die Konfiguration von ModuleStates ... 401
9.5.6 Anweisung GET_DIAG (Diagnoseinformationen lesen) ... 405
9.5.7 Anweisung Get_IM_Data (Identifikations- und Wartungsdaten lesen) ... 411

9.6 Impuls ... 413

9.6.1 Anweisung CTRL_PWM (Impulsdauermodulation) 413
9.6.2 Funktionsweise der Impulsausgänge 414
9.6.3 Konfigurieren eines Impulskanals für PWM 416

9.7 Rezepte und Datenprotokolle 419
9.7.1 Rezepte 419
9.7.1.1 Übersicht über Rezepte 419
9.7.1.2 Beispiel für ein Rezept 420
9.7.1.3 Programmanweisungen zum Übertragen von Rezeptdaten 424
9.7.1.4 Beispielprogramm für Rezepte 428
9.7.2 Datenprotokolle 431
9.7.2.1 Datensatzstruktur der Datenprotokolle 431
9.7.2.2 Programmanweisungen zum Steuern von Datenprotokollen 432
9.7.2.3 Arbeiten mit Datenprotokollen 444
9.7.2.4 Grenzwert für die Größe von Datenprotokolldateien 446
9.7.2.5 Beispielprogramm für Datenprotokolle 449

9.8 Datenbausteinsteuerung 453
9.8.1 Anweisungen READ_DBL und WRIT_DBL (Datenbaustein im Ladespeicher lesen/schreiben) 453

9.9 Adressverarbeitung 457
9.9.1 Anweisung GEO2LOG (Hardwarekennung über Steckplatz ermitteln) 457
9.9.2 Anweisung LOG2GEO (Steckplatz über Hardwarekennung ermitteln) 459
9.9.3 Anweisung IO2MOD (Hardwarekennung über E/A-Adresse ermitteln) 460
9.9.4 Anweisung RD_ADDR (E/A-Adressen über Hardwarekennung ermitteln) 461
9.9.5 Systemdatentyp GEOADDR 463

9.10 Gemeinsame Fehlercodes für die erweiterten Anweisungen 464

10 Technologieanweisungen 465

10.1 Schneller Zähler 465
10.1.1 Anweisung CTRL_HSC (Schnellen Zähler steuern) 465
10.1.2 Anweisung CTRL_HSC_EXT (Schnellen Zähler steuern (erweitert)) 468
10.1.3 Funktionsweise von schnellen Zählern 471
10.1.4 Konfiguration eines schnellen Zählers 479

10.2 PID-Regelung 481
10.2.1 Anweisung PID und Technologieobjekt einfügen 483
10.2.2 Operation PID_Compact 485
10.2.3 ErrorBit-Parameter der Anweisung PID_Compact 489
10.2.4 Warnparameter der Anweisung PID_Compact 491
10.2.5 Anweisung PID_3Step 492
10.2.6 ErrorBit-Parameter der Anweisung PID_3Step 499
10.2.7 Warnparameter der Anweisung PID_3Step 501
10.2.8 Anweisung PID_Temp 502
10.2.8.1 Funktionsweise des PID_Temp-Reglers 506
10.2.8.2 Kaskadieren von Reglern 509
10.2.9 Anweisung PID_Temp ErrorBit-Parameter 513
10.2.10 Warnparameter der Anweisung PID_Temp 515
10.2.11 PID_Compact- und PID_3Step-Regler konfigurieren 516
10.2.12 PID_Temp-Regler konfigurieren 519
10.2.13 PID_Compact- und PID_3Step-Regler in Betrieb nehmen 535
10.2.14 PID_Temp-Regler in Betrieb nehmen 537

10.3 Bewegungssteuerung ... 547
10.3.1 Phasenlage ... 552
10.3.2 Konfigurieren eines Impulsgenerators ... 555
10.3.3 Ungeregelte Bewegungssteuerung ... 556
10.3.3.1 Achse konfigurieren ... 556
10.3.3.2 Inbetriebnahme ... 560
10.3.4 Geregelte Bewegungssteuerung ... 566
10.3.4.1 Achse konfigurieren ... 566
10.3.5 TO-Befehlstabelle PTO konfigurieren ... 573
10.3.6 Funktionsweise der Bewegungssteuerung bei der S7-1200 ... 576
10.3.6.1 Für die Bewegungssteuerung verwendete CPU-Ausgänge ... 576
10.3.6.2 Hardware- und Software-Endschalter für die Bewegungssteuerung ... 578
10.3.6.3 Referenzpunktfahrt ... 582
10.3.6.4 Ruckbegrenzung ... 588
10.3.7 Bewegungssteuerungsanweisungen ... 589
10.3.7.1 Übersicht MC-Anweisungen ... 589
10.3.7.2 Anweisung MC_Power (Achse freigeben/sperren) ... 590
10.3.7.3 Anweisung MC_Reset (Fehler bestätigen) ... 593
10.3.7.4 Anweisung MC_Home (Referenzpunktfahrt der Achse durchführen) ... 594
10.3.7.5 Anweisung MC_Halt (Achse pausieren) ... 597
10.3.7.6 Anweisung MC_MoveAbsolute (Achse absolut positionieren) ... 599
10.3.7.7 Anweisung MC_MoveRelative (Achse relativ positionieren) ... 601
10.3.7.8 Anweisung MC_MoveVelocity (Achse mit vordefinierter Geschwindigkeit bewegen) ... 603
10.3.7.9 Anweisung MC_MoveJog (Achse im Tippbetrieb bewegen) ... 606
10.3.7.10 Anweisung MC_CommandTable (Achssteuerungsbefehle als Bewegungsfolge ausführen) ... 609
10.3.7.11 Anweisung MC_ChangeDynamic (Dynamikeinstellungen der Achse ändern) ... 612
10.3.7.12 Anweisung MC_WriteParam (Parameter des Technologieobjekts schreiben) ... 614
10.3.7.13 Anweisung MC_ReadParam (Parameter eines Technologieobjekts lesen) ... 616
10.3.8 Aktive Befehle überwachen ... 618
10.3.8.1 Bewegungssteuerungsanweisungen mit einem Ausgangsparameter "Done" überwachen ... 618
10.3.8.2 Anweisung MC_Velocity überwachen ... 622
10.3.8.3 Anweisung MC_MoveJog überwachen ... 626

11 Kommunikation ... 631

11.1 Asynchrone Kommunikationsverbindungen in V4.1 ... 633

11.2 PROFINET ... 636
11.2.1 Netzwerkverbindung erstellen ... 637
11.2.2 Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren ... 638
11.2.3 IP-Adressen zuweisen ... 642
11.2.3.1 IP-Adressen zu Programmier- und Netzwerkgeräten zuweisen ... 642
11.2.3.2 Ermitteln der IP-Adresse Ihres Programmiergeräts ... 644
11.2.3.3 Online eine IP-Adresse zu einer CPU zuweisen ... 645
11.2.3.4 IP-Adresse für eine CPU in Ihrem Projekt konfigurieren ... 646
11.2.4 Testen des PROFINET-Netzwerks ... 650
11.2.5 Ermitteln der Ethernet-Adresse (MAC-Adresse) der CPU ... 651
11.2.6 Konfigurieren der NTP-Synchronisierung ... 653
11.2.7 Anlaufzeit, Benennung und Adresszuweisung von PROFINET-Geräten ... 655
11.2.8 Offene Benutzerkommunikation ... 656
11.2.8.1 Protokolle ... 656
11.2.8.2 TCP und ISO on TCP ... 657

11.2.8.3 Kommunikationsdienste und verwendete Portnummern 658
11.2.8.4 Ad-hoc-Modus 659
11.2.8.5 Verbindungs-IDs für Anweisungen für die offene Benutzerkommunikation 659
11.2.8.6 Parameter für die PROFINET-Verbindung 663
11.2.8.7 Anweisungen TSEND_C und TRCV_C 667
11.2.8.8 Anweisungen TSEND_C und TRCV_C in älteren Systemen 678
11.2.8.9 Anweisungen TCON, TDISCON, TSEND und TRCV 687
11.2.8.10 Ältere Anweisungen TCON, TDISCON, TSEND und TRCV 697
11.2.8.11 Anweisung T_RESET (bestehende Verbindung beenden und neu aufbauen) 707
11.2.8.12 Anweisung T_DIAG (Status einer Verbindung prüfen und Informationen lesen) 709
11.2.8.13 Anweisung TMAIL_C (Email über die Ethernet-Schnittstelle der CPU senden) 714
11.2.8.14 UDP 724
11.2.8.15 TUSEND und TURCV 725
11.2.8.16 T_CONFIG 731
11.2.8.17 Gemeinsame Parameter für Anweisungen 739
11.2.9 Kommunikation mit einem Programmiergerät 741
11.2.9.1 Hardware-Kommunikationsverbindung herstellen 741
11.2.9.2 Konfigurieren der Geräte 742
11.2.9.3 IP-Adressen zuweisen 742
11.2.9.4 Testen Ihres PROFINET-Netzwerks 743
11.2.10 Kommunikation HMI/PLC 743
11.2.10.1 Konfigurieren der logischen Netzwerkverbindungen zwischen zwei Geräten 744
11.2.11 Kommunikation PLC/PLC 745
11.2.11.1 Konfigurieren der logischen Netzwerkverbindungen zwischen zwei Geräten 746
11.2.11.2 Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren 746
11.2.11.3 Sende- und Empfangsparameter konfigurieren 746
11.2.12 CPU und PROFINET IO-Gerät konfigurieren 750
11.2.12.1 PROFINET IO-Gerät hinzufügen 750
11.2.12.2 Konfigurieren der logischen Netzwerkverbindungen zwischen einer CPU und einem PROFINET IO-Gerät 751
11.2.12.3 CPUs und Gerätenamen zuweisen 751
11.2.12.4 IP-Adressen zuweisen 752
11.2.12.5 IO-Zykluszeit konfigurieren 752
11.2.13 CPU und PROFINET I-Device konfigurieren 754
11.2.13.1 I-Device-Funktionalität 754
11.2.13.2 Eigenschaften und Vorteile des I-Device 755
11.2.13.3 Merkmale eines I-Device 756
11.2.13.4 Datenaustausch zwischen über- und untergeordnetem E/A-System 758
11.2.13.5 Konfigurieren des I-Device 760
11.2.14 Shared Devices 762
11.2.14.1 Shared-Device-Funktion 762
11.2.14.2 Beispiel: Ein Shared Device konfigurieren (GSD-Konfiguration) 765
11.2.14.3 Beispiel: Ein I-Device als Shared Device konfigurieren 771
11.2.15 Diagnose 781
11.2.16 Anweisungen für die dezentrale Peripherie 781
11.2.17 Diagnoseanweisungen 781
11.2.18 Diagnoseereignisse für die dezentrale Peripherie 781

11.3 PROFIBUS 782
11.3.1 Kommunikationsdienste der PROFIBUS-CMs 784
11.3.2 Verweis auf die Benutzerhandbücher für PROFIBUS-CMs 785
11.3.3 DP-Master und -Slave konfigurieren 785
11.3.3.1 CM 1243-5 (DP-Master) und DP-Slave hinzufügen 785

11.3.3.2 Konfigurieren der logischen Netzwerkverbindungen zwischen zwei PROFIBUS-Geräten ... 786
11.3.3.3 PROFIBUS-Adressen zum CM 1243-5 und DP-Slave zuweisen ... 787
11.3.4 Anweisungen für die dezentrale Peripherie ... 789
11.3.5 Diagnoseanweisungen ... 789
11.3.6 Diagnoseereignisse für die dezentrale Peripherie ... 789

11.4 ASi ... 790
11.4.1 AS-i-Master und -Slavegeräte konfigurieren ... 791
11.4.1.1 AS-i-Mastermodul CM 1243-2 und AS-i-Slave hinzufügen ... 791
11.4.1.2 Logische Netzwerkverbindungen zwischen zwei AS-i-Geräten konfigurieren ... 792
11.4.1.3 Eigenschaften des AS-i-Masters CM1243-2 konfigurieren ... 792
11.4.1.4 Einem AS-i-Slave eine AS-i-Adresse zuweisen ... 793
11.4.2 Datenaustausch zwischen dem Anwenderprogramm und AS-i-Slaves ... 796
11.4.2.1 STEP 7 Basic konfigurieren ... 796
11.4.2.2 Slaves mit STEP 7 konfigurieren ... 797
11.4.3 Anweisungen für die dezentrale Peripherie ... 800
11.4.4 Mit AS-i-Online-Werkzeugen arbeiten ... 800

11.5 S7-Kommunikation ... 802
11.5.1 Anweisungen GET und PUT (Daten aus entfernter CPU auslesen/in entfernte CPU schreiben) ... 802
11.5.2 S7-Verbindung erstellen ... 806
11.5.3 Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren ... 807
11.5.4 Zuweisung von Verbindungsparametern für GET/PUT ... 808
11.5.4.1 Verbindungsparameter ... 809
11.5.4.2 S7-Verbindung von CPU zu CPU konfigurieren ... 811

12 Webserver ... 817

12.1 Webserver aktivieren ... 819

12.2 Konfigurieren von Webserver-Benutzern ... 821

12.3 Über den PC auf die Webseiten zugreifen ... 823

12.4 Über ein mobiles Gerät auf die Webseiten zugreifen ... 824

12.5 Verwenden eines CP-Moduls für den Zugriff auf die Webseiten ... 826

12.6 Standard-Webseiten ... 827
12.6.1 Aufbau der Standard-Webseiten ... 827
12.6.2 Anmeldung und Benutzerrechte ... 829
12.6.3 Einleitung ... 833
12.6.4 Start ... 834
12.6.5 Identifikation ... 835
12.6.6 Diagnosepuffer ... 836
12.6.7 Modulinformationen ... 837
12.6.8 Kommunikation ... 840
12.6.9 Variablenzustand ... 841
12.6.10 Dateibrowser ... 844

12.7 Benutzerdefinierte Webseiten ... 847
12.7.1 HTML-Seiten anlegen ... 848
12.7.2 Vom S7-1200 Webserver unterstützte AWP-Befehle ... 849
12.7.2.1 Variablen lesen ... 851
12.7.2.2 Variablen schreiben ... 852

12.7.2.3 Sondervariablen lesen 854
12.7.2.4 Sondervariablen schreiben 855
12.7.2.5 Alias für einen Variablenverweis nutzen 857
12.7.2.6 Enum-Typen definieren 858
12.7.2.7 CPU-Variablen mit einem Enum-Typ referenzieren 858
12.7.2.8 Fragmente erstellen 860
12.7.2.9 Fragmente importieren 861
12.7.2.10 Definitionen verbinden 862
12.7.2.11 Handhabung von Variablennamen mit Sonderzeichen 862
12.7.3 Verwendung von benutzerdefinierten Webseiten konfigurieren 865
12.7.4 WWW-Anweisung für benutzerdefinierte Webseiten programmieren 866
12.7.5 Programmbausteine in die CPU laden 868
12.7.6 Zugriff auf die benutzerdefinierten Webseiten 868
12.7.7 Einschränkungen bei benutzerdefinierten Webseiten 869
12.7.8 Beispiel für eine benutzerdefinierte Webseite 870
12.7.8.1 Webseite zum Beobachten und Steuern einer Windturbine 870
12.7.8.2 Steuerungsdaten lesen und anzeigen 873
12.7.8.3 Enum-Typ verwenden 873
12.7.8.4 Benutzereingaben in die Steuerung schreiben 875
12.7.8.5 Sondervariablen schreiben 876
12.7.8.6 Referenz: HTML-Code der Webseite "Remote Wind Turbine Monitor" 877
12.7.8.7 Konfiguration der Beispiel-Webseite in STEP 7 882
12.7.9 Benutzerdefinierte Webseiten in mehreren Sprachen einrichten 883
12.7.9.1 Ordnerstruktur anlegen 883
12.7.9.2 Sprachumschaltung programmieren 883
12.7.9.3 STEP 7 für die Verwendung einer mehrsprachigen Seitenstruktur konfigurieren 887
12.7.10 Erweiterte Steuerung von benutzerdefinierten Webseiten 887

12.8 Einschränkungen 892
12.8.1 Funktionseinschränkungen bei Deaktivierung von JavaScript in den Internetoptionen 893
12.8.2 Eingeschränkte Funktionen, wenn Cookies in den Internetoptionen nicht erlaubt sind 894
12.8.3 Siemens-Sicherheitszertifikat importieren 894
12.8.4 Datenprotokolle im CSV-Format in nicht amerikanische/englische Versionen von Microsoft Excel importieren 896

13 Kommunikationsprozessor und Modbus-TCP 897

13.1 Mit den seriellen Kommunikationsschnittstellen arbeiten 897

13.2 Abschließen eines RS485-Busanschlusssteckers 898

13.3 Punkt-zu-Punkt-Kommunikation (PtP) 900
13.3.1 Kommunikationsanschlüsse konfigurieren 902
13.3.1.1 Steuerung der Flusskontrolle 904
13.3.2 Sende- und Empfangsparameter konfigurieren 906
13.3.2.1 Sendeparameter konfigurieren 906
13.3.2.2 Empfangsparameter konfigurieren 907
13.3.3 Punkt-zu-Punkt-Anweisungen 915
13.3.3.1 Gemeinsame Parameter für Punkt-zu-Punkt-Operationen 915
13.3.3.2 Anweisung Port_Config (Kommunikationsparameter dynamisch konfigurieren) 918
13.3.3.3 Anweisung Send_Config (Serielle Übertragungsparameter dynamisch konfigurieren) 920
13.3.3.4 Anweisung Receive_Config (Serielle Empfangsparameter dynamisch konfigurieren) 923
13.3.3.5 Anweisung Send_P2P (Sendepufferdaten übertragen) 928
13.3.3.6 Anweisung Receive_P2P (Empfangsmeldungen aktivieren) 932
13.3.3.7 Anweisung Receive_Reset (Empfangspuffer löschen) 934

13.3.3.8 Anweisung Signal_Get (RS232-Signale abfragen) ... 935
13.3.3.9 Anweisung Signal_Set (RS232-Signale setzen) ... 936
13.3.3.10 Get_Features ... 937
13.3.3.11 Set_Features ... 938
13.3.4 Programmieren der PtP-Kommunikation ... 940
13.3.4.1 Abfragearchitektur ... 941
13.3.5 Beispiel: Punkt-zu-Punkt-Kommunikation ... 942
13.3.5.1 Kommunikationsmodul konfigurieren ... 943
13.3.5.2 Betriebsarten RS422 und RS485 ... 946
13.3.5.3 STEP 7-Programm programmieren ... 949
13.3.5.4 Terminalemulator konfigurieren ... 951
13.3.5.5 Beispielprogramm ausführen ... 951

13.4 Kommunikation über die universelle serielle Schnittstelle (USS) ... 952
13.4.1 Übersicht ... 952
13.4.2 Version der USS-Anweisungen auswählen ... 955
13.4.3 Voraussetzungen für den Einsatz des USS-Protokolls ... 956
13.4.4 USS-Anweisungen ... 959
13.4.4.1 Anweisung USS_Port_Scan (Kommunikation über USS-Netzwerk bearbeiten) ... 959
13.4.4.2 Anweisung USS_Drive_Control (Daten mit Antrieb tauschen) ... 960
13.4.4.3 Anweisung USS_Read_Param (Parameter aus dem Antrieb auslesen) ... 963
13.4.4.4 Anweisung USS_Write_Param (Parameter im Antrieb ändern) ... 964
13.4.5 USS-Zustandscodes ... 966
13.4.6 Allgemeine Voraussetzungen für die USS-Antriebseinrichtung ... 968
13.4.7 Beispiel: Allgemeine USS-Antriebsverbindung und -einrichtung ... 969

13.5 Modbus-Kommunikation ... 973
13.5.1 Überblick über die Modbus RTU- und TCP-Kommunikation, Modbus-TCP-Anweisungen V13 ... 973
13.5.2 Modbus TCP ... 976
13.5.2.1 Übersicht ... 976
13.5.2.2 Version der Modbus TCP-Anweisungen auswählen ... 977
13.5.2.3 Modbus TCP-Anweisungen ... 978
13.5.2.4 Beispiele für Modbus TCP ... 993
13.5.3 Modbus RTU ... 998
13.5.3.1 Übersicht ... 998
13.5.3.2 Version der Modbus RTU-Anweisungen auswählen ... 1000
13.5.3.3 Modbus RTU-Anweisungen ... 1001
13.5.3.4 Modbus RTU Beispiele ... 1018

13.6 PtP-Kommunikation in älteren Systemen (nur CM/CB 1241) ... 1022
13.6.1 Ältere Punkt-zu-Punkt-Anweisungen ... 1022
13.6.1.1 Anweisung PORT_CFG (Kommunikationsparameter dynamisch konfigurieren) ... 1022
13.6.1.2 Anweisung SEND_CFG (Serielle Übertragungsparameter dynamisch konfigurieren) ... 1024
13.6.1.3 RCV_CFG (Serielle Empfangsparameter dynamisch konfigurieren) ... 1025
13.6.1.4 Anweisung SEND_PTP (Sendepufferdaten übertragen) ... 1030
13.6.1.5 Anweisung RCV_PTP (Empfangsmeldungen aktivieren) ... 1033
13.6.1.6 Anweisung RCV_RST (Empfangspuffer löschen) ... 1035
13.6.1.7 Anweisung SGN_GET (RS232-Signale abfragen) ... 1036
13.6.1.8 Anweisung SGN_SET (RS232-Signale festlegen) ... 1037

13.7 USS-Kommunikation in älteren Systemen (nur CM/CB 1241) ... 1038
13.7.1 Version der USS-Anweisungen auswählen ... 1039
13.7.2 Voraussetzungen für den Einsatz des USS-Protokolls ... 1040

13.7.3 Ältere USS-Anweisungen 1042
13.7.3.1 Anweisung USS_PORT (Kommunikation über USS-Netzwerk bearbeiten)....1042
13.7.3.2 Anweisung USS_DRV (Daten mit Antrieb tauschen) 1044
13.7.3.3 Anweisung USS_RPM (Parameter aus dem Antrieb auslesen)....1047
13.7.3.4 Anweisung USS_WPM (Parameter im Antrieb ändern) 1049
13.7.4 Alte USS-Statuscodes 1050
13.7.5 Allgemeine Voraussetzungen für die Antriebseinrichtung mit der alten Anweisung USS ..1053

13.8 Modbus TCP-Kommunikation in älteren Systemen....1054
13.8.1 Übersicht....1054
13.8.2 Version der Modbus TCP-Anweisungen auswählen 1054
13.8.3 Ältere Modbus TCP-Anweisungen....1055
13.8.3.1 Anweisung MB_CLIENT (Über PROFINET als Modbus TCP-Client kommunizieren)....1055
13.8.3.2 Anweisung MB_SERVER (Über PROFINET als Modbus TCP-Server kommunizieren) ...1062
13.8.4 Beispiel: MB_SERVER für mehrere TCP-Verbindungen in älteren Systemen....1068
13.8.5 Beispiel: MB_CLIENT 1 in älteren Systemen: Mehrere Anforderungen mit gemeinsamer TCP-Verbindung 1069
13.8.6 Beispiel: MB_CLIENT 2 in älteren Systemen: Mehrere Anforderungen mit unterschiedlichen TCP-Verbindungen 1070
13.8.7 Beispiel: MB_CLIENT 3 in älteren Systemen: Schreibanforderung für das Prozessabbild der Ausgänge....1071
13.8.8 Beispiel: MB_CLIENT 4 in älteren Systemen: Mehrere Anforderungen koordinieren....1072

13.9 Modbus RTU-Kommunikation in älteren Systemen (nur CM/CB 1241) 1073
13.9.1 Übersicht....1073
13.9.2 Version der Modbus RTU-Anweisungen auswählen....1073
13.9.3 Ältere Modbus RTU-Anweisungen 1074
13.9.3.1 Anweisung MB_COMM_LOAD (Port am PtP-Modul für Modbus RTU konfigurieren) 1074
13.9.3.2 Anweisung MB_MASTER (Über PtP-Port als Modbus RTU-Master kommunizieren) 1077
13.9.3.3 Anweisung MB_SLAVE (Über PtP-Port als Modbus RTU-Slave kommunizieren)....1084
13.9.4 Beispiel: Beispielprogramm für einen Modbus RTU-Master in älteren Systemen 1091
13.9.5 Beispiel: Beispielprogramm für einen Modbus RTU-Slave in älteren Systemen 1093

13.10 Telecontrol und TeleService mit dem CP 1242-7....1094
13.10.1 Übersicht über Telecontrol-CPs....1094
13.10.2 Anschluss an ein GSM-Netz....1096
13.10.3 Anwendungen des CP 1242-7....1097
13.10.4 Weitere Eigenschaften des CP-1242-7 1098
13.10.5 Projektierung und elektrische Anschlüsse....1098
13.10.6 Weitere Informationen....1099
13.10.7 Zubehör....1100
13.10.8 Verweis auf Handbuch zur GSM-Antenne....1101
13.10.9 Konfigurationsbeispiele für Telecontrol....1101

14 TeleService-Kommunikation (SMTP-E-Mail)....1107

14.1 Anweisung TM_Mail (E-Mail senden) 1107

15 Online- und Diagnose-Tools....1115

15.1 Status-LEDs....1115

15.2 Online-Verbindung mit einer CPU herstellen....1119

15.3 Einem PROFINET IO-Gerät online einen Namen zuweisen 1120

15.4 Einstellen der IP-Adresse und der Uhrzeit....1122

15.5 Auf Werkseinstellungen zurücksetzen ... 1123
15.6 Firmware aktualisieren ... 1124
15.7 Bedienpanel für die Online-CPU ... 1125
15.8 Überwachung von Zykluszeit und Speicherauslastung ... 1125
15.9 Diagnoseereignisse in der CPU anzeigen ... 1126
15.10 Vergleichen von Offline- und Online-CPUs ... 1127
15.11 Werte in der CPU beobachten und steuern ... 1128
15.11.1 Online gehen, um die Werte in der CPU zu beobachten ... 1129
15.11.2 Zustand im Programmiereditor anzeigen ... 1130
15.11.3 Online-Werte eines DBs erfassen, um die Startwerte zurückzusetzen ... 1130
15.11.4 Werte in der CPU über die Beobachtungstabelle beobachten und steuern ... 1131
15.11.4.1 Variablen mit Trigger beobachten oder steuern ... 1133
15.11.4.2 Ausgänge im Betriebszustand STOP freischalten ... 1134
15.11.5 Werte in der CPU forcen ... 1135
15.11.5.1 Arbeiten mit der Forcetabelle ... 1135
15.11.5.2 Funktionsweise der Forcefunktion ... 1136
15.12 Laden im Betriebszustand RUN ... 1138
15.12.1 Voraussetzungen für "Laden im Betriebszustand RUN" ... 1139
15.12.2 Ändern des Programms im Betriebszustand RUN ... 1140
15.12.3 Ausgewählte Bausteine laden ... 1141
15.12.4 Einen einzelnen ausgewählten Baustein mit einem Übersetzungsfehler in einem anderen Baustein laden ... 1143
15.12.5 Bestehende Bausteine im Betriebszustand RUN ändern und ins Zielsystem laden ... 1144
15.12.6 Systemreaktion bei fehlgeschlagenem Ladevorgang ... 1147
15.12.7 Sicherheitsaspekte beim Laden im Betriebszustand RUN ... 1148
15.13 CPU-Daten bei Auslösebedingungen verfolgen und aufzeichnen ... 1150
A Technische Daten ... 1151
A.1 Allgemeine technische Daten ... 1151
A.2 CPU 1211C ... 1163
A.2.1 Allgemeine technische Daten und Leistungsmerkmale ... 1163
A.2.2 Von der CPU 1211C unterstützte Zeiten, Zähler und Codebausteine ... 1164
A.2.3 Digitale Eingänge und Ausgänge ... 1167
A.2.4 Analoge Eingänge ... 1169
A.2.4.1 Schrittantwort der integrierten analogen Eingänge der CPU ... 1169
A.2.4.2 Abtastzeit der integrierten analogen Ports der CPU ... 1170
A.2.4.3 Messbereiche der analogen Eingänge für Spannung (CPUs) ... 1170
A.2.5 Schaltpläne der CPU 1211 ... 1171
A.3 CPU 1212C ... 1175
A.3.1 Allgemeine technische Daten und Leistungsmerkmale ... 1175
A.3.2 Von der CPU 1212C unterstützte Zeiten, Zähler und Codebausteine ... 1177
A.3.3 Digitale Eingänge und Ausgänge ... 1179
A.3.4 Analoge Eingänge ... 1181
A.3.4.1 Schrittantwort der integrierten analogen Eingänge der CPU ... 1181
A.3.4.2 Abtastzeit der integrierten analogen Ports der CPU ... 1182
A.3.4.3 Messbereiche der analogen Eingänge für Spannung (CPUs) ... 1182
A.3.5 Schaltpläne der CPU 1212C ... 1183

A.4 CPU 1214C 1187
A.4.1 Allgemeine technische Daten und Leistungsmerkmale 1187
A.4.2 Von der CPU 1214C unterstützte Zeiten, Zähler und Codebausteine 1188
A.4.3 Digitale Eingänge und Ausgänge 1191
A.4.4 Analoge Eingänge 1193
A.4.4.1 Schrittantwort der integrierten analogen Eingänge der CPU 1193
A.4.4.2 Abtastzeit der integrierten analogen Ports der CPU 1194
A.4.4.3 Messbereiche der analogen Eingänge für Spannung (CPUs) 1194
A.4.5 Schaltpläne der CPU 1214C 1195

A.5 CPU 1215C 1199
A.5.1 Allgemeine technische Daten und Leistungsmerkmale 1199
A.5.2 Von der CPU 1215C unterstützte Zeiten, Zähler und Codebausteine 1201
A.5.3 Digitale Eingänge und Ausgänge 1203
A.5.4 Analogeingänge und -ausgänge 1205
A.5.4.1 Schrittantwort der integrierten analogen Eingänge der CPU 1205
A.5.4.2 Abtastzeit der integrierten analogen Ports der CPU 1206
A.5.4.3 Messbereiche der analogen Eingänge für Spannung (CPUs) 1206
A.5.4.4 Technische Daten der Analogausgänge 1206
A.5.5 Schaltpläne der CPU 1215C 1208

A.6 CPU 1217C 1213
A.6.1 Allgemeine technische Daten und Leistungsmerkmale 1213
A.6.2 Von der CPU 1217C unterstützte Zeiten, Zähler und Codebausteine 1215
A.6.3 Digitale Eingänge und Ausgänge 1217
A.6.4 Analogeingänge und -ausgänge 1221
A.6.4.1 Technische Daten der analogen Eingänge 1221
A.6.4.2 Schrittantwort der integrierten analogen Eingänge der CPU 1221
A.6.4.3 Abtastzeit der integrierten analogen Ports der CPU 1222
A.6.4.4 Messbereiche der analogen Eingänge für Spannung (CPUs) 1222
A.6.4.5 Technische Daten der Analogausgänge 1222
A.6.5 Schaltpläne der CPU 1217C 1224
A.6.6 CPU 1217C Differentialeingang (DI) Detail und Anwendungsbeispiel 1226
A.6.7 CPU 1217C Differentialausgang (DO) Detail und Anwendungsbeispiel 1227

A.7 Digitale Signalmodule (SM) 1228
A.7.1 Technische Daten für das digitale Eingangsmodul SM 1221 1228
A.7.2 Technische Daten für das digitale Ausgangsmodul SM 1222 mit 8 Ausgängen 1230
A.7.3 Technische Daten für das digitale Ausgangsmodul SM 1222 mit 16 Ausgängen 1232
A.7.4 Technische Daten des digitalen Ein-/Ausgangsmoduls SM 1223 (V DC) 1237
A.7.5 Technische Daten des digitalen Ein-/Ausgangsmoduls SM 1223 (AC) 1241

A.8 Analoge Signalmodule (SM) 1244
A.8.1 Technische Daten des SM 1231 Analogeingabemoduls 1244
A.8.2 Technische Daten des SM 1232 Analogausgabemoduls 1248
A.8.3 Technische Daten des SM 1234 Analogein-/Analogausgabemoduls 1250
A.8.4 Schrittantwort der analogen Eingänge 1253
A.8.5 Abtastzeit und Aktualisierungszeiten der Analogeingänge 1253
A.8.6 Messbereiche der analogen Eingänge für Spannung und Strom (SB und SM) 1254
A.8.7 Messbereiche der analogen Ausgänge für Spannung und Strom (SB und SM) 1255

A.9 Thermoelement- und RTD-Signalmodule (SMs) 1256
A.9.1 SM 1231 Thermoelement 1256
A.9.1.1 Grundlegende Funktionsweise eines Thermoelements 1259
A.9.1.2 Auswahltabellen für das SM 1231 Thermoelement 1260

A.9.2 SM 1231 RTD ... 1262
A.9.2.1 Auswahltabellen für das SM 1231 RTD ... 1265

A.10 Technologiemodule ... 1268
A.10.1 SM 1278 4xIO-Link-Master SM ... 1268
A.10.1.1 SM 1278 4xIO-Link-Master - Überblick ... 1271
A.10.1.2 Anschluss ... 1274
A.10.1.3 Parameter/Adressbereich ... 1276
A.10.1.4 Alarm-, Fehler- und Systemmeldungen ... 1278

A.11 Digitale Signalboards (SBs) ... 1282
A.11.1 Technische Daten des SB 1221 200 kHz Digitaleingabe ... 1282
A.11.2 Technische Daten des SB 1222 200 kHz Digitalausgabe ... 1284
A.11.3 Technische Daten des SB 1223 200 kHz Digitalein-/Digitalausgabe ... 1287
A.11.4 Technische Daten SB 1223 2 x 24 V DC Eingang / 2 x 24 V DC Ausgang ... 1290

A.12 Analoge Signalboards (SBs) ... 1293
A.12.1 Technische Daten des SB 1231 1 Analogeingang ... 1293
A.12.2 Technische Daten des SB 1232 1 Analogausgabe ... 1295
A.12.3 Messbereiche der analogen Eingänge und Ausgänge ... 1297
A.12.3.1 Schrittantwort der analogen Eingänge ... 1297
A.12.3.2 Abtastzeit und Aktualisierungszeiten der Analogeingänge ... 1297
A.12.3.3 Messbereiche der analogen Eingänge für Spannung und Strom (SB und SM) ... 1297
A.12.3.4 Messbereiche der analogen Ausgänge für Spannung und Strom (SB und SM) ... 1298
A.12.4 Thermoelement-Signalboards (SBs) ... 1300
A.12.4.1 Technische Daten des SB 1231 1 Analogeingang Thermoelement ... 1300
A.12.4.2 Grundlegende Funktionsweise eines Thermoelements ... 1301
A.12.5 RTD-Signalboards (SBs) ... 1304
A.12.5.1 Technische Daten des SB 1231 1 Analogeingang RTD ... 1304
A.12.5.2 Auswahltabellen für das SB 1231 RTD ... 1307

A.13 BB 1297 Batterieboard ... 1309

A.14 Kommunikationsschnittstellen ... 1311
A.14.1 PROFIBUS ... 1311
A.14.1.1 CM 1242-5 PROFIBUS DP-SLAVE ... 1311
A.14.1.2 Anschlussbelegung der Sub-D-Buchse des CM 1242-5 ... 1312
A.14.1.3 CM 1243-5 PROFIBUS DP-Master ... 1313
A.14.1.4 Anschlussbelegung der Sub-D-Buchse des CM 1243-5 ... 1315
A.14.2 CP 1242-7 ... 1315
A.14.2.1 CP 1242-7 GPRS ... 1316
A.14.2.2 GSM/GPRS-Antenne ANT794-4MR ... 1317
A.14.2.3 Flachantenne ANT794-3M ... 1318
A.14.3 CM 1243-2 AS-i-Master ... 1319
A.14.3.1 Technische Daten des AS-i Master CM 1243-2 ... 1319
A.14.3.2 Elektrischer Anschluss des AS-i-Masters ... 1320
A.14.4 RS232, RS422 und RS485 ... 1322
A.14.4.1 Technische Daten des CM 1241 RS485 ... 1322
A.14.4.2 Technische Daten des CM 1241 RS232 ... 1324
A.14.4.3 Technische Daten des CM 1241 RS422/485 ... 1325

A.15 TeleService (TS-Adapter und TS-Adaptermodul) ... 1327

A.16 SIMATIC Memory Cards ... 1327

A.17 Eingangssimulatoren ... 1327

A.18 S7-1200 Potentiometermodul 1329
A.19 Steckleitung für Erweiterungsmodule 1331
A.20 Zugehörige Produkte 1331
A.20.1 PM 1207 Stromversorgungsmodul 1331
A.20.2 CSM 1277 Compact Switch Module 1332
A.20.3 CM CANopen-Modul 1332
A.20.4 Kommunikationsmodul RF120C 1333
B Berechnen der Leistungsbilanz 1335
C Bestellinformationen 1339
C.1 CPU-Module 1339
C.2 Signalmodule (SMs), Signalboards (SBs) und Batterieboards (BBs) 1340
C.3 Kommunikation 1342
C.4 Fehlersichere CPUs und Signalmodule 1343
C.5 Sonstige Module 1344
C.6 Memory Cards 1344
C.7 Grundlegende HMI-Geräte 1344
C.8 Ersatzteile und sonstige Hardware 1345
C.9 Programmiersoftware 1348
D Geräteaustausch und Ersatzteilkompatibilität 1349
D.1 Ersatzklemmenblöcke S7-1200 V3.0 und V4.0 1355
Index 1357

Produktübersicht

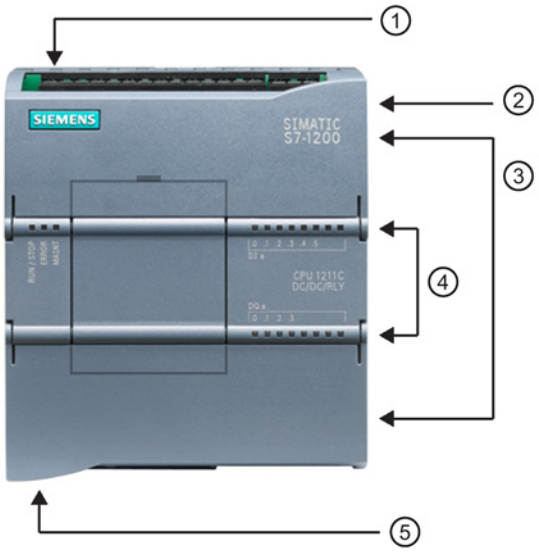
1

1.1 Einführung in die S7-1200 SPS

Die Steuerung S7-1200 bietet Ihnen die erforderliche Flexibilität und Leistung zur Steuerung einer breiten Palette von Geräten für Ihre Automatisierungslösungen. Durch das kompakte Design, die flexible Konfiguration und einen leistungsstarken Befehlssatz eignet sich die S7-1200 hervorragend für eine große Bandbreite von Steuerungsanwendungen.

Die CPU umfasst einen Mikroprozessor, eine integrierte Spannungsversorgung, Eingangskreise und Ausgangskreise, integriertes PROFINET, Peripherie zur Bewegungssteuerung in Hochgeschwindigkeit sowie integrierte Analogeingänge in einem kompakten Gehäuse und bildet somit eine leistungsstarke Steuerung. Nachdem Sie Ihr Programm geladen haben, enthält die CPU die erforderliche Logik, damit Sie die Geräte in Ihrer Anwendung beobachten und steuern können. Die CPU beobachtet Eingänge und ändert Ausgänge anhand der Befehle Ihres Anwenderprogramms, das Boolesche Verknüpfungen, Zähl- und Zeitfunktionen, komplexe arithmetische Operationen und Kommunikation mit anderen intelligenten Geräten umfassen kann.

Die CPU verfügt über einen PROFINET-Port zur Kommunikation über ein PROFINET-Netzwerk. Für die Kommunikation über PROFIBUS-, GPRS-, RS485- oder RS232-, IEC-, DNP3- und WDC-Netzwerke stehen weitere Module zur Verfügung.



① Stromanschluss
② Steckplatz für eine Memory Card unter der oberen Abdeckklappe
③ Steckbarer Klemmenblock für die Anwenderverdrahtung (hinter den Abdeckklappen)
④ Status-LEDs für die integrierten E/A
⑤ PROFINET-Anschluss (auf der Unterseite der CPU)

Verschiedene Sicherheitsfunktionen schützen den Zugriff auf die CPU und das Steuerungsprogramm:

- Jede CPU ist mit einem Passwortschutz (Seite 205) ausgestattet, mit dem der Zugriff auf die CPU-Funktionen nach Bedarf eingerichtet werden kann.
- Sie können mit dem Knowhow-Schutz (Seite 208) den Code in einem bestimmten Baustein verbergen.
- Mit dem Kopierschutz (Seite 209) können Sie Ihr Programm mit einer bestimmten Memory Card oder CPU verknüpfen.

Tabelle 1- 1 Vergleich der CPU-Varianten

Merkmal		CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C	CPU 1217C
Abmessungen (mm)		90 x 100 x 75		110 x 100 x 75	130 x 100 x 75	150 x 100 x 75
Anwenderspeicher	Arbeitsspeicher	50 KB	75 KB	100 KB	125 KB	150 KB
	Ladespeicher	1 MB		4 MB		
	Remanent	10 KB				
Integrierte E/A	Digital	6 Eingänge/4 Ausgänge	8 Eingänge/6 Ausgänge	14 Eingänge/10 Ausgänge		
	Analog	2 Eingänge			2 Eingänge/2 Ausgänge	
Größe des Prozessabbilds	Eingänge (E)	1024 Byte				
	Ausgang (A)	1024 Byte				
Merker (M)		4096 Byte		8192 Byte		

Merkmal		CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C	CPU 1217C
Erweiterung: Signalmodul (SM)		Keine	2	8		
Signalboard (SB), Batterieboard (BB) oder Kommunikationsboard (CB)		1				
Kommunikationsmodule (CM) (Anbau links)		3				
Schnelle Zähler	Summe	Bis zu 6 konfiguriert für die Verwendung integrierter oder SB-Eingänge				
	1 MHz	-				Eb.2 bis Eb.5
	100/[1]80 kHz	Ea.0 bis Ea.5				
	30/[1]20 kHz	--	Ea.6 bis Ea.7	Ea.6 bis Eb.5		Ea.6 bis Eb.1
	200 kHz[3]					
Impulsausgänge [2]	Summe	Bis zu 4 konfiguriert für die Verwendung integrierter oder SB-Ausgänge				
	1 MHz	--				Aa.0 bis Aa.3
	100 kHz	Aa.0 bis Aa.3				Aa.4 bis Ab.1
	20 kHz	--	Aa.4 bis Aa.5	Aa.4 bis Ab.		--
Memory Card		SIMATIC Memory Card (optional)				
Pufferung Echtzeituhr		20 Tage, typ./min. 12 Tage bei 40 Grad C (wartungsfreier Hochleistungskondensator)				
PROFINET Ethernet-Kommunikationsport		1			2	
Ausführungszeit arithm. Operationen		2,3 µs/Operation				
Ausführungszeit Boolesche Operationen		0,08 µs/Operation				

[1] Die niedrigere Geschwindigkeit ist gültig, wenn der HSC für den A/B-Zählerbetrieb konfiguriert ist.

[2] Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge zu verwenden.

[3] Bis zu 200 kHz sind verfügbar mit dem SB 1221 DI x 24 VDC 200 kHz und SB 1221 DI 4 x 5 VDC 200 kHz.

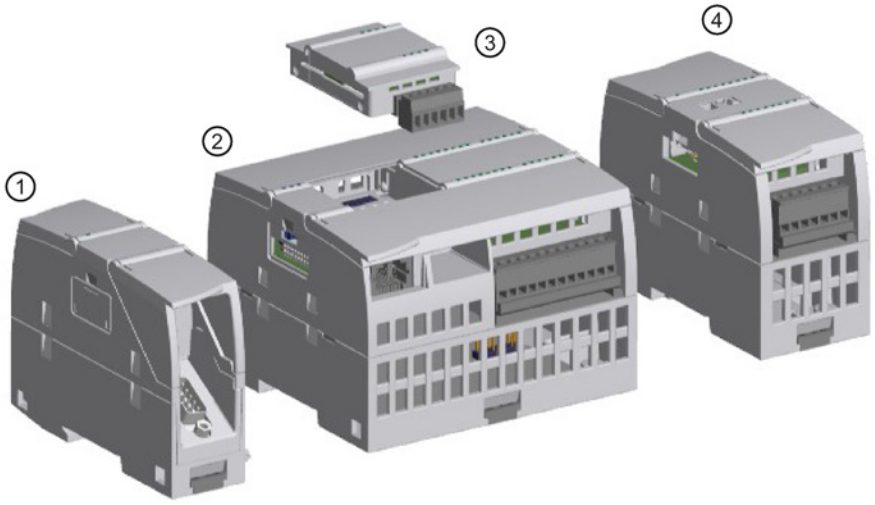
Die verschiedenen CPU-Ausführungen bieten eine Vielfalt an Leistungsmerkmalen und Funktionen, damit Sie effektive Lösungen für verschiedenste Anwendungen erstellen können. Ausführliche Informationen zu bestimmten CPUs finden Sie in den technischen Daten (Seite 1151).

Tabelle 1- 2 Von der S7-1200 unterstützte Bausteine, Zeiten und Zähler

Element		Beschreibung
Bausteine	Typ	OB, FB, FC, DB
	Größe	50 KB (CPU 1211C) 75 KB (CPU 1212C) 100 KB (CPU 1214C) 125 KB (CPU 1215C) 150 KB (CPU 1217C)
	Anzahl	Bis 1024 Bausteine gesamt (OBs + FBs + FCs + DBs)
	Schachtelungstiefe	16 aus Zyklus- oder Anlauf-OBs 6 aus beliebigen Alarm-OBs
	Überwachung	Der Zustand von 2 Codebausteinen kann gleichzeitig überwacht werden.
OBs	Programmzyklus	Multi
	Anlauf	Multi
	Verzögerungsalarme	4 (1 pro Ereignis)
	Weckalarme	4 (1 pro Ereignis)
	Prozessalarme	50 (1 pro Ereignis)
	Zeitfehleralarme	1
	Diagnosefehleralarme	1
	Ziehen oder Stecken von Modulen	1
	Fehler bei Baugruppenträger oder Station	1
	Tageszeit	Multi
	Zustand	1
	Update	1
	Profil	1
Zeiten	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, 16 Bytes pro Zeit
Zähler	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, Größe abhängig von der Zählart • SInt, USInt: 3 Byte • Int, UInt: 6 Byte • DInt, UDInt: 12 Byte

1.2 Erweiterung der CPU-Funktionen

Die Produktfamilie S7-1200 bietet eine Vielzahl von Modulen und steckbaren Boards zur Erweiterung der CPU um zusätzliche E/A oder andere Kommunikationsprotokolle. Ausführliche Informationen zu bestimmten Modulen finden Sie in den technischen Daten (Seite 1151).



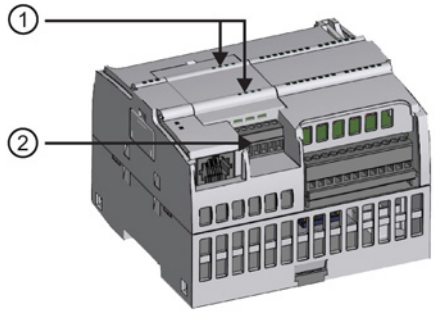
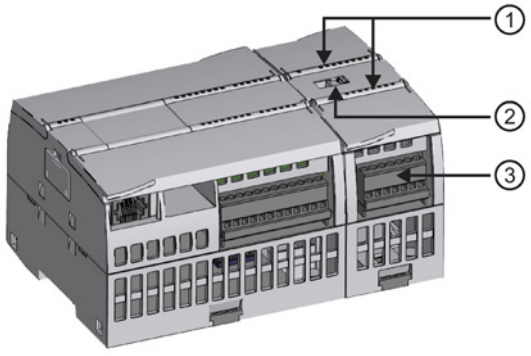
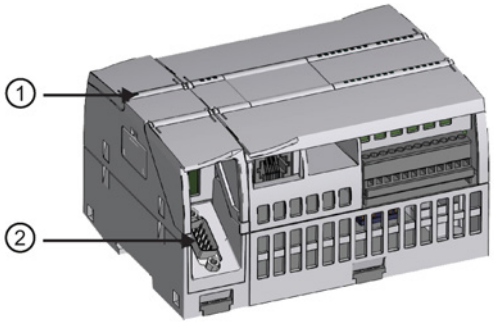
① Kommunikationsmodul (CM) oder Kommunikationsprozessor (CP) (Seite 1311)

② CPU (CPU 1211C (Seite 1163), CPU 1212C (Seite 1175), CPU 1214C (Seite 1187), CPU 1215C (Seite 1199), CPU 1217C (Seite 1213))

③ Signalboard (SB) (digitales SB (Seite 1282), analoges SB (Seite 1293)), Kommunikationsboard (CB) (Seite 1322) oder Batterieboard (BB) CPU (CPU 1211C, CPU 1212C, CPU 1214C, CPU 1215C, CPU 1217C) (Seite 1309)

④ Signalmodul (SM) (digitales SM (Seite 1228), analoges SM (Seite 1244), Thermoelement-SM (Seite 1256), RTD-SM (Seite 1262), Technologie-SM) (Seite 1268)

Tabelle 1- 3 S7-1200 Erweiterungsmodule

Art des Moduls	Beschreibung
Die CPU unterstützt ein steckbares Erweiterungsboard: • Ein Signalboard (SB) stellt zusätzliche E/A für Ihre CPU bereit. Das SB wird auf der Vorderseite der CPU angeschlossen. • Mit einem Kommunikationsboard (CB) können Sie Ihre CPU um einen Kommunikationsanschluss erweitern. • Ein Batterieboard (BB) gewährleistet die langfristige Pufferung der Echtzeituhr.	
	① Status-LEDs am SB
	② Steckbarer Klemmenblock für die Anwenderverdrahtung
Signalmodule (SMs) erweitern die CPU um zusätzliche Funktionalität. SMs werden an der rechten Seite der CPU angeschlossen. • Digitale E/A • Analoge E/A • RTD und Thermoelement • SM 1278 IO-Link-Master	
	① Status-LEDs
	② Schiebelasche Busstecker
	③ Steckbarer Klemmenblock für die Anwenderverdrahtung
Kommunikationsmodule (CMs) und Kommunikationsprozessoren (CPs) erweitern die CPU um verschiedene Kommunikationsmöglichkeiten, z. B. PROFIBUS oder RS232/RS485 (für PtP, Modbus oder USS) oder AS-i-Master. Ein CP bietet Möglichkeiten für andere Arten der Kommunikation, z. B. für den Anschluss der CPU über ein GPRS-, IEC-, DNP3- oder WDC-Netzwerk. • Die CPU unterstützt maximal 3 CMs oder CPs. • Jedes CM bzw. jeder CP wird an der linken Seite der CPU angeschlossen	
	① Status-LEDs

Art des Moduls	Beschreibung
(bzw. an der linken Seite eines anderen CMs oder CPs).	② Kommunikationsstecker

1.3 Grundlegende HMI-Panels

Die SIMATIC HMI Basic Panels bieten Geräte mit Touchscreen für grundlegende Aufgaben im Bereich Bedienen und Beobachten. Alle Panels weisen die Schutzklasse IP65 auf und sind nach CE, UL, cULus und NEMA 4x zertifiziert.

Die erhältlichen Basic HMI Panels (Seite 1344)sind nachstehend beschrieben:

- KTP400 Basic: 4"-Touchscreen mit 4 konfigurierbaren Tasten, Auflösung 480 x 272 und 800 Variablen
- KTP700 Basic: 7"-Touchscreen mit 8 konfigurierbaren Tasten, Auflösung 800 x 480 und 800 Variablen
- KTP700 Basic DP: 7" Touchscreen mit 8 konfigurierbaren Tasten, Auflösung 800 x 480 und 800 Variablen
- KTP900 Basic: 9"-Touchscreen mit 8 konfigurierbaren Tasten, Auflösung 800 x 480 und 800 Variablen
- KTP1200 Basic: 12" Touchscreen mit 10 konfigurierbaren Tasten, Auflösung 800 x 480 und 800 Variablen
- KTP 1200 Basic DP: 12" Touchscreen mit 10 konfigurierbaren Tasten, Auflösung 800 x 400 und 800 Variablen

Siehe auch

Kunden-Support (http://www.siemens.com/automation/)

Neue Funktionen

2

Folgende Funktionen sind in dieser Version neu hinzugekommen:

- Die funktionale Sicherheit ist in der Hardware und Firmware der fehlersicheren S7-1200 CPUs und Signalmodule (SM) zusammen mit dem von der Software (ES) heruntergeladenen Sicherheitsprogramm implementiert. Beachten Sie für weitere Informationen das S7-1200 Handbuch Funktionale Sicherheit (http://support.automation.siemens.com/WW/view/de/104547552).
- Simulation von S7-1200 CPUs mit Firmware-Version V4.0 und höher: Mithilfe von S7-PLCSIM V13 SP1 können Sie Ihre PLC-Programme auf einer simulierten PLC testen, ohne dazu Hardware zu benötigen. S7-PLCSIM ist eine separat installierte Anwendung, die zusammen mit STEP 7 in TIA Portal läuft. Sie können Ihre PLC und alle zugehörigen Module in STEP 7 konfigurieren, Ihre Anwendungslogik programmieren und dann die Hardwarekonfiguration und das Programm in den simulierten in S7-PLCSIM laden. Mit den Tools von S7-PLCSIM können Sie dann Ihr Programm simulieren und testen. Die vollständige Dokumentation finden Sie in der Online-Hilfe zu S7-PLCSIM. Fehlersichere CPUs können nicht simuliert werden.
- Konfigurationssteuerung (Optionsverwaltung) (Seite 157): Sie können die Hardware für den Maximalausbau Ihres Rechners konfigurieren, auch wenn Sie bestimmte Module derzeit in Ihrer Anwendung nicht nutzen. Die Konfiguration und Bezeichnung dieser flexiblen Module ist neu in dieser Version von STEP 7 und S7-1200. Module, die zwar konfiguriert, aber nicht vorhanden sind, verursachen keine Fehlermeldungen.
- Der Webserver (Seite 817) unterstützt jetzt den Zugriff über die IP-Adresse ausgewählter (Kommunikationsprozessor-) Module im lokalen Baugruppenträger sowie über die IP-Adresse der S7-1200 CPU.
- Verbesserte Motion-Funktionalität:
  - Analoge und PROFIdrive-Verbindungen
  - Erweiterte Modulo- und Regelparameter
- Zeitraummessungen mit schnellen Zählern (HSC) (Seite 465)
- Leistungsverbesserungen des SCL-Compilers
- Dynamischer Kopierschutz (Seite 209) für Programmbausteine mit obligatorischem Passwort
- Erweiterte PROFINET-Funktionalität, einschließlich Unterstützung für Shared Devices (Seite 762).

- Neue Programmieranweisungen:
  - EQ_Type, NE_Type, EQ_ElemType, NE_ElemType (Seite 245)
  - IS_NULL, NOT_NULL (Seite 246)
  - IS_ARRAY (Seite 246)
  - Deserialize (Seite 260), Serialize (Seite 263)
  - VariantGet (Seite 272), VariantPut (Seite 273), CountOfElements (Seite 274)
  - Variant_to_DB_Any (Seite 287), DB_Any_To_Variant (Seite 288)
  - GET_IM_DATA (Seite 411)
  - RUNTIME (Seite 303)
  - GEO2LOG (Seite 457), IO2MOD (Seite 460)
  - ReadLittle, WriteLittle, ReadBig, WriteBig (nur SCL) (Seite 271)
  - T_RESET (Seite 707), T_DIAG (Seite 709) und TMAIL_C (Seite 714)
  - PID_Temp (Seite 502)
  - Neue Modbus-Anweisungen (Seite 973)
  - Neue Anweisungen für Punkt-zu-Punkt-Kommunikation (PtP) (Seite 900)
  - Neue USS-Anweisungen (Seite 952)

Neue Module für die S7-1200

Neue Module erweitern die Leistungsfähigkeit der S7-1200 CPU und bieten Ihnen die erforderliche Flexibilität für Ihren Automatisierungsbedarf:

- Industrielle Kommunikationsmodule für die dezentrale Steuerung (Seite 1342): Diese CPs können als Kommunikationsmodule in der S7-1200 V4.1-CPU eingesetzt werden.
- Fehlersichere CPUs und Peripherie: In Zusammenhang mit der Version S7-1200 V4.1 oder neuer gibt es vier fehlersichere CPUs und drei fehlersichere Signalmodule (SMs):
  - CPU 1214FC DC/DC/DC (6ES7 214-1AF40-0XB0)
  - CPU 1214FC DC/DC/RLY (6ES7 214-1HF40-0XB0)
  - CPU 1215FC DC/DC/DC (6ES7 215-1AF40-0XB0)
  - CPU 1215FC DC/DC/RLY (6ES7 215-1HF40-0XB0)
  - SM 1226 F-DI 16 x 24 VDC (6ES7 226-6BA32-0XB0)
  - SM 1226 F-DQ 4 x 24 VDC (6ES7 226-6DA32-0XB0)
  - SM 1226 F-DQ 2 x Relay (6ES7 226-6RA32-0XB0)

  Die S7-1200 Standard-Signalmodule (SM), Kommunikationsmodule (CM) und Signalboards (SB) können in einem System zusammen mit fehlersicheren SMs für Steuerungsfunktionen eingesetzt werden, die keine funktionale Sicherheitsklasse erfordern. Standard-SMs, die für den Einsatz mit fehlersicheren SMs unterstützt werden, haben die Artikelnummern (6ES7 --- ---32 0XB0) oder neuer.

Austausch der V3.0-CPU durch eine V4.1-CPU

Wenn Sie eine S7-1200 V3.0-CPU durch eine S7-1200 V4.1-CPU austauschen, beachten Sie die dokumentierten Unterschiede (Seite 1349) zwischen den Versionen und die erforderlichen Maßnahmen.

STEP 7 Programmiersoftware

3

STEP 7 bietet eine bedienerfreundliche Umgebung zum Entwickeln, Bearbeiten und Beobachten der Logik zur Steuerung Ihrer Anwendung. Sie bietet auch die Werkzeuge zum Konfigurieren aller Geräte in Ihrem Projekt, wie PLC- und HMI-Geräte. Damit Sie die Informationen finden, die Sie benötigen, verfügt STEP 7 über eine umfangreiche Online-Hilfe.

STEP 7 bietet Standardprogrammiersprachen, mit denen Sie das Steuerungsprogramm für Ihre Anwendung bequem und effizient entwickeln können.

- KOP (Kontaktplan) (Seite 193) ist eine grafische Programmiersprache. Die Darstellung beruht auf Schaltplänen.
- FUP (Funktionsplan) (Seite 194) ist eine Programmiersprache, die auf den grafischen Logiksymbolen der Booleschen Algebra basiert.
- SCL (Structured Control Language) (Seite 195) ist eine textbasierte, höhere Programmiersprache.

Wenn Sie einen Codebaustein anlegen, müssen Sie die Programmiersprache für den Baustein auswählen. Ihr Anwenderprogramm kann mit Codebausteinen arbeiten, die in einer dieser Programmiersprachen angelegt wurden.

Hinweis

STEP 7 ist die Softwarekomponente für Programmierung und Konfiguration im TIA-Portal. Das TIA-Portal umfasst neben STEP 7 auch WinCC zum Entwerfen und Ausführen von Runtime-Prozessvisualisierung, und es bietet Online-Hilfe für WinCC ebenso wie für STEP 7.

3.1 Systemvoraussetzungen

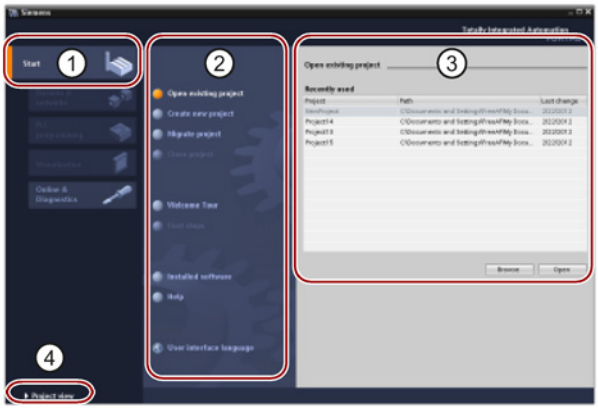
Sie müssen STEP 7 mit Administratorrechten installieren.

Tabelle 3- 1 Systemvoraussetzungen

Hardware/Software	Voraussetzungen
Prozessortyp	Intel® Core™ i5-3320M ab 3,3 GHz
RAM	8 GB
Freier Festplattenspeicher	2 GB auf Systemlaufwerk C:\
Betriebssysteme	Sie können STEP 7 mit den folgenden Betriebssystemen verwenden (64-Bit, Windows 7 auch 32-Bit) • Microsoft Windows 7 Home Premium SP1 oder höher (nur STEP 7 Basic, wird bei STEP 7 Professional nicht unterstützt) • Microsoft Windows 7 oder höher (Professional SP1, Enterprise SP1, Ultimate SP1) • Microsoft Windows 8.1 (nur STEP 7 Basic, wird bei STEP 7 Professional nicht unterstützt) • Microsoft Windows 8.1 (Professional, Enterprise) • Microsoft Server 2008 R2 StdE SP1 (nur STEP 7 Professional) • Microsoft Server 2012 R2 StdE
Grafikkarte	32 MB RAM-Speicher 24-Bit-Farbtiefe
Bildschirmauflösung	1920 x 1080 (empfohlen)
Netzwerk	20 MBit/s Ethernet oder schneller
Optisches Laufwerk	DVD-ROM

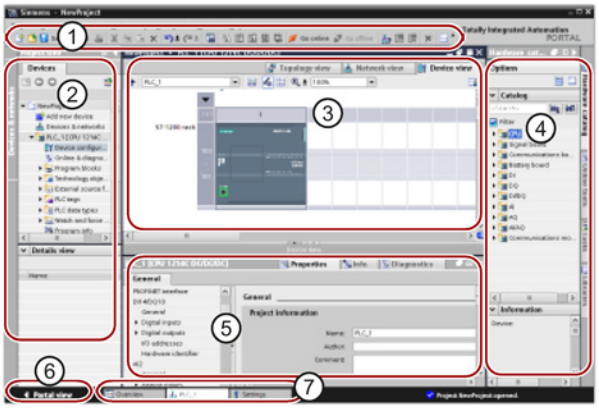
3.2 Einfaches Arbeiten mit unterschiedlichen Ansichten

STEP 7 stellt eine benutzerfreundliche Umgebung bereit, in der Sie die Steuerungslogik entwickeln, die HMI-Visualisierung konfigurieren und die Netzwerkkommunikation einrichten können. Zur Steigerung Ihrer Produktivität bietet STEP 7 zwei unterschiedliche Ansichten des Projekts: eine tätigkeitsorientierte Anzahl von Portalen für die einzelnen Funktionen (Portalansicht) und eine projektorientierte Ansicht der Elemente im Projekt (Projektansicht). Sie entscheiden, in welcher Ansicht Sie am effizientesten arbeiten können. Per Mausklick können Sie zwischen der Portalansicht und der Projektansicht wechseln.



Portalansicht

① Portale für die verschiedenen Aufgaben
② Aufgaben für das ausgewählte Portal
③ Auswahlpanel zur gewählten Aktion
④ Änderungen an der Projektansicht



Projektansicht

① Menüs und Funktionsleiste
② Projektnavigator
③ Arbeitsbereich
④ Taskcards
⑤ Inspektorfenster
⑥ Änderungen an der Portalansicht
⑦ Editorleiste

Da sich alle Komponenten an einer Stelle befinden, haben Sie schnellen Zugriff auf jeden Bereich Ihres Projekts. Beispielsweise zeigt das Inspektorfenster die Eigenschaften und weitere Informationen für das Objekt an, das im Arbeitsbereich ausgewählt wurde. Für die verschiedenen von Ihnen gewählten Objekte zeigt das Inspektorfenster jeweils die konfigurierbaren Eigenschaften. Das Inspektorfenster verfügt außerdem über Register, unter denen Diagnoseinformation und weitere Meldungen angezeigt werden.

In der Editorleiste werden alle derzeit geöffneten Editoren angezeigt. Mit der Editorleiste arbeiten Sie so schneller und effizienter. Zum Umschalten zwischen geöffneten Editoren klicken Sie einfach auf den gewünschten Editor. Sie können auch zwei Editoren gleichzeitig anzeigen und diese vertikal oder horizontal anordnen. Dadurch sind "Drag&Drop"-Operationen zwischen Editoren möglich.

3.3 Bedienerfreundliche Werkzeuge

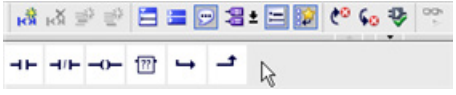
3.3.1 Anweisungen in Ihr Anwenderprogramm einfügen

STEP 7 bietet Taskcards mit den Anweisungen für Ihr Programm. Die Anweisungen sind nach Funktionen gegliedert.

Um Ihr Programm anzulegen, ziehen Sie die Anweisungen von der Taskcard in ein Netzwerk.

3.3.2 Zugreifen auf Anweisungen in der Funktionsleiste "Favoriten"

STEP 7 enthält eine Funktionsleiste "Favoriten" für den schnellen Zugriff auf Anweisungen, die Sie häufig verwenden. Um eine Anweisung in Ihr Netzwerk einzufügen, genügt ein Klick auf das entsprechende Symbol.



(Um im Anweisungsverzeichnis die "Favoriten" aufzurufen, doppelklicken Sie auf das Symbol.)

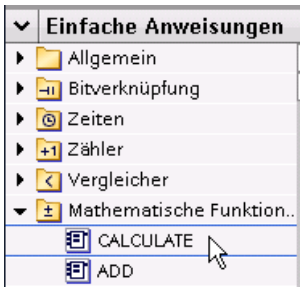
Die Funktionsleiste "Favoriten" lässt sich problemlos anpassen und durch neue Anweisungen erweitern.

Fügen Sie die gewünschte Anweisung einfach mit "Drag & Drop" den "Favoriten" hinzu.

Für den Zugriff auf die Anweisung genügt jetzt ein Klick!

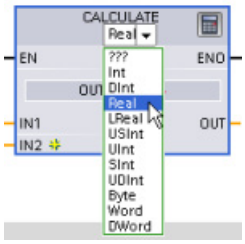
3.3.3 Erstellen einer komplexen Gleichung mit einer einfachen Anweisung

Mit der Anweisung Calculate (Seite 247) können Sie eine mathematische Funktion erstellen, die mehrere Eingangsparameter verarbeitet und das Ergebnis entsprechend der von Ihnen vorgegebenen Gleichung ausgibt.



Erweitern Sie im Basic-Anweisungsverzeichnis den Ordner der mathematischen Funktionen. Doppelklicken Sie auf die Anweisung Calculate, um sie in Ihr Anwenderprogramm einzufügen.

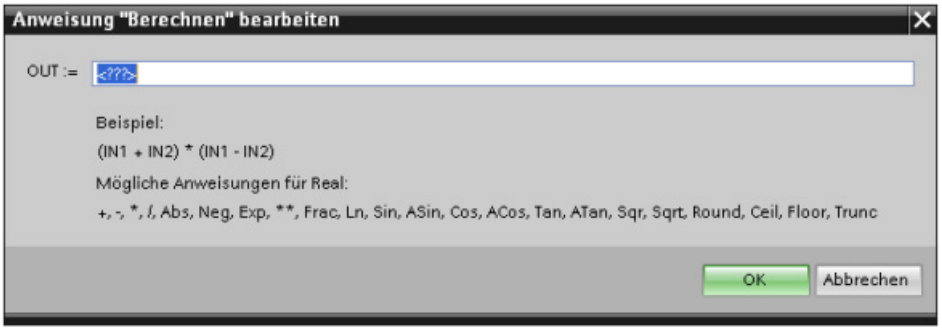
Die nicht konfigurierte Anweisung Calculate bietet zwei Eingangsparameter und einen Ausgangsparameter.



Klicken Sie auf "???" und wählen Sie die Datentypen für die Eingangs- und Ausgangsparameter aus. (Alle Eingangs- und Ausgangsparameter müssen denselben Datentyp haben.)

Wählen Sie für dieses Beispiel den Datentyp "Real" aus.

Klicken Sie auf das Symbol "Gleichung bearbeiten", um die Gleichung einzugeben.



Geben Sie in diesem Beispiel die folgende Gleichung zum Skalieren eines Rohanalogwerts ein. (Die Bezeichnungen "In" und "Out" entsprechen den Parametern der Anweisung Calculate.)

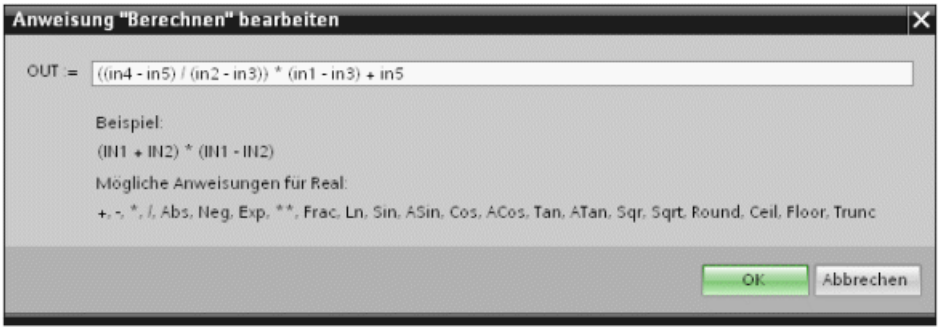
$Out_{value} = ((Out_{high} - Out_{low}) / (In_{high} - In_{low})) * (In_{value} - In_{low}) + Out_{low}$

Out = ((in4 - in5) / (in2 - in3)) * (in1 - in3) + in5

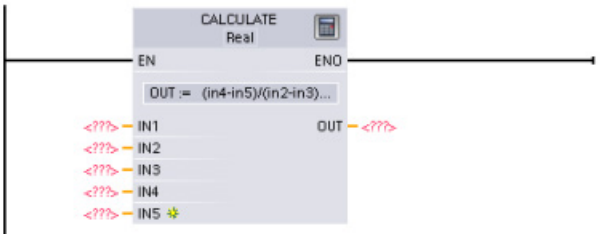
Erläuterung:			
	Out_{value}	(Out)	Skalierter Ausgangswert
	In_{value}	(in1)	Analogeingangswert
	In_{high}	(in2)	Oberer Grenzwert für den skalierten Eingangswert
	In_{low}	(in3)	Unterer Grenzwert für den skalierten Eingangswert
	Out_{high}	(in4)	Oberer Grenzwert für den skalierten Ausgangswert
	Out_{low}	(in5)	Unterer Grenzwert für den skalierten Ausgangswert

Geben Sie im Feld "Calculate bearbeiten" die Gleichung mit den Parameternamen ein:

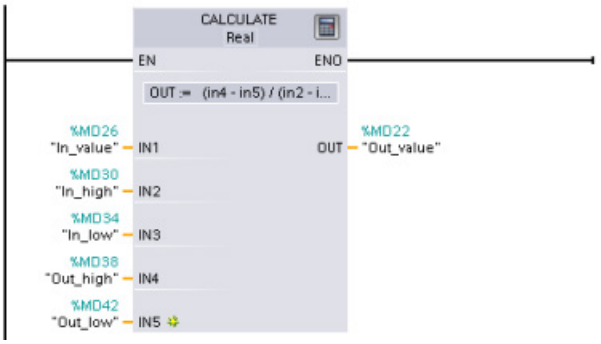
OUT = ((in4 - in5) / (in2 - in3)) * (in1 - in3) + in5



Wenn Sie auf "OK" klicken, erstellt die Anweisung Calculate die für die Anweisung erforderlichen Eingänge.



Geben Sie die Variablennamen für die Werte ein, die den Parametern entsprechen.



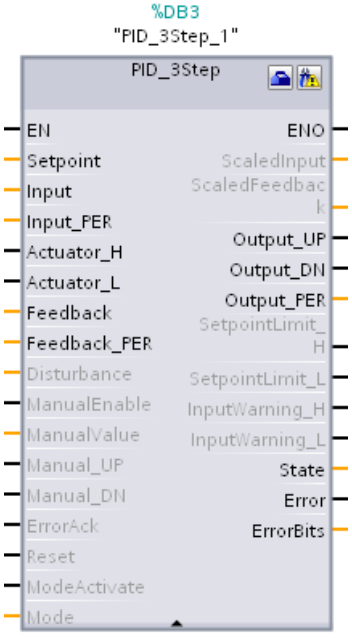
3.3.4 Ergänzen von Eingängen oder Ausgängen in einer KOP- oder FUP-Anweisung

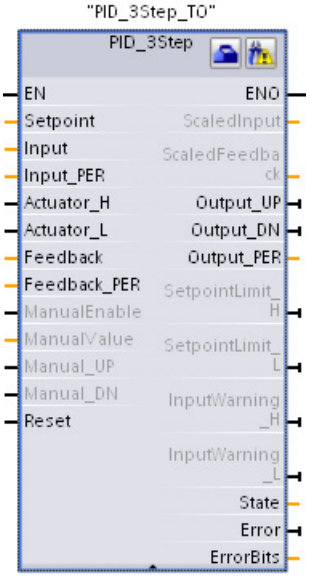
Bei einigen Anweisungen können Sie weitere Eingänge oder Ausgänge erstellen.

- Um einen Eingang oder Ausgang hinzuzufügen, klicken Sie auf das Symbol "Erstellen" oder an einem der vorhandenen Parameter IN oder OUT mit der rechten Maustaste auf den Eingangsanschluss und wählen den Befehl "Eingang einfügen".
- Um einen Eingang oder Ausgang zu löschen, klicken Sie bei einem der vorhandenen IN- oder OUTParameter mit der rechten Maustaste auf den Anschluss (sofern mehr als die zwei ursprünglichen Eingänge vorhanden sind) und wählen den Befehl "Löschen".

3.3.5 Erweiterbare Anweisungen

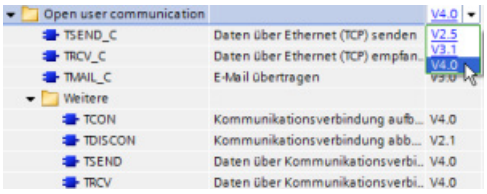
Einige der komplexeren Anweisungen sind erweiterbar und zeigen zunächst nur die wesentlichen Eingänge und Ausgänge an. Um alle Eingänge und Ausgänge vollständig anzuzeigen, klicken Sie auf den Pfeil im unteren Bereich der Anweisung.





3.3.6 Auswählen einer Version für eine Anweisung

Durch die Entwicklung und die Ausgabezyklen bestimmter Befehlssätze (z. B. Modbus, PID und Bewegungssteuerung) ist es inzwischen zu mehreren freigegebenen Versionen dieser Anweisungen gekommen. Um die Kompatibilität und Migration mit älteren Projekten sicherzustellen, können Sie in STEP 7 auswählen, welche Anweisungsversion Sie in Ihr Anwenderprogramm einfügen.



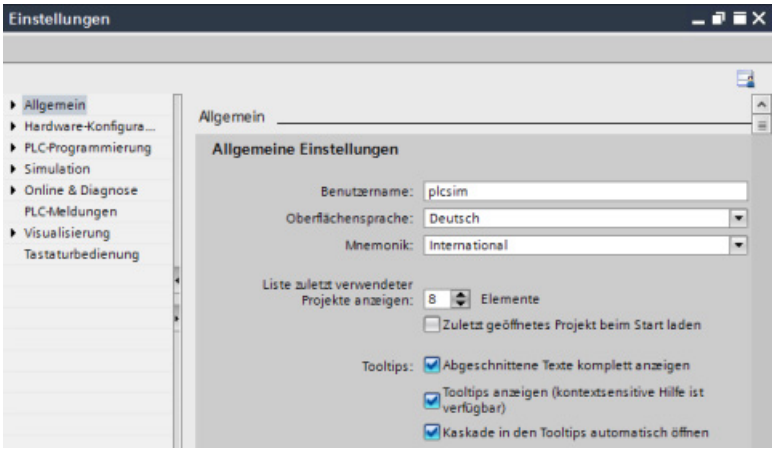
Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.

Um die Version einer Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus.

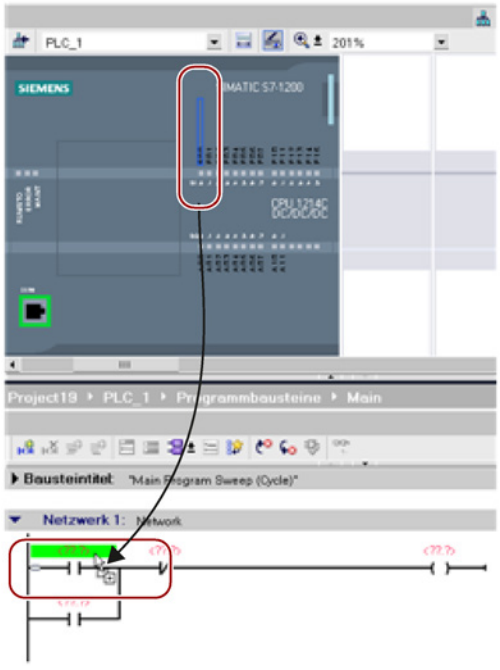
3.3.7 Ändern des Erscheinungsbilds und der Konfiguration von STEP 7

Sie haben zahlreiche Einstellmöglichkeiten; diese betreffen z. B. das Aussehen der Bedienoberfläche, die Sprache oder den Ordner zum Speichern Ihrer Arbeitsergebnisse.

Wählen Sie zum Ändern von Einstellungen im Menü "Optionen" den Befehl "Einstellungen".



3.3.8 Drag & Drop zwischen Editoren

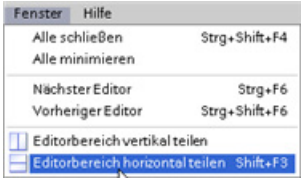


Damit Sie Aufgaben schnell und unkompliziert erledigen können, ermöglicht STEP 7 das Ziehen und Ablegen mit der Maus ("Drag & Drop") von Elementen zwischen den Editoren. So können Sie beispielsweise einen Eingang von der CPU an die Adresse einer Anweisung in Ihrem Anwenderprogramm ziehen.

Sie brauchen zum Auswählen der Eingänge oder Ausgänge der CPU mindestens den Zoomfaktor 200 %.

Beachten Sie, dass die Variablennamen nicht nur in der PLC-Variablentabelle, sondern auch in der CPU angezeigt werden.

Um zwei Editoren gleichzeitig anzuzeigen, verwenden Sie den Menübefehl "Editor teilen" oder die entsprechende Schaltfläche in der Funktionsleiste.



Zum Umschalten zwischen den geöffneten Editoren klicken Sie auf die jeweiligen Symbole in der Editorleiste.

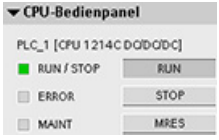
3.3.9 Wechseln des Betriebszustands der CPU

Die CPU verfügt nicht über einen physischen Schalter zum Ändern des Betriebszustands (STOP oder RUN).

Klicken Sie in der Funktionsleiste auf die Schaltfläche "CPU starten" oder "CPU stoppen", um den Betriebszustand der CPU zu ändern.

Beim Konfigurieren der CPU in der Gerätekonfiguration legen Sie das Anlaufverhalten der CPU über ihre Eigenschaften fest (Seite 168).

Im Portal "Online & Diagnose" steht Ihnen auch ein Bedienpanel zur Verfügung, in dem Sie den Betriebszustand der Online-CPU ändern können. Um das CPU-Bedienpanel verwenden zu können, müssen Sie mit der CPU online verbunden sein. Das in der Taskcard "Online-Tools" enthaltene Bedienpanel zeigt den Betriebszustand der Online-CPU an. Über das Bedienpanel können Sie auch den Betriebszustand der Online-CPU ändern.

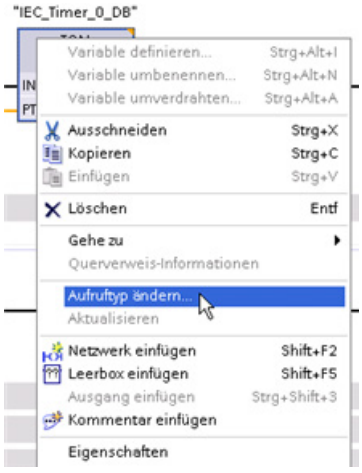


Mit der Schaltfläche auf dem Bedienpanel ändern Sie den Betriebszustand (STOP bzw. RUN). Außerdem enthält das Bedienpanel eine Schaltfläche MRES zum Urlöschen des Speichers.

Der aktuelle Betriebszustand der CPU wird durch die Farbe der RUN/STOP-Anzeige angegeben. Gelb steht für den Betriebszustand STOP, Grün für RUN.

In der Gerätekonfiguration in STEP 7 können Sie auch den Standardbetriebszustand beim Einschalten der CPU konfigurieren (Seite 85).

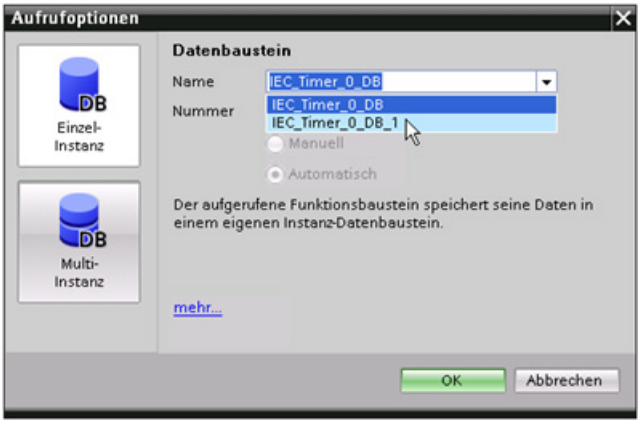
3.3.10 Aufruftyp eines DB ändern



In STEP 7 können Sie problemlos die Zuweisung einrichten oder ändern, die zwischen einem DB und einer Anweisung oder einem DB und einem FB besteht, der sich in einem FB befindet.

- Sie können die Zuweisung zwischen verschiedenen DBs umschalten.
- Sie können die Zuweisung zwischen einem Einzelinstanz-DB und einem Multiinstanz-DB wechseln.
- Sie können einen Instanz-DB erstellen (sofern ein Instanz-DB fehlt oder nicht verfügbar ist).

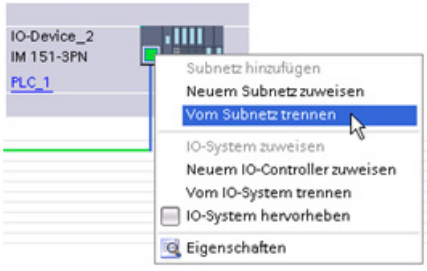
Sie können den Befehl "Aufruftyp ändern" auswählen, indem Sie entweder im Programmiereditor mit der rechten Maustaste auf die Anweisung oder den FB klicken oder indem Sie im Menü "Optionen" den Befehl "Bausteinaufruf" auswählen.



Im Dialog "Aufrufoptionen" können Sie einen Einzelinstanz- oder einen Multiinstanz-DB auswählen. Außerdem können Sie in einer Klappliste der verfügbaren DBs bestimmte DBs auswählen.

3.3.11 Geräte vorübergehend vom Netzwerk trennen

Sie können einzelne Netzwerkgeräte vom Subnetz trennen. Weil die Konfiguration des Geräts nicht aus dem Projekt entfernt wird, können Sie die Verbindung des Geräts mühelos wiederherstellen.



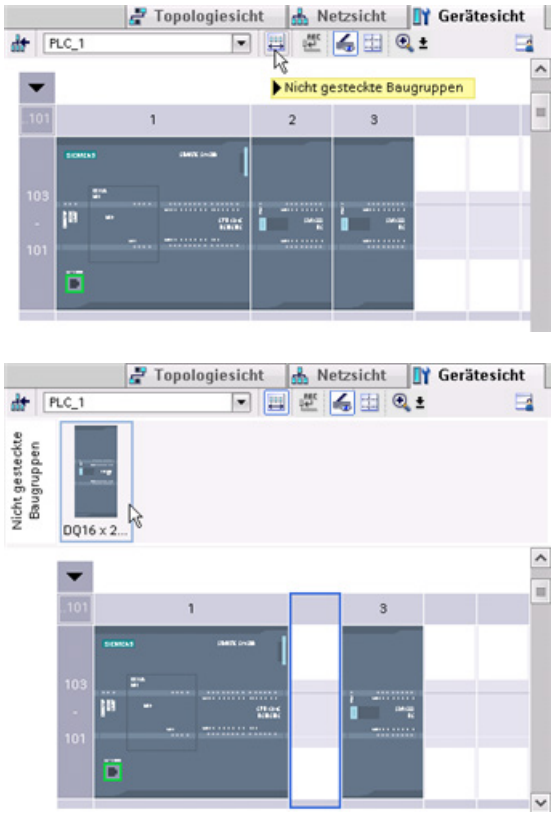
Klicken Sie mit der rechten Maustaste auf den Schnittstellenanschluss des Netzwerkgeräts und wählen Sie im Kontextmenü den Befehl "Vom Subnetz trennen".

STEP 7 konfiguriert die Netzwerkverbindungen neu, entfernt das getrennte Gerät jedoch nicht aus dem Projekt. Die Netzwerkverbindung wird zwar gelöscht, doch die Schnittstellenadressen werden nicht verändert.

Wenn Sie die neuen Netzwerkverbindungen laden, müssen Sie die CPU in den Betriebszustand STOP versetzen.

Um das Gerät wieder anzuschließen, erstellen Sie einfach eine neue Netzwerkverbindung mit dem Anschluss des Geräts.

3.3.12 Virtuelles Abziehen von Geräten aus der Konfiguration



STEP 7 bietet Ihnen eine virtuelle Ablage für "nicht gesteckte" Baugruppen. Sie können eine Baugruppe vom Baugruppenträger "abziehen" und dabei die Konfiguration der Baugruppe speichern. Diese abgezogenen Baugruppen werden mit dem Projekt zusammen gespeichert; dadurch können Sie sie später wieder einsetzen, ohne die Parameter erneut konfigurieren zu müssen.

Diese Funktion wird z. B. bei kurzfristigen Wartungsarbeiten eingesetzt. Denken Sie beispielsweise an einen Fall, in dem Sie auf eine bestimmte Baugruppe warten, die ausgetauscht werden soll, und eine andere Baugruppe vorübergehend als Ersatz verwenden wollen. In diesem Fall können Sie die konfigurierte Baugruppe vom Baugruppenträger in die "Ablage nicht gesteckter Baugruppen" ziehen und anschließend die vorübergehende Ersatzbaugruppe einsetzen.

3.4 Rückwärtskompatibilität

STEP 7 V13 SP1 unterstützt die Konfiguration und Programmierung der S7-1200 V4.1-CPU.

Sie können aber Konfigurationen und Programme für die S7-1200 V4.0 von STEP 7 V13 in eine S7-1200 V4.1-CPU laden. Ihre Konfigurationen und Programme sind dann auf die Funktionen und Anweisungen beschränkt, die von STEP 7 V13 und S7-1200 V4.0 unterstützt wurden.

Durch diese Rückwärtskompatibilität können auf den neuen S7-1200 V4.1-CPU-Versionen Programme laufen, die für ältere Versionen entwickelt und programmiert wurden.

Einbau

4

4.1 Richtlinien für den Einbau von S71200 Geräten

Die S7-1200 Geräte wurden so ausgelegt, dass sie einfach einzubauen sind. Sie können eine S7-1200 entweder in einer Schalttafel oder auf einer Standard-Hutschiene einbauen; die S71200 kann horizontal oder vertikal eingebaut werden. Die kompakte Größe der S7-1200 macht eine effiziente Platzausnutzung möglich.

Die fehlersicheren S7-1200 CPUs unterstützen die fehlersichere dezentrale PROFINET- und PROFIBUS-Peripherie nicht.

Das SIMATIC S7-1200 System ist anhand von Normen für elektrische Geräte als offenes Betriebsmittel klassifiziert. Sie müssen die S7-1200 in einem Gehäuse, Schaltschrank oder in einer Schaltzentrale einbauen. Nur berechtigtes Personal darf Zugang zum Gehäuse, Schaltschrank oder der Schaltzentrale haben.

Bei der Installation ist eine trockene Umgebung für die S7-1200 vorzusehen. SELV/PELV-Schaltungen sollen in trockenen Umgebungen Schutz vor elektrischem Schlag bieten.

Die Installation muss für offene Betriebsmittel in Ihrer spezifischen Standortkategorie gemäß den geltenden elektrischen Vorschriften und den Gebäudevorschriften genehmigten mechanischen Schutz und Umgebungsschutz bieten.

Verschmutzung leitfähiger Teile durch Staub, Feuchtigkeit und Luftverschmutzung kann zu Betriebsfehlern und elektrischen Fehlern in der PLC führen.

Wenn sich der PLC in einem Bereich befindet, in dem Verschmutzung von leitfähigen Teilen auftreten kann, dann muss der PLC durch ein Gehäuse mit entsprechender Schutzklasse geschützt werden. IP54 ist eine Schutzklasse, die im Allgemeinen für elektronische Anlagen in stark verunreinigten Umgebungen verwendet wird und möglicherweise für Ihre Anwendung geeignet ist.

⚠ WARNUNG

Nicht ordnungsgemäße Installation der S7-1200 kann zu elektrischen Fehlern oder unerwartetem Betrieb der Maschine führen.

Elektrische Fehler oder unerwarteter Betrieb der Maschine kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Alle Anweisungen für die Installation und Wartung einer ordnungsgemäßen Betriebsumgebung sind zu befolgen, um einen sicheren Betrieb der Geräte zu gewährleisten.

Halten Sie die S7-1200 Geräte fern von Wärme, Hochspannung und elektrischen Störungen

Als allgemeine Regel für die Anordnung von Geräten in Ihrem System gilt, dass Sie Geräte, die Hochspannung oder hohe elektrische Störungen erzeugen, von den elektronischen Niederspannungsgeräten wie der S7-1200 fernhalten.

Wenn Sie das Layout der S71200 in Ihrer Schalttafel planen, berücksichtigen Sie wärmeerzeugende Geräte und ordnen Sie die elektronischen Geräte in den kühleren Bereichen Ihres Schaltschranks an. Je weniger Sie ein elektronisches Gerät Umgebungen mit hohen Temperaturen aussetzen, desto länger ist die Betriebsdauer des Geräts.

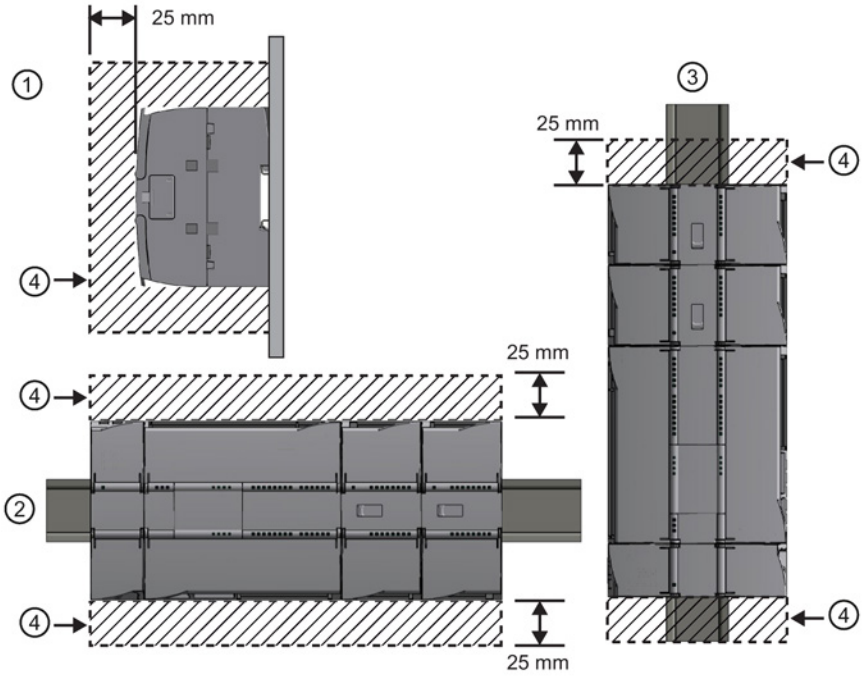
Berücksichtigen Sie auch, wie Sie die Verdrahtung der Geräte in der Schalttafel verlegen. Vermeiden Sie es, Niederspannungssignalleitungen und Kommunikationskabel in der gleichen Kabelbahn wie AC-Versorgungsleitungen und schnellschaltende Hochgeschwindigkeits-DC-Leitungen zu verlegen.

Lassen Sie genügend Abstand für Kühlung und Verdrahtung

Die S71200 Geräte sind für natürliche Wärmeabfuhr durch Konvektion ausgelegt. Lassen Sie deshalb oberhalb und unterhalb der Geräte jeweils mindestens 25 mm Platz, um die Wärmeabfuhr zu gewährleisten. Achten Sie ferner darauf, dass zwischen der Modulfront und der Innenseite des Gehäuses eine Tiefe von mindestens 25 mm bleibt.

⚠ VORSICHT
Bei vertikalem Einbau ist die maximal zulässige Umgebungstemperatur um 10 °C niedriger.
Richten Sie ein vertikal eingebautes S71200 System wie in der folgenden Abbildung gezeigt aus.
Stellen Sie sicher, dass das S7-1200 System richtig montiert wird.

Wenn Sie das Layout für Ihr S71200 System planen, lassen Sie genügend Abstand für die Verdrahtung und die Kommunikationskabelanschlüsse.



① Seitenansicht
② Waagerechter Einbau
③ Senkrechter Einbau
④ Freiraum

4.2 Leistungsbilanz

Ihre CPU besitzt eine interne Spannungsversorgung, die neben der CPU die Signalmodule, Signalboards, Kommunikationsmodule und andere 24VDCVerbraucher speist.

In den Technischen Daten (Seite 1151) finden Sie Informationen zur 5-V-DC-Leistungsbilanz Ihrer CPU und zum 5-V-DC-Leistungsbedarf der Signalmodule, Signalboards und Kommunikationsmodule. Mit Hilfe der Informationen unter "Berechnen der Leistungsbilanz" (Seite 1335) können Sie berechnen, wieviel Leistung (bzw. Strom) die CPU für Ihre Konfiguration liefern kann.

Die CPU verfügt auch über eine 24-V-DC-Geberversorgung, die 24 V DC für Eingänge, für die Relaisspulen der Signalmodule und für sonstige Verbraucher liefert. Wenn der 24-V-DC-Leistungsbedarf die Leistung der Geberversorgung übersteigt, dann müssen Sie eine externe 24-V-DC-Spannungsversorgung an Ihr System anschließen. In den Technischen Daten (Seite 1151) finden Sie die Leistungsbilanz für die 24-V-DC-Geberversorgung für Ihre jeweilige S7-1200 CPU.

Hinweis

Das CM 1243-5 (PROFIBUS-Mastermodul) muss über die 24-V-DC-Geberversorgung der CPU gespeist werden.

Wenn Sie eine externe 24-V-DC-Spannungsversorgung benötigen, müssen Sie darauf achten, dass die Spannungsversorgung nicht parallel zur Geberversorgung der CPU angeschlossen ist. Die beste Störfestigkeit erreichen Sie, wenn Sie die Leitungen (M) der jeweiligen Spannungsversorgungen miteinander verbinden.

⚠ WARNUNG

Wenn Sie parallel zu der 24-V-DCGeberversorgung eine externe 24VDCSpannungsquelle anschließen, kann es sein, dass die beiden Spannungsquellen sich beim Aufbauen der geeigneten Ausgangsspannung gegenseitig beeinträchtigen.

Als Folge kann sich die Lebensdauer verkürzen bzw. eine oder beide Spannungsquellen können sofort ausfallen. Unvorhersehbarer Betrieb kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Die DC-Geberversorgung und eine externe Spannungsquelle müssen die Spannung an unterschiedlichen Punkten liefern.

Einige der 24-V-DC-Eingangsports des S7-1200 Systems sind miteinander verbunden, wobei ein logischer Bezugsleiter mehrere M-Klemmen verbindet. Die folgenden Stromkreise sind beispielsweise miteinander verbunden, sofern sie in den Datenblättern als "nicht potentialgetrennt" angegeben sind: die 24-V-DC-Versorgung der CPU, der Leistungseingang für die Relaisspule eines SM oder die Versorgung eines nicht potentialgetrennten Analogeingangs. Alle nicht potentialgetrennten M-Klemmen müssen an dasselbe externe Bezugspotential angeschlossen werden.

⚠ WARNUNG

Wenn Sie nicht potentialgetrennte M-Klemmen an verschiedene Bezugspotentiale anschließen, verursacht dies unbeabsichtigten Stromfluss, der zu Beschädigung oder unvorhersehbarem Betrieb des Zielsystems und angeschlossener Geräte führen kann.

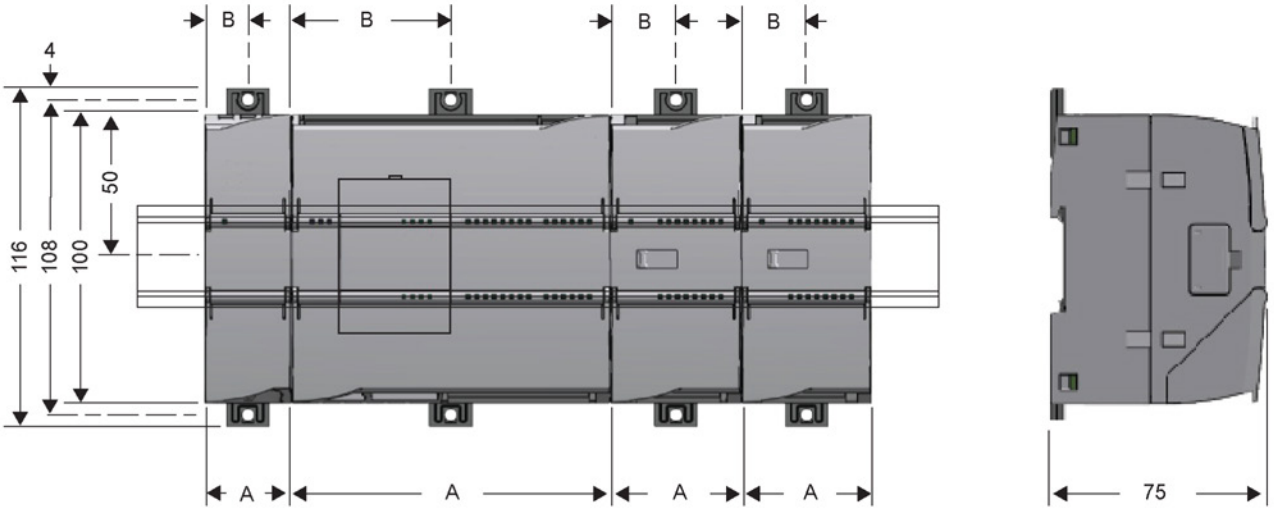
Die Nichteinhaltung dieser Richtlinien kann Schaden oder unvorhersehbaren Betrieb verursachen, was zu Tod oder schwerer Körperverletzung und/oder Sachschaden führen kann.

Schließen Sie stets alle nicht potentialgetrennten M-Klemmen in einem S7-1200 System an dasselbe Bezugspotential an.

4.3 Ein- und Ausbauanweisungen

4.3.1 Einbaumaße der S7-1200 Geräte

CPU 1211C, CPU 1212C, CPU 1214C
(measurements in mm)



CPU 1215C, CPU 1217C

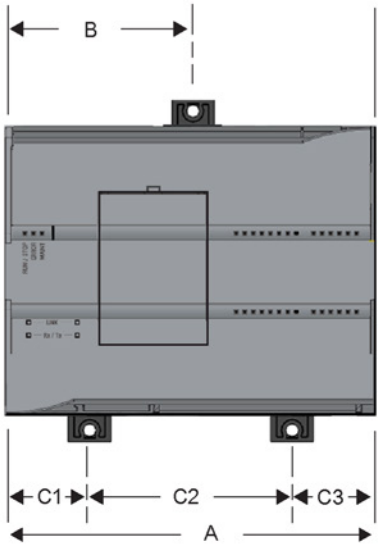


Tabelle 4- 1 Abmessungen für die Montage (mm)

S71200 Geräte		Breite A (mm)	Breite B (mm)	Breite C (mm)
CPU	CPU 1211C und CPU 1212C	90	45	--
	CPU 1214C	110	55	--
	CPU 1215C	130	65 (oben)	Unten: C1: 32.5 C2: 65 C3: 32.5
	CPU 1217C	150	75	Unten: C1: 37.5 C2: 75 C3: 37.5
Signalmodule	8 oder 16 digitale E/A 2, 4 oder 8 analoge E/A Thermoelement, 4 oder 8 E/A RTD, 4 E/A SM 1278 IO Link-Master	45	22.5	--
	8 digitale Ausgänge x Relais (Umschaltung)	70	35	--
	16 analoge E/A RTD, 8 E/A	70	35	--
Kommunikati-onsschnittstellen	CM 1241 RS232 und CM 1241 RS422/485 CM 1243-5 PROFIBUS-Master und CM 1242-5 PROFIBUS-Slave CM 1242-2 AS-i-Master CP 1242-7 GPRS V2 CP 1243-7 LTE-EU CP 1243-1 DNP3 CP 1243-1 IEC CP 1243-1 CP1243-1 PCC CP 1243-8 ST7 RF120C	30	15	--
	TS (TeleService) Adapter IE Advanced [1] TS (TeleService)-Adapter IE Basic [1] TS-Adapter TS-Modul	 30 30	 15 15	 -- --

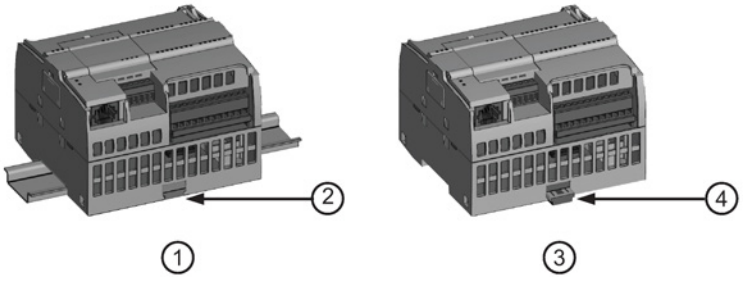
[1] Bevor Sie den TS (TeleService)-Adapter IE Advanced oder IE Basic einbauen, müssen Sie zunächst den TS-Adapter und ein TS-Modul anschließen. Die Gesamtbreite ("Breite A") beträgt 60 mm.

Alle CPUs, SMs, CMs und CPs können auf der DIN-Schiene oder im Schaltschrank montiert werden. Verwenden Sie die Hutschienenklemmen für die Befestigung des Geräts auf der Hutschiene. Diese Klemmen rasten auch in einer ausgezogenen Position ein, um den Einbau des Geräts in einer Schalttafel zu ermöglichen. Das Innenmaß der Bohrung für die Hutschienenklemmen am Gerät beträgt 4,3 mm.

Ober- und unterhalb des Geräts muss ein Freiraum von 25 mm für die Luftzirkulation als Schutz vor Überhitzung eingehalten werden.

Einbau und Ausbau der S71200 Geräte

Die CPU kann auf einfache Weise auf einer Standard-Hutschiene oder in einer Schalttafel eingebaut werden. Für die Befestigung des Geräts auf der Hutschiene werden passende Hutschienenklemmen mitgeliefert. Diese Klemmen können auch in einer ausgezogenen Position einrasten und ermöglichen dann das Festschrauben des Geräts in einer Schalttafel.



① Hutschienenmontage
② Standard-Hutschienenklemme in verriegelter Position
③ Schalttafelmontage
④ Klemme in ausgezogener Position für die Schalttafelmontage

Bevor Sie ein elektrisches Gerät einbauen oder ausbauen, müssen Sie sicherstellen, dass die Spannungsversorgung der Geräte ausgeschaltet ist. Achten Sie außerdem darauf, dass auch alle angeschlossenen Geräte ausgeschaltet sind.

⚠WARNUNG

Wenn Sie die S7-1200 oder daran angeschlossene Geräte in eingeschaltetem Zustand ein- oder ausbauen, kann es passieren, dass Sie einen elektrischen Schlag bekommen oder die Geräte unerwartet arbeiten.

Ist die Spannungsversorgung der S7-1200 und aller daran angeschlossenen Geräte während des Einbaus bzw. Ausbaus von Geräten nicht abgeschaltet, so kann dies aufgrund von elektrischem Schlag oder unerwartetem Betrieb der Geräte zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Treffen Sie alle notwendigen Sicherheitsvorkehrungen und vergewissern Sie sich, dass vor dem Einbau bzw. Ausbau eines Geräts die Spannungsversorgung der S71200 CPUs abgeschaltet ist.

Achten Sie immer darauf, dass Sie das richtige Modul bzw. das richtige Gerät verwenden, wenn Sie ein S7-1200 Gerät einbauen bzw. auswechseln.

⚠ WARNUNG

Falscher Einbau eines S7-1200 Moduls kann zu unvorhersehbarer Funktionsweise des Programms der S7-1200 führen.

Wird ein S7-1200 Gerät durch eine andere Variante ersetzt, nicht richtig ausgerichtet oder in der falschen Reihenfolge eingebaut, so kann dies aufgrund von unerwartetem Betrieb der Geräte zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Wechseln Sie ein S7-1200 Gerät immer mit der gleichen Ausführung aus, richten Sie das Gerät korrekt aus und bauen Sie es an der richtigen Stelle ein.

⚠ WARNUNG

Trennen Sie keine Geräte, wenn eine entflammbare oder brennbare Atmosphäre vorliegt.

Das Trennen von Geräten in einer entflammbaren oder brennbaren Atmosphäre kann ein Feuer oder eine Explosion verursachen, was zu Tod, schweren Verletzungen und/oder Sachschaden führen kann.

Achten Sie immer auf angemessene Sicherheitsmaßnahmen, wenn eine entflammbare oder brennbare Atmosphäre vorliegt.

Hinweis

Elektrostatische Entladungen können das Gerät oder den dafür vorgesehenen Schacht in der CPU beschädigen.

Sie müssen auf einem leitfähigen, geerdeten Boden stehen und/oder ein geerdetes Armband tragen, wenn Sie mit dem Gerät arbeiten.

4.3.2 Einbau und Ausbau der CPU

Sie können die CPU in einer Schalttafel oder auf einer Standard-Hutschiene einbauen.

Hinweis

Schließen Sie die Kommunikationsmodule an die CPU an und bauen Sie alle Module gemeinsam ein. Installieren Sie die Signalmodule erst nach dem Einbau der CPU.

Beachten Sie beim Einbau der Geräte auf der Hutschiene oder in einer Schalttafel Folgendes:

- Achten Sie beim Einbau in eine Hutschiene darauf, dass sich die obere Hutschienenklemme der CPU und der angeschlossenen CMs in der eingerasteten (inneren) Stellung und die untere Hutschienenklemme in der ausgezogenen Stellung befindet.
- Bringen Sie nach dem Einbau der Geräte auf der Hutschiene die unteren Hutschienenklemmen in die eingerastete Stellung, um die Geräte auf der Hutschiene zu befestigen.
- Achten Sie bei der Schalttafelmontage darauf, dass sich die Hutschienenklemmen in der ausgezogenen Stellung befinden.

Um die CPU in eine Schalttafel einzubauen, gehen Sie folgendermaßen vor:

1. Bringen Sie die Bohrungen (M4) mit den in den Abmessungen in der Tabelle Abmessungen für die Montage (mm) (Seite 56) gezeigten Vorgaben an.
2. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind.
3. Ziehen Sie die Montageklemmen aus dem Modul heraus. Achten Sie darauf, dass sich die Hutschienenklemmen oben und unten an der CPU in der ausgezogenen Position befinden.
4. Schrauben Sie das Modul mit einer Zylinderkopfschraube M4 sowie einem Federring und einer flachen Unterlegscheibe fest. Verwenden Sie keine Senkkopfschraube.

Hinweis

Welcher Schraubentyp erforderlich ist, hängt von der Art des Materials ab, auf dem das Modul montiert wird. Ziehen Sie die Schraube mit dem entsprechenden Drehmoment fest, bis der Federring flachgedrückt ist. Ziehen Sie die Schrauben nicht mit übermäßigem Drehmoment fest. Verwenden Sie keine Senkkopfschrauben.

Hinweis

In Umgebungen, in denen starke Schwingungen auftreten, oder bei vertikalem Einbau der CPU kann es nützlich sein, die CPU mit DIN-Schienenklemmen auf der Hutschiene zu sichern. Montieren Sie einen Endhalter (8WA1808 oder 8WA1805) auf der DIN-Schiene, um sicherzustellen, dass die Module miteinander verbunden bleiben. In Umgebungen, in denen starke Schwingungen auftreten, bietet die Schalttafelmontage der CPU einen besseren Schutz vor Schwingungen.

Tabelle 4- 2 Einbau der CPU auf einer DIN-Schiene

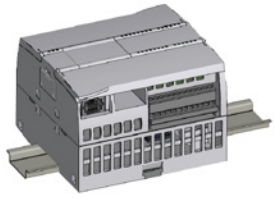
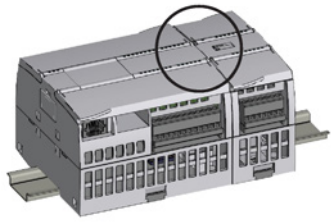
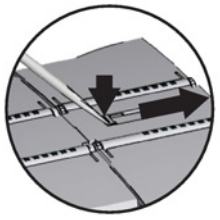
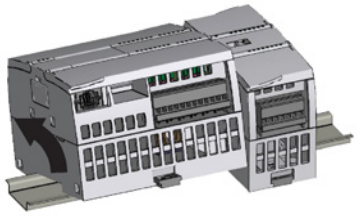
Aufgabenstellung	Vorgehensweise
	1. Montieren Sie die Hutschiene. Verschrauben Sie die Hutschiene in Abständen von jeweils 75 mm mit der Schalttafel. 2. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 3. Hängen Sie die CPU oben an der Hutschiene ein. 4. Ziehen Sie die Hutschienenklemme auf der Unterseite der CPU heraus, damit die CPU über die Schiene passt. 5. Schwenken Sie die CPU nach unten in ihre Einbauposition auf der Schiene. 6. Drücken Sie die Klemmen herunter, so dass die CPU fest auf der Schiene sitzt.

Tabelle 4- 3 Ausbau der CPU auf einer DIN-Schiene

Aufgabenstellung	Vorgehensweise
  	1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Trennen Sie die E/A-Steckverbinder, die Verdrahtung und Kabel von der CPU (Seite 67). 3. Bauen Sie die CPU und die zugehörigen Kommunikationsmodule gemeinsam aus. Alle Signalmodule müssen eingebaut bleiben. 4. Wenn ein SM an die CPU angeschlossen ist, ziehen Sie den Busstecker zurück: – Setzen Sie einen Schraubendreher an der Lasche an der Oberseite des Signalmoduls an. – Drücken Sie nach unten, um den Klemmenblock von der CPU zu lösen. – Schieben Sie die Lasche ganz nach rechts. 5. Nehmen Sie die CPU ab: – Ziehen Sie die Hutschienenklemme heraus, damit die CPU von der Schiene gelöst wird. – Schwenken Sie die CPU nach oben von der Schiene herunter und nehmen Sie die CPU aus dem System heraus.

4.3.3 Ein- und Ausbau eines SBs, CBs oder BBs

Tabelle 4- 4 Einbau eines SBs, CBs oder BBs 1297

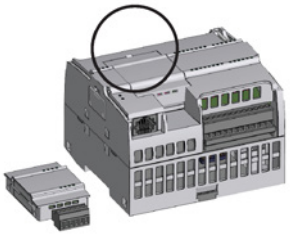
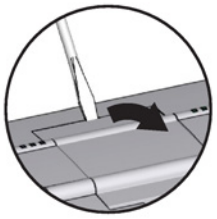
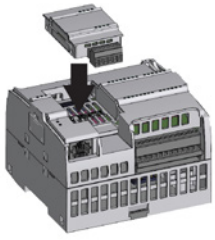
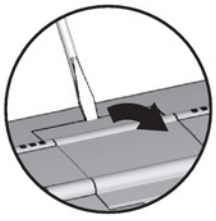
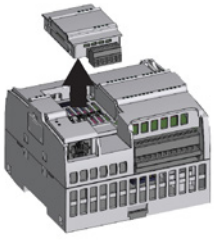
Aufgabenstellung	Vorgehensweise
  	1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Nehmen Sie die obere und untere Abdeckung des Klemmenblocks von der CPU ab. 3. Führen Sie einen Schraubendreher in den Schlitz oben auf der CPU an der Hinterseite der Abdeckung ein. 4. Hebeln Sie die Abdeckung vorsichtig aus ihrer Halterung und nehmen Sie sie von der CPU ab. 5. Setzen Sie das Modul gerade von oben in seine Montageposition in der CPU ein. 6. Drücken Sie das Modul fest herunter, bis es einrastet. 7. Setzen Sie die Klemmenabdeckungen wieder ein.

Tabelle 4- 5 Ausbau eines SBs, CBs oder BBs 1297

Aufgabenstellung	Vorgehensweise
 	1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Nehmen Sie die obere und untere Abdeckung des Klemmenblocks von der CPU ab. 3. Setzen Sie einen Schraubendreher in den Schlitz auf der Oberseite des Moduls ein. 4. Hebeln Sie das Modul vorsichtig aus seiner Halterung und nehmen Sie es von der CPU ab. 5. Nehmen Sie das Modul gerade von oben aus seiner Montageposition aus der CPU heraus. 6. Bringen Sie die Abdeckung der CPU wieder an. 7. Setzen Sie die Klemmenabdeckungen wieder ein.

Einsetzen oder Austauschen der Batterie des Batterieboards BB 1297

Das BB 1297 benötigt eine Batterie des Typs CR1025. Die Batterie ist nicht im Lieferumfang des BB 1297 enthalten und muss separat erworben werden. Zum Einbauen oder Austauschen der Batterie gehen Sie folgendermaßen vor:

1. Setzen Sie eine neue Batterie im BB 1297 mit der positiven Seite der Batterie nach oben und der negativen Seite neben der gedruckten Beschaltungsplatte ein.
2. Das BB 1297 kann jetzt in die CPU eingesetzt werden. Zur Installation des BB 1297 gehen Sie wie oben beschrieben vor.

So ersetzen Sie die Batterie des BBs 1297:

1. Bauen Sie das BB 1297 wie oben beschrieben aus der CPU aus.
2. Entnehmen Sie die alte Batterie vorsichtig mit Hilfe eines kleinen Schraubendrehers. Schieben Sie die Batterie unter der Klemme heraus.
3. Setzen Sie eine neue Batterie CR1025 mit der positiven Seite der Batterie nach oben und der negativen Seite neben der gedruckten Beschaltungsplatte ein.
4. Bauen Sie das Batterieboard BB 1297 wieder wie oben beschrieben ein.

⚠ WARNUNG

Die Installation einer nicht spezifizierten Batterie im BB 1297 oder der Anschluss einer nicht spezifizierten Batterie an den Schaltkreis auf andere Weise kann zu Brand oder Bauteilbeschädigung und unvorhersehbarem Betrieb von Maschinen führen.

Brand oder unvorhersehbarer Betrieb von Maschinen können tödliche oder schwere Verletzungen oder Sachschäden verursachen.

Verwenden Sie für die Pufferung der Echtzeituhr nur die spezifizierte Batterie CR1025.

4.3.4 Einbau und Ausbau eines SMs

Tabelle 4- 6 Einbau eines SMs

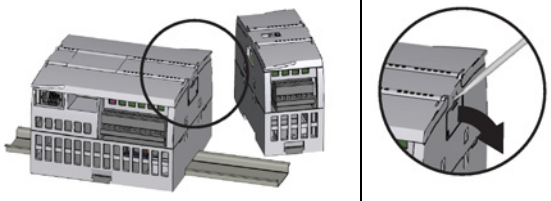
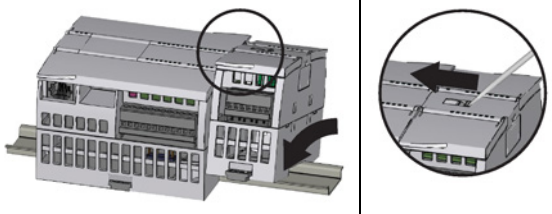
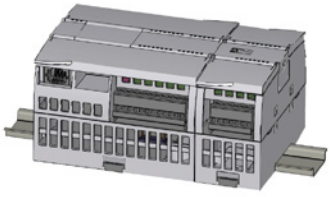
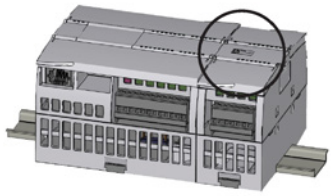
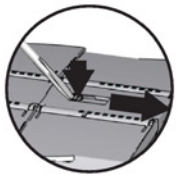
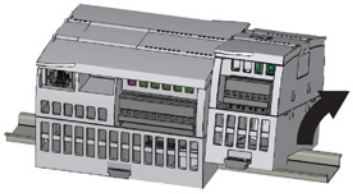
Aufgabenstellung	Vorgehensweise
	Bauen Sie Ihr SM nach der Montage der CPU ein. 1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Nehmen Sie die Abdeckung für den Anschluss an der rechten Seite der CPU ab: – Führen Sie einen Schraubendreher in den Schlitz über der Abdeckung ein. – Hebeln Sie die Abdeckung vorsichtig oben aus ihrer Halterung und nehmen Sie sie ab. 3. Bewahren Sie die Abdeckung zur späteren Verwendung auf.
	SM an die CPU anschließen: 1. Positionieren Sie das SM neben der CPU. 2. Hängen Sie das SM oben an der Hutschiene ein. 3. Ziehen Sie die untere Hutschienenklemme heraus, damit das SM über die Schiene passt. 4. Schwenken Sie das SM nach unten in die Position neben der CPU und drücken Sie die untere Klemme herunter, so dass das SM fest auf der Hutschiene sitzt.
	Der Busstecker stellt dadurch die mechanische und elektrische Verbindung für das SM her. 1. Setzen Sie einen Schraubendreher an der Lasche an der Oberseite des SM an. 2. Schieben Sie die Lasche ganz nach links, um den Busstecker in die CPU zu schieben. Gehen Sie genauso vor, um ein Signalmodul an einem Signalmodul einzubauen.

Tabelle 4- 7 Ausbau eines SMs

Aufgabenstellung	Vorgehensweise
 	Sie können jedes SM ausbauen, ohne die CPU oder andere SMs ausbauen zu müssen. 1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Ziehen Sie die E/A-Steckverbinder und die Verdrahtung vom SM (Seite 67) ab. 3. Ziehen Sie den Busstecker zurück. – Setzen Sie einen Schraubendreher an der Lasche an der Oberseite des SM an. – Drücken Sie nach unten, um den Klemmenblock von der CPU zu lösen. – Schieben Sie die Lasche ganz nach rechts. Ist auf der rechten Seite ein weiteres SM eingebaut, wiederholen Sie diese Vorgehensweise für das SM.
	Nehmen Sie das SM ab: 1. Ziehen Sie die untere Hutschienenklemme heraus, damit das SM von der Schiene gelöst wird. 2. Schwenken Sie das SM nach oben und von der Schiene herunter. Nehmen Sie das SM aus dem System heraus. 3. Bringen Sie ggf. eine Abdeckung am Busstecker der CPU an, um Verschmutzung zu vermeiden. Gehen Sie genauso vor, um ein Signalmodul von einem Signalmodul auszubauen.

4.3.5 Einbau und Ausbau eines CMs oder CPs

Schließen Sie die Kommunikationsmodule an die CPU an und bauen Sie alle Module gemeinsam ein. Dies wird unter Einbau und Ausbau der CPU (Seite 60) gezeigt.

Tabelle 4- 8 Einbau eines CMs oder CPs

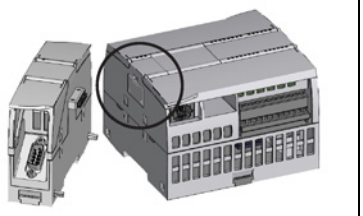
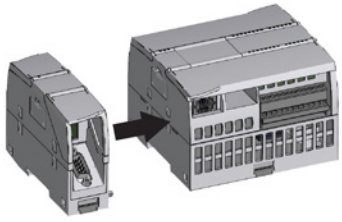
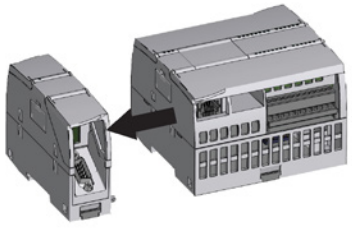
Aufgabenstellung	Vorgehensweise
 	1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Schließen Sie das CM an die CPU an und bauen Sie alle Module gemeinsam auf der Hutschiene oder in der Schalttafel ein. 3. Nehmen Sie die Busabdeckung an der linken Seite der CPU ab: – Führen Sie einen Schraubendreher in den Schlitz über der Busabdeckung ein. – Hebeln Sie die Abdeckung vorsichtig oben aus ihrer Halterung. 4. Nehmen Sie die Busabdeckung ab. Bewahren Sie die Abdeckung zur späteren Verwendung auf. 5. CM oder CP an die CPU anschließen: – Richten Sie den Busstecker und die Stifte des CMs zu den Bohrungen in der CPU aus. – Drücken Sie die Bauteile fest zusammen, bis die Stifte einrasten. 6. Bauen Sie die CPU und den CP auf einer DIN-Schiene oder in eine Schalttafel ein.

Tabelle 4- 9 Ausbau eines CMs oder CPs

Aufgabenstellung	Vorgehensweise
	Bauen Sie die CPU und das CM gemeinsam aus der Standard-Hutschiene bzw. der Schattafel aus. 1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Ziehen Sie die E/A-Steckverbinder, die Verdrahtung und die Kabel von der CPU und den CMs ab. 3. Bringen Sie bei der Hutschienenmontage die unteren Hutschienenklemmen an der CPU und den CMs in die ausgezogene Stellung. 4. Bauen Sie die CPU und die CMs aus der Standard-Hutschiene bzw. der Schattafel aus. 5. Halten Sie die CPU und die CMs fest und ziehen Sie sie auseinander.

ACHTUNG
Trennen Sie die Module ohne Einsatz eines Werkzeugs. Verwenden Sie zum Trennen der Module kein Werkzeug, da sonst die Teile beschädigt werden können.

4.3.6 Ausbau und Wiedereinbau des S7-1200 Klemmenblocks

Die CPU, das SB und SM verfügen über abnehmbare Steckverbinder, um die Verdrahtung zu vereinfachen.

Tabelle 4- 10 Ausbau des Klemmenblocks

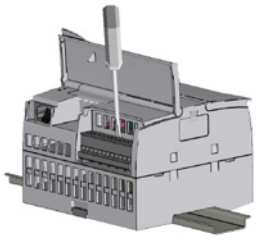
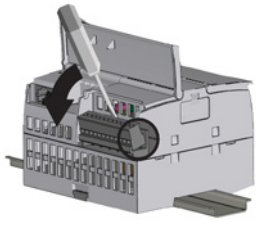
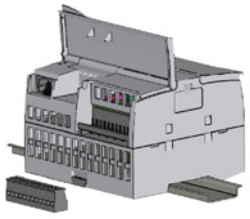
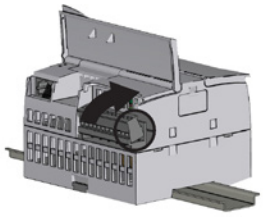
Aufgabenstellung	Vorgehensweise
	Bereiten Sie das System für den Ausbau des Klemmenblocks vor, indem Sie die Spannung der CPU ausschalten und die Abdeckung oberhalb des Klemmenblocks öffnen. 1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Suchen Sie auf der Oberseite des Klemmenblocks den Schlitz für die Spitze des Schraubendrehers. 3. Führen Sie einen Schraubendreher in den Schlitz ein. 4. Hebeln Sie die Oberseite des Klemmenblocks vorsichtig aus der Halterung in der CPU. Es ist ein Schnappgeräusch zu hören, wenn sich der Klemmenblock löst. 5. Fassen Sie den Klemmenblock mit der Hand und ziehen Sie ihn aus der CPU heraus.
	

Tabelle 4- 11 Einbau des Klemmenblocks

Aufgabenstellung	Vorgehensweise
 	Bereiten Sie die Komponenten für den Einbau des Klemmenblocks vor, indem Sie die Spannung der CPU ausschalten und die Abdeckung für den Steckverbinder öffnen. 1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Richten Sie ihn zu den Stiften im Gerät aus. 3. Richten Sie die Verdrahtungsseite des Klemmenblocks im Stecksockel richtig aus. 4. Drücken Sie den Klemmenblock mit einer Drehbewegung fest nach unten, bis er einrastet. Prüfen Sie, ob der Klemmenblock richtig ausgerichtet ist und fest sitzt.

4.3.7 Einbau und Ausbau des Erweiterungskabels

Das S7-1200 Erweiterungskabel sorgt für zusätzliche Flexibilität beim Konfigurieren des Layouts Ihres S7-1200 Systems. Pro CPU-System dürfen Sie ein Erweiterungskabel verwenden. Sie bauen das Erweiterungskabel entweder zwischen der CPU und dem ersten SM oder zwischen zwei SMs ein.

Tabelle 4- 12 Ein- und Ausbau des Steckers des Erweiterungskabels

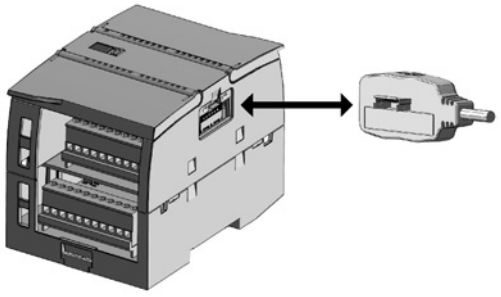
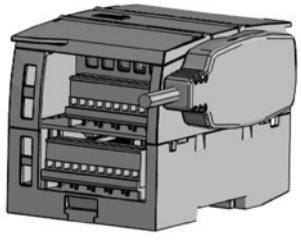
Aufgabenstellung	Vorgehensweise
 	Stecker einbauen: 1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Drücken Sie den Steckverbinder in den Busanschluss auf der rechten Seite des Signalmoduls oder der CPU. Stecker ausbauen: 1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Ziehen Sie den Stecker vom Signalmodul oder von der CPU ab.

Tabelle 4- 13 Einbau der Buchse des Erweiterungskabels

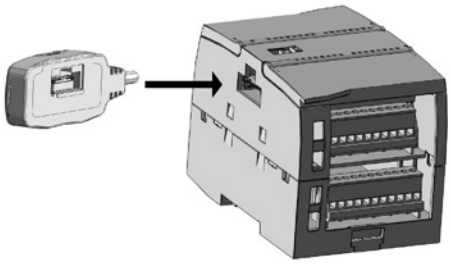
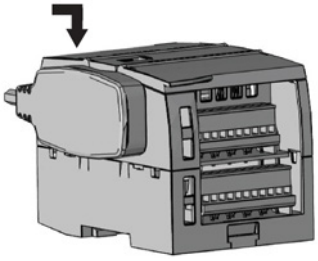
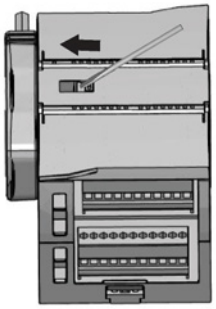
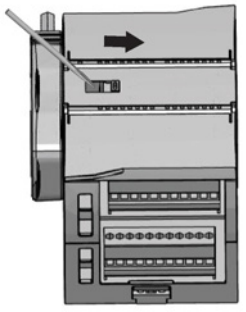
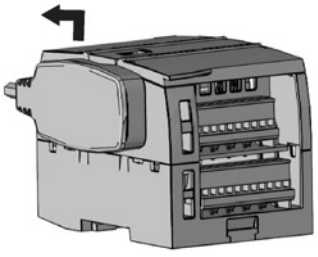
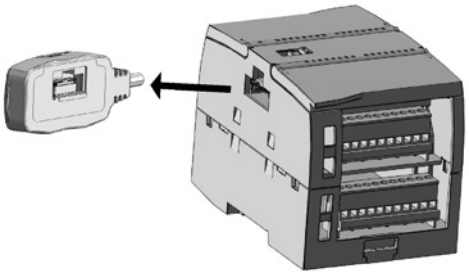
Aufgabenstellung	Vorgehensweise
  	1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Stecken Sie die Buchse auf den Busanschluss an der linken Seite des Signalmoduls. 3. Schieben Sie den Hakenfortsatz der Buchse in das Gehäuse am Busanschluss und drücken Sie vorsichtig, damit der Haken einrastet. 4. Befestigen Sie die Steckverbindung: – Setzen Sie einen Schraubendreher an der Lasche an der Oberseite des Signalmoduls an. – Schieben Sie die Lasche ganz nach links. Damit der Steckverbinder einrastet, müssen Sie die Lasche des Steckverbinders ganz nach links schieben. Die Steckverbinderlasche muss vollständig einrasten.

Tabelle 4- 14 Ausbau der Buchse des Erweiterungskabels

Aufgabenstellung	Vorgehensweise
  	1. Stellen Sie sicher, dass die CPU und alle S7-1200 Geräte von der elektrischen Leistung getrennt sind. 2. Lösen Sie den Steckverbinder: – Setzen Sie einen Schraubendreher an der Lasche an der Oberseite des Signalmoduls an. – Drücken Sie ihn leicht nach unten und schieben Sie die Lasche ganz nach rechts. 3. Heben Sie den Steckverbinder leicht an, damit sich der Haken löst. 4. Nehmen Sie die Buchse ab.

Hinweis

Einbau des Erweiterungskabels in schwingungsbelasteten Umgebungen

Wird das Erweiterungskabel an Module angeschlossen, die sich bewegen oder nicht fest montiert sind, kann sich die Einrastverbindung am Stecker des Kabels im Lauf der Zeit lockern.

Um für zusätzliche Zugentlastung zu sorgen, fixieren Sie in diesem Fall das steckerseitige Kabelende mit einem Kabelbinder auf der DIN-Schiene (oder an einem anderen Träger).

Vermeiden Sie übermäßige Kraftanwendung, wenn das Kabel während des Einbaus gezogen werden muss. Stellen Sie nach beendetem Einbau sicher, dass die Verbindung zwischen Kabel und Modul sich an der korrekten Position befindet.

4.3.8 TS (TeleService)-Adapter

4.3.8.1 Anschließen des TeleService-Adapters

Bevor Sie den TS (TeleService)-Adapter IE Basic oder TS (TeleService)-Adapter IE Advanced einbauen, müssen Sie zunächst den TS-Adapter und ein TS-Modul anschließen.

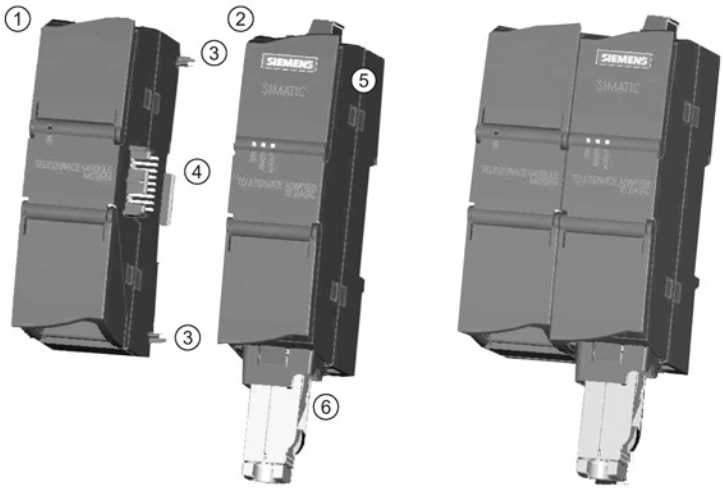
Verfügbare TS-Module:

- TS-Modul RS232
- TS-Modul Modem
- TS-Modul GSM
- TS-Modul ISDN

Hinweis

Das TS-Modul kann beschädigt werden, wenn Sie die Kontakte des Steckverbinders ④ des TS-Moduls berühren.

Befolgen Sie die EGB-Schutzmaßnahmen, um eine Beschädigung des TS-Moduls durch elektrostatische Entladung zu verhindern. Vergewissern Sie sich, bevor Sie ein TS-Modul und einen TS-Adapter anschließen, dass sich beide im Ruhezustand befinden.



①	TS-Modul	④	Steckverbinder des TS-Moduls
②	TS-Adapter	⑤	Kann nicht geöffnet werden
③	Elemente	⑥	Ethernet-Port

Hinweis

Stellen Sie vor dem Anschließen eines TS-Moduls und eines TS-Adapters sicher, dass die Kontaktstifte ④ nicht verbogen sind.

Stellen Sie beim Anschließen sicher, dass der Stecker und die Führungsstifte korrekt positioniert sind.

Schließen Sie nur ein TS-Modul an den TS-Adapter an. Stecken Sie den TS-Adapter nicht mit Gewalt auf ein anderes Gerät, z. B. eine S7-1200 CPU. Ändern Sie nicht die mechanische Konstruktion des Steckverbinders und entfernen oder beschädigen Sie nicht die Führungsstifte.

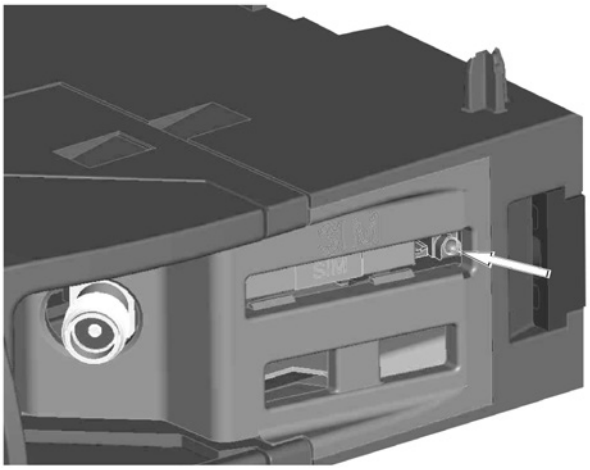
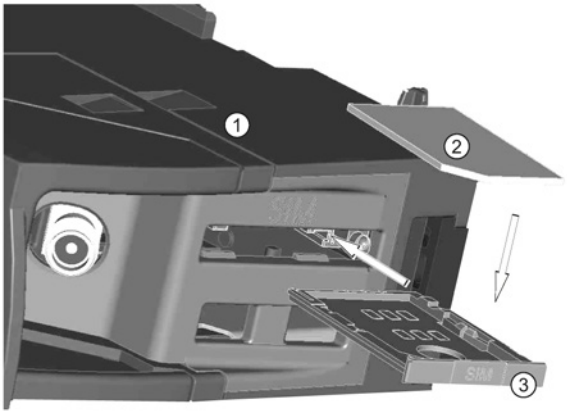
4.3.8.2 Einlegen der SIM-Karte

Lokalisieren Sie den SIM-Kartensteckplatz auf der Unterseite des TS module GSM.

Hinweis

Die SIM-Karte darf nur entnommen oder gesteckt werden, wenn das TS module GSM ausgeschaltet ist.

Tabelle 4- 15 Einlegen der SIM-Karte

Aufgabenstellung	Vorgehensweise
	Drücken Sie mit einem spitzen Objekt auf den Auswurfknopf des SIM-Kartenfachs (in Pfeilrichtung) und nehmen Sie das SIM-Kartenfach heraus.
	Legen Sie die SIM-Karte wie gezeigt in das SIM-Kartenfach ein und stecken Sie das SIM-Kartenfach wieder in den Steckplatz. ① TS Module GSM ② SIM-Karte ③ SIM-Kartenfach

Hinweis

Stellen Sie sicher, dass die SIM-Karte in der richtigen Richtung ins Kartenfach eingelegt wird. Andernfalls stellt die Karte keine Verbindung mit dem Modul her und das Kartenfach lässt sich möglicherweise nicht mehr über den Auswurfknopf herausnehmen.

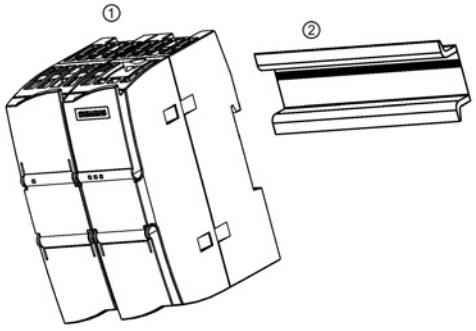
4.3.8.3 Einbau des TS-Adapters auf einer DIN-Schiene

Voraussetzungen: Sie müssen den TS-Adapter und das TS-Modul miteinander verbunden haben und die DIN-Schiene muss installiert sein.

Hinweis

Wenn Sie das TS-Gerät vertikal oder in einer Umgebung mit starken Schwingungen einbauen, kann sich das TS-Modul vom TS-Adapter lösen. Montieren Sie einen Endhalter 8WA1808 auf der DIN-Schiene, um sicherzustellen, dass die Module miteinander verbunden bleiben.

Tabelle 4- 16 Einbau und Ausbau des TS-Adapters

Aufgabenstellung	Vorgehensweise
	Einbau: 1. Haken Sie den TS-Adapter mit dem angeschlossenen TS-Modul ① auf der DIN-Schiene ein ②. 2. Drehen Sie das Gerät nach hinten, bis es einrastet. 3. Drücken Sie den DIN-Schienen-Clip an jedem Modul nach innen, um die Module auf der DIN-Schiene zu befestigen.
	Ausbau: 1. Ziehen Sie das Analogkabel und das Ethernet-Kabel von der Unterseite des TS-Adapters ab. 2. Schalten Sie die Spannungsversorgung des TS-Adapters aus. 3. Lösen Sie die DIN-Schienen-Clips an beiden Modulen mit einem Schraubendreher. 4. Drehen Sie das Gerät nach oben, um es von der DIN-Schiene abzunehmen.

⚠ WARNUNG

Sicherheitshinweise für den Ein- und Ausbau des TS-Adapters.

Bevor Sie die Versorgung des Geräts ausschalten, heben Sie die Erdung des TS-Adapters auf, indem Sie das Analogkabel und das Ethernet-Kabel abziehen. Wird dies nicht beachtet, kann es durch unerwartete Reaktion des Geräts zu tödlichen oder schweren Verletzungen und/oder Sachschäden kommen.

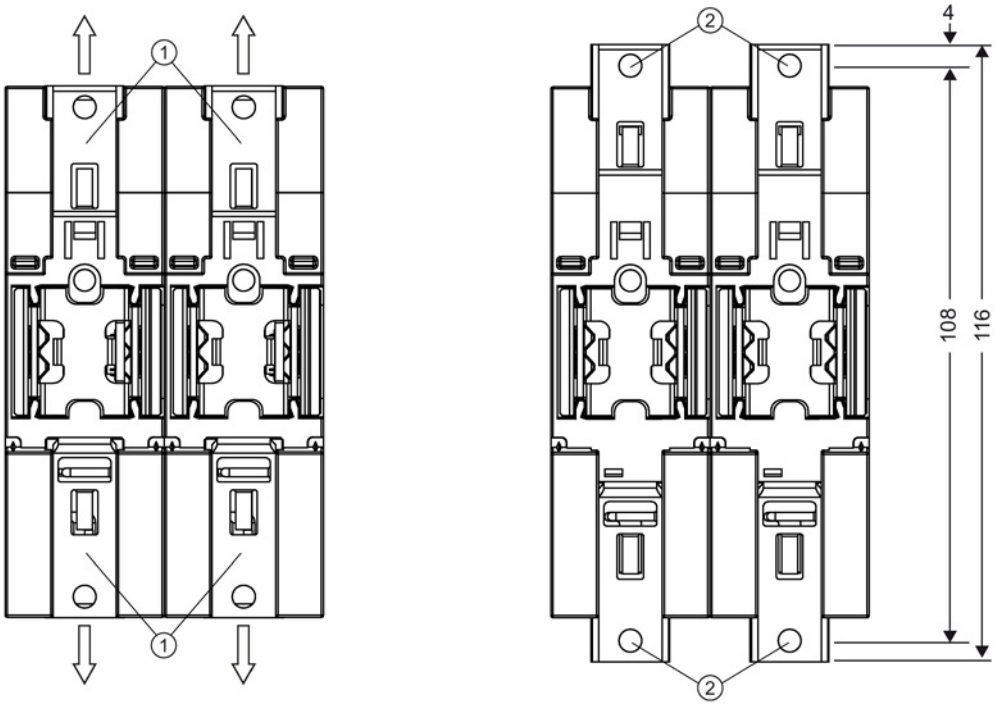
Beachten Sie bei jedem Ein- oder Ausbau des TS-Adapters diese Sicherheitshinweise.

4.3.8.4 Schalttafeleinbau des TS-Adapters

Voraussetzungen: Sie müssen den TS-Adapter und das TS-Modul miteinander verbunden haben.

1. Schieben Sie den Schieber für Anbauten ① zur Rückseite des TS-Adapters und des TS-Moduls in Pfeilrichtung, bis der Schieber einrastet.
2. Schrauben Sie den TS-Adapter und das TS-Modul an der mit ② gekennzeichneten Position an der gewünschten Montagewand fest.

Die folgende Abbildung zeigt den TS-Adapter von hinten, mit den Schiebern ① in beiden Positionen:



① Schieber für Anbauten
② Bohrungen zur Wandmontage

4.4 Verdrahtungsrichtlinien

Die ordnungsgemäße Erdung und Verdrahtung aller elektrischen Geräte ist wichtig für den optimalen Betrieb Ihres Systems und für die zusätzliche Störfestigkeit Ihrer Anwendung und der S71200. Beachten Sie die technischen Daten (Seite 1151) für die S7-1200-Schaltpläne.

Voraussetzung

Bevor Sie ein elektrisches Gerät erden oder verdrahten, müssen Sie sicherstellen, dass die Spannungsversorgung der Geräte ausgeschaltet ist. Achten Sie außerdem darauf, dass auch alle angeschlossenen Geräte ausgeschaltet sind.

Stellen Sie sicher, dass Sie bei der Verdrahtung der S7-1200 und aller angeschlossenen Geräte alle geltenden und verbindlichen Normen befolgen. Beachten Sie bei Einbau und Betrieb der Geräte die entsprechenden nationalen und regionalen Vorschriften. Erfragen Sie bei den Behörden vor Ort die Normen und Vorschriften, die für Ihren speziellen Fall zu befolgen sind.

⚠WARNUNG

Wenn Sie die S7-1200 oder daran angeschlossene Geräte in eingeschaltetem Zustand einbauen oder verdrahten, kann es passieren, dass Sie einen elektrischen Schlag bekommen oder die Geräte unerwartet arbeiten.

Ist die Spannungsversorgung der S7-1200 und aller daran angeschlossenen Geräte während des Einbaus bzw. Ausbaus von Geräten nicht abgeschaltet, so kann dies aufgrund von elektrischem Schlag oder unerwartetem Betrieb der Geräte zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Treffen Sie alle notwendigen Sicherheitsvorkehrungen und vergewissern Sie sich, dass vor dem Einbau bzw. Ausbau eines Geräts die Spannungsversorgung der S71200 abgeschaltet ist.

Denken Sie beim Planen von Erdung und Verdrahtung Ihres S7-1200 Systems immer an die Sicherheit. Elektronische Steuerungsgeräte wie die S7-1200 können ausfallen und dadurch unerwarteten Betrieb der gesteuerten oder beobachteten Geräte hervorrufen. Deshalb sollten Sie Sicherheitseinrichtungen implementieren, die von der S7-1200 unabhängig sind und vor möglichen Personen- und/oder Sachschäden schützen.

⚠WARNUNG

Steuerungen können bei unsicheren Betriebszuständen ausfallen und dadurch den unkontrollierten Betrieb der gesteuerten Geräte verursachen.

Unvorhersehbarer Betrieb kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Sorgen Sie daher für eine NOTAUSFunktion, elektromechanische oder andere redundante Sicherheitseinrichtungen, die von Ihrer S7-1200 unabhängig sind.

Richtlinien für die Potentialtrennung

Die Grenzwerte der AC-Spannungsversorgung und die E/A-Grenzen zu AC-Stromkreisen der S7-1200 sind dafür konzipiert und zugelassen, sichere elektrische Trennung zwischen AC-Leitungsspannungen und Niederspannungskreisen zu bieten. Je nach Norm umfassen diese Grenzen doppelte oder verstärkte Isolierung bzw. grundlegende plus zusätzliche Isolierung. Komponenten, die diese Grenzen kreuzen, z.B. optische Koppler, Kondensatoren, Transformatoren und Relais, haben die Zulassung, sichere elektrische Trennung zu bieten. Nur für AC-Leitungsspannung ausgelegte Schaltungen umfassen die sicherheitsbezogene Potenzialtrennung zu anderen Schaltungen. Die Potenzialtrennungsgrenzen zwischen 24-V-DC-Schaltungen sind rein funktional und Sie sollten sich bezüglich der Sicherheit nicht auf diese Grenzen verlassen.

Geberversorgungsausgang, Kommunikationsstromkreise und Stromkreise der internen Logik einer S7-1200 mit integrierter AC-Spannungsversorgung werden als Sicherheitskleinspannung (SELV) nach EN 61131-2 gespeist.

Um die sichere Eigenschaft der Niederspannungskreise der S7-1200 zu erhalten, müssen externe Anschlüsse an Kommunikationsports, analoge Stromkreise sowie sämtliche 24-V-Nennspannungsversorgung und alle E/A-Stromkreise aus zugelassenen Quellen gespeist werden, die die Anforderungen nach verschiedenen Normen für SELV, PELV, Klasse 2, spannungsbegrenzte oder leistungsbegrenzte Quellen erfüllen.

⚠WARNUNG

Der Einsatz nicht potentialgetrennter oder einfach isolierter Versorgungen für die Niederspannungskreise aus AC-Leitungen kann zu gefährlichen Spannungen in Kreisen führen, die als berührungssicher gelten, z.B. Kommunikationskreise oder Niederspannungsgeberverdrahtung.

Unerwartet hohe Spannungen können einen elektrischen Schlag verursachen, der zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen kann.

Verwenden Sie nur solche Hochspannungs-/Niederspannungswandler, die als Quellen berührungssicherer Kreise mit begrenzter Spannung zugelassen sind.

Richtlinien für die Erdung der S7-1200

Am besten erden Sie Ihre Anwendung, indem Sie darauf achten, dass alle gemeinsamen Anschlüsse und alle Erdanschlüsse Ihrer S7-1200 und aller angeschlossenen Geräte an einer einzigen Stelle geerdet werden. Diese Stelle muss direkt mit der Systemerde verbunden werden.

Erdleitungen sollten möglichst kurz sein und einen großen Aderquerschnitt, z. B. 2 mm^2 (14 AWG) haben.

Beachten Sie beim Auswählen von Erdungspunkten die entsprechenden Sicherheitsvorschriften, und stellen Sie die einwandfreie Funktion von stromkreisunterbrechenden Schutzeinrichtungen sicher.

Richtlinien für die Verdrahtung der S71200

Wenn Sie die Verdrahtung Ihrer S7-1200 planen, richten Sie einen Einzeltrennschalter ein, der gleichzeitig die Spannung der Spannungsversorgung für die S7-1200 CPU, die Spannung aller Eingangskreise und die Spannung aller Ausgangskreise trennt. Sorgen Sie für Überstromschutz, z. B. durch eine Sicherung oder einen Schutzschalter, um Fehlerstrom in der Versorgungsverdrahtung zu begrenzen. Sie können zusätzlichen Schutz durch Sicherungen oder andere Strombegrenzungen in den einzelnen Ausgangskreisen implementieren.

Versehen Sie blitzschlaggefährdete Leitungen mit einem geeigneten Überspannungsschutz. Weitere Informationen finden Sie im Abschnitt "Allgemein" der technischen Daten unter Störfestigkeit gegen Stoßspannungen (Seite 1151).

Vermeiden Sie es, Niederspannungssignalleitungen und Kommunikationskabel in der gleichen Kabelbahn wie AC-Versorgungsleitungen und schnellschaltende Hochgeschwindigkeits-DC-Leitungen zu verlegen. Leitungen sollten Sie paarweise verlegen: den Neutral oder Nullleiter zusammen mit dem Phasenleiter oder der Signalleitung.

Verlegen Sie Leitungen so kurz wie möglich und achten Sie darauf, dass der Leitungsquerschnitt dem benötigten Strom entspricht. Am Klemmenblock von CPU und SM können Drahtquerschnitte von 2 mm^2 bis 0,3 mm^2 (14 AWG bis 22 AWG) angeschlossen werden. Isolieren Sie die Ader auf 6,4 mm ab. Am Klemmenblock des SBs können Drahtquerschnitte von 1,3 mm^2 bis 0,3 mm^2 (16 AWG bis 22 AWG) angeschlossen werden. Isolieren Sie die Ader auf 6,3 bis 7 mm ab.

Leitungen und Kabel müssen für eine Temperatur von 30 °C höher als die Umgebungstemperatur der S7-1200 ausgelegt sein (Beispiel: Leitungen, die bei einer Umgebungstemperatur von 55 °C für mindestens 85 °C ausgelegt sind). Ermitteln Sie andere Verdrahtungsarten und Materialanforderungen anhand der spezifischen Bemessungswerte der elektrischen Schaltungen und Ihrer Installationsumgebung.

Verwenden Sie als optimalen Schutz vor Störfestigkeit geschirmte Leitungen. Typischerweise erreichen Sie durch Erdung des Schirms an der S7-1200 die besten Ergebnisse. Sie müssen den Schirm von Kommunikationskabeln mit S7-1200 Kommunikationssteckergehäusen über Steckverbinder erden, die leitend mit dem Kabelschirm verbunden sind, oder indem Sie den Schirm des Kommunikationskabels an separate Erde anschließen. Erden Sie andere Kabelschirme mittels Klemmen oder Kupferband um den Schirm, um eine große Oberfläche mit dem Erdungspunkt zu verbinden.

Wenn Sie einen Eingangskreis verdrahten, der durch eine externe Spannungsversorgung versorgt wird, implementieren Sie ein Überstromschutzgerät in dem Eingangskreis. Bei Kreisen, die durch die 24-V-DC-Geberversorgung der S71200 versorgt werden, ist kein externer Schutz erforderlich, weil die Geberversorgung bereits strombegrenzt ist.

Alle S7-1200 Module verfügen über steckbare Klemmenblöcke für die Anwenderverdrahtung. Achten Sie darauf, dass der Klemmenblock fest sitzt und die Leitungen fest im Klemmenblock verdrahtet sind, damit es nicht zu losen Anschlüssen kommt. Siemens empfiehlt, die Drahtisolierung auf etwa 6 mm zu entfernen, um einen sicheren Anschluss zu gewährleisten. Ziehen Sie die Schrauben jedoch nicht zu fest an, damit Sie den Klemmenblock nicht beschädigen. Das maximale Drehmoment für die Schrauben des Klemmenblocks von CPU und SM beträgt 0,56 Nm. Das maximale Drehmoment für die Schrauben des Klemmenblocks von SB, Simulator und Potentiometer beträgt 0,33 Nm.

Die S7-1200 arbeitet mit Potentialtrennungsgrenzen, wodurch das Entstehen unerwünschter Ströme in Ihrer Anlage verhindert wird. Wenn Sie die Verdrahtung für Ihr System planen, berücksichtigen Sie diese Potentialtrennungsgrenzen. Im Kapitel Technische Daten (Seite 1151) finden Sie die Werte für die zur Verfügung gestellte Potentialtrennung und die Anordnung der Potentialtrennungsgrenzen. Für AC-Leitungsspannung ausgelegte Schaltungen umfassen die sicherheitsbezogene Potenzialtrennung zu anderen Schaltungen. Die Potenzialtrennungsgrenzen zwischen 24-V-DC-Schaltungen sind rein funktional und Sie sollten sich bezüglich der Sicherheit nicht auf diese Grenzen verlassen.

Siehe auch

Allgemeine technische Daten und Leistungsmerkmale (Seite 1213)

Richtlinien für Lampenlasten

Lampenlasten schädigen Relaiskontakte aufgrund des hohen Einschaltstoßstroms. Dieser Stoßstrom ist nominal 10 bis 15 Mal so hoch wie der stationäre Strom einer Wolframlampe. Für Lampenlasten, die während der Lebensdauer der Anwendung sehr häufig geschaltet werden, wird ein austauschbares Koppelrelais oder ein Stoßstrombegrenzer empfohlen.

Richtlinien für induktive Lasten

Verwenden Sie bei induktiven Lasten Schutzbeschaltungen, um den Spannungsanstieg beim Ausschalten eines Steuerungsausgangs zu begrenzen. Schutzbeschaltungen schützen die Ausgänge vor frühzeitigem Ausfall aufgrund der Hochspannungstransiente, die auftritt, wenn der Stromfluss durch eine induktive Last unterbrochen wird.

Außerdem begrenzen Schutzbeschaltungen die elektrischen Störungen, die beim Schalten induktiver Lasten entstehen. Hochfrequente Störungen aus schlecht entstörten induktiven Lasten können den Betrieb des PLCs unterbrechen. Am effektivsten verringern Sie elektrische Störungen durch Anordnen einer externen Schutzbeschaltung parallel zur Last und in der Nähe der Last.

Die DC-Ausgänge der S7-1200 umfassen interne Schutzbeschaltungen, die für die induktiven Lasten in den meisten Anwendungen adäquat sind. Da die Relais-Ausgangskontakte der S7-1200 zum Schalten einer DC-Last oder einer AC-Last verwendet werden können, ist kein interner Schutz vorhanden.

Eine gute Lösung ist die Verwendung von Schützen und anderen induktiven Lasten, für die Schutzbeschaltungen angeboten werden, die vom Hersteller ins Lastgerät integriert oder als optionales Zubehör erhältlich sind. Einige vom Hersteller gelieferte Schutzbeschaltungen sind für Ihre Anwendung jedoch möglicherweise ungeeignet. Gegebenenfalls ist eine zusätzliche Schutzbeschaltung notwendig, um eine optimale Rauschminderung und die optimale Lebensdauer der Kontakte zu erreichen.

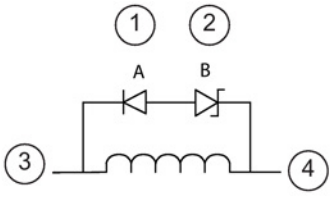
Bei AC-Lasten kann ein Metalloxidvaristor (MOV) oder ein anderer Spannungsbegrenzer mit einer parallelen RC-Beschaltung verwendet werden, diese Vorrichtungen alleine sind jedoch nicht so effektiv. Ein MOV ohne parallele RC-Beschaltung führt häufig zu erheblichen Hochfrequenzstörungen bis zur Höhe der Klemmspannung.

Eine gut beherrschte Ausschalttransiente hat eine Ringfrequenz von maximal 10 kHz, bevorzugt weniger als 1 kHz. Die Spitzenspannung für AC-Leitungen muss innerhalb von +/- 1200 V von Masse liegen. Die negative Spitzenspannung für DC-Lasten mit interner PLC-Schutzbeschaltung liegt ~40 V unterhalb der 24-V-DC-Versorgungsspannung. Eine externe Schutzbeschaltung muss die Transiente auf einen Wert innerhalb von 36 V der Versorgung begrenzen, um die interne Schutzbeschaltung zu entlasten.

Hinweis

Die Effektivität einer Schutzbeschaltung hängt von der jeweiligen Anwendung ab und muss immer für den Einzelfall geprüft werden. Achten Sie darauf, dass alle Komponenten richtig bemessen sind, und beobachten Sie die Ausschalttransiente mittels eines Oszilloskops.

Typische Schutzbeschaltungen für DC- oder Relaisausgänge, die induktive DC-Lasten schalten



① Diode 1N4001 oder gleichwertig

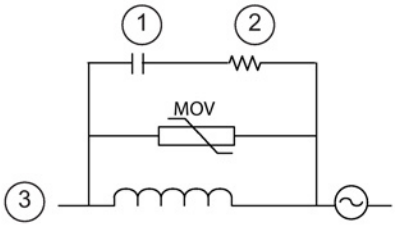
② Zener-Diode 8,2 V (DC-Ausgänge),
Zener-Diode 36 V (Relaisausgänge)

③ Ausgang

④ M, 24-V-Referenz

In den meisten Anwendungen ist der zusätzliche Einsatz einer Diode (A) quer zur induktiven DC-Last geeignet, doch wenn für Ihre Anwendung schnellere Ausschaltzeiten erforderlich sind, ist der Einsatz einer Zener-Diode (B) empfehlenswert. Bemessen Sie die Zener-Diode gemäß dem Strom im Ausgangskreis.

Typische Schutzbeschaltungen für Relaisausgänge, die induktive AC-Lasten schalten



① Den Wert für C finden Sie in der Tabelle

② Den Wert für R finden Sie in der Tabelle

③ Ausgang

Achten Sie darauf, dass die Arbeitsspannung des Metalloxidvaristors (MOV) mindstens 20 % höher ist als die Nennspannung.

Verwenden Sie für Impulsanwendungen empfohlene impulsbemessene, nicht induktive Widerstände und Kondensatoren (üblicherweise Metallschicht). Achten Sie darauf, dass die Komponenten die Anforderungen an die durchschnittliche Leistung, Spitzenleistung und Spitzenspannung erfüllen.

Wenn Sie Ihre eigene Schutzbeschaltung entwerfen, finden Sie in der folgenden Tabelle Werte für Widerstand und Kondensator für eine Vielzahl von AC-Lasten. Diese Werte basieren auf Berechnungen mit idealen Komponentenparametern. Die Angabe I effektiv in der Tabelle bezieht sich auf den Beharrungsstrom der Last im vollständig eingeschalteten Zustand.

Tabelle 4- 17 Widerstands- und Kondensatorwerte für AC-Schutzbeschaltungen

Induktive Last			Schutzbeschaltungswerte		
I effektiv	230 V AC	120 V AC	Widerstand		Kondensator
A	VA	VA	Ω	W (Nennleistung)	nF
0.02	4.6	2.4	15000	0.1	15
0.05	11.5	6	5600	0.25	470
0.1	23	12	2700	0.5	100
0.2	46	24	1500	1	150
0.05	115	60	560	2.5	470
1	230	120	270	5	1000
2	460	240	150	10	1500

Von den Tabellenwerten erfüllte Bedingungen:

Maximaler Ausschalttransitionsschritt < 500 V
Spitzenspannung des Widerstands < 500 V
Spitzenspannung des Kondensators < 1250 V
Strom der Schutzbeschaltung < 8 % des Laststroms (50 Hz)
Strom der Schutzbeschaltung < 11 % des Laststroms (60 Hz)
Kondensator dV/dt < 2 V/µs
Impulsverlust des Kondensators: $\int(dv/dt)^2\ dt < 10000\ V^2/\mu s$
Resonanzfrequenz < 300 Hz
Widerstandsleistung bei max. 2 Hz Schaltfrequenz
Angenommener Leistungsfaktor von 0,3 bei typischen induktiven Lasten

Richtlinien für Differenzein- und -ausgänge

Differenzeingänge und -ausgänge verhalten sich anders als Standardein- und -ausgänge. Jeder Differenzein- und -ausgang besitzt zwei Anschlussstifte. Um festzustellen, ob ein Differenzeingang oder -ausgang ein- oder ausgeschaltet ist, muss die Spannungsdifferenz zwischen diesen beiden Anschlussstiften gemessen werden.

Beachten Sie die ausführlichen technischen Daten zur CPU 1217C in Anhang A (Seite 1213).

SPS-Grundlagen

5

5.1 Ausführung des Anwenderprogramms

Die CPU unterstützt die folgenden Bausteinarten für den Aufbau einer geeigneten Struktur Ihres Anwenderprogramms:

- Organisationsbausteine (OBs) legen die Struktur des Programms fest. Für einige OBs gibt es vordefiniertes Verhalten und Startereignisse, Sie können aber auch OBs mit Ihren eigenen Startereignissen anlegen.
- Funkionen (FCs) und Funktionsbausteine (FBs) enthalten den Programmcode, der den jeweiligen Aufgaben oder Parametrierungen entspricht. Jede FC bzw. jeder FB stellt eine Anzahl Ein- und Ausgangsparameter für die gemeinsame Nutzung der Daten mit dem aufrufenden Baustein bereit. Ein FB verwendet ferner einen weiteren Datenbaustein (den Instanz-DB) für die Speicherung von Datenwerten für die jeweilige Instanz des FB-Aufrufs. Sie können einen FB mehrere Male aufrufen, jedes Mal mit einem eindeutigen Instanz-DB. Aufrufe desselben FB mit unterschiedlichen Instanz-DBs wirken sich nicht auf die Datenwerte in einem der anderen Instanz-DBs aus.
- Datenbausteine (DBs) speichern Daten, die von den Programmbausteinen verwendet werden können.

Die Ausführung des Anwenderprogramms beginnt mit einem oder mehreren optionalen Anlauf-OBs, die nach dem Wechsel in den Betriebszustand RUN einmal abgearbeitet werden, gefolgt von einem oder mehreren Zyklus-OBs, die zyklisch abgearbeitet werden. Sie können einen OB auch einem Alarmereignis zuordnen, bei dem es sich um ein Standard- oder ein Fehlerereignis handeln kann. Diese OBs werden ausgeführt, wenn das entsprechende Standard- oder Fehlerereignis auftritt.

Eine Funktion (FC) oder ein Funktionsbaustein (FB) ist ein Baustein mit Programmcode, der aus einem OB oder einer anderen FC oder einem FB aufgerufen werden kann. Folgende Schachtelungstiefen sind dabei möglich:

- 16 aus Zyklus- oder Anlauf-OBs
- 6 aus beliebigen Alarmereignis-OBs

FCs sind keinem bestimmten Datenbaustein (DB) zugeordnet. FBs sind direkt mit einem DB verbunden und nutzen diesen für die Übergabe von Parametern und die Speicherung von Zwischenwerten und -ergebnissen.

Die Größe des Anwenderprogramms, der Daten und der Konfiguration ist durch den verfügbaren Ladespeicher und den Arbeitsspeicher in der CPU begrenzt. Die Anzahl der einzelnen OBs, FCs, FBs und DBs ist nicht begrenzt. Die Gesamtzahl der Bausteine darf jedoch 1024 nicht überschreiten.

Jeder Zyklus umfasst das Schreiben der Ausgänge, das Lesen der Eingänge, das Bearbeiten der Anweisungen des Anwenderprogramms und die Durchführung der Hintergrundverarbeitung. Der Zyklus wird auch als Abtastzyklus oder Abtastung bezeichnet.

Ihre S7-1200 Automatisierungslösung kann aus einem zentralen Baugruppenträger mit der S7-1200 CPU und weiteren Modulen bestehen. Der Begriff "zentraler Baugruppenträger" bezieht sich auf den Hutschienen- oder Schaltschrankeinbau der CPU und der zugehörigen Module. Die Module (SM, SB, BB, CB, CM oder CP) werden erkannt und erst beim Anlauf angemeldet.

- Das Stecken oder Ziehen eines Moduls im zentralen Baugruppenträger bei eingeschaltetem Gerät ist nicht möglich. Stecken oder ziehen Sie niemals ein Modul im zentralen Baugruppenträger, wenn die CPU eingeschaltet ist.

  ⚠ WARNUNG

  Sicherheitsanforderungen zum Stecken oder Ziehen von Modulen

  Das Stecken oder Ziehen eines Moduls (SM, SB, BB, CD, CM oder CP) im zentralen Baugruppenträger bei eingeschalteter CPU kann unvorhersehbares Verhalten verursachen, was wiederum zu Sachschaden und/oder Verletzungen führen kann.

  Trennen Sie die Spannungsversorgung von CPU und zentralem Baugruppenträger und befolgen Sie die entsprechenden Sicherheitsvorkehrungen, bevor Sie ein Modul im zentralen Baugruppenträger stecken oder ziehen.

- Eine SIMATIC Memory Card können Sie bei eingeschalteter CPU stecken oder ziehen. Das Stecken oder Ziehen einer Memory Card im Betriebszustand RUN der CPU verursacht jedoch, dass die CPU in STOP geht.

  ACHTUNG

  Risiken beim Entfernen der Memory Card, wenn sich die CPU in RUN befindet

  Durch Stecken oder Ziehen einer Memory Card im Betriebszustand RUN der CPU geht die CPU in den Betriebszustand STOP, was zu Sachschaden an den gesteuerten Geräten oder im gesteuerten Prozess führen kann.

  Wenn Sie eine Memory Card stecken oder ziehen, geht die CPU sofort in den Betriebszustand STOP. Stellen Sie vor dem Stecken oder Ziehen einer Memory Card stets sicher, dass die CPU nicht aktiv eine Maschine oder einen Prozess steuert. Installieren Sie immer einen NOT-AUS-Schaltkreis für Ihre Anwendung bzw. Ihren Prozess.

- Wenn Sie in einem dezentralen E/A-Baugruppenträger (AS-i, PROFINET oder PROFIBUS) im Betriebszustand RUN der CPU ein Modul stecken oder ziehen, generiert die CPU einen Eintrag im Diagnosepuffer, führt den OB "Ziehen oder Stecken von Modulen" aus (sofern vorhanden) und bleibt standardmäßig im Betriebszustand RUN.

Aktualisierung von Prozessabbildern und Teilprozessabbildern

Die CPU aktualisiert die lokalen digitalen und analogen Ein- und Ausgänge synchron zum Zyklus mit einem internen Speicherbereich, dem so genannten Prozessabbild. Das Prozessabbild enthält ein Momentabbild der physischen Ein- und Ausgänge (physische E/A von CPU, Signalboard und Signalmodulen).

Sie können die E/A so einrichten, dass sie in jedem Zyklus im Prozessabbild aktualisiert werden oder immer dann, wenn ein spezifisches Alarmereignis auftritt. Sie können die E/A auch so einrichten, dass sie aus der Aktualisierung des Prozessabbilds ausgeschlossen werden. Beispiel: Ihr Prozess benötigt bestimmte Datenwerte möglicherweise immer dann, wenn ein Ereignis wie ein Prozessalarm auftritt. Indem Sie die Aktualisierung des Prozessabbilds für diese E/A einem Teilprozessabbild zuweisen, das Sie wiederum einem Prozessalarm-OB zuweisen, aktualisiert die CPU diese Datenwerte nicht unnötigerweise in jedem Zyklus, da Ihr Prozess die ständige Aktualisierung nicht erfordert.

Für E/A, die in jedem Zyklus aktualisiert werden, führt die CPU in jedem Zyklus die folgenden Aufgaben durch:

- Die CPU schreibt die Ausgänge aus dem Prozessabbild der Ausgänge in die physischen Ausgänge.
- Die CPU liest die physischen Eingänge unmittelbar vor der Ausführung des Anwenderprogramms und speichert die Eingangswerte im Prozessabbild der Eingänge. Diese Werte bleiben somit während der Ausführung der Anwenderanweisungen konsistent.
- Die CPU führt die Logik der Anwenderanweisungen durch und aktualisiert die Ausgangswerte im Prozessabbild der Ausgänge, statt in die tatsächlichen physischen Ausgänge zu schreiben.

  Dieser Vorgang sorgt während der gesamten Ausführung der Anwenderanweisungen in dem jeweiligen Zyklus für eine konsistente Logik und verhindert ein Pendeln der physischen Ausgänge mit mehrmaligen Zustandswechseln im Prozessabbild der Ausgänge.

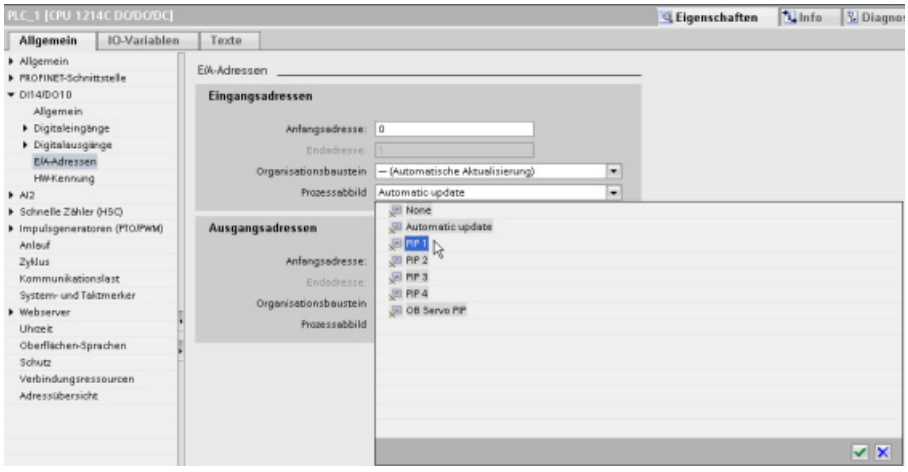
Um zu steuern, ob Ihr Prozess die E/A automatisch in jedem Zyklus oder bei der Auslösung von Ereignissen aktualisiert, stellt die S7-1200 fünf Teilprozessabbilder bereit. Das erste Teilprozessabbild, TPA0, ist für E/A bestimmt, die automatisch in jedem Zyklus aktualisiert werden. Hierbei handelt es sich um die Standardzuweisung. Den übrigen vier Teilprozessabbildern, TPA1, TPA2, TPA3 und TPA4, können Sie die Aktualisierung des Prozessabbilds bei verschiedenen Alarmereignissen zuweisen. Die E/A weisen Sie den Teilprozessabbildern in der Gerätekonfiguration zu, und die Teilprozessabbilder wiederum weisen Sie den Alarmereignissen zu, wenn Sie Alarm-OBs erstellen (Seite 182) oder OB-Eigenschaften bearbeiten (Seite 182).

Standardmäßig stellt STEP 7 beim Einfügen eines Moduls in der Gerätesicht für die Aktualisierung des Prozessabbilds die Option "Automatische Aktualisierung" ein. Bei E/A, für die die "Automatische Aktualisierung" vorgegeben ist, bearbeitet die CPU den Datenaustausch zwischen dem Modul und dem Prozessabbild automatisch in jedem Zyklus.

Um digitale oder analoge E/A einem Teilprozessabbild zuzuweisen oder um E/A aus der Aktualisierung des Prozessabbilds auszuschließen, gehen Sie wie folgt vor:

1. Öffnen Sie für das entsprechende Gerät das Register "Eigenschaften" in der Gerätekonfiguration.
2. Erweitern Sie ggf. die Auswahl unter "Allgemein", um die gewünschten E/A anzuzeigen.

3. Wählen Sie "E/A-Adressen".
4. Wählen Sie optional einen spezifischen OB aus der Klappliste "Organisationsbaustein" aus.
5. Ändern Sie über die Klappliste "Prozessabbild" die Einstellung "Automatische Aktualisierung" in "TPA1", "TPA2", "TPA3", "TPA4" oder "Keine". Die Auswahl "Keine" bedeutet, dass Sie diese E/A nur über direkte Anweisungen lesen und schreiben können. Um die Eingänge und Ausgänge wieder in die automatische Aktualisierung des Prozessabbilds aufzunehmen, stellen Sie für diese Option wieder "Automatische Aktualisierung" ein.



Sie können bei Ausführung einer Anweisung die Werte physischer Eingänge direkt lesen und auch direkt Werte in physische Ausgänge schreiben. Beim direkten Lesen wird auf den aktuellen Zustand des physischen Eingangs zugegriffen. Das Prozessabbbild der Eingänge wird, unabhängig davon, ob der Eingang für die Speicherung im Prozessabbild konfiguriert ist, nicht aktualisiert. Beim direkten Schreiben in den physischen Ausgang werden sowohl das Prozessabbild der Ausgänge (sofern der Ausgang für die Speicherung im Prozessabbild konfiguriert ist) als auch der physische Ausgang aktualisiert. Hängen Sie die Endung ":P" an die E/A-Adresse an, wenn Sie möchten, dass das Programm direkt über den physischen Eingang bzw. Ausgang auf die E/A-Daten zugreift und nicht über das Prozessabbild.

Hinweis

Verwenden von Teilprozessabbildern

Wenn Sie einem der Teilprozessabbilder TPA1 bis TPA4 einige E/A zuweisen, dem Teilprozessabbild jedoch keinen OB zuweisen, dann aktualisiert die CPU diese E/A niemals im Prozessabbild. Die Zuweisung von E/A zu einem TPA ohne entsprechende OB-Zuweisung kommt der Einstellung des Prozessabbilds für "Keine" Aktualisierung gleich. Sie können die E/A mit einer direkten Leseanweisung direkt aus den physischen E/A lesen oder mit einer direkten Schreibanweisung direkt in die physische E/A schreiben. Die CPU aktualisiert das Prozessabbild nicht.

Die CPU unterstützt dezentrale E/A in PROFINET- und PROFIBUS-Netzen (Seite 631).

5.1.1 Betriebszustände der CPU

Die CPU hat drei Betriebszustände: Betriebszustand STOP, Betriebszustand STARTUP und Betriebszustand RUN. Die Status-LEDs auf der Vorderseite der CPU geben den aktuellen Betriebszustand an.

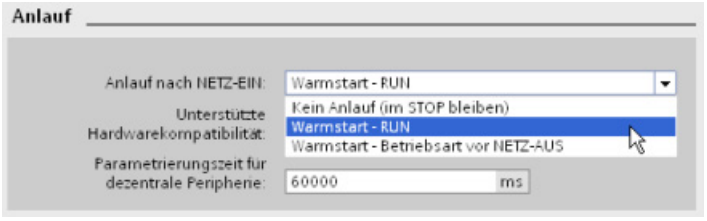
- Im Betriebszustand STOP führt die CPU das Programm nicht aus. Sie können ein Projekt in die CPU laden.
- Im Betriebszustand STARTUP werden die Anlauf-OBs (sofern vorhanden) einmal abgearbeitet. In diesem Betriebszustand verarbeitet die CPU keine Prozessalarmereignisse.
- Im Betriebszustand RUN werden die Programmzyklus-OBs wiederholt ausgeführt. Es können Alarmereignisse auftreten und die entsprechenden Alarmereignis-OBs können jederzeit im Betriebszustand RUN ausgeführt werden. Sie können Teile eines Projekts im Betriebszustand RUN laden (Seite 1138).

Die CPU unterstützt den Warmstart, um in den Betriebszustand RUN zu gehen. Während eines Warmstarts wird kein Urlöschen ausgeführt. Die CPU initialisiert alle nicht remanenten System- und Anwenderdaten bei einem Warmstart und puffert die Werte aller remanenten Anwenderdaten.

Beim Urlöschen werden der Arbeitsspeicher sowie alle remanenten und nicht remanenten Speicherbereiche gelöscht, der Ladespeicher in den Arbeitsspeicher kopiert und die Ausgänge in die konfigurierte "Reaktion auf CPU-STOP" versetzt. Der Diagnosepuffer und die dauerhaft gespeicherten Werte der IP-Adresse werden beim Urlöschen nicht gelöscht.

Sie können die Einstellung "Anlauf nach NETZ-EIN" der CPU konfigurieren. Diese Einstellungen finden Sie in der Gerätekonfiguration der CPU unter "Anlauf". Beim Einschalten führt die CPU eine Reihe von Diagnoseprüfungen und anschließend die Systeminitialisierung durch. Während der Initialisierung des Systems löscht die CPU den gesamten nicht remanenten Speicherbereich der Merker und setzt alle nicht remanenten DB-Inhalte auf die Anfangswerte aus dem Ladespeicher zurück. Die CPU puffert den remanenten Speicherbereich der Merker und remanente DB-Inhalte und nimmt dann den entsprechenden Betriebszustand ein. Bestimmte Fehler verhindern, dass die CPU in den Betriebszustand RUN geht. Die CPU unterstützt die folgenden Konfigurationsoptionen:

- Kein Neustart (in STOP bleiben)
- Warmstart - RUN
- Warmstart - Betriebsart vor NETZ-AUS



ACHTUNG

Behebbare Fehler können die CPU veranlassen, in den Betriebszustand STOP zu gehen.

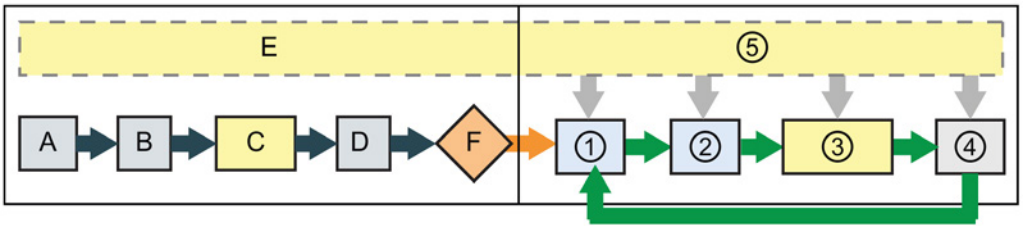
Die CPU kann aufgrund von behebbaren Fehlern in den Betriebszustand STOP gehen. Dies können z.B. der Ausfall eines ersetzbaren Signalmoduls oder temporäre Fehler sein wie Störungen der Netzleitung oder ein unvorhersehbares Anlaufereignis. Solche Situationen können zu Sachschäden führen.

Wenn für die CPU die Einstellung "Warmstart - Betriebsart vor NETZ-AUS" konfiguriert ist, kehrt sie in den Betriebszustand zurück, in dem sie vor dem Netzausfall oder Fehler war. Befand sich die CPU zum Zeitpunkt des Netzausfalls oder Fehlers im Betriebszustand STOP, kehrt sie beim Anlaufen in STOP zurück und bleibt in diesem Betriebszustand, bis sie einen Befehl für den Wechsel nach RUN erhält. War die CPU zum Zeitpunkt des Netzausfalls oder Fehlers in Betriebszustand RUN, kehrt sie beim nächsten Anlaufen in RUN zurück, sofern sie keine Fehler erkennt, die den Wechsel in RUN verhindern würden.

CPUs, die unabhängig von einer STEP 7-Verbindung laufen sollen, sind für "Warmstart - RUN" zu konfigurieren, damit die CPU beim nächsten Anlaufen in den Betriebszustand RUN zurückkehren kann.

Sie können den aktuellen Betriebszustand mit den Befehlen "STOP" und "RUN" (Seite 1125) in den Online-Tools der Programmiersoftware ändern. Sie können ferner eine STP-Anweisung (Seite 298) in Ihr Programm einfügen, um die CPU in den Betriebszustand STOP zu versetzen. Auf diese Weise können Sie die Bearbeitung Ihres Programms abhängig von der Programmlogik unterbrechen.

- Im Betriebszustand STOP bearbeitet die CPU Kommunikationsanforderungen und führt Selbstdiagnosen durch. Die CPU führt das Anwenderprogramm nicht aus und es finden keine automatischen Aktualisierungen des Prozessabbilds statt.
- In den Betriebszuständen STARTUP und RUN führt die CPU die im folgenden Bild gezeigten Aufgaben aus.



STARTUP

A	Speicherbereich E (Abbild) wird gelöscht
B	Speicherbereich A (Bild) wird je nach Konfiguration mit null, dem letzten Wert oder dem Ersatzwert initialisiert und PB-, PN- und AS-i-Ausgänge werden auf null gesetzt
C	Nicht remanente Speicherbereiche der Merker und Datenbausteine werden mit ihrem Anfangswert initialisiert und die konfigurierten Weckalarm- und Uhrzeitereignisse werden aktiviert. Die Anlauf-OBs werden ausgeführt.
D	Der Zustand der physischen Eingänge wird in den Speicherbereich E kopiert
E	Alle Alarmereignisse werden in der Warteschlange für die Verarbeitung im Betriebszustand RUN gespeichert
F	Das Schreiben von Speicherbereich A in die physischen Ausgänge wird aktiviert

RUN

①	Speicherbereich A wird in die physischen Ausgänge geschrieben
②	Der Zustand der physischen Eingänge wird in den Speicherbereich E kopiert
③	Die Programmzyklus-OBs werden ausgeführt
④	Führt Selbstdiagnose durch
⑤	Alarme und Kommunikation werden in allen Teilen des Zyklus bearbeitet

STARTUP-Verarbeitung

Immer wenn der Betriebszustand von STOP nach RUN wechselt, löscht die CPU das Prozessabbild der Eingänge, initialisiert das Prozessabbild der Ausgänge und verarbeitet die Anlauf-OBs. Alle Lesezugriffe auf das Prozessabbild der Eingänge von Anweisungen in den Anlauf-OBs ergeben den Wert null und nicht den aktuellen Wert des physischen Eingangs. Deshalb müssen Sie, um den aktuellen Zustand eines physischen Eingangs im Betriebszustand STARTUP zu lesen, den Eingang direkt lesen. Dann werden die Anlauf-OBs und zugehörige FCs und FBs ausgeführt. Sind mehrere Anlauf-OBs vorhanden, so werden diese fortlaufend nach OB-Nummer, beginnend mit der niedrigsten OB-Nummer, abgearbeitet.

Jeder Anlauf-OB enthält Anlaufinformationen, damit Sie die Gültigkeit der remanenten Daten und der Echtzeituhr ermitteln können. Sie können in den Anlauf-OBs Anweisungen programmieren, um diese Anlaufwerte zu untersuchen und entsprechende Maßnahmen zu ergreifen. Die folgenden Anlaufadressen werden von Anlauf-OBs unterstützt:

Tabelle 5- 1 Vom Anlauf-OB unterstützte Anlaufadressen

Eingang	Datentyp	Beschreibung
LostRetentive	Bool	Dieses Bit ist wahr, wenn die Speicherbereiche der remantenten Daten verloren gegangen sind.
LostRTC	Bool	Dieses Bit ist wahr, wenn die Echtzeituhr verloren gegangen ist.

Die CPU führt während der Anlaufverarbeitung auch die folgenden Aufgaben aus.

- Alarme werden während der Anlaufphase in eine Warteschlage gestellt und nicht bearbeitet
- In der Anlaufphase findet keine Zykluszeitüberwachung statt
- Beim Anlaufen kann die Konfiguration der schnellen Zähler (HSC), der Impulsdauermodulation (PWM) und der Punkt-zu-Punkt-Kommunikationsmodule geändert werden
- Die eigentlichen Funktionsabläufe von HSC, PWM und PTP-Modulen finden nur im Betriebszustand RUN statt

Nach der Ausführung der Anlauf-OBs geht die CPU in den Betriebszustand RUN und bearbeitet die Steuerungsaufgaben in einem fortlaufenden Zyklus.

5.1.2 Bearbeitung des Zyklus im Betriebszustand RUN

In jedem Zyklus schreibt die CPU in die Ausgänge, sie liest die Eingänge, führt das Anwenderprogramm aus, aktualisiert die Kommunikationsmodule und antwortet auf Anwenderalarmereignisse und Kommunikationsanfragen. Kommunikationsanfragen werden während des Zyklus regelmäßig bearbeitet.

Diese Aktionen (außer den Anwenderalarmereignissen) werden zyklisch fortlaufend bearbeitet. Anwenderalarmereignisse, die aktiviert sind, werden nach der Priorität in der Reihenfolge ihres Auftretens abgearbeitet. Bei Alarmereignissen liest die CPU die Eingänge, führt den OB aus und schreibt in die Ausgänge, ggf. mit dem zugehörigen Teilprozessabbild (PIP).

Das System gewährleistet, dass der Zyklus innerhalb der maximalen Zykluszeit abgearbeitet wird, sonst wird ein Zeitfehler erzeugt.

- Jeder Zyklus beginnt mit der Abfrage der aktuellen Werte der digitalen und analogen Ausgänge im Prozessabbild und dem Schreiben dieser Werte in die physischen Ausgänge von CPU, SB und SMs, die für die automatische E/A-Aktualisierung konfiguriert sind (Standardkonfiguration). Greift eine Anweisung auf einen physischen Ausgang zu, so werden der Ausgang im Prozessabbild und der physische Ausgang aktualisiert.
- Im weiteren Verlauf des Zyklus werden die aktuellen Werte der digitalen und analogen Eingänge aus der CPU, der SB und den SMs, die für die automatische E/A-Aktualisierung konfiguriert sind (Standardkonfiguration), ausgelesen und diese Werte in das Prozessabbild geschrieben. Greift eine Anweisung auf einen physischen Eingang zu, so wird der Wert des physischen Eingangs geändert, der Eingang im Prozessabbild jedoch nicht aktualisiert.
- Nach dem Lesen der Eingänge wird das Anwenderprogramm von der ersten Anweisung bis zur letzten Anweisung ausgeführt. Darin enthalten sind alle Programmzyklus-OBs sowie alle zugehörigen FCs und FBs. Die Programmzyklus-OBs werden fortlaufend nach der OB-Nummer, beginnend mit der niedrigsten OB-Nummer, abgearbeitet.

Die Kommunikationsbearbeitung tritt während des Zyklus regelmäßig auf und unterbricht möglicherweise die Ausführung des Anwenderprogramms.

Zu den Selbstdiagnosen gehören regelmäßige Prüfungen des System und die Abfrage des Zustands der E/A-Module.

Alarme können in jedem Teil des Zyklus auftreten, sie sind ereignisgesteuert. Tritt ein Ereignis auf, so unterbricht die CPU den Zyklus und ruft den OB für die Verarbeitung des Ereignisses auf. Wenn der OB das Ereignis abgearbeitet hat, setzt die CPU die Ausführung des Anwenderprogramms an der Stelle fort, an der es zuvor unterbrochen wurde.

5.1.3 Organisationsbausteine (OBs)

OBs steuern die Ausführung des Anwenderprogramms. Die Ausführung eines Organisationsbausteins wird durch bestimmte Ereignisse in der CPU angestoßen. OBs können sich nicht gegenseitig aufrufen und sie können auch nicht aus einer FC oder einem FB aufgerufen werden. Nur ein Ereignis wie zum Beispiel ein Diagnosealarm oder ein Zeitintervall kann die Ausführung eines OB anstoßen. Die CPU bearbeitet die OBs dann entsprechend ihren Prioritätsklassen, wobei OBs mit höheren Prioritäten zuerst ausgeführt werden. Die niedrigste Prioritätsklasse ist 1 (für den Hauptprogrammzyklus), die höchste ist 26.

5.1.3.1 Programmzyklus-OB

Programmzyklus-OBs werden zyklisch ausgeführt, wenn die CPU im Betriebszustand RUN ist. Der Hauptbaustein des Programms ist ein Programmzyklus-OB. Er enthält die Anweisungen für die Steuerung Ihrer Anwendung und aus ihm heraus werden weitere Anwenderbausteine aufgerufen. Sie können mehrere Programmzyklus-OBs anlegen, die die CPU in numerischer Reihenfolge ausführt. Der Hauptbaustein (OB1) ist der Standardbaustein.

Programmzyklusereignisse

Das Programmzyklusereignis tritt einmal in jedem Programmzyklus auf. Während des Programmzyklus schreibt die CPU in die Ausgänge, liest die Eingänge und führt Programmzyklus-OBs aus. Das Programmzyklusereignis ist erforderlich und immer aktiviert. Möglicherweise haben Sie für das Programmzyklusereignis keinen Programmzyklus-OB oder Sie haben mehrere OBs. Nachdem das Programmzyklusereignis eingetreten ist, führt die CPU den Programmzyklus-OB mit der niedrigsten Nummer (in der Regel OB1) aus. Die anderen Programmzyklus-OBs werden von der CPU sequentiell (in numerischer Reihenfolge) innerhalb des Programmzyklus ausgeführt. Das Programm wird zyklisch ausgeführt, so dass das Programmzyklusereignis zu den folgenden Zeitpunkten eintritt:

- Wenn der letzte Anlauf-OB abgearbeitet ist
- Wenn der letzte Programmzyklus-OB abgearbeitet ist

Tabelle 5- 2 Anlaufinformationen für einen Programmzyklus-OB

Eingang	Datentyp	Beschreibung
Initial_Call	Bool	Wahr beim ersten Aufruf des OB
Remanence	Bool	Wahr, wenn remanente Daten vorhanden sind

5.1.3.2 Anlauf-OB

Anlauf-OBs werden einmal ausgeführt, wenn der Betriebszustand der CPU von STOP nach RUN wechselt, beim Hochfahren in den Betriebszustand RUN und bei einem vorgegebenen Wechsel von STOP nach RUN. Anschließend beginnt die Ausführung des Programmzyklus.

Anlaufereignisse

Das Anlaufereignis tritt einmal bei einem Wechsel von STOP in RUN auf und verursacht die Ausführung des Anlauf-OBs durch die CPU. Für das Anlaufereignis können Sie mehrere OBs konfigurieren. Die Anlauf-OBs werden in numerischer Reihenfolge ausgeführt.

Tabelle 5- 3 Startinformationen für einen Anlauf-OB

Eingang	Datentyp	Beschreibung
LostRetentive	Bool	Wahr, wenn remanente Daten verloren gegangen sind
LostRTC	Bool	Wahr, wenn Datum und Uhrzeit verloren gegangen sind

5.1.3.3 Verzögerungsalarm-OB

Verzögerungsalarm-OBs werden nach einer von Ihnen konfigurierten Zeitverzögerung ausgeführt.

Verzögerungsalarmereignisse

Verzögerungsalarmereignisse konfigurieren Sie für das Auftreten nach einer angegebenen Verzögerungszeit. Die Verzögerungszeit weisen Sie mit der Anweisung SRT_DINT zu. Die Verzögerungsereignisse unterbrechen den Programmzyklus, um den entsprechenden Verzögerungsalarm-OB auszuführen. Einem Verzögerungsereignis können Sie nur einen Verzögerungsalarm-OB zuordnen. Die CPU unterstützt vier Verzögerungsereignisse.

Tabelle 5- 4 Startinformationen für einen Verzögerungsalarm-OB

Eingang	Datentyp	Beschreibung
Sign	Word	An auslösenden Aufruf von SRT_DINT übergebene Kennung

5.1.3.4 Weckalarm-OB

Weckalarm-OBs werden in bestimmten Abständen ausgeführt. Sie können bis zu vier Weckalarmereignisse konfigurieren, wobei jedem Ereignis jeweils ein OB entspricht.

Weckalarmereignisse

Die Weckalarmereignisse ermöglichen Ihnen, die Ausführung eines Alarm-OBs zu einer konfigurierten Zykluszeit einzurichten. Die Anfangszykluszeit wird beim Anlegen des Weckalarm-OBs konfiguriert. Ein Weckalarmereignis unterbricht den Programmzyklus und führt den entsprechenden Weckalarm-OB aus. Zu beachten ist, dass ein Weckalarmereignis eine höhere Prioritätsklasse als das Programmzyklusereignis hat.

Einem Weckalarmereignis kann nur genau ein Weckalarm-OB zugewiesen werden.

Sie können jedem Weckalarm eine Phasenverschiebung zuweisen, so dass die Ausführung von Weckalarmen um den Wert der Phasenverschiebung gegeneinander verschoben werden kann. Ist beispielsweise ein Weckalarmereignis für 1 ms und ein weiteres für 2 ms projektiert, treten alle 2 ms beide Ereignisse gleichzeitig auf. Wird das 1-ms-Ereignis um 500 µsec und das 2-ms-Ereignis um 0 µs phasenverschoben, treten die beiden Ereignisse nie gleichzeitig auf.

Die Phasenverschiebung ist standardmäßig auf 0 eingestellt. Um die anfängliche Phasenverschiebung zu ändern oder um die anfängliche Zykluszeit eines zyklischen Ereignisses zu ändern, klicken Sie in der Projektnavigation mit der rechten Maustaste auf den Weckalarm-OB, dann auf "Eigenschaften" und auf "Weckalarm" und geben Sie hier die neuen Anfangswerte ein. Sie können die Zykluszeit und Phasenverschiebung auch aus Ihrem Programm abfragen und ändern. Verwenden Sie dazu die Anweisungen Weckalarm-Parameter abfragen (QRY_CINT) und Weckalarm-Parameter setzen (SET_CINT). Die über die Anweisung SET_CINT eingegebenen Werte für Zykluszeit und Phasenverschiebung werden bei Ausschalten oder Wechsel in STOP nicht gespeichert; beim Wiedereinschalten oder bei Rückkehr in RUN werden wieder die ursprünglichen Werte verwendet. Die CPU unterstützt insgesamt vier Weckalarmereignisse.

5.1.3.5 Prozessalarm-OB

Prozessalarm-OBs werden ausgeführt, sobald das entsprechende Prozessereignis auftritt. Ein Prozessalarm-OB unterbricht den normalen Programmablauf als Reaktion auf ein Signal eines Prozessereignisses.

Prozessalarmereignisse

Prozessalarmereignisse werden durch eine Veränderung in der Hardware ausgelöst, z. B. eine steigende oder fallende Flanke an einem Eingang oder ein Ereignis an einem HSC (schnellen Zähler). Die S7-1200 unterstützt genau einen Alarm-OB für jedes Prozessalarmereignis. Sie können die Prozessereignisse in der Gerätekonfiguration aktivieren und einem Ereignis in der Gerätekonfiguration oder mit einer ATTACH-Anweisung im Anwenderprogramm einen OB zuweisen. Die CPU unterstützt mehrere Prozessalarmereignisse. Welche Ereignisse verfügbar sind, hängt vom CPU-Modell und der Anzahl der Eingänge ab.

Für Prozessalarmereignisse gelten die folgenden Grenzwerte:

Flanken:

- Ereignisse steigende Flanke: maximal 16
- Ereignisse fallende Flanke: maximal 16

HSC-Ereignisse:

- CV=PV: maximal 6
- Richtungswechsel: maximal 6
- Externes Rücksetzen: maximal 6

5.1.3.6 Zeitfehler-OB

Sofern konfiguriert, wird der Zeitfehler-OB (OB 80) ausgeführt, wenn die maximale Zykluszeit überschritten wird oder ein Zeitfehlerereignis auftritt. Wird dieser OB ausgelöst, beginnt die Ausführung, die den normalen Programmablauf oder auch einen anderen Ereignis-OB unterbricht.

Das Auftreten eines dieser Ereignisse erzeugt einen Eintrag im Diagnosepuffer, der das Ereignis beschreibt. Der Eintrag im Diagnosepuffer wird unabhängig davon erzeugt, ob ein Zeitfehler-OB vorhanden ist oder nicht.

Zeitfehleralarmereignisse

Das Auftreten eines beliebigen Zeitfehlers führt zu einem Zeitfehlerereignis:

- Überschreiten der maximalen Zykluszeit

  Der Fehler "Maximale Zykluszeit überschritten" tritt auf, wenn der Programmzyklus nicht innerhalb der angegebenen maximalen Zykluszeit beendet wird. Weitere Informationen zum Fehler "Maximale Zykluszeit überschritten", zum Konfigurieren der maximalen Zykluszeit in den Eigenschaften der CPU und zum Zurücksetzen der Zykluszeit finden Sie im Abschnitt "Überwachen der Zykluszeit" im S7-1200 Systemhandbuch (Seite 104).

- CPU kann den angeforderten OB nicht starten, weil ein zweiter Alarm (Weck- oder Verzögerungsalarm) gestartet wurde, bevor die CPU die Ausführung des ersten Alarm-OBs beendet hat

- Warteschlangenüberlauf

  Der Fehler "Warteschlangenüberlauf" tritt ein, wenn die Alarme schneller auftreten, als sie von der CPU verarbeitet werden können. Die Zahl anstehender Ereignisse wird von der CPU begrenzt, indem jedem Ereignistyp eine eigene Warteschlange zugewiesen wird. Tritt ein Ereignis auf, wenn die entsprechende Warteschlange voll ist, wird von der CPU ein Zeitfehlerereignis erzeugt.

Alle Zeitfehlerereignisse lösen die Ausführung des Zeitfehler-OBs (sofern vorhanden) aus. Ist kein Zeitfehler-OB vorhanden, hängt die Reaktion der CPU auf den Zeitfehler von der Gerätekonfiguration der CPU ab:

- Bei der Standardkonfiguration für Zeitfehler, wenn z. B. ein zweiter Weckalarm gestartet wird, bevor die CPU die Ausführung des ersten beendet hat, bleibt die CPU in RUN.
- Bei der Standardkonfiguration für das Überschreiten der maximalen Zykluszeit wird die CPU in STOP versetzt.

Durch Ausführen der Anweisung RE_TRIGR (Seite 297) zum Neustarten der Zykluszeitüberwachung kann das Anwenderprogramm die Ausführung des Programmzyklus bis zum Zehnfachen der konfigurierten maximalen Zykluszeit verlängern. Wenn jedoch der Fehler "Maximale Zykluszeit überschritten" zweimal in demselben Programmzyklus auftritt, ohne dass die Zykluszeit zurückgesetzt wird, geht die CPU in STOP, unabhängig davon, ob der Zeitfehler-OB vorhanden ist. Weitere Informationen finden Sie im Abschnitt "Überwachen der Zykluszeit" im S7-1200 Systemhandbuch (Seite 104).

Der Zeitfehler-OB enthält Anlaufinformationen, anhand derer Sie ermitteln können, welches Ereignis und welcher OB den Zeitfehler erzeugt hat. Sie können im OB Anweisungen programmieren, um diese Anlaufwerte zu untersuchen und entsprechende Maßnahmen zu ergreifen.

Tabelle 5- 5 Anlaufinformationen für den Zeitfehler-OB (OB 80)

Eingang	Datentyp	Beschreibung
fault_id	BYTE	16#01 - Maximale Zykluszeit überschritten 16#02 - Gewünschter OB kann nicht gestartet werden 16#07 und 16#09 - Warteschlangenüberlauf
csg_OBnr	OB_ANY	Nummer des OBs, der bei Auftreten des Fehlers ausgeführt wurde
csg_prio	UINT	Priorität des fehlerverursachenden OBs

Um einen Zeitfehler-OB in Ihr Projekt einzufügen, müssen Sie einen Zeitfehleralarm hinzufügen, indem Sie in der Projektnavigation unter "Programmbausteine" auf "Neuen Baustein hinzufügen" doppelklicken und dann "Organisationsbaustein" und "Zeitfehler" auswählen.

Die Priorität bei einer neuen V4.0-CPU ist 22. Wenn Sie eine V3.0-CPU durch eine V4.0-CPU ersetzen (Seite 1349), ist die Priorität 26, also die Priorität, die für V3.0 gültig war. In beiden Fällen kann das Prioritätsfeld geändert werden und Sie können für die Priorität einen beliebigen Wert im Bereich von 22 bis 26 festlegen.

5.1.3.7 Diagnosefehler-OB

Der Diagnosefehler-OB wird ausgeführt, wenn die CPU einen Diagnosefehler erkennt oder wenn ein diagnosefähiges Modul einen Fehler erkennt und der Diagnosefehleralarm für das Modul aktiviert ist. Ein Diagnosefehler-OB unterbricht den normalen Programmablauf. Wenn die CPU in den Betriebszustand STOP gehen soll, sobald dieser Fehlertyp erkannt wird, können Sie in den Diagnosefehler-OB eine STP-Anweisung einfügen.

Wenn Sie keinen Diagnosefehler-OB in Ihr Programm einfügen, ignoriert die CPU den Fehler und bleibt in RUN.

Diagnosefehlerereignisse

Analoge (lokale), PROFINET-, PROFIBUS- und einige digitale (lokale) Geräte können Diagnosefehler erkennen und melden. Das Auftreten bzw. Verschwinden eines von verschiedenen Diagnosefehlern führt zu einem Diagnosefehlerereignis. Die folgenden Diagnosefehler werden unterstützt:

- Keine Anwenderspannung
- Oberer Grenzwert überschritten
- Unterer Grenzwert überschritten
- Drahtbruch
- Kurzschluss

Diagnosefehlerereignisse lösen die Ausführung des Diagnosefehler-OBs (OB 82) aus, sofern dieser vorhanden ist. Ist er nicht vorhanden, ignoriert die CPU den Fehler.

Um einen Diagnosefehler-OB in Ihr Projekt einzufügen, müssen Sie einen Diagnosefehleralarm hinzufügen. Hier doppelklicken Sie in der Projektnavigation unter "Programmbausteine" auf "Neuen Baustein hinzufügen" und wählen dann "Organisationsbaustein" und "Diagnosefehler" aus.

Hinweis

Diagnosefehler bei mehrkanaligen lokalen Analoggeräten (E/A, RTD und Thermoelement)

Der Diagnosefehler-OB kann nur die Diagnosefehler jeweils eines Kanals verarbeiten.

Wenn in zwei Kanälen eines mehrkanaligen Geräts Fehler auftreten, löst der zweite Fehler den Diagnosefehler-OB nur unter folgenden Bedingungen aus: Der Fehler des ersten Kanals wird behoben, die vom ersten Fehler ausgelöste Ausführung des Diagnosefehler-OBs ist beendet und der zweite Fehler liegt weiterhin vor.

Der Diagnosefehler-OB enthält Anlaufinformationen, anhand derer Sie ermitteln können, ob das Ereignis wegen des Auftretens oder Verschwindens eines Fehlers ausgelöst wurde und welches Gerät und welcher Kanal den Fehler gemeldet haben. Sie können im Diagnosefehler-OB Anweisungen programmieren, um diese Anlaufwerte zu untersuchen und entsprechende Maßnahmen zu ergreifen.

Hinweis

Die Anlaufinformationen des Diagnosefehler-OBs beziehen sich auf das Submodul als Ganzes, wenn kein Diagnoseereignis ansteht

In V3.0 gaben die Anlaufinformationen eines gehenden Diagnosefehlerereignisses stets die Quelle des Ereignisses an. In V4.0 beziehen sich die Anlaufinformationen, sofern durch das gehende Ereignis beim Submodul keine Diagnose mehr ansteht, auch dann auf das Submodul als Ganzes (16#8000), wenn es sich bei der Quelle des Ereignisses um einen bestimmten Kanal handelte.

Beispiel: Wenn ein Drahtbruch einen Diagnosefehler an Kanal 2 auslöst, der Fehler dann behoben und das Diagnosefehlerereignis gelöscht wird, beziehen sich die Anlauftinformationen nicht auf Kanal 2, sondern auf das Submodul (16#8000).

Tabelle 5- 6 Anlaufinformationen für den Diagnosefehler-OB

Eingang	Datentyp	Beschreibung
IOstate	WORD	E/A-Zustand des Geräts: • Bit 0 = 1, wenn die Konfiguration korrekt ist, und Bit 0 = 0, wenn die Konfiguration nicht mehr korrekt ist. • Bit 4 = 1, wenn ein Fehler vorliegt (Beispiel: Drahtbruch). (Bit 4 = 0, wenn kein Fehler vorliegt.) • Bit 5 = 1, wenn die Konfiguration **nicht** korrekt ist, und Bit 5 = 0, wenn die Konfiguration wieder korrekt ist. • Bit 6 = 1, wenn ein E/A-Zugriffsfehler aufgetreten ist. Die Hardwarekennung der E/A mit dem Zugriffsfehler finden Sie in LADDR. (Bit 6 = 0, wenn kein Fehler vorliegt.)
LADDR	HW_ANY	Hardwarekennung des Geräts oder der Funktionseinheit, das bzw. die den Fehler gemeldet hat[1]
Channel	UINT	Kanalnummer
MultiError	BOOL	WAHR, wenn mehrere Fehler vorliegen

[1] Der Eingang LADDR enthält die Hardwarekennung des Geräts bzw. der Funktionseinheit, das bzw. die den Fehler ausgegeben hat. Die Hardwarekennung wird automatisch zugewiesen, wenn Komponenten in die Geräte- oder Netzsicht eingefügt werden. Sie wird im Register "Konstanten" von PLC-Variablen angezeigt. Der Hardwarekennung wird zudem automatisch ein Name zugewiesen. Diese Einträge im Register "Konstanten" der PLC-Variablen können nicht geändert werden.

5.1.3.8 OB "Ziehen oder Stecken von Modulen"

Der OB "Ziehen oder Stecken von Modulen" wird ausgeführt, wenn ein konfiguriertes und nicht deaktiviertes dezentrales E/A-Modul oder -Submodul (PROFIBUS, PROFINET, AS-i) ein Ereignis in Bezug auf das Ziehen oder Stecken eines Moduls generiert.

Ereignis "Ziehen oder Stecken von Modulen"

Bei den folgenden Bedingungen wird ein Ereignis "Ziehen oder Stecken von Modulen" generiert:

- Ein Anwender zieht oder steckt ein konfiguriertes Modul
- Ein konfiguriertes Modul ist physisch nicht im Erweiterungsbaugruppenträger vorhanden
- Ein inkompatibles Modul, das nicht dem konfigurierten Modul entspricht, befindet sich in einem Erweiterungsbaugruppenträger
- Ein mit einem konfigurierten Modul kompatibles Modul befindet sich in einem Erweiterungsbaugruppenträger, doch die Konfiguration gestattet keine Ersatzgeräte
- Ein Modul oder Submodul weist Parametrierungsfehler auf

Wurde dieser OB nicht programmiert, bleibt die CPU in RUN, wenn eine dieser Bedingungen für ein konfiguriertes und nicht deaktiviertes dezentrales E/A-Modul eintritt.

Unabhängig davon, ob dieser OB programmiert wurde, wechselt die CPU in STOP, wenn eine dieser Bedingungen für ein Modul im zentralen Baugruppenträger eintritt.

Tabelle 5- 7 Anlaufinformationen für den OB "Ziehen oder Stecken von Modulen"

Eingang	Datentyp	Beschreibung
LADDR	HW_IO	Hardwarekennung
Event_Class	Byte	16#38: Modul gesteckt 16#29: Modul gezogen
Fault_ID	Byte	Fehler-ID

5.1.3.9 Baugruppenträger- oder Stationsfehler-OB ("Rack or station failure OB")

Der Baugruppenträger- oder Stationsfehler-OB ("Rack or station failure OB") wird ausgeführt, wenn die CPU einen Fehler oder Kommunikationsausfall an einem dezentralen Baugruppenträger/einer dezentralen Station erkennt.

Baugruppenträger- oder Stationsfehlerereignis

Die CPU erzeugt ein Baugruppenträger- oder Stationsfehlerereignis, wenn eine der folgenden Situationen erkannt wird:

- Ausfall eines DP-Mastersystems oder eines PROFINET IO-Systems (bei einem kommenden oder gehenden Ereignis)
- Ausfall eines DP-Slaves oder eines IO-Geräts (bei einem kommenden oder gehenden Ereignis)
- Ausfall eines Submoduls eines PROFINET I-Device

Ist dieser OB nicht programmiert, bleibt die CPU in RUN, wenn eine dieser Bedingungen eintritt.

Tabelle 5- 8 Anlaufinformationen für den Baugruppenträger- oder Stationsfehler-OB

Eingang	Datentyp	Beschreibung
LADDR	HW_IO	Hardware-Kennung
Event_Class	Byte	16#32: Aktivierung eines DP-Slaves oder eines IO-Device 16#33: Deaktivierung eines DP-Slaves oder eines IO-Geräts 16#38: gehendes Ereignis 16#39: kommendes Ereignis
Fault_ID	Byte	Fehlerkennung

5.1.3.10 Uhrzeit-OB

Uhrzeit-OBs werden anhand konfigurierter Uhrzeitbedingungen ausgeführt. Die CPU unterstützt zwei Uhrzeit-OBs.

Uhrzeitereignisse

Sie können ein Uhrzeitalarmereignis so konfigurieren, dass es zu einer bestimmten Uhrzeit oder zyklisch auftritt, wobei Sie einen der folgenden Zyklen einstellen können:

- Minütlich: Der Alarm tritt einmal pro Minute auf.
- Stündlich: Der Alarm tritt einmal pro Stunde auf.
- Täglich: Der Alarm tritt täglich zu einer bestimmten Zeit (Stunde und Minute) auf.
- Wöchentlich: Der Alarm tritt wöchentlich zu einer bestimmten Zeit an einem bestimmten Wochentag auf (zum Beispiel jeden Dienstag um 16:30 h).
- Monatlich: Der Alarm tritt monatlich zu einer bestimmten Zeit an einem bestimmten Tag des Monats auf. Für den Tag muss eine Zahl im Bereich von 1 bis 28 festgelegt werden.
- Am Monatsende: Der Alarm tritt am letzten Tag jedes Monats zu einer bestimmten Zeit auf.
- Jährlich: Der Alarm tritt einmal jährlich an einem bestimmten Datum (Monat und Tag) auf. Der 29. Februar kann nicht als Datum eingegeben werden.

Tabelle 5- 9 Anlaufinformationen für den Uhrzeit-OB

Eingang	Datentyp	Beschreibung
CaughtUp	Bool	Aufruf des OB wurde abgefangen, weil die Zeit vorgestellt wurde
SecondTimes	Bool	Aufruf des OB startet erneut, weil die Zeit zurückgestellt wurde

5.1.3.11 Zustands-OB

Zustands-OBs werden ausgeführt, wenn ein DPV1- oder PNIO-Slave einen Zustandsalarm auslöst. Dies kann vorkommen, wenn eine Komponente (Modul oder BG-Träger) eines DPV1- oder PNIO-Slaves ihren Betriebszustand, zum Beispiel von RUN nach STOP, ändert.

Zustandsereignisse

Ausführlichere Informationen zu Ereignissen, die einen Zustandsalarm auslösen, sind in der Herstellerdokumentation des DPV1- oder PNIO-Slaves zu finden.

Tabelle 5- 10 Anlaufinformationen für einen Zustands-OB

Eingang	Datentyp	Beschreibung
LADDR	HW_IO	Hardware-Kennung
Slot	UInt	Steckplatznummer
Specifier	Word	Alarm Specifier

5.1.3.12 Aktualisierungs-OB

Aktualisierungs-OBs werden ausgeführt, wenn ein DPV1- oder PNIO-Slave einen Aktualisierungsalarm auslöst.

Aktualisierungsereignisse

Ausführlichere Informationen zu Ereignissen, die einen Aktualisierungsalarm auslösen, sind in der Herstellerdokumentation des DPV1- oder PNIO-Slaves zu finden.

Tabelle 5- 11 Anlaufinformationen für einen Aktualisierungs-OB

Eingang	Datentyp	Beschreibung
LADDR	HW_IO	Hardware-Kennung
Slot	UInt	Steckplatznummer
Specifier	Word	Alarm Specifier

5.1.3.13 Profil-OB

Profil-OBs werden ausgeführt, wenn ein DPV1- oder PNIO-Slave einen profilspezifischen Alarm auslöst.

Profilereignisse

Ausführlichere Informationen zu Ereignissen, die einen Profilalarm auslösen, sind in der Herstellerdokumentation des DPV1- oder PNIO-Slaves zu finden.

Tabelle 5- 12 Anlaufinformationen für einen Profil-OB

Eingang	Datentyp	Beschreibung
LADDR	HW_IO	Hardware-Kennung
Slot	UInt	Steckplatznummer
Specifier	Word	Alarm Specifier

5.1.3.14 Prioritäten und Warteschlange für die Ausführung von Ereignissen

Die CPU-Bearbeitung wird durch Ereignisse gesteuert. Ein Ereignis löst die Ausführung eines Alarm-OBs aus. Sie können während der Erstellung des Bausteins, während der Gerätekonfiguration oder über eine Anweisung ATTACH oder DETACH den Alarm-OB für ein Ereignis angeben. Einige Ereignisse wie das Programmzyklusereignis oder zyklische Ereignisse treten regelmäßig auf. Andere Ereignisse wie das Anlaufereignis oder Zeitverzögerungsereignisse treten einmalig auf. Einige Ereignisse treten auf, wenn die Hardware ein Ereignis auslöst, z. B. ein Flankenereignis an einem Eingang oder ein Ereignis eines schnellen Zählers. Ereignisse wie das Diagnosefehler- und das Zeitfehlerereignis treten nur im Fehlerfall auf. Die Ereignisprioritäten und Warteschlangen dienen zum Festlegen der Verarbeitungsreihenfolge der Alarm-OBs.

Die CPU verarbeitet die Ereignisse in der Reihenfolge ihrer Priorität, wobei 1 die niedrigste Priorität und 26 die höchste Priorität ist. Vor der Version 4.0 der S7-1200 CPUs gehörte jeder OB-Typ zu einer festen Prioritätsklasse (1 bis 26). Ab V4.0 können Sie jedem von Ihnen konfigurierten OB eine Prioritätsklasse zuweisen. Sie konfigurieren die Prioritätsklasse in den OB-Eigenschaften.

Unterbrechbare und nicht unterbrechbare Ausführungsarten

OBs (Seite 90) werden in der Reihenfolge der Priorität der Ereignisse ausgeführt, die die OBs auslösen. Ab Version 4.0 können Sie konfigurieren, ob die OB-Ausführung unterbrechbar oder nicht unterbrechbar sein soll. Beachten Sie, dass Programmzyklus-OBs immer unterbrechbar sind. Alle anderen OBs können Sie jedoch als unterbrechbar oder als nicht unterbrechbar konfigurieren.

Wenn Sie die unterbrechbare Ausführungsart festlegen, wird die Ausführung eines OBs unterbrochen, falls ein Ereignis mit höherer Priorität vor dem Ausführungsende des OBs auftritt, um die Ausführung des OBs höherer Priorität zu ermöglichen. Daraufhin wird das Ereignis mit höherer Priorität ausgeführt und anschließend der unterbrochene OB fortgesetzt. Treten während der Ausführung eines unterbrechbaren OBs mehrere Ereignisse auf, verarbeitet die CPU diese Ereignisse in der Reihenfolge der Priorität.

Wenn Sie nicht die unterbrechbare Ausführungsart festlegen, wird die Ausführung eines OBs unabhängig von anderen während dieses Zeitraums ausgelösten Ereignissen beendet.

Nachfolgend sind zwei Fälle beschrieben, in denen Alarmereignisse einen Weckalarm-OB und einen Zeitverzögerungs-OB auslösen. In beiden Fällen ist der Zeitverzögerungs-OB (OB201) keinem Teilprozessabbild zugeordnet (Seite 81) und wird mit Priorität 4 ausgeführt. Der Weckalarm-OB (OB200) ist Teilprozessabbild PIP1 zugeordnet und wird mit Priorität 2 ausgeführt. Die folgenden Abbildungen zeigen die unterschiedliche Ausführung unterbrechbarer und nicht unterbrechbarer OBs:

Bild 5-1 Fall 1: Nicht unterbrechbare OB-Ausführung

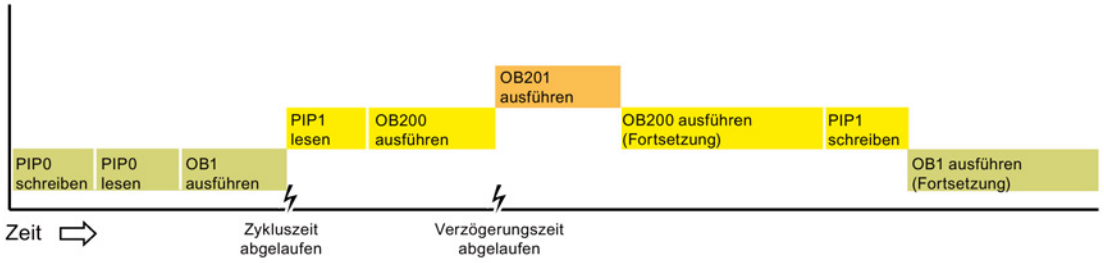


Bild 5-2 Fall 2: Unterbrechbare OB-Ausführung

Hinweis

Wenn Sie die OB-Ausführungsart als nicht unterbrechbar festlegen, kann ein Zeitfehler-OB lediglich Programmzyklus-OBs unterbrechen. In den Versionen vor V4.0 der S7-1200 CPU konnte ein Zeitfehler-OB jede OB-Ausführung unterbrechen. Ab V4.0 muss die OB-Ausführung als unterbrechbar konfiguriert werden, wenn ein Zeitfehler-OB (oder ein anderer OB mit höherer Priorität) die Ausführung von OBs, die keine Programmzyklus-OBs sind, unterbrechen soll.

Wissenswertes zu Prioritäten und Warteschlange für die Ausführung von Ereignissen

Die CPU begrenzt die Anzahl anstehender Ereignisse aus einer einzigen Quelle, indem jedem Ereignistyp eine eigene Warteschlange zugewiesen wird. Sobald die maximale Zahl anstehender Ereignisse eines bestimmten Typs erreicht ist, wird das nächste Ereignis nicht mehr bearbeitet und geht verloren. Mit einem Zeitfehleralarm-OB (Seite 93) können Sie auf einen Überlauf der Warteschlange reagieren.

Jedes Ereignis einer CPU hat eine Priorität. Die Ereignisse werden von der CPU im Allgemeinen in der Reihenfolge der Priorität (höchste zuerst) bearbeitet. Ereignisse mit gleicher Priorität werden von der CPU nach dem First-In-First-Out-Prinzip bearbeitet.

Tabelle 5- 13 OB-Ereignisse

Ereignis	Zulässige Anzahl	OB-Standardpriorität
Programmzyklus	1 Programmzyklusereignis Mehrere OBs zulässig	1[4]
Anlauf	1 Anlaufereignis [1] Mehrere OBs zulässig	1[4]
Verzögerung	Bis zu 4 Zeitereignisse 1 OB je Ereignis	3
Weckalarm	Bis zu 4 Ereignisse 1 OB je Ereignis	8
Prozessalarm	Bis zu 50 Prozessalarmereignisse[2]	18
	1 OB je Ereignis, Sie können jedoch den gleichen OB für mehrere Ereignisse verwenden	18
Zeitfehler	1 Ereignis (nur wenn konfiguriert)[3]	22 oder 26[4]
Diagnosefehler	1 Ereignis (nur wenn konfiguriert)	5
Ziehen oder Stecken von Modulen	1 Ereignis	6
Fehler bei Baugruppenträger oder Station	1 Ereignis	6
Tageszeit	Bis zu 2 Ereignisse	2
Zustand	1 Ereignis	4
Update	1 Ereignis	4
Profil	1 Ereignis	4

[1] Das Anlauf- und das Programmzyklusereignis treten nie gleichzeitig auf, weil der Anlauf zuerst beendet sein muss, bevor der Programmzyklus gestartet wird.

[2] Bei Verwendung der Anweisungen DETACH und ATTACH sind mehr als 50 Prozessalarm-OBs möglich.

[3] Sie können die CPU so konfigurieren, dass sie in RUN bleibt, wenn die maximale Zykluszeit überschritten wird, oder Sie können mit der Anweisung RE_TRIGR die Zykluszeit zurücksetzen. Wenn die maximale Zykluszeit jedoch in einem Zyklus zum zweiten Mal überschritten wird, geht die CPU in den Betriebszustand STOP.

[4] Die Priorität bei einer neuen CPU V4.0 oder V4.1 ist 22. Wenn Sie eine CPU V3.0 durch eine CPU V4.0 oder V4.1 ersetzen, ist die Priorität 26: also die Priorität, die für V3.0 gültig war. In beiden Fällen kann das Prioritätsfeld geändert werden und Sie können für die Priorität einen beliebigen Wert im Bereich von 22 bis 26 festlegen.

Weitere Informationen hierzu finden Sie unter "Austauschen Ihrer CPU V3.0 durch eine CPU V4.1 (Seite 1349)".

Außerdem erkennt die CPU andere Ereignisse, die nicht über zugehörige OBs verfügen. Die folgende Tabelle zeigt diese Ereignisse und die entsprechenden Aktionen der CPU:

Tabelle 5- 14 Zusätzliche Ereignisse

Ereignis	Beschreibung	CPU-Aktion
Peripheriezugriffsfehler	Fehler beim direkten Lesen/Schreiben der E/A	Die CPU schreibt das erste Auftreten in den Diagnosepuffer und bleibt im Betriebszustand RUN.
Fehler bezüglich der maximalen Zykluszeit	Die CPU überschreitet die konfigurierte Zykluszeit zum zweiten Mal	Die CPU schreibt den Fehler in den Diagnosepuffer und geht in den Betriebszustand STOP.
Peripheriezugriffsfehler	E/A-Fehler beim Aktualisieren des Prozessabbilds	Die CPU schreibt das erste Auftreten in den Diagnosepuffer und bleibt im Betriebszustand RUN.
Programmierfehler	Fehler beim Ausführen des Programms	Falls der Baustein mit dem Fehler über eine Fehlerbehandlungsroutine verfügt, aktualisiert er die Fehlerstruktur; falls nicht, schreibt die CPU den Fehler in den Diagnosepuffer und bleibt im Betriebszustand RUN.

Latenzzeit

Die Ereignis-Latenzzeit (d.h. die Zeit zwischen der Mitteilung der CPU über das Auftreten eines Ereignisses und dem Start der Ausführung der ersten Anweisung im OB für die Ereignisbearbeitung) beträgt ca. 175 µs, wenn zum Zeitpunkt des Alarmereignisses nur ein Programmzyklus-OB als Bearbeitungsroutine aktiv ist.

5.1.4 Überwachen und Konfigurieren der Zykluszeit

Die Zykluszeit ist die Zeit, die das Betriebssystem der CPU benötigt, um die zyklische Phase des Betriebszustands RUN auszuführen. Die CPU bietet zwei Verfahren zum Überwachen der Zykluszeit:

- Maximale Zykluszeit
- Mindestzykluszeit

Die Zyklusüberwachung beginnt nach abgeschlossenem Anlaufereignis. Projektiert werden kann diese Funktion in der CPU unter "Gerätekonfiguration > Zykluszeit".

Die CPU überwacht den Zyklus ständig und reagiert, wenn die maximale Zykluszeit überschritten wird. Wird die eingestellte maximale Zykluszeit überschritten, so wird ein Fehler erzeugt und auf eine der beiden folgenden Arten behandelt:

- Wenn das Anwenderprogramm keinen Zeitfehleralarm-OB enthält, generiert das Zeitfehlerereignis einen Diagnosepuffereintrag, doch die CPU bleibt im Betriebszustand RUN. (Sie können die Konfiguration der CPU ändern, sodass die CPU bei Erkennen eines Zeitfehlers in STOP wechselt, doch bei der Standardkonfiguration bleibt die CPU im Betriebszustand RUN.)
- Wenn das Anwenderprogramm einen Zeitfehleralarm-OB enthält, führt die CPU ihn aus.

Die Anweisung RE_TRIGR (Seite 297) (Zykluszeitüberwachung neu starten) ermöglicht das Rücksetzen des Zeitgebers für die Messung der Zykluszeit. Wenn die für die aktuelle Programmzyklusausführung abgelaufene Zeit weniger als das Zehnfache der konfigurierten maximalen Zykluszeit beträgt, stößt die Anweisung RE_TRIGR die Zykluszeitüberwachung erneut an und gibt ENO = WAHR zurück. Ist dies nicht der Fall, stößt die Anweisung RE_TRIGR die Zykluszeitüberwachung nicht erneut an. Sie gibt ENO = FALSCH zurück.

Typischerweise wird der Zyklus so schnell wie möglich ausgeführt und der nächste Zyklus beginnt, sobald der vorherige beendet ist. Je nach Anwenderprogramm und Kommunikationsaufgaben kann die Zykluszeit jedoch variieren. Um solche Schwankungen zu vermeiden, unterstützt die CPU eine optionale Mindestzykluszeit. Wenn Sie diese optionale Funktion aktivieren und eine Mindestzykluszeit in ms vorgeben, verzögert die CPU nach der Ausführung der Programmzyklus-OBs die Wiederholung des Programmzyklus, bis die Mindestzykluszeit abgelaufen ist.

Führt die CPU einen Zyklus schneller aus als mit der Mindestzykluszeit festgelegt ist, so nutzt die CPU die verbleibende Zeit für Laufzeitdiagnosen und/oder für die Bearbeitung von Kommunikationsanforderungen.

Wird der Zyklus nicht innerhalb der vorgegebenen Mindestzykluszeit beendet, so wird er normal bis zum Ende ausgeführt (einschließlich der Kommunikationsbearbeitung) und die Überschreitung der Mindestzeit erzeugt keine Systemreaktion. In der folgenden Tabelle sind die Bereiche und Voreinstellungen für die Zykluszeitüberwachung angegeben.

Tabelle 5- 15 Bereich für die Zykluszeit

Zykluszeit	Bereich (ms)	Voreinstellung
Maximale Zykluszeit[1]	1 bis 6000	150 ms
Mindestzykluszeit[2]	1 bis maximale Zykluszeit	Deaktiviert

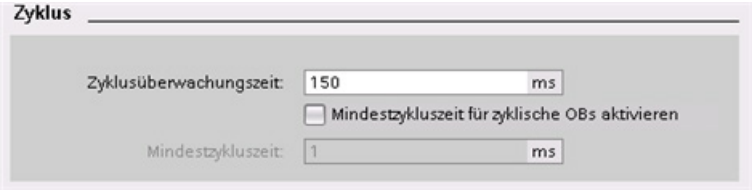
1 Die maximale Zykluszeit ist immer aktiviert. Richten Sie eine Zykluszeit zwischen 1 ms und 6000 ms ein. Standard sind 150 ms.

2 Die Mindestzykluszeit ist optional und standardmäßig deaktiviert. Richten Sie ggf. eine Zykluszeit zwischen 1 ms und der maximalen Zykluszeit ein.

Konfigurieren von Zykluszeit und Kommunikationslast

In den CPU-Eigenschaften der Gerätekonfiguration können Sie die folgenden Parameter einstellen:

- Zyklus: Hier können Sie eine maximale Zyklusüberwachungszeit eingeben. Sie können auch eine Mindestzykluszeit eingeben bzw. aktivieren.



- Kommunikationslast: Sie können einen prozentualen Anteil der Zeit für Kommunikationsaufgaben festlegen.

Ausführliche Informationen über den Zyklus finden Sie unter "Überwachung des Zyklus" (Seite 104).

5.1.5 CPU-Speicher

Speicherverwaltung

Die CPU stellt die folgenden Speicherbereiche für Anwenderprogramm, Daten und Konfiguration bereit:

- Der Ladespeicher ist ein nicht-flüchtiger Speicher für Anwenderprogramm, Daten und Konfiguration. Beim Laden eines Projekts in die CPU speichert die CPU das Programm zunächst im Ladespeicher. Dieser Speicher befindet sich entweder auf einer Memory Card (sofern vorhanden) oder in der CPU. Die CPU speichert diesen nicht-flüchtigen Speicherbereich auch bei einem Spannungsausfall. Die Memory Card unterstützt einen größeren Speicherbereich als den in der CPU integrierten.
- Der Arbeitsspeicher ist ein flüchtiger Speicher für einige Elemente des Anwenderprojekts während der Bearbeitung des Anwenderprogramms. Die CPU kopiert einige Elemente des Projekts vom Ladespeicher in den Arbeitsspeicher. Dieser flüchtige Speicherbereich geht bei Spannungsausfall verloren und wird bei Spannungsrückkehr von der CPU wiederhergestellt.
- Der remanente Speicher ist ein nicht-flüchtiger Speicher für eine begrenzte Menge an Arbeitsspeicherwerten. Die CPU nutzt den remanenten Speicherbereich zum Speichern der Werte ausgewählter Adressen des Anwenderspeichers bei Spannungsausfall. Wenn eine Abschaltung oder ein Spannungsausfall auftritt, stellt die CPU diese remanenten Werte beim Anlauf wieder her.

Zur Anzeige der Speicherauslastung durch einen übersetzten Programmbaustein doppelklicken Sie auf den Baustein im Ordner "Programmbausteine" der STEP 7-Projektnavigation und wählen Sie im Kontextmenü Option "Speicherauslastung". In den Übersetzungseigenschaften werden der Ladespeicher und der Arbeitsspeicher für den übersetzten Baustein angezeigt.

Um die Speicherauslastung der Online-CPU anzuzeigen, doppelklicken Sie in STEP 7 auf "Online & Diagnose", erweitern "Diagnose" und wählen "Speicher".

Remanenter Speicher

Sie können Datenverlust nach Spannungsausfall dadurch vermeiden, dass Sie bestimmte Daten als remanent definieren. Die folgenden Daten können Sie in der CPU als remanent definieren:

- Merker (M): Sie können die Größe des remanenten Speichers für Merker in der PLC-Variablentabelle oder in der Zuweisungsliste definieren. Der remanente Merkerspeicher beginnt immer an MB0 und läuft über die angegebene Anzahl von Bytes ununterbrochen weiter. Geben Sie diesen Wert in der PLC-Variablentabelle oder in der Zuweisungsliste an, indem Sie in der Funktionsleiste auf das Symbol "Remanent" klicken. Geben Sie die Anzahl der Merkerbytes an, die ab MB0 remanent sein sollen.

  Hinweis: Für jeden Baustein kann die Zuweisungsliste durch Auswahl eines Bausteins im Ordner Programmbausteine und den Menübefehl **Werkzeuge > Belegungsplan** angezeigt werden.

- Variablen eines Funktionsbausteins (FB): Wenn ein FB mit aktivierter Option "Optimiert" erstellt wurde, enthält der Schnittstelleneditor für diesen FB eine Spalte "Remanent". In dieser Spalte können Sie für jede Variable einzeln entweder "Remanent", "Nicht remanent" oder "Im IDB setzen" auswählen. Ein Instanz-DB, der beim Einfügen des FBs in den Programmiereditor angelegt wurde, weist ebenfalls diese Spalte auf. Sie können den remanenten Zustand einer Variable im Schnittstelleneditor des Instanz-DBs nur ändern, wenn Sie im optimierten FB im Bereich "Remanent" für die Variable die Option "Im IDB setzen" (Im Instanzdatenbaustein setzen) ausgewählt haben.

  Wenn ein FB mit aktivierter Option "Standard - kompatibel mit S7-300/400" erstellt wurde, enthält der Schnittstelleneditor für diesen FB keine Spalte "Remanent". Ein Instanz-DB, der beim Einfügen des FBs in den Programmiereditor angelegt wurde, weist eine Spalte "Remanent" auf, die bearbeitet werden kann. In diesem Fall wirkt sich die Auswahl der Option "Remanent" für eine Variable so aus, dass **alle** Variablen ausgewählt werden. Genauso wirkt es sich aus, wenn Sie die Option für eine Variable deaktivieren. Dann werden **alle** Variablen deaktiviert. Bei einem FB, für den die Option "Standard - kompatibel mit S7-300/400" aktiviert ist, können Sie den remanenten Zustand im Editor des Instanz-DBs ändern, doch es werden alle Variablen gleichzeitig auf denselben Zustand gesetzt.

  Nachdem Sie den FB angelegt haben, können Sie die Option "Standard - kompatibel mit S7-300/400" nicht mehr ändern. Sie können diese Option nur beim Anlegen des FBs auswählen. Um zu ermitteln, ob ein vorhandener FB mit aktivierter Option "Optimiert" oder "Standard - kompatibel mit S7-300/400" angelegt wurde, klicken Sie in der Projektnavigation mit der rechten Maustaste auf den FB, wählen "Eigenschaften" und dann "Attribute". Ist das Kontrollkästchen "Optimiert" aktiviert, wurde der Baustein mit optimiertem Bausteinzugriff angelegt. Andernfalls ist es ein Standardbaustein und kompatibel mit den S7-300/400 CPUs.

- Variablen eines globalen Datenbausteins: Das Verhalten eines globalen DBs in Bezug auf die Zuweisung des remanenten Zustands ähnelt dem eines FBs. Je nach Einstellung des Bausteinzugriffs können Sie den remanenten Zustand entweder für einzelne oder für alle Variablen eines globalen Datenbausteins definieren.
  - Wenn Sie beim Anlegen des DBs "Optimiert" auswählen, können Sie den remanenten Zustand für jede Variable einzeln festlegen.
  - Wenn Sie beim Anlegen des DBs "Standard - kompatibel mit S7-300/400" auswählen, gilt die Einstellung des remanenten Zustands für alle Variablen des DBs, d. h., es sind entweder alle Variablen remanent oder es ist keine Variable remanent.

Die CPU unterstützt insgesamt 10240 Byte remanenter Daten. Um zu ermitteln, wieviele Bytes verfügbar sind, klicken Sie in der PLC-Variablentabelle oder in der Zuweisungsliste in der Funktionsleiste auf das Symbol "Remanent". Hier geben Sie zwar den remanenten Bereich für den Merkerspeicher an, doch die zweite Zeile gibt den verbleibenden Gesamtspeicher für M und DB zusammen an. Beachten Sie, dass Sie alle Datenbausteine mit remanenten Variablen übersetzen müssen, damit dieser Wert korrekt ist.

Hinweis

Durch das Laden eines Programms werden die Werte im remanenten Speicher weder gelöscht noch verändert. Soll der remanente Speicher vor dem Laden gelöscht werden, ist vor dem Laden des Programms die CPU auf die Werkseinstellungen zurückzusetzen.

5.1.5.1 System- und Taktmerker

In den CPU-Eigenschaften können Sie Bytes für "Systemspeicher" und "Taktmerker" aktivieren. Die einzelnen Bits dieser Funktionen können in Ihrer Programmlogik über die Variablennamen referenziert werden.

- Sie können im Merkerbereich ein Byte als Systemspeicher zuweisen. Das Byte des Systemmerkers bietet die folgenden vier Bits, die von Ihrem Anwenderprogramm über die folgenden Variablennamen referenziert werden können:
  - Erster Zyklus: Das Bit (Variablenname "FirstScan") wird nach dem Ende des Anlauf-OBs für den ersten Zyklus auf 1 gesetzt. (Nach dem Ende des ersten Zyklus wird das Bit "Erster Zyklus" auf 0 gesetzt.)
  - Diagnosezustand geändert: (Variablenname: "DiagStatusUpdate") wird einen Zyklus lang auf 1 gesetzt, nachdem die CPU ein Diagnoseereignis erfasst hat. Weil die CPU das Bit "DiagStatusUpdate" erst am Ende der ersten Ausführung des Programmzyklus-OBs setzt, kann Ihr Anwenderprogramm nicht erkennen, ob während der Ausführung des Anlauf-OBs oder während der ersten Ausführung des Programmzyklus-OBs eine Diagnoseänderung auftrat.
  - Immer 1 (high): Das Bit (Variablenname "AlwaysTRUE") ist immer auf 1.
  - Immer 0 (low): Das Bit (Variablenname "AlwaysFALSE") ist immer auf 0.
- Sie können im Merkerbereich ein Byte als Taktmerker zuweisen. Jedes Bit dieses als Taktmerker konfigurierten Bytes erzeugt einen Rechteckimpuls. Das Byte des Taktmerkers bietet 8 verschiedene Frequenzen, von 0,5 Hz (langsam) bis 10 Hz (schnell). Sie können diese Bits, besonders in Verbindung mit Flankenanweisungen, als Steuerbits für die zyklische Auslösung von Aktionen im Anwenderprogramm verwenden.

Die CPU initialisiert diese Bytes beim Wechsel von STOP in STARTUP. Die Bits des Taktmerkers wechseln während der Betriebsarten STARTUP und RUN synchron zum CPU-Takt.

⚠ VORSICHT

Gefahren beim Überschreiben der Bits von Systemspeicher oder Taktmerker

Ein Überschreiben der Bits von Systemspeicher oder Taktmerker kann die Daten in diesen Funktionen beschädigen und bewirken, dass Ihr Anwenderprogramm fehlerhaft arbeitet, was zu Sachschaden und Verletzungen des Personals führen kann.

Weil Taktmerker und Systemspeicher nicht reservierter Speicher im Bereich der Merker sind, können Anweisungen und Kommunikation in diese Adressen schreiben und Daten beschädigen.

Vermeiden Sie das Schreiben von Daten in diese Adressen, um sicherzustellen, dass diese Funktionen einwandfrei arbeiten, und implementieren Sie immer eine Not-Aus-Schaltung für Ihren Prozess oder Ihre Maschine.

Der Systemspeicher konfiguriert ein Byte mit Bits, die bei einem spezifischen Ereignis eingeschaltet werden (Wert = 1).

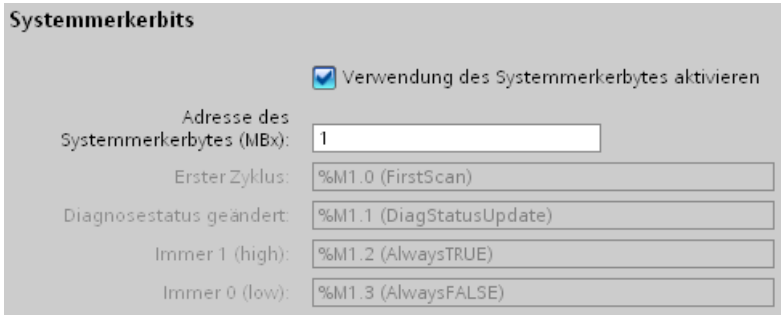


Tabelle 5- 16 Systemspeicher

7	6	5	4	3	2	1	0
Reserviert Wert 0				Immer ausgeschaltet Wert 0	Immer eingeschaltet Wert 1	Diagnosestatusanzeige • 1: Änderung • 0: Keine Änderung	Anzeige erster Zyklus • 1: Erster Zyklus nach Anlauf • 0: Nicht der erste Zyklus

Der Taktmerker konfiguriert ein Byte, das die einzelnen Bits in bestimmten Abständen ein- und ausschaltet. Jeder Taktmerker erzeugt einen Rechteckimpuls im entsprechenden Merkerspeicher M. Diese Bits können, vor allem in Verbindung mit Flankenanweisungen, als Steuerbits für die zyklische Auslösung von Aktionen im Anwendercode verwendet werden.

Tabelle 5- 17 Taktmerker

Bitnummer	7	6	5	4	3	2	1	0
Variablenname								
Period(en)	2.0	1.6	1.0	0.8	0.5	0.4	0.2	0.1
Frequenz (Hz)	0.5	0.625	1	1.25	2	2.5	5	10

Weil der Taktmerker asynchron zum CPU-Zyklus läuft, kann sich der Zustand des Taktmerkers während eines langen Zyklus mehrere Male ändern.

5.1.6 Diagnosepuffer

Die CPU unterstützt einen Diagnosepuffer, der für jedes Diagnoseereignis einen Eintrag enthält. Jeder Eintrag umfasst das Datum und die Uhrzeit, zu denen das Ereignis aufgetreten ist, eine Ereigniskategorie und eine Ereignisbeschreibung. Die Einträge werden in chronologischer Reihenfolge angezeigt, wobei das jüngste Ereignis an oberster Stelle steht. In diesem Puffer werden bis zu 50 aktuelle Ereignisse gespeichert. Ist der Puffer voll, wird immer das älteste Ereignis durch ein neues Ereignis überschrieben. Bei einer Unterbrechung der Stromversorgung werden die Ereignisse gespeichert.

Im Diagnosepuffer werden die folgenden Ereignistypen erfasst:

- Jedes Diagnoseereignis des Systems, CPU-Fehler und Modulfehler
- Jeder Zustandswechsel der CPU (Anlauf, Wechsel in Betriebszustand STOP oder RUN)

Für den Zugriff auf den Diagnosepuffer (Seite 1126) müssen Sie online sein. Den Puffer finden Sie unter "Online & Diagnose > Diagnose > Diagnosepuffer".

5.1.7 Echtzeituhr

Die CPU verfügt über eine Echtzeituhr. Wenn die CPU ausgeschaltet ist, wird die Uhr über einen Hochleistungskondensator mit Spannung versorgt. Der Hochleistungskondensator wird aufgeladen, wenn die CPU eingeschaltet ist. War die CPU mindestens 24 Stunden eingeschaltet, so reicht die Ladung des Hochleistungskondensators für den Betrieb der Uhr typischerweise für 20 Tage.

STEP 7 stellt für die Echtzeituhr die Systemzeit ein, die einen werkseitig voreingestellten Standardwert hat bzw. den Standardwert durch Rücksetzen auf die Werkseinstellungen erhält. Um die Echtzeituhr zu nutzen, müssen Sie sie einstellen. Zeitstempel wie die von Diagnosepuffereinträgen, Datenprotokolldateien und Datenprotokolleinträgen beruhen auf der Systemzeit. Sie stellen die Echtzeituhr über die Funktion "Uhrzeit einstellen" (Seite 1122) in der Ansicht "Online & Diagnose" der Online-CPU ein. STEP 7 berechnet dann die Systemzeit anhand der von Ihnen eingestellten Zeit plus oder minus dem Zeitunterschied des Windows-Betriebssystems zur UTC (Coordinated Universal Time). Wird für die Echtzeituhr die Ortszeit eingestellt, erzeugt dies eine Systemzeit, die der UTC entspricht, sofern die Einstellungen für Zeitzone und Sommer-/Winterzeitumschaltung Ihres Windows-Betriebssystems Ihrem Gebietsschema entsprechen.

STEP 7 enthält Anweisungen (Seite 326) zum Lesen und Schreiben der Systemzeit (RD_SYS_T und WR_SYS_T), zum Lesen der Ortszeit (RD_LOC_T) und zum Einstellen der Zeitzone (SET_TIMEZONE). Die Anweisung RD_LOC_T berechnet die Ortszeit anhand der Zeitzone und der Zeitverschiebung für die Sommer-/Winterzeit, die Sie in der Konfiguration der Echtzeituhr in den allgemeinen Eigenschaften der CPU (Seite 168) eingegeben haben. Über diese Einstellungen können Sie die Zeitzone für Ihre Ortszeit einrichten, Sie können optional die Sommerzeitumschaltung aktivieren und die Anfangs- und Endezeiten für die Sommer-/Winterzeitumschaltung eingeben. Diese Einstellungen können Sie auch mit den Anweisungen SET_TIMEZONE konfigurieren.

5.1.8 Konfigurieren der Ausgänge für den Wechsel von RUN in STOP

Sie können das Verhalten der digitalen und analogen Ausgänge für den Betriebszustand STOP der CPU konfigurieren. Für jeden Ausgang einer CPU, eines SB oder SM können Sie die Werte der Ausgänge einfrieren oder einen Ersatzwert aufschalten:

- Durch einen vorgegebenen Ausgangswert ersetzen (Voreinstellung): Sie geben für jeden Ausgang (Kanal) der CPU, des SB oder SM einen Ersatzwert ein.

  Der voreingestellte Ersatzwert für digitale Ausgänge ist AUS, für analoge Ausgänge ist er 0.

- Einfrieren der Ausgänge im letzten Zustand: Die Ausgänge behalten ihre aktuellen Werte, die sie zu der Zeit hatten, als der Wechsel von RUN nach STOP auftrat. Nach dem Anlauf werden die Ausgänge auf den Standardersatzwert gesetzt.

Sie konfigurieren das Verhalten der Ausgänge in der Gerätekonfiguration. Wählen Sie die einzelnen Geräte und öffnen Sie das Register "Eigenschaften", um die Ausgänge des jeweiligen Geräts zu konfigurieren.

Hinweis

Einige dezentrale E/A-Module bieten zusätzliche Einstellungen für die Reaktion auf die Betriebsart STOP der CPU. Wählen Sie für diese Module die gewünschte Option aus der Liste in der Gerätekonfiguration aus.

Wenn die CPU von RUN nach STOP wechselt, speichert die CPU das Prozessabbild und schreibt die entsprechenden Werte für digitale und analoge Ausgänge entsprechend der Konfiguration.

5.2 Datenspeicher, Speicherbereiche, E/A und Adressierung

5.2.1 Zugriff auf Daten der S7-1200

STEP 7 vereinfacht die symbolische Programmierung. Dazu erstellen Sie für die Adressen der Daten symbolische Namen oder "Variablen", die entweder in Form von PLC-Variablen für Speicheradressen und E/A oder in Form von lokalen Variablen innerhalb eines Codebausteins vorkommen. Zum Einfügen dieser Variablen in Ihr Anwenderprogramm geben Sie einfach den Variablennamen für den gewünschten Anweisungsparameter ein.

Zur Verdeutlichung, wie die CPU Speicherbereiche strukturiert und adressiert, wird im Folgenden dargestellt, wie PLC-Variablen auf die "absolute" Adressierung der Daten verweisen. Die CPU bietet mehrere Möglichkeiten für die Datenspeicherung während der Ausführung des Anwenderprogramms:

- Globaler Speicher: Die CPU bietet eine Vielzahl von spezialisierten Speicherbereichen, einschließlich Eingänge (E), Ausgänge (A) und Merker (M). Dieser Speicher ist für alle Codebausteine ohne Einschränkung zugänglich.
- PLC-Variablentabelle: In der STEP 7-PLC-Variablentabelle können Sie für bestimmte Adressen im Speicher symbolische Namen eingeben. Dieser Variablen gelten global im gesamten STEP 7-Programm und gestatten die Programmierung mit aussagekräftigen Namen.
- Datenbaustein (DB): Sie können in Ihr Anwenderprogramm DBs zum Speichern von Daten für die Codebausteine einfügen. Die gespeicherten Daten bleiben nach der Ausführung des zugehörigen Codebausteins erhalten. In einem "globalen" DB werden Daten gespeichert, die von allen Codebausteinen verwendet werden können, in einem Instanz-DB werden jedoch nur Daten für einen bestimmten FB gespeichert, und er ist entsprechend der Parameter des FBs strukturiert.
- Temporärer Speicher: Bei jedem Aufruf eines Codebausteins gibt das Betriebssystem der CPU temporären bzw. lokalen Speicherplatz (L) frei, der bei der Ausführung des Bausteins genutzt werden kann. Ist die Ausführung des Codebausteins beendet, weist die CPU den lokalen Speicher für die Ausführung anderer Codebausteine zu.

Jeder Speicherplatz hat eine eindeutige Adresse. Anhand dieser Adresse kann Ihr Anwenderprogramm auf die Informationen an diesem Speicherplatz zugreifen. Durch Verweise auf Speicherbereiche für Eingänge (E) oder Ausgänge (A), z. B. E0.3 oder A1.7, erfolgt ein Zugriff auf das Prozessbild. Für den direkten Zugriff auf den physischen Eingang oder Ausgang fügen Sie den Verweis ":P" (z. B. E0.3:P, A1.7:P oder "Stop:P") als Anhang hinzu.

Tabelle 5- 18 Speicherbereiche

Speicherbereich	Beschreibung	Forcen	Remanent
E Prozessabbild der Eingänge E_:P [1] (physischer Eingang)	Wird zu Beginn des Zyklus aus den physischen Eingängen kopiert	Nein	Nein
	Direktes Lesen der physischen Eingänge von CPU, SB oder SM	Ja	Nein

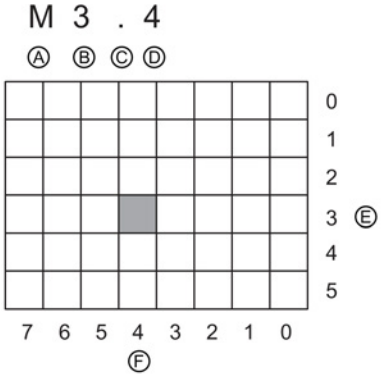
Speicherbereich	Beschreibung	Forcen	Remanent
A Prozessabbild der Ausgänge A_:P [1] (physischer Ausgang)	Wird zu Beginn des Zyklus in die physischen Ausgänge kopiert	Nein	Nein
	Direktes Schreiben in die physischen Ausgänge von CPU, SB oder SM	Ja	Nein
M Merker	Steuerung und Datenspeicher	Nein	Ja (optional)
L temporärer Speicher	Temporäre, lokale Daten für einen Baustein	Nein	Nein
DB Datenbaustein	Datenspeicher und auch Parameterspeicher für FBs	Nein	Ja (optional)

[1] Für den direkten Zugriff (Lesen oder Schreiben) auf die physischen Eingänge und Ausgänge hängen Sie den Code ":P" an die Adresse oder Variable an (z. B. E0.3:P, A1.7:P oder "Stop:P").

Jeder Speicherplatz hat eine eindeutige Adresse. Anhand dieser Adresse kann Ihr Anwenderprogramm auf die Informationen an diesem Speicherplatz zugreifen. Die absolute Adresse setzt sich aus den folgenden Elementen zusammen:

- Speicherbereichskennung (wie E, A oder M)
- Größe der Daten, auf die zugegriffen werden soll (wie "B" für Byte, "W" für Word oder "D" für DWord)
- Anfangsadresse der Daten (wie Byte 3 oder Wort 3)

Beim Zugriff auf ein Bit in der Adresse eines Booleschen Werts geben Sie keine Mnemonik für die Größe ein. Sie geben nur den Speicherbereich, die Byte-Adresse und die Bitadresse der Daten ein (wie E0.0, A0.1 oder M3.4).



A	Speicherbereichskennung	E	Bytes des Speicherbereichs
B	Adresse des Byte: Byte 3	F	Bits des ausgewählten Byte
C	Trennzeichen ("Byte.Bit")		
D	Bitadresse im Byte (Bit 4 von 8)		

In dem Beispiel folgt auf den Speicherbereich und die Adresse des Bytes (M = Bereich der Merker und 3 = Byte 3) ein Punkt ("."), um die Adresse des Bits (Bit 4) abzutrennen.

Zugriff auf Daten in den Speicherbereichen der CPU

STEP 7 vereinfacht die symbolische Programmierung. Typischerweise werden Variablen entweder in PLC-Variablen, einem Datenbaustein oder in der Schnittstelle oberhalb von OB, FC oder FB angelegt. Diese Variablen umfassen einen Namen, Datentyp, Versatz und Kommentar. Außerdem kann in einem Datenbaustein ein Startwert angegeben werden. Diese Variablen können Sie während der Programmierung nutzen, indem Sie den Variablennamen als Parameter für die Anweisung eingeben. Optional können Sie den absoluten Operanden (Speicher, Bereich, Größe und Versatz) als Anweisungsparameter eingeben. Die Beispiele in den folgenden Abschnitten zeigen, wie Sie absolute Operanden eingeben. Das Zeichen % wird vom Programmiereditor automatisch vor dem absoluten Operanden eingefügt. Im Programmiereditor haben Sie die Auswahl zwischen folgenden Ansichten: symbolisch, symbolisch und absolut, absolut.

E (Prozessabbild der Eingänge): Die CPU fragt die Peripherieeingänge (physikalischen Eingänge) unmittelbar vor der Ausführung eines Zyklus-OB in jedem Zyklus ab und schreibt diese Werte in das Prozessabbild der Eingänge. Auf das Prozessabbild der Eingänge können Sie im Bit, Byte, Wort und Doppelwortformat zugreifen. Es ist sowohl der Lese- als auch der Schreibzugriff erlaubt, jedoch werden die Eingänge des Prozessabbilds typischerweise nur gelesen.

Tabelle 5- 19 Absolute Adressierung für den Speicherbereich E

Bit	E[Byteadresse].[Bitadresse]	E0.1
Byte, Wort oder Doppelwort	E[Größe][Adresse des Anfangs-byte]	EB4, EW5 oder ED12

Wenn Sie den Verweis ":P" an die Adresse anhängen, können Sie die digitalen und analogen Eingänge von CPU, SB, SM oder des dezentralen Moduls direkt lesen. Der Unterschied zwischen dem Zugriff über E_:P statt über E liegt darin, dass die Daten direkt von den angesprochenen Eingängen kommen und nicht aus dem Prozessabbild der Eingänge. Dieser Zugriff über E_:P wird auch als direkter Lesezugriff bezeichnet, weil die Daten direkt aus der Quelle statt aus einer Kopie, die bei der letzten Aktualisierung des Prozessabbilds der Eingänge erstellt wurde, gelesen werden.

Weil die physikalischen Eingänge ihre Werte direkt aus den mit ihnen verbundenen Feldgeräten erhalten, darf in diese Eingänge nicht geschrieben werden. Zugriffe über E_:P sind somit reine Lesezugriffe, im Gegensatz zu Zugriffen auf Eingänge, die gelesen oder geschrieben werden können.

Zugriffe über E_:P sind ferner durch die Größe der Eingänge begrenzt, die von CPU, SB oder SM unterstützt werden (gerundet auf das nächste Byte). Sind beispielweise die Eingänge eines SB mit 2 DE/2 DA so konfiguriert, dass sie bei E4.0 beginnen, kann mit E4.0:P and E4.1:P oder EB4:P darauf zugegriffen werden. Der Zugriff auf E4.2:P bis E4.7:P wird nicht als Fehler erkannt, ist aber sinnlos, weil diese Adressen nicht belegt sind. Zugriffsversuche auf EW4:P und ED4:P sind verboten, weil sie den Byteversatz dieses SB überschreiten.

Zugriffe über E_:P haben keinen Einfluss auf den im Prozessabbild der Eingänge gespeicherten Wert.

Tabelle 5- 20 Absolute Adressierung für den Speicherbereich E (direkt)

Bit	E[Byteadresse].[Bitadresse]:P	E0.1:P
Byte, Wort oder Doppelwort	E[Größe][Anfangsadresse des Byte]:P	EB4:P, EW5:P oder ED12:P

A (Prozessabbild der Ausgänge): Die CPU kopiert die im Prozessabbild der Ausgänge gespeicherten Werte in die physischen Ausgänge. Auf das Prozessabbild der Ausgänge können Sie im Bit, Byte, Wort und Doppelwortformat zugreifen. Auf die Ausgänge des Prozessabbilds ist sowohl der Lese- als auch der Schreibzugriff erlaubt.

Tabelle 5- 21 Absolute Adressierung für den Speicherbereich A

Bit	A[Byteadresse].[Bitadresse]	A1.1
Byte, Wort oder Doppelwort	A[Größe][Adresse des Anfangs-byte]	AB5, AW10, AD40

Wenn Sie den Verweis ":P" an die Adresse anhängen, können Sie direkt in die digitalen und analogen Ausgänge von CPU, SB, SM oder des dezentralen Moduls schreiben. Der Unterschied zwischen dem Zugriff über A_:P statt über A liegt darin, dass die Daten direkt zu den angesprochenen Ausgängen gelangen und zusätzlich zum Prozessabbild der Ausgänge (es wird in beide Bereiche geschrieben). Dieser Zugriff über A_:P wird auch als direkter Schreibzugriff bezeichnet, weil die Daten direkt an die Zieladresse gesendet werden und diese nicht auf die nächste Aktualisierung aus dem Prozessabbild der Ausgänge warten muss.

Weil die physikalischen Ausgänge direkt die mit ihnen verbundenen Feldgeräte steuern, dürfen diese Ausgänge nicht gelesen werden. Zugriffe über A_:P sind somit reine Schreibzugriffe, im Gegensatz zu Zugriffen auf Ausgänge, die gelesen oder geschrieben werden können.

Zugriffe über A_:P sind ferner durch die Größe der Ausgänge begrenzt, die von CPU, SB oder SM unterstützt werden (gerundet auf das nächste Byte). Sind beispielweise die Ausgänge eines SB mit 2 DE/2 DA so konfiguriert, dass sie bei A4.0 beginnen, kann mit A4.0:P and A4.1:P oder AB4:P darauf zugegriffen werden. Der Zugriff auf A4.2:P bis A4.7:P wird nicht als Fehler erkannt, ist aber sinnlos, weil diese Adressen nicht belegt sind. Zugriffsversuche auf AW4:P und AD4:P sind verboten, weil sie den Byteversatz dieses SB überschreiten.

Zugriffe über A_:P beeinflussen den physischen Ausgang sowie den im Prozessabbild der Ausgänge gespeicherten Wert.

Tabelle 5- 22 Absolute Adressierung für den Speicherbereich A (direkt)

Bit	A[Byteadresse].[Bitadresse]:P	A1.1:P
Byte, Wort oder Doppelwort	A[Größe][Anfangsadresse des Byte]:P	AB5:P, AW10:P oder AD40:P

M (Speicherbereich der Merker): Den Speicherbereich der Merker (M) können Sie für Steuerungsrelais und Daten verwenden, um Zwischenergebnisse von Anweisungen oder andere Steuerungsinformationen zu speichern. Auf den Speicherbereich der Merker können Sie im Bit, Byte, Wort und Doppelwortformat zugreifen. Auf den Merkerspeicher ist sowohl der Lese- als auch der Schreibzugriff erlaubt.

Tabelle 5- 23 Absolute Adressierung für den Speicherbereich M

Bit	M[Byteadresse].[Bitadresse]	M26.7
Byte, Wort oder Doppelwort	M[Größe][Adresse des Anfangsbyte]	MB20, MW30, MD50

Temp (temporärer Speicher): Die CPU ordnet den temporären Speicher nach Bedarf zu. Die CPU ordnet dem Codebaustein den temporären Speicher zu und initialisiert die Speicheradressen auf 0, wenn der Codebaustein gestartet (bei einem OB) bzw. aufgerufen wird (bei einer FC oder einem FB).

Der temporäre Speicher ist dem Merkerspeicher (M) ähnlich. Es gibt jedoch einen wichtigen Unterschied: Der Merkerspeicher hat "globalen" Geltungsbereich, während der temporäre Speicher "lokalen" Geltungsbereich hat:

- Speicherbereich der Merker: Jeder OB, FB und jede FC kann auf die Daten im Speicherbereich der Merker zugreifen, d.h. die Daten stehen global für alle Elemente des Anwenderprogramms zur Verfügung.
- Temporärer Speicher: Die CPU beschränkt den Zugriff auf die Daten des temporären Speichers auf den OB, FB oder die FC, der bzw. die die Adresse im temporären Speicher erstellt bzw. deklariert hat. Adressen im temporären Speicher bleiben lokal und werden nicht von verschiedenen Codebausteinen gemeinsam genutzt, auch dann nicht, wenn der Codebaustein einen anderen Codebaustein aufruft. Beispiel: Wenn ein OB eine FC aufruft, kann die FC nicht auf den temporären Speicher des OBs, von dem sie aufgerufen wurde, zugreifen.

Die CPU stellt für jede OB-Prioritätsklasse temporären (lokalen) Speicher zur Verfügung:

- 16 KB für Anlauf und Programmzyklus, einschließlich der zugehörigen FBs und FCs
- 6 KB für jeden zusätzlichen Alarmereignis-Thread einschließlich der zugehörigen FBs und FCs

Auf den temporären Speicher können Sie nur über die symbolische Adressierung zugreifen.

DB (Datenbaustein): Nutzen Sie Datenbausteine zum Speichern verschiedener Arten von Daten, auch Zwischenergebnisse einer Anweisung oder andere Steuerungsparameter für FBs und Datenstrukturen für viele Anweisungen wie Zeiten und Zähler. Auf Datenbausteine können Sie im Bit, Byte, Wort und Doppelwortformat zugreifen. Für nicht schreibgeschützte Datenbausteine ist sowohl der Lese- als auch der Schreibzugriff erlaubt. Für schreibgeschützte Datenbausteine ist nur der Lesezugriff erlaubt.

Tabelle 5- 24 Absolute Adressierung für den Speicherbereich DB

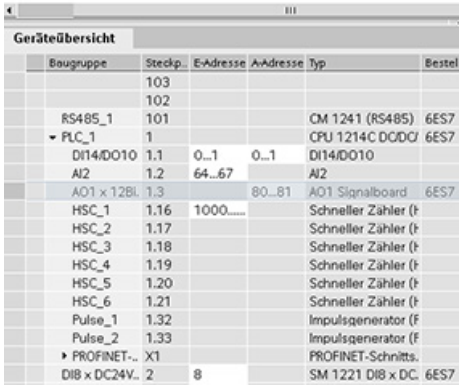
Bit	DB[Datenbausteinnummer].DBX[Byteadresse].[Bitadresse]	DB1.DBX2.3
Byte, Wort oder Doppelwort	DB[Datenbausteinnummer].DB [Größe].[Anfangsbyteadresse]	DB1.DBB4, DB10.DBW2, DB20.DBD8

Hinweis

Wenn Sie in KOP oder FUP eine absolute Adresse angeben, stellt STEP 7 dieser Adresse das Zeichen "%" voran, um kenntlich zu machen, dass es sich um eine absolute Adresse handelt. Bei der Programmierung können Sie eine absolute Adresse entweder mit dem oder ohne das Zeichen "%" eingeben (Beispiel: %E0.0 oder E.0). Wenn Sie das Zeichen "%" weglassen, fügt STEP 7 es ein.

In SCL muss das Zeichen "%" vor der Adresse eingegeben werden, um kenntlich zu machen, dass es sich um eine absolute Adresse handelt. Fehlt das "%", erzeugt STEP 7 beim Übersetzen einen undefinierten Variablenfehler.

E/A in der CPU und in E/A-Modulen konfigurieren



Wenn Sie eine CPU und E/A-Module in Ihre Gerätekonfiguration einfügen, weist STEP 7 E- und A-Adressen automatisch zu. Sie können die voreingestellte Adressierung ändern, indem Sie in der Gerätekonfiguration das Adressfeld auswählen und neue Zahlen eingeben.

- Digitale Eingänge und Ausgänge werden in STEP 7 in Gruppen zu 8 Punkten (1 Byte) zugewiesen, unabhängig davon, ob das Modul alle Ein- bzw. Ausgänge (Punkte) nutzt oder nicht.
- STEP 7 weist analoge Eingänge und Ausgänge in 2er-Gruppen zu, wobei jeder analoge Punkt 2 Bytes (16 Bit) belegt.

Das Bild zeigt ein Beispiel für eine CPU 1214C mit zwei SMs und einem SB. In diesem Beispiel können Sie in der Adresse des Moduls DE8 die 8 in 2 ändern. Das Werkzeug unterstützt Sie, indem Adressbereiche, die die falsche Größe haben oder mit anderen Adressen in Konflikt stehen, geändert werden.

5.3 Verarbeitung von Analogwerten

Analoge Signalmodule liefern Eingangssignale oder erwarten Ausgangswerte, die entweder einen Spannungsbereich oder einen Strombereich darstellen. Diese Bereiche sind ±10 V, ±5 V, ±2,5 V oder 0 bis 20 mA. Die von den Modulen ausgegebenen Werte sind ganzzahlige Werte. Dabei stellt 0 bis 27648 den Nennbereich für Strom und -27648 bis 27648 den für Spannung dar. Werte außerhalb dieser Bereiche stellen entweder einen Überlauf oder einen Unterlauf dar. Einzelheiten zu den Wertetypen, die außerhalb des Bereichs liegen, finden Sie in den Tabellen Darstellung Analogeingang (Seite 1254) und Darstellung Analogausgang (Seite 1255).

In Ihrem Steuerungsprogramm müssen Sie diese Werte wahrscheinlich in physikalischen Einheiten verwenden, um beispielsweise Volumen, Temperatur, Gewicht oder einen anderen quantitativen Wert darzustellen. Hierfür müssen Sie den Analogeingang zunächst in eine Realzahl (Gleitpunktwert) zwischen 0,0 und 1,0 normalisieren. Dann müssen Sie den Wert für den Mindest- und Höchstwert der darzustellenden physikalischen Einheit skalieren. Bei Werten in physikalischen Einheiten, die Sie in einen Analogausgangswert umwandeln müssen, normalisieren Sie zunächst den Wert in physikalischen Einheiten in einen Wert zwischen 0,0 und 1,0. Danach skalieren Sie den Wert zwischen 0 und 27648 oder -27648 und 27648, je nach Bereiche des Analogmoduls. STEP 7 bietet zu diesem Zweck die Anweisungen NORM_X und SCALE_X (Seite 284). Sie können auch die Anweisung CALCULATE (Seite 247) verwenden, um die Analogwerte (Seite 41) zu skalieren.

Beispiel: Analogwertverarbeitung

Betrachten wir beispielsweise einen analogen Eingang mit einem Strombereich von 0 - 20 mA. Das analoge Eingangsmodul gibt Messwerte im Bereich von 0 bis 24768 zurück. Gehen wir davon aus, dass dieser analoge Eingangswert für die Messung eines Temperaturbereichs von 50 °C bis 100 °C verwendet wird. Folgende Beispielwerte würden dann den angegebenen Werten entsprechen:

Analogeingangswert	Technische Einheiten
0	50 °C
6192	62,5 °C
12384	75 °C
18576	87,5 °C
24768	100 °C

Die physikalischen Einheiten in diesem Beispiel würden anhand des analogen Eingangswerts wie folgt berechnet:

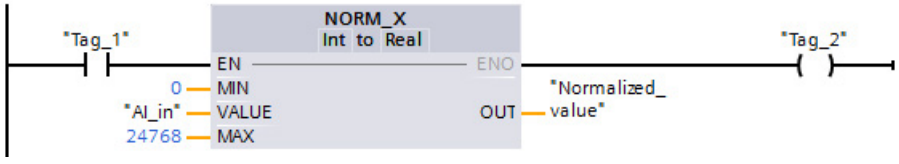
Wert in physikalischen Einheiten = 50 + (Analogeingangswert) * (100 - 50) / (24768 - 0)

Allgemein würde die Gleichung wie folgt aussehen:

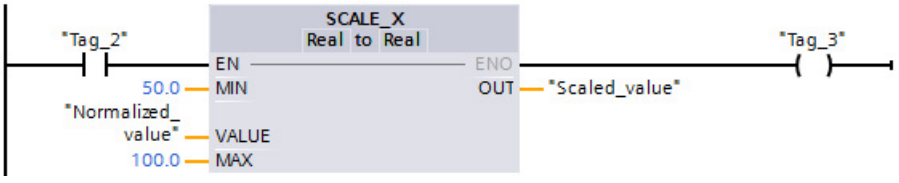
Wert in physikalischen Einheiten = (unterer Bereich der physikalischen Einheiten) +
(Analogeingangswert) *
(oberer Bereich der physikalischen Einheiten - unterer Bereich der physikalischen Einheiten) /
(maximaler Analogeingangsbereich - minimaler Analogeingangsbereich)

In PLC-Anwendungen wird üblicherweise der Analogeingangswert in einen Gleitpunktwert zwischen 0.0 und 1.0 normiert. Das Ergebnis würde dann in einen Gleitpunktwert im Bereich der physikalischen Einheiten umgewandelt. Der Einfachheit halber nutzen die folgenden KOP-Anweisungen konstante Bereichswerte; es können jedoch auch Variablen verwendet werden.

Netzwerk 1



Netzwerk 2



5.4 Datentypen

Datentypen geben die Größe eines Datenelements und die Art der Auswertung der Daten an. Jeder Anweisungsparameter unterstützt mindestens einen Datentyp, einige Parameter unterstützen mehrere Datentypen. Halten Sie den Mauszeiger auf dem Parameterfeld einer Anweisung, damit Ihnen angezeigt wird, welche Datentypen für den jeweiligen Parameter unterstützt werden.

Ein Formalparameter ist die Kennung an einer Anweisung, die die Adresse der von der Anweisung zu verwendenden Daten angibt (Beispiel: Eingang IN1 einer Anweisung ADD). Ein Aktualparameter ist die Adresse (mit vorangestelltem Zeichen "%") oder Konstante, die die von der Anweisung zu verwendenden Daten enthält (Beispiel: %MD400 "Anzahl_Widgets"). Der Datentyp des von Ihnen angegebenen Aktualparameters muss einem der von der Anweisung angegebenen unterstützten Datentypen des Formalparameters entsprechen.

Wenn Sie einen Aktualparameter angeben, müssen Sie entweder eine Variable (Symbol) oder eine absolute (direkte) Adresse angeben. Variablen weisen einem symbolischen Namen (Variablennamen) einen Datentyp, Speicherbereich, Speicherversatz und Kommentar zu und können entweder im PLC-Variableneditor oder im Schnittstelleneditor eines Bausteins (OB, FC, FB und DB) angegeben werden. Wenn Sie eine absolute Adresse eingeben, die keine zugewiesene Variable hat, müssen Sie eine Größe verwenden, die einem unterstützen Datentyp entspricht, dann wird bei der Eingabe eine Standardvariable angelegt.

Alle Datentypen, mit Ausnahme von String, Struct, Array und DTL sind im PLC-Variableneditor und in den Bausteinschnittstelleneditoren verfügbar. Die Datentypen String, Struct, Array und DTL stehen nur in den Bausteinschnittstelleneditoren zur Verfügung. Für viele der Eingangsparameter können Sie auch einen konstanten Wert eingeben.

- Bit und Bitfolgen (Seite 121): Bool (Boolescher Wert oder Bitwert), Byte (8-Bit-Bytewert), Word (16-Bit-Wortwert), DWord (32-Bit-Doppelwortwert)
- Ganzzahl (Seite 122)
  - USInt (vorzeichenlose 8-Bit-Ganzzahl), SInt (vorzeichenbehaftete 8-Bit-Ganzzahl),
  - UInt (vorzeichenlose 16-Bit-Ganzzahl), Int (vorzeichenbehaftete 16-Bit-Ganzzahl),
  - UDInt (vorzeichenlose 32-Bit-Ganzzahl), DInt (vorzeichenbehaftete 32-Bit-Ganzzahl),
- Gleitpunktzahl/Realzahl (Seite 122): Real (32-Bit-Real- oder Gleitpunktwert), LReal (64-Bit-Real- oder Gleitpunktwert)
- Uhrzeit und Datum (Seite 123): Time (32-Bit-IEC-Zeitwert), Date (16-Bit-Datumswert), TOD (32-Bit-Uhrzeitwert), DTL (12-Byte-Datums- und-Uhrzeitstruktur)
- Zeichen und Zeichenfolge (Seite 125): Char (8-Bit-Einzelzeichen), String (Zeichenfolge variabler Länge mit bis zu 254 Zeichen)
- Array (Seite 127)
- Datenstruktur (Seite 128): Struct
- PLC-Datentyp (Seite 128)
- Pointer (Seite 129): Pointer, Any, Variant

Ferner wird das folgende numerische BCD-Format von den Umwandlungsanweisungen unterstützt, obwohl es nicht als Datentyp zur Verfügung steht.

Tabelle 5- 25 Größe und Bereich des BCD-Formats

Format	Größe (Bit)	Bereich	Beispiel für konstanten Eintrag
BCD16	16	-999 bis 999	123, -123
BCD32	32	-9999999 bis 9999999	1234567, -1234567

5.4.1 Datentypen Bool, Byte, Word und DWord

Tabelle 5- 26 Bit- und Bitfolge-Datentypen

Daten-typ	Bit-größe	Zahlen-typ	Zahlen-bereich	Beispiele für Konstanten	Beispiele für Adressen
Bool	1	Boolesch	FALSCH oder WAHR	WAHR, 1,	I1.0 Q0.1 M50.7 DB1.DBX2.3 Variablenname
		Binär	0 oder 1	0, 2#0	
		Oktal	8#0 oder 8#1	8#1	
		Hexadezimal	16#0 oder 16#1	16#1	
Byte	8	Binär	2#0 bis 2#11111111	2#00001111	IB2 MB10 DB1.DBB4 Variablenname
		Vorzeichenlose Ganzzahl	0 bis 255	15	
		Oktal	8#0 bis 8#377	8#17	
		Hexadezimal	B#16#0 bis B#16#FF	B#16#F, 16#F	
Word	16	Binär	2#0 bis 2#1111111111111111	2#1111000011110000	MW10 DB1.DBW2 Variablenname
		Vorzeichenlose Ganzzahl	0 bis 65535	61680	
		Oktal	8#0 bis 8#177777	8#170360	
		Hexadezimal	W#16#0 bis W#16#FFFF, 16#0 bis 16#FFFF	W#16#F0F0, 16#F0F0	
DWord	32	Binär	2#0 bis 2#11111111111111111111111111111111	2#11110000111111110000111111	MD10 DB1.DBD8 Variablenname
		Vorzeichenlose Ganzzahl	0 bis 4294967295	15793935	
		Oktal	8#0 bis 8#37777777777	8#74177417	
		Hexadezimal	DW#16#0000_0000 bis DW#16#FFFF_FFFF, 16#0000_0000 bis 16#FFFF_FFFF	DW#16#F0FF0F, 16#F0FF0F	

5.4.2 Ganzzahlige Datentypen

Tabelle 5- 27 Ganzzahlige Datentypen (U = vorzeichenlos, S = kurz, D = doppelt)

Datentyp	Bitgröße	Zahlenbereich	Beispiele für Konstanten	Beispiele für Adressen
USInt	8	0 bis 255	78, 2#01001110	MB0, DB1.DBB4, Variablenname
SInt	8	-128 bis 127	+50, 16#50	
UInt	16	0 bis 65.535	65295, 0	MW2, DB1.DBW2, Variablenname
Int	16	-32.768 bis 32.767	30000, +30000	
UDInt	32	0 bis 4.294.967.295	4042322160	MD6, DB1.DBD8, Variablenname
DInt	32	-2.147.483.648 bis 2.147.483.647	-2131754992	

5.4.3 Gleitpunktzahl/Realzahl-Datentypen

Realzahlen (bzw. Gleitpunktzahlen) werden als einfachgenaue 32-Bit-Zahlen (Real) oder als doppeltgenaue 64-Bit-Zahlen (LReal) dargestellt, wie in der Norm ANSI/IEEE7541985 beschrieben. Einfachgenaue Gleitpunktzahlen sind bis zu 6 signifikante Ziffern genau und doppeltgenaue Gleitpunktzahlen sind bis zu 15 signifikante Ziffern genau. Sie können maximal 6 (Real) oder 15 (LReal) signifikante Ziffern angeben, wenn Sie eine Gleitpunktkonstante eingeben.

Tabelle 5- 28 Gleitpunktzahl/Realzahl-Datentypen (L = lang)

Datentyp	Bitgröße	Zahlenbereich	Beispiele für Konstanten	Beispiele für Adressen
Real	32	-3.402823e+38 bis -1.175 495e-38, ±0, +1.175 495e-38 bis +3.402823e+38	123.456, -3.4, 1,0e-5	MD100, DB1.DBD8, Variablenname
LReal	64	-1.7976931348623158e+308 bis -2.2250738585072014e-308, ±0, +2.2250738585072014e-308 bis +1.7976931348623158e+308	12345,123456789e40, 1,2E+40	DB-Name.Var-Name Regeln: • Direkte Adressierung wird nicht unterstützt • Kann in der Bausteinschnittstelle eines OBs, FBs oder FCs zugewiesen werden

Bei Berechnungen, die eine lange Reihe von Werten einschließlich sehr großen und sehr kleinen Zahlen benötigen, kann es zu ungenauen Ergebnissen kommen. Dies kann auftreten, wenn sich die Zahlen um 10 hoch x unterscheiden, wobei x > 6 (Real) oder 15 (LReal) ist. Beispiel (Real): 100 000 000 + 1 = 100 000 000.

5.4.4 Uhrzeit- und Datums-Datentypen

Tabelle 5- 29 Uhrzeit- und Datums-Datentypen

Datentyp	Größe	Bereich	Beispiele für konstanten Eintrag
Time	32 Bits	T#-24d_20h_31m_23s_648ms bis T#24d_20h_31m_23s_647ms Gespeichert als: -2.147.483.648 ms bis +2.147.483.647 ms	T#5m_30s T#1d_2h_15m_30s_45ms TIME#10d20h30m20s630ms 500h10000ms 10d20h30m20s630ms
Date	16 Bits	D#1990-1-1 bis D#2168-12-31	D#2009-12-31 DATE#2009-12-31 2009-12-31
Time_of_Day	32 Bit	TOD#0:0:0.0 bis TOD#23:59:59.999	TOD#10:20:30.400 TIME_OF_DAY#10:20:30.400 23:10:1
DTL (Date and Time Long)	12 Byte	Min.: DTL#1970-01-01-00:00:00.0 Max.: DTL#2262-04-11:23:47:16.854 775 807	DTL#2008-12-16-20:30:20.250

Zeit

TIME-Daten werden als vorzeichenbehaftete doppelte Ganzzahl gespeichert, die als Millisekunden ausgewertet wird. Das Editorformat kann Angaben für Tag (d), Stunden (h), Minuten (m), Sekunden (s) und Millisekunden (ms) verarbeiten.

Es ist nicht erforderlich, alle Zeiteinheiten anzugeben. So sind z. B. T#5h10s und 500h gültig.

Der Gesamtwert aller angegebenen Werte in den einzelnen Einheiten darf den oberen bzw. unteren Grenzwert in Millisekunden für den Zeit-Datentyp nicht über- bzw. unterschreiten (-2.147.483.648 ms bis +2.147.483.647 ms).

Datum

DATE-Daten werden als vorzeichenloser Ganzzahlenwert gespeichert, der als Anzahl der Tage ausgewertet wird, die zum Basisdatum 01.01.1990 addiert werden, um das angegebene Datum zu erhalten. Das Editorformat muss ein Jahr, einen Monat und einen Tag angeben.

TOD

TOD-Daten (TIME_OF_DAY) werden als vorzeichenlose doppelte Ganzzahl gespeichert, die als Anzahl der Millisekunden seit Mitternacht für die angegebene Uhrzeit ausgewertet wird (Mitternacht = 0 ms). Es müssen die Stunden (24 h/Tag), Minuten und Sekunden angegeben werden. Die Angabe der Nachkommastellen der Sekunde ist optional.

DTL

Der DTL-Datentyp (Date and Time Long) nutzt eine 12-Byte-Struktur, um Angaben zum Datum und zur Uhrzeit zu speichern. Sie können den Datentyp DTL entweder im temporären Speicher eines Bausteins oder in einem DB definieren. Für alle Komponenten muss in der Spalte für den Startwert im DB-Editor ein Wert eingegeben werden.

Tabelle 5- 30 Größe und Bereich von DTL

Länge (Bytes)	Format	Wertebereich	Beispiel für Werteingabe
12	Uhrzeit und Kalender Jahr-Monat-Tag:Stunde:Minute:Sekunde.Nanosekunden	Min.: DTL#1970-01-01-00:00:00.0 Max.: DTL#2554-12-31-23:59:59.999 999 999	DTL#2008-12-16-20:30:20.250

Jede Komponente des Datentyps DTL enthält einen unterschiedlichen Datentyp und Wertebereich. Der Datentyp eines angegebenen Werts muss dem Datentyp der jeweiligen Komponenten entsprechen.

Tabelle 5- 31 Elemente der DTL-Struktur

Byte	Komponente	Datentyp	Wertebereich
0	Jahr	UINT	1970 bis 2554
1			
2	Monat	USINT	1 bis 12
3	Tag	USINT	1 bis 31
4	Wochentag [1]	USINT	1 (Sonntag) bis 7 (Samstag)[1]
5	Stunde	USINT	0 bis 23
6	Minute	USINT	0 bis 59
7	Sekunde	USINT	0 bis 59
8	Nanosekunden	UDINT	0 bis 999.999.999
9			
10			
11			

[1] Das Format Jahr-Monat-Tag:Stunde:Minute:Sekunde.Nanosekunde enthält nicht den Wochentag.

5.4.5 Zeichen- und Zeichenfolge-Datentypen

Tabelle 5- 32 Zeichen- und Zeichenfolge-Datentypen

Datentyp	Größe	Bereich	Beispiel für konstanten Eintrag
Char	8 Bit	16#00 bis 16#FF	'A', 't', '@', 'ä', '∑'
WChar	16 Bits	16#0000 bis 16#FFFF	'A', 't', '@', 'ä', '∑', Asiatische Zeichen, kyrillische Zeichen und andere
String	n+ 2 Byte	n = (0 bis 254 Bytes)	"ABC"
WString	n+ 2 Wörter	n = (0 bis 65534 Wörter)	"ä123@XYZ.COM"

Char und WChar

Ein Char (Zeichen) belegt ein Byte im Speicher und speichert ein einzelnes Zeichen im ASCII-Format, einschließlich der erweiterten ASCII-Zeichencodes. Ein WChar belegt ein Wort im Speicher und kann eine beliebige Doppelbyte-Zeichendarstellung enthalten.

In der Editorsyntax wird ein einzelnes Anführungszeichen vor und hinter dem Zeichen eingegeben. Sie können sichtbare Zeichen und Steuerzeichen verwenden.

String und WString

Die CPU unterstützt den Datentyp String zum Speichern einer Folge von Einzelbyte-Zeichen. Der Datentyp String enthält die Gesamtzeichenzahl (Anzahl der Zeichen in der Zeichenkette) und die tatsächliche Zeichenzahl. Der Datentyp String bietet bis zu 256 Bytes zum Speichern der maximalen Gesamtzeichenzahl (1 Byte), der tatsächlichen Zeichenzahl (1 Byte) und bis zu 254 Bytes in der Zeichenfolge. Jedes Byte in einem Datentyp String kann einen beliebigen Wert von 16#00 bis 16#FF annehmen.

Der Datentyp WString stellt längere Zeichenfolgen für Werte mit einem Wort (Doppelbyte) bereit. Das erste Wort enthält die maximale Gesamtzeichenzahl; das nächste Wort enthält die tatsächliche Gesamtzeichenzahl, die folgende Zeichenfolge kann bis zu 65334 Wörter enthalten. Jedes Wort in einem Datentyp WString kann einen beliebigen Wert von 16#0000 bis 16#FFFF annehmen.

Sie können literale Zeichenketten (Konstanten) für Anweisungsparameter vom Typ IN in einzelnen Hochkommata angeben. ‘ABC’ zum Beispiel ist eine Zeichenkette aus drei Zeichen, die als Eingang für Parameter IN der Anweisung S_CONV genutzt werden kann. Sie können auch Zeichenkettenvariablen erstellen, indem Sie in den Bausteinschnittstelleneditoren für OB, FC, FB und DB den Datentyp "String" oder "WString" auswählen. Im PLC-Variableneditor können Sie keine Zeichenkette erstellen.

Sie können die maximale Zeichenfolgegröße in Bytes (String) oder in Wörtern (WString) mithilfe von eckigen Klammern nach dem Schlüsselwort "String" oder "WString" angeben, nachdem Sie den jeweiligen Datentyp in der Klappliste für die Datentypauswahl ausgewählt haben. Beispiel: "MeinString String[10]" gibt eine maximale Größe von 10 Byte für die Zeichenfolge MeinString an. Wenn Sie die eckigen Klammern mit der Angabe der maximalen Größe weglassen, wird die Größe 254 für Datentyp String und 65534 für Datentyp WString angenommen. "MyWString WString[1000]" stünde somit für einen Datentyp WString mit 1000 Wörtern.

Das folgende Beispiel zeigt eine Zeichenfolge mit der maximalen Zeichenzahl 10 und der tatsächlichen Zeichenzahl 3. Die Zeichenfolge enthält somit 3 Zeichen mit je einem Byte, kann aber bis auf 10 Zeichen mit je einem Byte erweitert werden.

Tabelle 5- 33 Beispiel für einen Datentyp String

Gesamtzeichenzahl	Tatsächliche Zeichenzahl	Zeichen 1	Zeichen 2	Zeichen 3	...	Zeichen 10
10	3	'C' (16#43)	'A' (16#41)	'T' (16#54)	...	-
Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	...	Byte 11

Das folgende Beispiel zeigt eine Zeichenfolge des Datentyps WString mit einer maximalen Wortzahl 500 und einer tatsächlichen Wortzahl 300. Die Zeichenfolge enthält somit 300 Zeichen mit je einem Wort, kann aber bis auf 500 Zeichen mit je einem Wort erweitert werden.

Tabelle 5- 34 Beispiel für einen Datentyp WString

Gesamtzeichenzahl	Tatsächliche Zeichenzahl	Zeichen 1	Zeichen 2 bis 299	Zeichen 300	...	Zeichen 500
500	300	'ä' (16#0084)	ASCII-Zeichenwörter	'M' (16#004D)	...	-
Wort 0	Wort 1	Wort 2	Wörter 3 bis 300	Wort 301	...	Wort 501

ASCII-Steuerzeichen können in Char-, WChar-, String- und WString-Daten verwendet werden. Die folgende Tabelle zeigt Beispiele für die Syntax von Steuerzeichen.

Tabelle 5- 35 Gültige ASCII-Steuerzeichen

Steuerzeichen	ASCII-Hexadezimalwert (Char)	ASCII-Hexadezimalwert (WChar)	Regelungsfunktion	Beispiele
$L oder $l	16#0A	16#000A	Zeilenvorschub	'$LText', '$0AText'
$N oder $n	16#0A und 16#0D	16#000A und 16#000D	Zeilenumbruch Die neue Zeile zeigt zwei Zeichen in der Zeichenfolge.	'$NText', '$0A$0DText'
$P oder $p	16#0C	16#000C	Formularvorschub	'$PText', '$0CText'
$R oder $r	16#0D	16#000D	Zeilenschaltung (CR)	'$RText','$0DText'
$T oder $t	16#09	16#0009	Tabulator	'$TText', '$09Text'
$$	16#24	16#0024	Dollarzeichen	'100$$', '100$24'
$'	16#27	16#0027	Einzelnes Hochkomma	'$'Text$'','$27Text$27'

5.4.6 Datentyp ARRAY

Arrays

Sie können ein Array erstellen, das mehrere Elemente des gleichen Datentyps enthält. Arrays können in der Bausteinschnittstelle von OB, FC, FB und DB angelegt werden. Im PLC-Variableneditor können Sie kein Array erstellen.

Um ein Array in der Bausteinschnittstelle zu erstellen, benennen Sie das Array und wählen den Datentyp "Array [lo .. hi] of type", dann ändern Sie "lo", "hi" und "type" wie folgt:

- lo - Anfangsindex (niedrigster Index) für Ihr Array
- hi - Abschlussindex (höchster Index) für Ihr Array
- type - einer der Datentypen wie BOOL, SINT, UDINT

Tabelle 5- 36 Regeln für den Datentyp ARRAY

Datentyp	Array-Syntax		
ARRAY	Name [index1_min..index1_max, index2_min..index2_max] of <Datentyp> • Alle Array-Elemente müssen den gleichen Datentyp haben. • Der Index kann negativ sein, doch der untere Grenzwert muss kleiner oder gleich dem oberen Grenzwert sein. • Arrays können bis zu sechs Dimensionen umfassen. • Multidimensionale Indexdeklarationen min..max werden durch Kommazeichen voneinander getrennt. • Verschachtelte Arrays bzw. Arrays aus Arrays sind nicht zulässig. • Dei Speichergröße eines Arrays = (Größe eines Elements * Gesamtzahl der Elemente im Array)		
	Arrayindex	**Gültige Index-Datentypen**	**Regeln für den Arrayindex**
	Konstante oder Variable	USInt, SInt, UInt, Int, UDInt, DInt	• Grenzwerte: -32768 bis +32767 • Gültig: Gemischte Konstanten und Variablen • Gültig: Konstante Ausdrücke • Ungültig: Variable Ausdrücke

Beispiel: Array-Deklarationen	ARRAY[1..20] of REAL	Eine Dimension, 20 Elemente
	ARRAY[-5..5] of INT	Eine Dimension, 11 Elemente
	ARRAY[1..2, 3..4] of CHAR	Zwei Dimensionen, 4 Elemente
Beispiel: Array-Adressen	ARRAY1[0]	ARRAY1 Element 0
	ARRAY2[1,2]	ARRAY2 Element [1,2]
	ARRAY3[i,j]	Wenn i = 3 und j = 4, dann wird ARRAY3 Element [3, 4] angesprochen

5.4.7 Datentyp Struktur

Mit dem Datentyp "Struct" können Sie eine aus anderen Datentypen bestehende Datenstruktur definieren. Der Datentyp Struct kann genutzt werden, um eine Gruppe zusammengehöriger Prozessdaten als eine Dateneinheit zu behandeln. Der Datentyp Struct wird benannt und die interne Datenstruktur im Datenbausteineditor oder in einem Bausteinschnittstelleneditor deklariert.

Arrays und Strukturen können auch zu einer größeren Struktur zusammengefügt werden. Eine Struktur kann bis zu acht Ebenen tief verschachtelt werden. Sie können z. B. eine Struktur aus Strukturen erstellen, die wiederum Arrays enthalten.

5.4.8 PLC-Datentyp

Im PLC-Datentypeditor können Sie Datenstrukturen definieren, die Sie mehrmals in Ihrem Programm verwenden können. Sie erstellen einen PLC-Datentyp durch Öffnen von "PLC-Datentypen" in der Projektnavigation und Doppelklick auf den Befehl "Neuen Datentyp hinzufügen". Klicken Sie zweimal einzeln auf den neu erstellen PLC-Datentyp, um den Standardnamen zu ändern. Doppelklicken Sie dann, um den PLC-Datentypeditor zu öffnen.

Zum Erstellen einer benutzerdefinierten PLC-Datentypstruktur können Sie auf dieselbe Weise vorgehen wie im Datenbausteineditor. Fügen Sie für alle zusätzlich erforderlichen Datentypen neue Zeilen ein, um die gewünschte Datenstruktur anzulegen.

Wenn Sie einen PLC-Datentyp anlegen, erscheint der Name des neuen PLC-Datentyps in der Auswahl-Klappliste im DB-Editor und im Codebaustein-Schnittstelleneditor.

Mögliche Verwendung von PLC-Datentypen:

- PLC-Datentypen können direkt als Datentyp in einer Codebausteinschnittstelle oder in Datenbausteinen verwendet werden.
- PLC-Datentypen können als Vorlage für die Erstellung von mehreren globalen Datenbausteinen mit der gleichen Datenstruktur verwendet werden.

Ein PLC-Datentyp kann beispielsweise ein Rezept zum Mischen von Farben sein. Sie können diesen PLC-Datentyp dann mehreren Datenbausteinen zuweisen. In jedem Datenbaustein können die Variablen dann entsprechend angepasst werden, um eine bestimmte Farbe zu erstellen.

5.4.9 Pointer-Datentypen

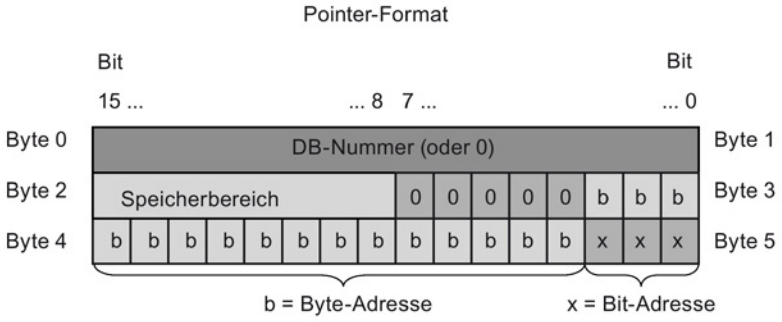
Die Pointer-Datentypen (Pointer, Any und Variant) können in den Bausteinschnittstellentabellen für FB- und FC-Codebausteine verwendet werden. Sie können in der Datentyp-Klappliste der Bausteinschnittstelle einen Pointer-Datentyp auswählen.

Der Datentyp Variant wird auch für Anweisungsparameter verwendet.

5.4.9.1 Pointer-Datentyp "Pointer"

Der Datentyp Pointer zeigt auf eine bestimmte Variable. Der Datentyp belegt 6 Bytes (48 Bits) im Speicher und kann die folgenden Informationen enthalten:

- DB-Nummer oder 0, wenn die Daten nicht in einem DB gespeichert werden
- Speicherbereich in der CPU
- Variablenadresse



Je nach Anweisung können Sie die folgenden drei Typen von Pointern deklarieren:

- Bereichsinterner Pointer: enthält Daten zur Adresse einer Variable
- Bereichsübergreifender Pointer: enthält Daten zum Speicherbereich und zur Adresse einer Variable
- DB-Pointer: enthält eine Datenbausteinnummer und die Adresse einer Variable

Tabelle 5- 37 Pointer-Typen:

Typ	Format	Beispieleintrag
Bereichsinterner Pointer	P#Byte.Bit	P#20.0
Bereichsübergreifender Pointer	P#Speicherbereich_Byte.Bit	P#M20.0
DB-Pointer	P#Datenbaustein.Datenelement	P#DB10.DBX20.0

Einen Parameter vom Typ Pointer können Sie ohne Präfix (P #) eingeben. Ihre Eingabe wird automatisch ins Pointer-Format umgewandelt.

Tabelle 5- 38 Codierung des Speicherbereichs in den Pointer-Daten:

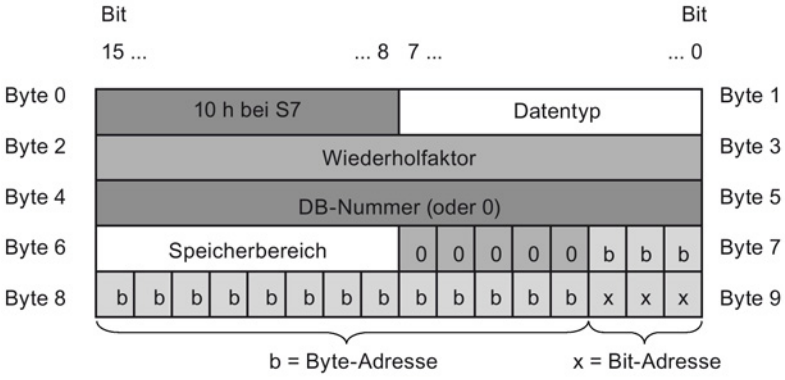
Hexadezimalcode	Datentyp	Beschreibung
b#16#81	E	Speicherbereich der Eingänge
b#16#82	A	Speicherbereich der Ausgänge
b#16#83	M	Speicherbereich der Merker
b#16#84	DBX	Datenbaustein
b#16#85	DIX	Instanz-Datenbaustein
b#16#86	L	Lokaldaten
b#16#87	V	Vorherige Lokaldaten

5.4.9.2 Pointer-Datentyp "Any"

Der Pointer-Datentyp ANY ("Any") zeigt auf den Anfang eines Datenbereichs und gibt dessen Länge an. Der Pointer ANY belegt 10 Bytes im Speicher und kann die folgenden Informationen enthalten:

- Datentyp: Datentyp der Datenelemente
- Wiederholfaktor: Anzahl der Datenelemente
- DB-Nummer: Datenbaustein, in dem Datenelemente gespeichert werden
- Speicherbereich: Speicherbereich der CPU, in der die Datenelemente gespeichert werden
- Anfangsadresse: "Byte.Bit"-Anfangsadresse der Daten

Die folgende Abbildung zeigt die Struktur des Pointes ANY:



Ein Pointer kann keine ANY-Strukturen erkennen. Er kann nur lokalen Variablen zugewiesen werden.

Tabelle 5- 39 Format und Beispiele des Pointers ANY:

Format	Beispieleintrag	Beschreibung
P#Datenbaustein.Speicherbereich Datenadresse Typ Nummer	P#DB 11.DBX 20.0 INT 10	10 Wörter im globalen DB 11 mit Beginn an DBB 20.0
P#Speicherbereich Datenadresse Typ Nummer	P#M 20.0 BYTE 10	10 Byte mit Beginn an MB 20.0
	P#E 1.0 BOOL 1	Eingang E1.0

Tabelle 5- 40 Codierung des Datentyps im Pointer ANY

Hexadezimalcode	Datentyp	Beschreibung
b#16#00	Null	Null-Pointer
b#16#01	Bool	Bits
b#16#02	Byte	Bytes, 8 Bits
b#16#03	Char	8-Bit-Zeichen
b#16#04	Word	16-Bit-Wort
b#16#05	Int	16-Bit-Ganzzahl
b#16#37	SInt	8-Bit-Ganzzahl
b#16#35	UInt	vorzeichenlose 16-Bit-Ganzzahl
b#16#34	USInt	vorzeichenlose 8-Bit-Ganzzahl
b#16#06	DWord	32-Bit-Doppelwort
b#16#07	DInt	32-Bit-Ganzzahl
b#16#36	UDInt	vorzeichenlose 32-Bit-Ganzzahl
b#16#08	Real	32-Bit-Gleitpunkt
b#16#0B	Time	Zeit
b#16#13	String	Zeichenfolge

Tabelle 5- 41 Codierung des Speicherbereichs im Pointer ANY:

Hexadezimalcode	Speicherbereich	Beschreibung
b#16#81	E	Speicherbereich der Eingänge
b#16#82	A	Speicherbereich der Ausgänge
b#16#83	M	Speicherbereich der Merker
b#16#84	DBX	Datenbaustein
b#16#85	DIX	Instanz-Datenbaustein
b#16#86	L	Lokaldaten
b#16#87	V	Vorherige Lokaldaten

5.4.9.3 Pointer-Datentyp "Variant"

Der Datentyp Variant kann auf Variablen verschiedener Datentypen oder Parameter verweisen. Der Pointer Variant kann auf Strukturen und einzelne Strukturkomponenten zeigen. Der Pointer Variant belegt keinen Platz im Speicher.

Tabelle 5- 42 Eigenschaften des Pointers Variant

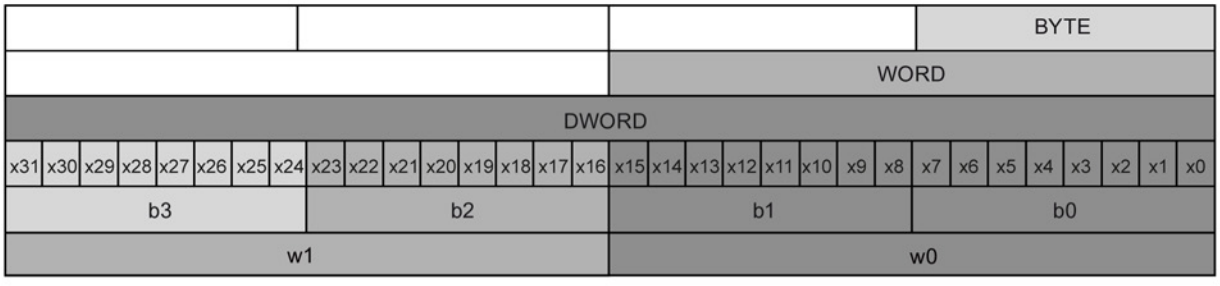
Länge (Byte)	Darstellung	Format	Beispieleintrag
0	Symbolischer	Operand	MeineVariable
		DB-Name.Strukturname.Elementname	MeinDB.Strukt1.Druck1
	Absoluter	Operand	%MW10
		DB-Nummer.Operand Typ Länge	P#DB10.DBX10.0 INT 12

5.4.10 Zugriff auf eine "Scheibe" eines Variablendatentyps

Auf PLC-Variablen und Datenbausteinvariablen kann je nach ihrer Größe auf Bit-, Byte- oder Wortebene zugegriffen werden. Die Syntax für den Zugriff auf eine Datenscheibe lautet wie folgt:

- "<PLC-Variablenname>".xn (Bitzugriff)
- "<PLC-Variablenname>".bn (Bytezugriff)
- "<PLC-Variablenname>".wn (Wortzugriff)
- "<Datenbausteinname>".<Variablenname>.xn (Bitzugriff)
- "<Datenbausteinname>".<Variablenname>.bn (Bytezugriff)
- "<Datenbausteinname>".<Variablenname>.wn (Wortzugriff)

Auf eine Variable mit Doppelwortgröße kann über die Bits 0-31, Bytes 0-3 oder die Wörter 0-1 zugriffen werden. Auf eine Variable mit Wortgröße kann über die Bits 0-15, Bytes 0-1 oder das Wort 0 zugegriffen werden. Auf eine Variable mit Bytegröße kann über die Bits 0-7 oder das Byte 0 zugegriffen werden. Bit-, Byte- und Wort-Scheiben können überall dort verwendet werden, wo Bits, Bytes oder Wörter erwartete Operanden sind.



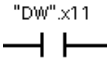
Hinweis

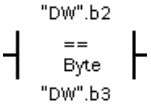
Auf folgende Datentypen kann über Scheiben zugegriffen werden: Byte, Char, Conn_Any, Date, DInt, DWord, Event_Any, Event_Att, Hw_Any, Hw_Device, HW_Interface, Hw_Io, Hw_Pwm, Hw_SubModule, Int, OB_Any, OB_Att, OB_Cyclic, OB_Delay, OB_WHINT, OB_PCYCLE, OB_STARTUP, OB_TIMEERROR, OB_Tod, Port, Rtm, SInt, Time, Time_Of_Day, UDInt, UInt, USInt und Word. Auf PLC-Variablen vom Datentyp Real kann über Scheiben zugegriffen werden, auf Datenbausteinvariablen vom Typ Real jedoch nicht.

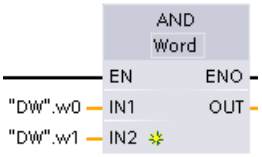
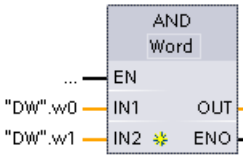
Beispiele

In der PLC-Variablentabelle ist "DW" eine deklarierte Variable vom Typ DWORD. Die Beispiele zeigen den Zugriff in Form von Bit-, Byte- und Wortscheiben:

	KOP	FUP	SCL

Bitzugriff			`IF "DW".x11 THEN`
			`...`
			`END_IF;`

Bytezugriff			`IF "DW".b2 = "DW".b3`
			`THEN`
			`...`
			`END_IF;`

Wortzugriff			`out:= "DW".w0 AND` `"DW".w1;`

5.4.11 Zugriff auf eine Variable mit einer AT-Überlagerung

Mit Hilfe der AT-Variablenüberlagerung können Sie mit einer überlagerten Deklaration eines unterschiedlichen Datentyps auf eine bereits deklarierte Variable eines Standardzugriffsbausteins zugreifen. Sie können beispielsweise die einzelnen Bits einer Variable vom Datentyp Byte, Word oder DWord mit einem Bool-Array adressieren.

Deklaration

Um einen Parameter zu überlagern, deklarieren Sie einen zusätzlichen Parameter direkt nach dem zu überlagernden Parameter und wählen den Datentyp "AT". Der Editor legt die Überlagerung an und Sie können dann den Datentyp, die Struktur oder das Array für die Überlagerung wählen.

Beispiel

Dieses Beispiel zeigt die Eingangsparameter eines FBs mit Standardzugriff. Die Bytevariable B1 wird mit einem Booleschen Array überlagert:

	B1		Byte	0.0
	OV	AT"B1"	Array[0..7] of Bool	0.0
	OV[0]		Bool	0.0
	OV[1]		Bool	0.1
	OV[2]		Bool	0.2
	OV[3]		Bool	0.3
	OV[4]		Bool	0.4
	OV[5]		Bool	0.5
	OV[6]		Bool	0.6
	OV[7]		Bool	0.7

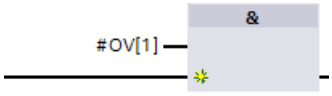
Ein weiteres Beispiel ist eine Variable vom Typ DWord, die mit einem Datentyp Struct überlagert wird, der ein Wort, ein Byte und zwei Boolesche Werte enthält:

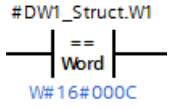
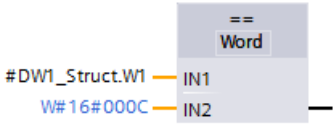
	DW1		DWord	2.0
	DW1_Struct	AT"DW1"	Struct	2.0
	W1		Word	0.0
	B1		Byte	2.0
	BO1		Bool	3.0
	BO2		Bool	3.1

Die Offset-Spalte der Bausteinschnittstelle zeigt die Lage der überlagerten Datentypen relativ zur Originalvariablen.

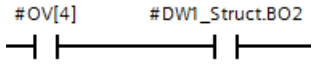
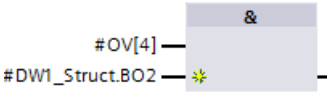
Die Überlagerungstypen können in der Programmlogik direkt angesprochen werden:

KOP	FUP	SCL

		`IF #OV[1] THEN`
		`...`
		`END_IF;`

		`IF #DW1_Struct.W1 = W#16#000C THEN`
		`...`
		`END_IF;`

		`out1 := #DW1_Struct.B1;`

		`IF #OV[4] AND #DW1_Struct.BO2 THEN`
		`...`
		`END_IF;`

Regeln

- Die Überlagerung von Variablen ist nur in FB- und FC-Bausteinen mit Standardzugriff (kein optimierter Zugriff) möglich.
- Sie können Parameter für alle Bausteintypen und alle Deklarationsabschnitte überlagern.
- Ein überlagerter Parameter kann wie jeder andere Bausteinparameter verwendet werden.
- Parameter vom Typ VARIANT können Sie nicht überlagern.
- Die Größe des überlagernden Parameters muss kleiner oder gleich der Größe des überlagerten Parameters sein.
- Die überlagernde Variable muss sofort nach der Variablen deklariert werden, die sie überlagert, und muss durch das Schlüsselwort "AT" als Ausgangsdatentyp gekennzeichnet sein.

5.5 Memory Card verwenden

Hinweis

Die CPU unterstützt nur die vorformatierten SIMATIC Memory Cards (Seite 1327).

Löschen Sie vor dem Kopieren von Programmen auf die formatierte Memory Card alle zuvor gespeicherten Programme von der Memory Card.

Sie können die Memory Card als Übertragungskarte oder als Programmkarte nutzen. Jedes Programm, das Sie auf die Memory Card kopieren, enthält alle Codebausteine und Datenbausteine, alle Technologieobjekte und die Gerätekonfiguration. Ein kopiertes Programm enthält **keine** geforcten Werte.

- Mit einer Übertragungskarte (Seite 139) kopieren Sie ein Programm in den internen Ladespeicher der CPU, ohne dafür STEP 7 zu verwenden. Nachdem Sie die Übertragungskarte gesteckt haben, löscht die CPU zunächst das Anwenderprogramm und alle geforcten Werte aus dem internen Ladespeicher und kopiert dann das Programm von der Übertragungskarte in den internen Ladespeicher. Wenn der Übertragungsvorgang beendet ist, müssen Sie die Übertragungskarte ziehen.

  Mit einer leeren Übertragungskarte können Sie auf eine passwortgeschützte CPU zugreifen, wenn Sie das Passwort verloren oder vergessen haben (Seite 149). Durch Stecken der leeren Übertragungskarte wird das passwortgeschützte Programm im internen Ladespeicher der CPU gelöscht. Dann können Sie ein neues Programm in die CPU laden.

- Eine Programmkarte (Seite 142) nutzen Sie als externen Ladespeicher für die CPU. Wenn Sie eine Programmkarte in die CPU stecken, wird der gesamte interne Ladespeicher der CPU gelöscht (Anwenderprogramm und ggf. geforcte Werte). Dann führt die CPU das Programm im externen Ladespeicher aus (Programmkarte). Beim Laden in eine CPU mit Programmkarte wird nur der externe Ladespeicher (die Programmkarte) aktualisiert.

  Weil der interne Ladespeicher der CPU beim Stecken der Programmkarte gelöscht wurde, **muss** die Programmkarte in der CPU gesteckt bleiben. Wenn Sie die Programmkarte ziehen, geht die CPU in den Betriebszustand STOP. (Die Fehler-LED blinkt, um anzuzeigen, dass die Programmkarte entfernt wurde.)

Das kopierte Programm auf einer Memory Card umfasst die Codebausteine, Datenbausteine, Technologieobjekte und die Gerätekonfiguration. Die Memory Card enthält **keine** geforcten Werte. Die geforcten Werte sind nicht Teil des Programms, doch sie werden im Ladespeicher abgelegt. Dies kann der interne Ladespeicher der CPU oder der externe Ladespeicher (eine Programmkarte) sein. Wenn eine Programmkarte in der CPU steckt, wendet STEP 7 die geforcten Werte nur auf den externen Ladespeicher auf der Programmkarte an.

Sie verwenden die Memory Card auch dann, wenn Sie Firmware-Updates (Seite 146) herunterladen.

5.5.1 Memory Card in die CPU stecken

ACHTUNG

Schutz von Memory Card und Schacht vor elektrostatischer Entladung

Elektrostatische Entladungen können die Memory Card oder den dafür vorgesehenen Schacht in der CPU beschädigen.

Sie müssen auf einem leitfähigen, geerdeten Boden stehen und/oder ein geerdetes Armband tragen, wenn Sie mit der Memory Card arbeiten. Die Memory Card ist in einem leitfähigen Behälter aufzubewahren.

Stellen Sie sicher, dass die Memory Card nicht schreibgeschützt ist. Schieben Sie dazu den Schutzschalter aus der Verriegelungsposition heraus.

⚠ WARNUNG

Vergewissern Sie sich vor dem Einlegen der Memory Card, dass die CPU keinen Prozess ausführt.

Wenn Sie eine Memory Card (unabhängig davon, ob als Programm-, Übertragungs- oder Firmware-Aktualisierungskarte genutzt) in eine laufende CPU stecken, geht die CPU sofort in den Betriebszustand STOP, was zu Prozessunterbrechung und dadurch zu Tod oder schweren Personenschäden führen kann.

Stellen Sie vor dem Stecken oder Ziehen einer Memory Card stets sicher, dass die CPU nicht aktiv eine Maschine oder einen Prozess steuert. Installieren Sie einen NOT-AUS-Schaltkreis für Ihre Anwendung bzw. Ihren Prozess.

Hinweis

Stecken Sie keine Übertragungskarte mit einem Programm V3.0 in S7-1200 CPUs V4.0.

Übertragungskarten mit einem Programm der Version 3.0 sind nicht mit S7-1200 CPUs der Version 4.0 kompatibel. Wenn Sie eine Memory Card mit einem Programm V3.0 stecken, wird dadurch ein CPU-Fehler verursacht.

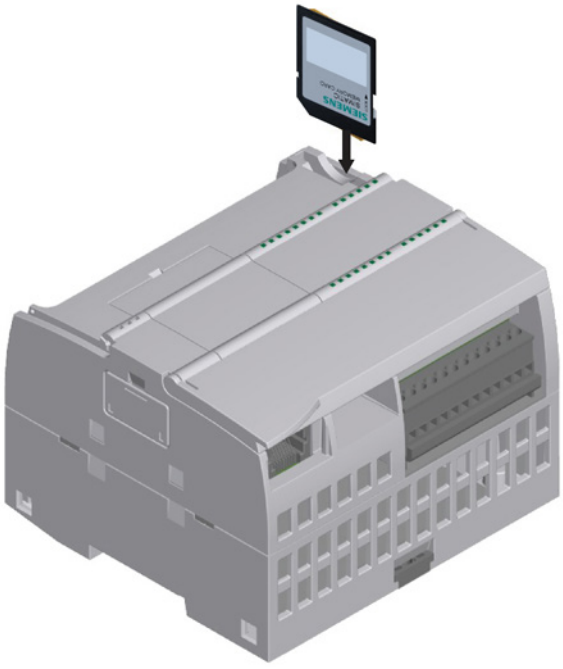
Wenn Sie eine Übertragungskarte (Seite 139) mit einer ungültigen Programmversion stecken, müssen Sie die Karte entnehmen und die CPU von STOP in RUN versetzen, ein Urlöschen (MRES) durchführen oder die CPU aus- und wieder einschalten. Nachdem Sie den Fehlerzustand der CPU behoben haben, können Sie ein gültiges CPU-Programm V4.0 laden.

Um ein V3.0-Programm in ein V4.0-Programm zu übertragen, müssen Sie das TIA Portal verwenden und in der Hardware-Konfiguration das Gerät ändern.

Hinweis

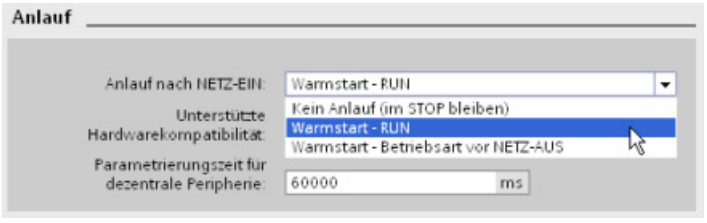
Wenn Sie eine Memory Card im Betriebszustand STOP der CPU stecken, zeigt der Diagnosepuffer die Meldung an, dass die Auswertung der Memory Card gestartet wurde. Die CPU wertet die Memory Card aus, wenn Sie entweder die CPU in den Betriebszustand RUN versetzen, den Speicher der CPU über MRES urlöschen oder die CPU aus- und wieder einschalten.

Tabelle 5- 43 Memory Card einsetzen

	Zum Einsetzen einer Memory Card öffnen Sie die obere Abdeckung der CPU und stecken die Memory Card in den Steckplatz. Ein Steckverbinder ermöglicht einfaches Stecken und Ziehen des Moduls. Die Memory Card ist so geformt, dass sie nur in eine Richtung in den Schacht gesteckt werden kann.

5.5.2 Anlaufparameter der CPU vor dem Kopieren des Projekts auf die Memory Card konfigurieren

Wenn Sie ein Programm auf eine Übertragungskarte oder eine Programmkarte kopieren, enthält das Programm die Anlaufparameter für die CPU. Stellen Sie stets vor dem Kopieren des Prgramm auf die Memory Card sicher, dass Sie den Betriebszustand der CPU nach dem Aus- und wieder Einschalten konfiguriert haben. Sie können wählen, ob die CPU im Betriebszustand STOP oder RUN oder im letzten Betriebszustand (vor dem Neustart) starten soll.



5.5.3 Einsatz der Memory Card als Übertragungskarte

ACHTUNG
Schutz von Memory Card und Schacht vor elektrostatischer Entladung Elektrostatische Entladungen können die Memory Card oder den dafür vorgesehenen Schacht in der CPU beschädigen. Sie müssen auf einem leitfähigen, geerdeten Boden stehen und/oder ein geerdetes Armband tragen, wenn Sie mit der Memory Card arbeiten. Die Memory Card ist in einem leitfähigen Behälter aufzubewahren.

Übertragungskarte anlegen

Denken Sie daran, die Anlaufparameter der CPU zu konfigurieren (Seite 139), bevor Sie ein Programm auf die Übertragungskarte kopieren. Um eine Übertragungskarte anzulegen, gehen Sie folgendermaßen vor:

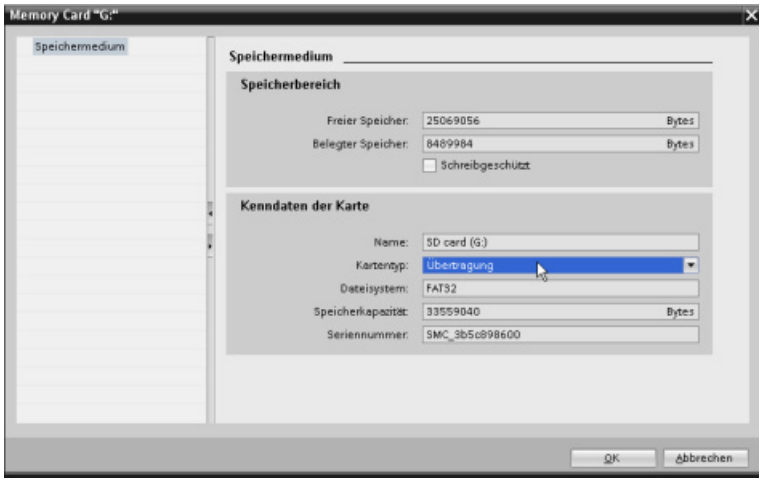
1. Stecken Sie eine leere, nicht schreibgeschützte SIMATIC Memory Card in einen an Ihren Computer angeschlossenen SD-Kartenleser. (Falls die Memory Card schreibgeschützt ist, schieben Sie den Schutzschalter aus der Verriegelungsposition heraus.)

   Wenn Sie eine SIMATIC Memory Card verwenden, die bereits ein Anwenderprogramm oder ein Firmware-Update enthält, **müssen** Sie die Programmdateien löschen, bevor Sie die Karte erneut verwenden. Rufen Sie den Windows Explorer auf und zeigen Sie die Inhalte der Memory Card an. Löschen Sie die Datei "S7_JOB.S7S" sowie alle vorhandenen Ordner mit Datenprotokollen und alle Verzeichnisordner (wie "SIMATIC.S7S" oder "FWUPDATE.S7S").

ACHTUNG
Löschen Sie NICHT die versteckten Dateien "__LOG__" und "crdinfo.bin" von der Memory Card. Die Dateien "__LOG__" und "crdinfo.bin" werden von der Memory Card benötigt. Wenn Sie diese Dateien löschen, können Sie die Memory Card nicht mehr mit der CPU nutzen.

2. Erweitern Sie in der Projektnavigation (Projektansicht) den Ordner "SIMATIC Kartenleser" und wählen Sie Ihren Kartenleser aus.
3. Klicken Sie mit der rechten Maustaste auf den Laufwerksbuchstaben der Memory Card im Kartenleser und wählen Sie im Kontextmenü die Option "Eigenschaften". Daraufhin wird der Dialog "Memory Card" angezeigt.
4. Wählen Sie im Dialog "Memory Card" in der Klappliste "Kartentyp" die Option "Übertragen" aus.

   Daraufhin erstellt STEP 7 die leere Übertragungskarte. Wenn Sie eine leere Übertragungskarte anlegen, weil Sie Ihr CPU-Passwort verloren haben (Seite 149), entnehmen Sie die Übertragungskarte aus dem Kartenleser.



5. Fügen Sie das Programm hinzu, indem Sie in der Projektnavigation die CPU (z.B. PLC_1 [CPU 1214C DC/DC/DC]) auswählen und mit der Maus auf die Memory Card ziehen. (Alternativ können Sie die CPU kopieren und in die Memory Card einfügen.) Durch Kopieren der CPU in die Memory Card wird der Dialog "Vorschau laden" geöffnet.
6. Klicken Sie im Dialog "Vorschau laden" auf die Schaltfläche "Laden", um die CPU in die Memory Card zu kopieren.
7. Wenn der Dialog eine Meldung anzeigt, dass die CPU (das Programm) fehlerfrei geladen wurde, klicken Sie auf die Schaltfläche "Fertig stellen".

Übertragungskarte verwenden

⚠ WARNUNG

Vergewissern Sie sich vor dem Einlegen der Memory Card, dass die CPU keinen Prozess ausführt.

Durch das Einlegen einer Memory Card geht die CPU in den Betriebszustand STOP, was sich auf den Betrieb eines Online-Prozesses oder einer Maschine auswirken kann. Unvorhersehbarer Betrieb eines Prozesses oder einer Maschine kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Stellen Sie vor dem Einlegen einer Übertragungskarte sicher, dass die CPU im Betriebszustand STOP ist und sich der Prozess in einem sicheren Zustand befindet.

Hinweis

Stecken Sie keine Übertragungskarte mit einem Programm V3.0 in S7-1200 CPUs V4.0.

Übertragungskarten mit einem Programm der Version 3.0 sind nicht mit S7-1200 CPUs der Version 4.0 kompatibel. Wenn Sie eine Memory Card mit einem Programm V3.0 stecken, wird dadurch ein CPU-Fehler verursacht.

Wenn Sie eine Übertragungskarte mit einer ungültigen Programmversion stecken, entnehmen Sie die Karte und versetzen die CPU von STOP in RUN, führen ein Urlöschen (MRES) durch oder schalten die CPU aus und wieder ein. Nachdem Sie den Fehlerzustand der CPU behoben haben, können Sie ein gültiges CPU-Programm V4.0 laden.

Um das Programm in die CPU zu übertragen, gehen Sie folgendermaßen vor:

1. Stecken Sie die Übertragungskarte in die CPU (Seite 137). Wenn sich die CPU in RUN befindet, geht die CPU in den Betriebszustand STOP. Die Wartungs-LED (MAINT) blinkt, um anzuzeigen, dass die Memory Card ausgewertet werden muss.
2. Schalten Sie die CPU aus und wieder ein, um die Memory Card auszuwerten. Alternative Methoden zum Neustarten der CPU sind ein Wechsel von STOP in RUN oder ein Urlöschen (MRES) in STEP 7.

3. Nach dem Neustart und der Auswertung der Memory Card kopiert die CPU das Programm in den internen Ladespeicher der CPU.

   Die RUN/STOP-LED blinkt abwechselnd grün und gelb, um kenntlich zu machen, dass das Programm kopiert wird. Wenn die RUN/STOP-LED eingeschaltet wird (und dauerhaft gelb leuchtet) und die MAINT-LED blinkt, ist der Kopiervorgang beendet. Nun können Sie die Memory Card entnehmen.

4. Starten Sie die CPU neu (entweder durch Einschalten oder durch eine der alternativen Methoden zum Neustarten), um das neue Programm, das in den internen Ladespeicher übertragen wurde, auszuwerten.

Die CPU geht dann in den Betriebszustand für den Anlauf (RUN oder STOP), den Sie für das Projekt eingerichtet haben.

Hinweis

Sie müssen die Übertragungskarte ziehen, bevor Sie die CPU in RUN versetzen.

5.5.4 Einsatz der Memory Card als Programmkarte

ACHTUNG
Elektrostatische Entladungen können die Memory Card oder den dafür vorgesehenen Schacht in der CPU beschädigen. Sie müssen auf einem leitfähigen, geerdeten Boden stehen und/oder ein geerdetes Armband tragen, wenn Sie mit der Memory Card arbeiten. Die Memory Card ist in einem leitfähigen Behälter aufzubewahren.

Stellen Sie sicher, dass die Memory Card nicht schreibgeschützt ist. Schieben Sie dazu den Schutzschalter aus der Verriegelungsposition heraus.

Löschen Sie vor dem Kopieren von Programmelementen auf die Programmkarte alle zuvor gespeicherten Programme von der Memory Card.

Programmkarte anlegen

Bei Einsatz als Programmkarte funktioniert die Memory Card als externer Ladespeicher der CPU. Wenn Sie die Programmkarte ziehen, ist der interne Ladespeicher der CPU leer.

Hinweis

Wenn Sie eine leere Memory Card in die CPU stecken und eine Auswertung der Memory Card durchführen, indem Sie die CPU aus- und wieder einschalten, die CPU von STOP nach RUN versetzen oder ein Urlöschen (MRES) durchführen, werden das Programm und die geforcten Werte aus dem internen Ladespeicher der CPU in die Memory Card kopiert. (Die Memory Card ist nun eine Programmkarte.) Nach Abschluss des Kopiervorgangs wird das Programm im internen Ladespeicher der CPU gelöscht. Die CPU geht dann in den konfigurierten Betriebszustand für den Anlauf (RUN oder STOP).

Denken Sie daran, die Anlaufparameter der CPU zu konfigurieren (Seite 139), bevor Sie ein Projekt auf die Programmkarte kopieren. Um eine Programmkarte anzulegen, gehen Sie folgendermaßen vor:

1. Stecken Sie eine leere, nicht schreibgeschützte SIMATIC Memory Card in einen an Ihren Computer angeschlossenen SD-Kartenleser. (Falls die Memory Card schreibgeschützt ist, schieben Sie den Schutzschalter aus der Verriegelungsposition heraus.)

   Wenn Sie eine SIMATIC Memory Card verwenden, die bereits ein Anwenderprogramm oder ein Firmware-Update enthält, **müssen** Sie die Programmdateien löschen, bevor Sie die Karte erneut verwenden. Rufen Sie den Windows Explorer auf und zeigen Sie die Inhalte der Memory Card an. Löschen Sie die Datei "S7_JOB.S7S" sowie alle vorhandenen Ordner mit Datenprotokollen und alle Verzeichnisordner (wie "SIMATIC.S7S" oder "FWUPDATE.S7S").

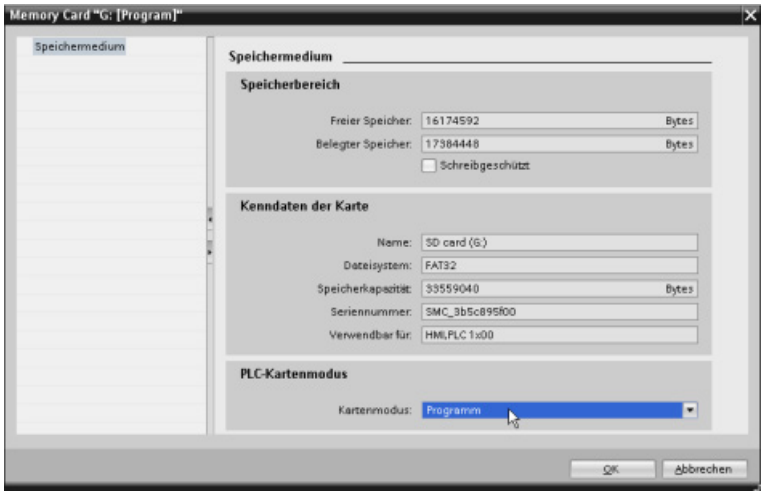
   ACHTUNG

   Löschen Sie NICHT die versteckten Dateien "__LOG__" und "crdinfo.bin" von der Memory Card.

   Die Dateien "__LOG__" und "crdinfo.bin" werden von der Memory Card benötigt. Wenn Sie diese Dateien löschen, können Sie die Memory Card nicht mehr mit der CPU nutzen.

2. Erweitern Sie in der Projektnavigation (Projektansicht) den Ordner "SIMATIC Kartenleser" und wählen Sie Ihren Kartenleser aus.
3. Klicken Sie mit der rechten Maustaste auf den Laufwerksbuchstaben der Memory Card im Kartenleser und wählen Sie im Kontextmenü die Option "Eigenschaften". Daraufhin wird der Dialog "Memory Card" angezeigt.

4. Wählen Sie im Dialog "Memory Card" in der Klappliste die Option "Programm" aus.



5. Fügen Sie das Programm hinzu, indem Sie in der Projektnavigation die CPU (z.B. PLC_1 [CPU 1214C DC/DC/DC]) auswählen und mit der Maus auf die Memory Card ziehen. (Alternativ können Sie die CPU kopieren und in die Memory Card einfügen.) Durch Kopieren der CPU in die Memory Card wird der Dialog "Vorschau laden" geöffnet.
6. Klicken Sie im Dialog "Vorschau laden" auf die Schaltfläche "Laden", um die CPU in die Memory Card zu kopieren.
7. Wenn der Dialog eine Meldung anzeigt, dass die CPU (das Programm) fehlerfrei geladen wurde, klicken Sie auf die Schaltfläche "Fertig stellen".

Programmkarte als externen Ladespeicher für die CPU nutzen

⚠ WARNUNG

Risiken beim Stecken einer Programmkarte

Vergewissern Sie sich vor dem Einlegen der Memory Card, dass die CPU keinen Prozess ausführt.

Durch das Einlegen einer Memory Card geht die CPU in den Betriebszustand STOP, was sich auf den Betrieb eines Online-Prozesses oder einer Maschine auswirken kann. Unvorhersehbarer Betrieb eines Prozesses oder einer Maschine kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Stellen Sie vor dem Einlegen einer Memory Card sicher, dass die CPU offline und in einem sicheren Zustand ist.

Um mit Ihrer CPU eine Programmkarte zu nutzen, gehen Sie folgendermaßen vor:

1. Schieben Sie die Programmkarte in die CPU. Befindet sich die CPU im Betriebszustand RUN, geht sie in STOP. Die Wartungs-LED (MAINT) blinkt, um anzuzeigen, dass die Memory Card ausgewertet werden muss.
2. Schalten Sie die CPU aus und wieder ein, um die Memory Card auszuwerten. Alternative Methoden zum Neustarten der CPU sind ein Wechsel von STOP in RUN oder ein Urlöschen (MRES) in STEP 7.
3. Nach dem Neustart der CPU und der Auswertung der Programmkarte löscht die CPU den internen Ladespeicher der CPU.

Die CPU geht dann in den Betriebszustand für den Anlauf (RUN oder STOP), den Sie für die CPU eingerichtet haben.

Die Programmkarte muss in der CPU gesteckt bleiben. Wenn Sie die Programmkarte ziehen, hat die CPU kein Programm mehr im internen Ladespeicher.

⚠ WARNUNG

Risiken beim Ziehen einer Programmkarte

Wenn Sie die Programmkarte ziehen, verliert die CPU den externen Ladespeicher und erzeugt einen Fehler. Die CPU geht dann in den Betriebszustand STOP und die Fehler-LED blinkt.

Steuerungen können bei unsicheren Betriebszuständen ausfallen und dadurch den unkontrollierten Betrieb der gesteuerten Geräte verursachen. Daraus resultiert ein unvorhersehbarer Betrieb des Automatisierungssystems, der zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen kann.

Ziehen Sie die Programmkarte nur, wenn Sie sich darüber im Klaren sind, dass Sie damit das Programm aus der CPU entfernen.

5.5.5 Firmware-Update

Mit einer Memory Card können Sie auch eine Firmware-Aktualisierung durchführen. Alternativ können Sie eine Firmware-Aktualisierung über die Modulinformationsseite (Seite 837) des Webservers oder über die Online- und Diagnosefunktionen in STEP 7 (Seite 1124) durchführen. In diesem Kapitel wird die Vorgehensweise bei Verwendung einer Memory Card erläutert.

ACHTUNG
Schutz von Memory Card und Schacht vor elektrostatischer Entladung Elektrostatische Entladungen können die Memory Card oder den dafür vorgesehenen Schacht in der CPU beschädigen. Sie müssen auf einem leitfähigen, geerdeten Boden stehen und/oder ein geerdetes Armband tragen, wenn Sie mit der Memory Card arbeiten. Die Memory Card ist in einem leitfähigen Behälter aufzubewahren.

Sie verwenden die Memory Card auch zum Herunterladen von Firmware-Updates vom Kunden-Support (http://www.siemens.com/tiaportal). Wählen Sie auf dieser Website den Befehl **Automatisierungstechnik > Automatisierungssysteme > Industrie-Automatisierungssysteme SIMATIC > Steuerungen > Modulare Controller SIMATIC S7 > SIMATIC S7-1200.** Von dort navigieren Sie zu dem speziellen Modultyp, den Sie aktualisieren möchten. Klicken Sie unter "Support" auf den Link für "Software Downloads", um fortzufahren.

Alternativ können Sie die S7-1200 Website für Downloads (http://support.automation.siemens.com/WW/view/de/34612486/133100) direkt aufrufen.

Hinweis

Sie können eine S7-1200 CPU V3.0 oder eine Vorgängerversion davon nicht auf die Firmware S7-1200 V4.0 oder V4.1 aktualisieren.

ACHTUNG
Verwenden Sie zum Neuformatieren der Memory Card nicht das Windows-Formatierungsprogramm oder ein anderes Formatierungsprogramm. Wenn eine Siemens Memory Card mit dem Windows-Formatierungsprogramm neu formatiert wird, kann die Memory Card nicht mehr von einer S7-1200 CPU verwendet werden.

Um das Firmware-Update auf Ihre Memory Card zu laden, gehen Sie folgendermaßen vor:

1. Stecken Sie eine leere SIMATIC Memory Card ohne Schreibschutz in einen SD-Kartenleser, der an Ihren Computer angeschlossen ist. (Ist die Karte schreibgeschützt, schieben Sie den Schutzschalter aus der Verriegelungsposition heraus.)

   Sie können eine SIMATIC Memory Card verwenden, die bereits ein Anwenderprogramm oder ein anderes Firmware-Update enthält, doch Sie müssen einige der Dateien auf der Memory Card löschen.

   Um eine Memory Card wiederzuverwenden, **müssen** Sie die Datei "S7_JOB.S7S" sowie alle vorhandenen Ordner mit Datenprotokollen und alle Verzeichnisordner (wie "SIMATIC.S7S" oder "FWUPDATE.S7S") löschen, bevor Sie das Firmware-Update herunterladen. Mit dem Windows Explorer können Sie den Inhalt der Memory Card anzeigen und die Datei und Ordner löschen.

   ACHTUNG

   Löschen Sie NICHT die versteckten Dateien "__LOG__" und "crdinfo.bin" von der Memory Card.

   Die Dateien "__LOG__" und "crdinfo.bin" werden von der Memory Card benötigt. Wenn Sie diese Dateien löschen, können Sie die Memory Card nicht mehr mit der CPU nutzen.

2. Wählen Sie die selbst extrahierende Datei (*.exe) des Ihrem Modul entsprechenden Firmware-Updates aus und laden Sie sie auf Ihren Computer herunter. Doppelklicken Sie auf die Update-Datei, geben Sie als Zielpfad das Stammverzeichnis der SIMATIC Memory Card an und starten Sie die Extraktion. Wenn die Extraktion beendet ist, enthält das Stammverzeichnis (Ordner) der Memory Card das Verzeichnis "FWUPDATE.S7S" und die Datei "S7_JOB.S7S".
3. Nehmen Sie die Karte vorsichtig aus dem Lesegerät.

Zum Installieren des Firmware-Updates gehen Sie folgendermaßen vor:

⚠ WARNUNG

Stellen Sie vor der Installation des Firmware-Updates sicher, dass die CPU keinen aktiven Prozess ausführt.

Durch die Installation des Firmware-Updates geht die CPU in den Betriebszustand STOP, was sich auf den Betrieb eines Online-Prozesses oder einer Maschine auswirken kann. Unerwarteter Betrieb eines Prozesses oder einer Maschine kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Bevor Sie die Memory Card stecken, stellen Sie unbedingt sicher, dass die CPU offline und in einem sicheren Zustand ist.

1. Schieben Sie die Memory Card in die CPU. Befindet sich die CPU im Betriebszustand RUN, geht sie in STOP. Die Wartungs-LED (MAINT) blinkt, um anzuzeigen, dass die Memory Card ausgewertet werden muss.
2. Starten Sie die CPU neu, um das Firmware-Update zu starten. Alternative Methoden zum Neustarten der CPU sind ein Wechsel von STOP in RUN oder ein Urlöschen (MRES) in STEP 7.

   Hinweis

   Damit das Firmware-Upgrade für das Modul abgeschlossen werden kann, achten Sie darauf, dass die externe 24-V-DC-Versorgung des Moduls eingeschaltet bleibt.

   Nach dem Neustart der CPU beginnt das Firmware-Update. Die RUN/STOP-LED blinkt grün und gelb, um anzuzeigen, dass das Update kopiert wird. Wenn die RUN/STOP-LED dauerhaft gelb leuchtet und die MAINT-LED blinkt, ist der Kopiervorgang beendet. Anschließend müssen Sie die Memory Card entnehmen.
3. Nachdem Sie die Memory Card entnommen haben, starten Sie die CPU erneut neu (entweder durch Einschalten oder mit einem anderen Verfahren zum Neustarten), um die neue Firmware zu laden.

Das Anwenderprogramm und die Hardware-Konfiguration sind vom Firmware-Update nicht betroffen. Wenn die CPU eingeschaltet wird, geht die CPU in den konfigurierten Anlaufzustand. (Wenn als Anlaufbetriebszustand für die CPU "Warmstart - Betriebsart vor NETZ-AUS" konfiguriert wurde, befindet sich die CPU im Betriebszustand STOP, weil STOP der letzte Betriebszustand der CPU war.)

Hinweis

Mehrere an eine CPU angeschlossene Module aktualisieren

Wenn Ihre Hardwarekonfiguration mehrere Module enthält, auf die ein Firmware-Update von der Memory Card angewendet werden kann, aktualisiert die CPU alle betroffenen Module (CM, SM und SB) in der Konfigurationsreihenfolge, d. h. in aufsteigender Reihenfolge der Modulposition in der Gerätekonfiguration in STEP 7.

Wenn Sie mehrere Firmware-Updates für mehrere Module auf die Memory Card geladen haben, führt die CPU die Updates in der Reihenfolge durch, in der Sie sie auf die Memory Card geladen haben.

5.6 Vorgehensweise bei verlorenem Passwort

Wenn Sie das Passwort für eine passwortgeschützte CPU verloren haben, löschen Sie das passwortgeschützte Programm mit einer leeren Übertragungskarte. Die leere Übertragungskarte löscht den internen Ladespeicher der CPU. Dann können Sie ein neues Anwenderprogramm aus STEP 7 in die CPU laden.

Informationen zum Erstellen und Nutzen einer leeren Übertragungskarte finden Sie im Abschnitt Übertragungskarten (Seite 139).

⚠ WARNUNG

Vergewissern Sie sich vor dem Einlegen der Memory Card, dass die CPU keinen Prozess ausführt

Wenn Sie eine Übertragungskarte in eine laufende CPU stecken, geht die CPU in STOP. Steuerungen können bei unsicheren Betriebszuständen ausfallen und dadurch den unkontrollierten Betrieb der gesteuerten Geräte verursachen. Daraus resultiert ein unvorhersehbarer Betrieb des Automatisierungssystems, der zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen kann.

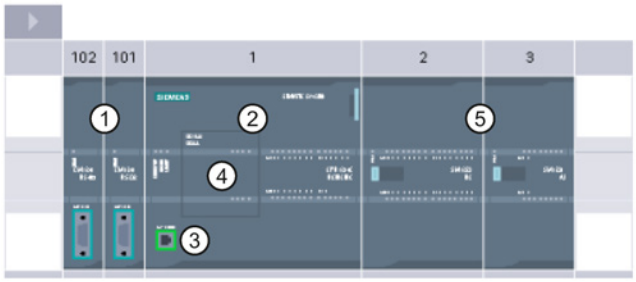
Stellen Sie vor dem Einlegen einer Übertragungskarte sicher, dass die CPU sich im Betriebszustand STOP und Ihr Prozess sich in einem sicheren Zustand befindet.

Sie müssen die Übertragungskarte ziehen, bevor Sie die CPU in RUN versetzen.

Gerätekonfiguration

6

Sie können die Gerätekonfiguration für Ihr PLC-Gerät durch Hinzufügen einer CPU und weiterer Module zu Ihrem Projekt erstellen.



① Kommunikationsmodul (CM) oder Kommunikationsprozessor (CP): bis zu 3, in Steckplätzen 101, 102 und 103

② CPU: Steckplatz 1

③ PROFINET-Port der CPU

④ Signalboard (SB), Kommunikationsboard (CB) oder Batterieboard (BB): max. 1, in CPU gesteckt

⑤ Signalmodul (SM) für digitale oder analoge E/A: bis zu 8, in Steckplätzen 2 bis 9
(8 bei CPU 1214C, CPU 1215C und CPU 1217C, 2 bei CPU 1212C, keines bei der CPU 1211C)

Konfigurationssteuerung

Die Gerätekonfiguration für die S7-1200 unterstützt auch die "Konfigurationssteuerung (Seite 157)", mit der eine Maximalkonfiguration für ein Projekt mit Modulen projektiert werden kann, die möglicherweise nicht alle verwendet werden. Mit dieser Funktion, manchmal auch "Optionsverwaltung" genannt, kann ein Maximalausbau projektiert werden, der die Verwendung variabler Module in zahlreichen Anwendungen gestattet.

6.1 CPU einfügen

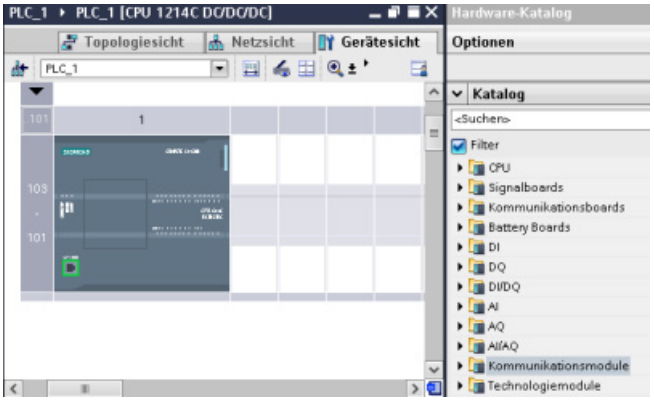
Zum Erstellen Ihrer Gerätekonfiguration fügen Sie eine CPU in Ihr Projekt ein.

- Wählen Sie in der Portalansicht das Portal "Geräte & Netze" und klicken Sie auf "Neues Gerät hinzufügen".
- Doppelklicken Sie in der Projektansicht unter dem Projektnamen auf "Neues Gerät hinzufügen".

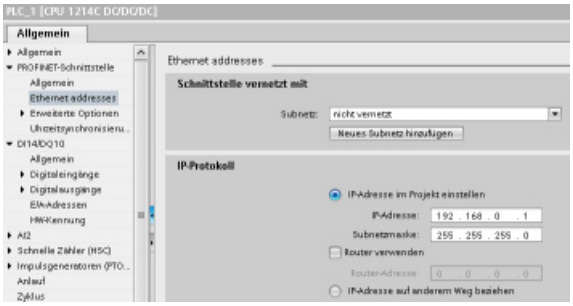
Achten Sie darauf, dass Sie die richtige Variante und Firmware-Version aus der Liste auswählen. Durch die Auswahl der CPU im Dialog "Neues Gerät hinzufügen" werden der Baugruppenträger und die CPU ausgewählt.

Dialog "Neues Gerät hinzufügen"

Gerätesicht der Hardwarekonfiguration



Nach der Auswahl der CPU in der Gerätesicht werden die Eigenschaften der CPU im Inspektorfenster angezeigt.



Hinweis

Die CPU hat keine vorkonfigurierte IP-Adresse. Sie müssen der CPU daher bei der Gerätekonfiguration manuell eine IP-Adresse zuweisen. Ist Ihre CPU an einen Router im Netzwerk angeschlossen, so muss auch die IP-Adresse des Routers eingegeben werden.

6.2 Konfiguration aus einer angeschlossenen CPU laden

STEP 7 ermöglicht das Laden der Hardwarekonfiguration einer angeschlossenen CPU auf zweierlei Art:

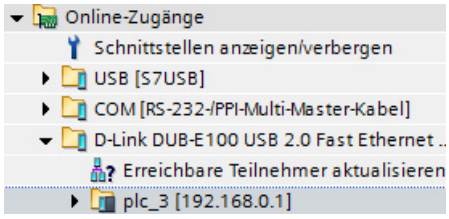
- das Laden des angeschlossenen Geräts als neue Station
- das Konfigurieren einer nicht spezifizierten CPU mittels Hardwareerkennung der angeschlossenen CPU

Zu beachten ist jedoch, dass im ersten Fall sowohl die Hardwarekonfiguration als auch die Software der angeschlossenen CPU geladen werden.

Gerät als neue Station laden

Um ein angeschlossenes Gerät als neue Station zu laden, gehen Sie wie folgt vor:

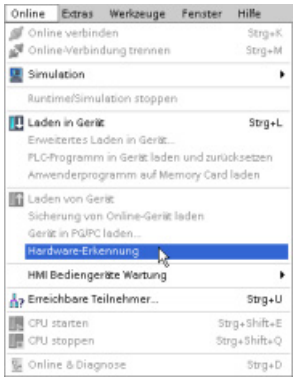
1. Erweitern Sie in der Projektnavigation Ihre Kommunikationsschnittstelle über den Knoten "Online-Zugänge".
2. Doppelklicken Sie auf "Erreichbare Teilnehmer aktualisieren".
3. Wählen Sie die PLC aus den erkannten Geräten.



4. Wählen Sie im Online-Menü von STEP 7 Befehl "Gerät als neue Station laden (Hardware und Software)".

STEP 7 lädt die Hardwarekonfiguration und die Programmbausteine.

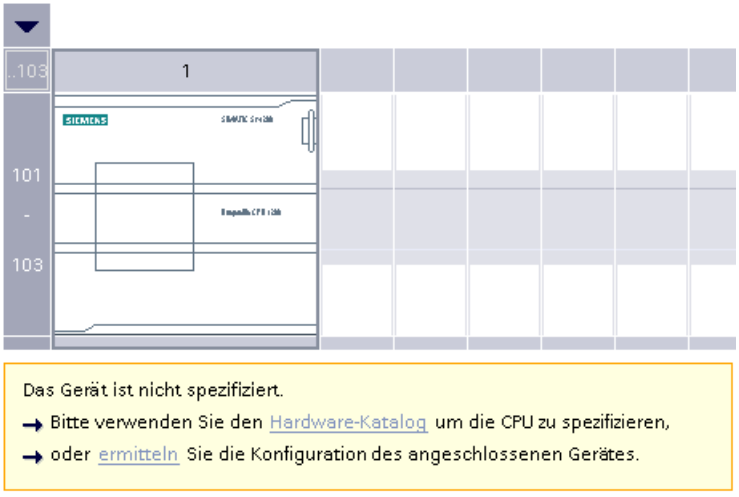
Hardwarekonfiguration einer nicht spezifizierten CPU erkennen



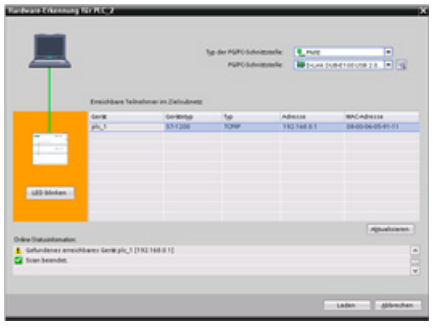
Wenn eine Verbindung zu einer CPU besteht, können Sie die Konfiguration dieser CPU einschließlich evtl. vorhandener Module aus dem Gerät in Ihr Projekt laden. Legen Sie dazu einfach ein neues Projekt an und wählen Sie anstelle einer bestimmten CPU die "nicht spezifizierte CPU". (Sie können auch die Gerätekonfiguration ganz umgehen, indem Sie unter "Erste Schritte" auf "Ein PLC-Programm erstellen" klicken. STEP 7 legt dann automatisch eine nicht spezifizierte CPU an.)

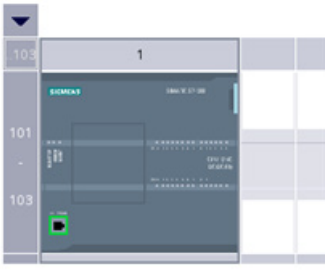
Wählen Sie im Programmiereditor im Menü "Online" den Befehl "Hardwareerkennung".

Wählen Sie im Gerätekonfigurationseditor die Option zum Erkennen der Konfiguration des angeschlossenen Geräts.



Nachdem Sie im Online-Dialog die CPU ausgewählt und auf die Schaltfläche zum Laden geklickt haben, lädt STEP 7 die Hardwarekonfiguration einschließlich möglicher Module (SM, SB oder CM) aus der CPU. Sie können dann die Parameter für die CPU und die Module (Seite 168) konfigurieren.





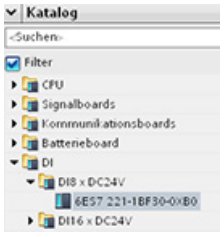
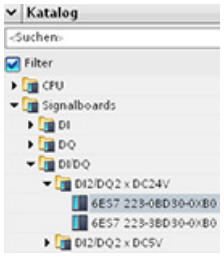
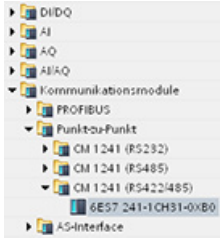
6.3 Module zur Konfiguration hinzufügen

Im Hardwarekatalog können Sie Module zur CPU hinzufügen:

- Signalmodule (SMs) für zusätzliche digitale oder analoge Ein- und Ausgänge. Diese Module werden an der rechten Seite der CPU angeschlossen.
- Signalboards (SBs) bieten eine begrenzte Zahl von zusätzlichen Ein-/Ausgängen für die CPU. Das SB wird auf der Vorderseite der CPU gesteckt.
- Das Batterieboard 1297 (BB) bietet eine Langzeitpufferung der Echtzeituhr. Das BB wird auf der Vorderseite der CPU gesteckt.
- Kommunikationsboards (CBs) bieten einen zusätzlichen Kommunikationsanschluss (z. B. RS485). Das CB wird auf der Vorderseite der CPU gesteckt.
- Kommunikationsmodule (CMs) und Kommunikationsprozessoren (CPs) bieten einen zusätzlichen Kommunikationsanschluss, z. B. für PROFIBUS oder GPRS. Diese Module werden an der linken Seite der CPU angeschlossen.

Um ein Modul in die Gerätekonfiguration einzufügen, wählen Sie das Modul im Hardwarekatalog aus und klicken doppelt darauf oder ziehen Sie es in den markierten Steckplatz. Sie müssen die Module in die Gerätekonfiguration aufnehmen und die Hardwarekonfiguration in die CPU laden, damit die Module funktionsfähig sind.

Tabelle 6- 1 Modul zur Gerätekonfiguration hinzufügen

Modul	Modul auswählen	Modul einsetzen	Ergebnis
SM			
SB, BB oder CB			
CM oder CP			

Mit der Funktion "Konfigurationssteuerung" (Seite 157) können Signalmodule und Signalboards in der Gerätekonfiguration hinzugefügt werden, die möglicherweise nicht der aktuellen Hardware für eine bestimmte Anwendung entsprechen, aber in anderen Anwendungen verwendet werden, die das gleiche Anwenderprogramm, CPU-Modell und u. U. auch einige der konfigurierten Module teilen.

6.4 Konfigurationssteuerung

6.4.1 Vorteile und Nutzung der Konfigurationssteuerung

Die Konfigurationssteuerung kann hilfreich sein, wenn eine Automatisierungslösung entwickelt wird, die mit Variationen in unterschiedlichen Aufbauten eingesetzt werden soll.

Eine STEP 7-Gerätekonfiguration und ein Anwenderprogramm können in verschiedene installierte PLC-Konfigurationen geladen werden. Dafür sind lediglich einige wenige Anpassungen des STEP 7-Projekts für den neuen Aufbau erforderlich.

6.4.2 Den zentralen Aufbau und optionale Module konfigurieren

Die Konfigurationssteuerung in STEP 7 und die S7-1200 ermöglichen die Projektierung einer Maximalkonfiguration für ein Standardsystem mit unterschiedlichen Versionen (Optionen), die jeweils eine Teilmenge dieser Konfiguration nutzen. Im Handbuch PROFINET mit STEP 7 (http://support.automation.siemens.com/WW/view/en/49948856) werden diese Projekttypen als "Serienmaschinen-Projekte" bezeichnet.

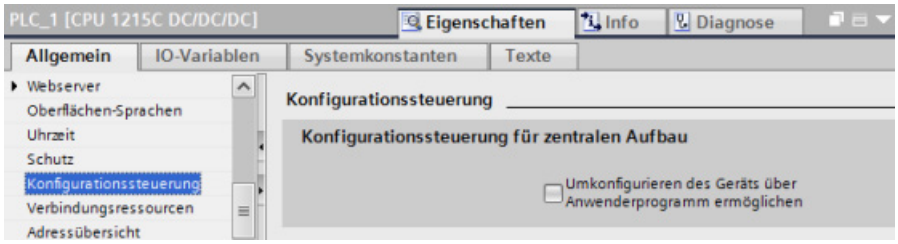
Ein Steuerdatensatz, der im Anlauf-OB programmiert wird, teilt der CPU mit, welche Module im realen Aufbau abweichend von der Projektierung fehlen oder welche Module sich abweichend von der Projektierung in einem anderen Steckplatz befinden. Die Konfigurationssteuerung hat keine Auswirkung auf die Parametrierung der Module.

Mit der Konfigurationssteuerung kann der Aufbau flexibel verändert werden, solange der reale Aufbau in der maximalen Gerätekonfiguration in STEP 7 enthalten ist.

Gehen Sie wie folgt vor, um die Konfigurationssteuerung zu aktivieren und den erforderlichen Steuerdatensatz zu strukturieren:

1. Optional kann die CPU auf die Werkseinstellungen zurückgesetzt werden, um sicherzustellen, dass kein inkompatibler Steuerdatensatz in der CPU vorhanden ist.
2. Wählen Sie die CPU in der Gerätekonfiguration in STEP 7.

3. Im Knoten "Konfigurationssteuerung" in den CPU-Eigenschaften aktivieren Sie Option "Umkonfigurieren des Geräts über Anwenderprogramm ermöglichen".

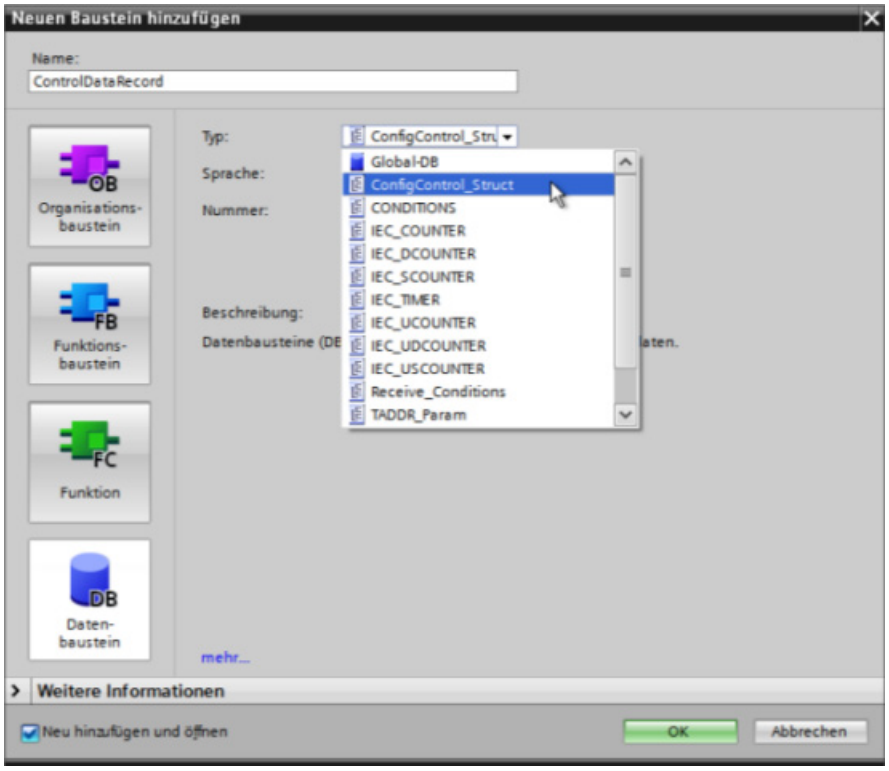


4. Erstellen Sie einen PLC-Datentyp für den Steuerdatensatz. Konfigurieren Sie ihn als Struct mit vier USints für die Konfigurationssteuerungsdaten und weiteren USints entsprechend der Anzahl der Steckplätze für den S7-1200 Maximalausbau wie folgt:

ConfigControl_Struct

	Name	Datentyp	Defaultwert	Kommentar
1	ConfigControl	Struct		
2	Block_length	USInt	16	Length of control data record, including header
3	Block_ID	USInt	196	Data record number
4	Version	USInt	5	
5	Subversion	USInt	0	
6	Slot_1	USInt	255	Assignment for CPU annex card/Actual annex card
7	Slot_2	USInt	255	Configured slot 2 / Assigned "real" slot
8	Slot_3	USInt	255	Configured slot 3 / Assigned "real" slot
9	Slot_4	USInt	255	Configured slot 4 / Assigned "real" slot
10	Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
11	Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
12	Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
13	Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
14	Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
15	Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
16	Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
17	Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

5. Legen Sie einen Datenbaustein mit dem gewählten PLC-Datentyp an.



6. Konfigurieren Sie in diesem Datenbaustein die Parameter Block_length, Block_ID, Version und Subversion wie unten gezeigt. Konfigurieren Sie die Werte für die Steckplätze je nachdem, ob sie belegt sind oder nicht und nach ihrer Lage im tatsächlichen Aufbau:
   - 0: Konfiguriertes Modul in der tatsächlichen Konfiguration nicht vorhanden. (Steckplatz ist leer.)
   - 1 bis 9, 101 bis 103: Tatsächliche Position des konfigurierten Steckplatzes
   - 255: Die STEP 7-Gerätekonfiguration enthält kein Modul an diesem Steckplatz.

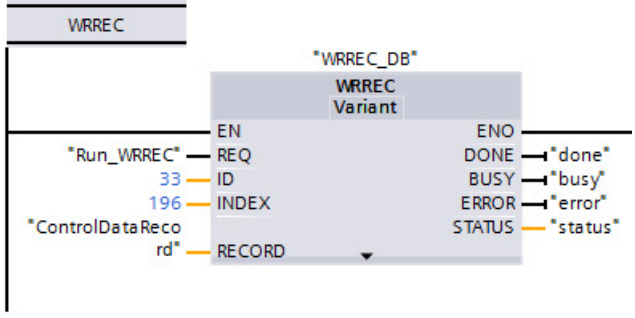
ControlDataRecord

	Name	Datentyp	Startwert	Kommentar
1	Static			
2	ConfigControl	Struct		
3	Block_length	USInt	16	Length of control data record, including header
4	Block_ID	USInt	196	Data record number
5	Version	USInt	5	
6	Subversion	USInt	0	
7	Slot_1	USInt	255	Assignment for CPU annex card/ Actual annex ...
8	Slot_2	USInt	255	Configured slot 2 / Assigned "real" slot
9	Slot_3	USInt	255	Configured slot 3 / Assigned "real" slot
10	Slot_4	USInt	255	Configured slot 4 / Assigned "real" slot
11	Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
12	Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
13	Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
14	Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
15	Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
16	Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
17	Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
18	Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

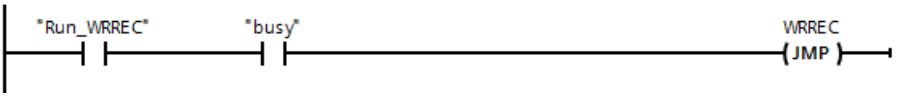
Zur Zuweisung der Werte für die Steckplätze, siehe Beispiel für die Konfigurationssteuerung (Seite 164).

7. Rufen Sie im Anlauf-OB die erweiterte Anweisung WRREC (Datensatz schreiben) auf, um den Steuerdatensatz zu Index 196 der Hardware-ID 33 zu übertragen. Verwenden Sie eine Sprungmarke (JMP), um zu warten, bis die Anweisung WRREC beendet ist.

Netzwerk 1:



Netzwerk 2:



Hinweis

Die Konfigurationssteuerung ist erst wirksam, wenn Anweisung WRREC den Steuerdatensatz in den Anlauf-OB übertragen hat. Ist die Konfigurationssteuerung aktiviert und der Steuerdatensatz ist in der CPU nicht vorhanden, geht diese nach dem Anlauf direkt in STOP. Daher ist es wichtig, dass der Anlauf-OB so programmiert ist, dass der Steuerdatensatz übertragen wird.

Modulanordnung

Die folgende Tabelle zeigt die Zuweisung der Steckplätze:

Steckplatz	Module
1	Signalboard oder Kommunikationsboard (CPU-Zusatzkarte)
2 bis 9	Signalmodule
101 bis 103	Kommunikationsmodule

Steuerdatensatz

Ein Steuerdatensatz 196 enthält die Steckplatzbelegung und die tatsächliche Konfiguration wie unten gezeigt:

Byte	Element	Wert	Erklärung
0	Bausteinlänge	16	Header
1	Baustein-ID	196	
2	Version	5	
3	Subversion	0	
4	Zuweisung einer CPU-Zusatzkarte	Tatsächliche Zusatzkarte, 0 oder 255*	Steuerelement Beschreibt in jedem Element, welcher reale Steckplatz im Gerät dem konfigurierten Steckplatz zugewiesen ist.
5	Zuweisung konfigurierter Steckplatz 2	Tatsächlicher Steckplatz, 0 oder 255*	
...	...	...	
12	Zuweisung konfigurierter Steckplatz 9	Tatsächlicher Steckplatz, 0 oder 255*	
13	Zuweisung konfigurierter Steckplatz 101	Tatsächlicher Steckplatz oder 255*	Anders als bei Signalmodulen muss der tatsächliche Steckplatz bei physisch vorhandenen Kommunikationsmodulen mit dem konfigurierten Steckplatz übereinstimmen.
14	Zuweisung konfigurierter Steckplatz 102	Tatsächlicher Steckplatz oder 255*	
15	Zuweisung konfigurierter Steckplatz 103	Tatsächlicher Steckplatz oder 255*	

***Steckplatzwerte:**

0: Konfiguriertes Modul in der tatsächlichen Konfiguration nicht vorhanden. (Steckplatz ist leer.)

1 bis 9, 101 bis 103: Tatsächliche Position des konfigurierten Steckplatzes

255: Die STEP 7-Gerätekonfiguration enthält kein Modul an diesem Steckplatz.

Hinweis

Alternative zum Erstellen eines PLC-Variablentyps

Als Alternative zum Erstellen eines benutzerdefinierten PLC-Variablentyps können Sie einen Datenbaustein direkt mit sämtlichen Strukturelementen eines Steuerdatensatzes erstellen. Sie könnten sogar mehrere Strukturen in diesem Datenbaustein konfigurieren, die als verschiedene Steuerdatensatzkonfigurationen dienen. Beide Implementierungen sind ein effektiver Weg, um den Steuerdatensatz während des Anlaufs zu übertragen.

Regeln

Die folgenden Regeln sind zu beachten:

- Die Konfigurationssteuerung unterstützt keine Änderung der Position für Kommunikationsmodule. Die im Steuerdatensatz angegebenen Positionen der Steckplätze 101 bis 103 müssen dem tatsächlichen Aufbau entsprechen. Ist für einen Steckplatz kein Modul konfiguriert, geben Sie 255 für diese Position ein.
- Es sind keine unbenutzten Steckplätze zwischen belegten Steckplätzen zulässig. Ist in der tatsächlichen Konfiguration beispielsweise ein Modul in Steckplatz 4 vorhanden, dann müssen auch Module in Steckplätzen 2 und 3 gesteckt sein. Ist in der tatsächlichen Konfiguration ein Kommunikationsmodul in Steckplatz 102 vorhanden, muss dementsprechend auch ein Modul in Steckplatz 101 gesteckt sein.
- Ist die Konfigurationssteuerung aktiviert, ist die CPU erst betriebsbereit, wenn ein Steuerdatensatz geladen wurde. Die CPU wechselt aus dem Anlauf in STOP, wenn der Anlauf-OB keinen gültigen Steuerdatensatz überträgt. Die zentralen E/A werden dann von der CPU nicht initialisiert und der Grund für den Wechsel in STOP wird in den Diagnosepuffer geschrieben.
- Ein erfolgreich übertragener Steuerdatensatz wird von der CPU im remanenten Speicher abgelegt; der Steuerdatensatz 196 muss daher nach einem Neustart nicht neu geschrieben werden, sofern die Konfiguration nicht geändert wurde.
- Jeder reale Steckplatz darf nur einmal im Steuerdatensatz enthalten sein.
- Jedem konfigurierten Steckplatz kann nur ein realer Steckplatz zugewiesen werden.

Hinweis

Ändern einer Konfiguration

Wird ein Steuerdatensatz mit einer geänderten Konfiguration geschrieben, wird in der CPU der folgende Ablauf automatisch angestoßen: Urlöschen und anschließender Neustart mit der geänderten Konfiguration.

Die CPU löscht daraufhin den ursprünglichen Steuerdatensatz und speichert den neuen Datensatz im remanenten Speicher.

Verhalten im Betrieb

Für die Online-Anzeige und die Anzeige im Diagnosepuffer (Modul OK oder Modul fehlerhaft) nutzt STEP 7 die Gerätekonfiguration, nicht die abweichende reale Konfiguration.

Beispiel: Ein Modul gibt Diagnosedaten aus. Dieses Modul steckt laut Konfiguration in Steckplatz 4, tatsächlich aber in Steckplatz 3. In der Online-Ansicht wird ein Fehler in Steckplatz 4 angezeigt. In der realen Konfiguration meldet das Modul in Steckplatz 3 einen Fehler an seiner LED-Anzeige.

Wurden im Steuerdatensatz Module als fehlend (0) konfiguriert, verhält sich das Automatisierungssystem wie folgt:

- Module, die im Steuerdatensatz als nicht vorhanden konfiguriert sind, melden keine Diagnosedaten und ihr Status ist immer OK. Der Wertstatus ist OK.
- Ein direkter Schreibzugriff auf die Ausgänge oder Schreibzugriff auf das Prozessbild von Ausgängen, die nicht vorhanden sind, hat keine Wirkung; die CPU meldet keinen Zugriffsfehler.
- Ein direkter Lesezugriff auf die Eingänge oder Lesezugriff auf das Prozessbild von Eingängen, die nicht vorhanden sind, bewirkt einen Wert "0" für jeden Eingang; die CPU meldet keinen Zugriffsfehler.
- Das Schreiben eines Datensatzes in ein Modul, das nicht vorhanden ist, hat keine Wirkung; die CPU meldet keinen Fehler.
- Wird versucht, Daten aus einem nicht vorhandenen Modul zu lesen, so wird eine Fehlermeldung erzeugt, weil die CPU keinen gültigen Datensatz zurückgeben kann.

Fehlermeldungen

Die CPU gibt die folgenden Fehlermeldungen zurück, wenn beim Schreiben des Steuerdatensatzes ein Fehler auftritt:

Fehlercode	Bedeutung
16#80B1	Ungültige Länge; im Steuerdatensatz ist eine falsche Länge angegeben.
16#80B5	Konfigurationssteuerungsparameter nicht zugewiesen
16#80E2	Datensatz wurde im falschen OB übertragen. Der Datensatz muss im Anlauf-OB übertragen werden.
16#80B0	Bausteintyp (Byte 2) des Steuerdatensatzes ist nicht 196.
16#80B8	Parameterfehler; Modul meldet ungültige Parameter, zum Beispiel: • Der Steuerdatensatz versucht, die Konfiguration eines Kommunikationsmoduls oder einer Kommunikationszusatzkarte zu ändern. Die tatsächliche Konfiguration für Kommunikationsmodule und eine Kommunikationszusatzkarte muss mit der Konfiguration in STEP 7 übereinstimmen. • Der zugewiesene Wert für einen nicht konfigurierten Steckplatz im STEP 7-Projekt ist nicht 255. • Der zugewiesene Wert für einen konfigurierten Steckplatz liegt nicht im zulässigen Bereich. • Die zugewiesene Konfiguration hat einen leeren "internen" Steckplatz, z. B. Steckplatz n ist zugewiesen und Steckplatz n-1 ist nicht zugewiesen.

6.4.3 Beispiel für die Konfigurationssteuerung

Dieses Beispiel beschreibt eine Konfiguration bestehend aus einer CPU und drei E/A-Modulen. Das Modul an Steckplatz 3 ist im ersten tatsächlichen Aufbau nicht vorhanden und wird daher mit Hilfe der Konfigurationssteuerung "verborgen".

Im zweiten Aufbau wird das anfangs verborgene Modul zwar verwendet, aber nun im letzten Steckplatz konfiguriert. Ein geänderter Steuerdatensatz überträgt die Informationen über die Zuordnung der Module zu den Steckplätzen.

Beispiel: Tatsächlicher Aufbau mit einem konfigurierten, aber nicht benutzten Modul

Die Gerätekonfiguration enthält alle Module, die in einem tatsächlichen Aufbau vorhanden sein können (Maximalausbau). In diesem Fall ist das in der Gerätekonfiguration Steckplatz 3 zugewiesene Modul im realen Aufbau nicht vorhanden.

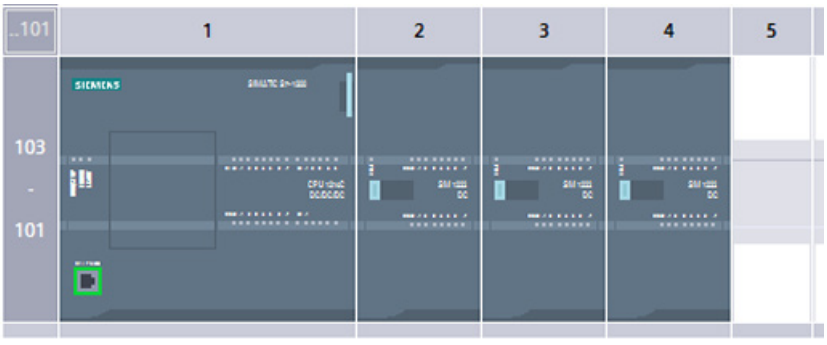


Bild 6-1 Gerätekonfiguration für Maximalausbau

Bild 6-2 Tatsächlicher Aufbau mit fehlendem Modul für Steckplatz 3 und dem für Steckplatz 4 konfigurierten Modul in Steckplatz 3

Um anzuzeigen, dass ein Modul fehlt, muss Steckplatz 3 im Steuerdatensatz mit 0 konfiguriert werden.

ControlDataRecord

	Name	Datentyp	Startwert	Kommentar
1	Static			
2	ConfigControl	Struct		
3	Block_length	USInt	16	Length of control data record, including header
4	Block_ID	USInt	196	Data record number
5	Version	USInt	5	
6	Subversion	USInt	0	
7	Slot_1	USInt	255	Assignment for CPU annex card/Actual annex ca.
8	Slot_2	USInt	2	Configured slot 2 / Assigned "real" slot
9	Slot_3	USInt	0	Configured slot 3 / Assigned "real" slot
10	Slot_4	USInt	3	Configured slot 4 / Assigned "real" slot
11	Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
12	Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
13	Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
14	Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
15	Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
16	Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
17	Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
18	Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

Beispiel: Tatsächlicher Aufbau mit einem Modul, das nachträglich einem anderen Steckplatz hinzugefügt wurde

Im zweiten Beispiel ist das Modul in Steckplatz 3 der Gerätekonfigurtion im tatsächlichen Aufbau vorhanden, es befindet sich jedoch in Steckplatz 4.

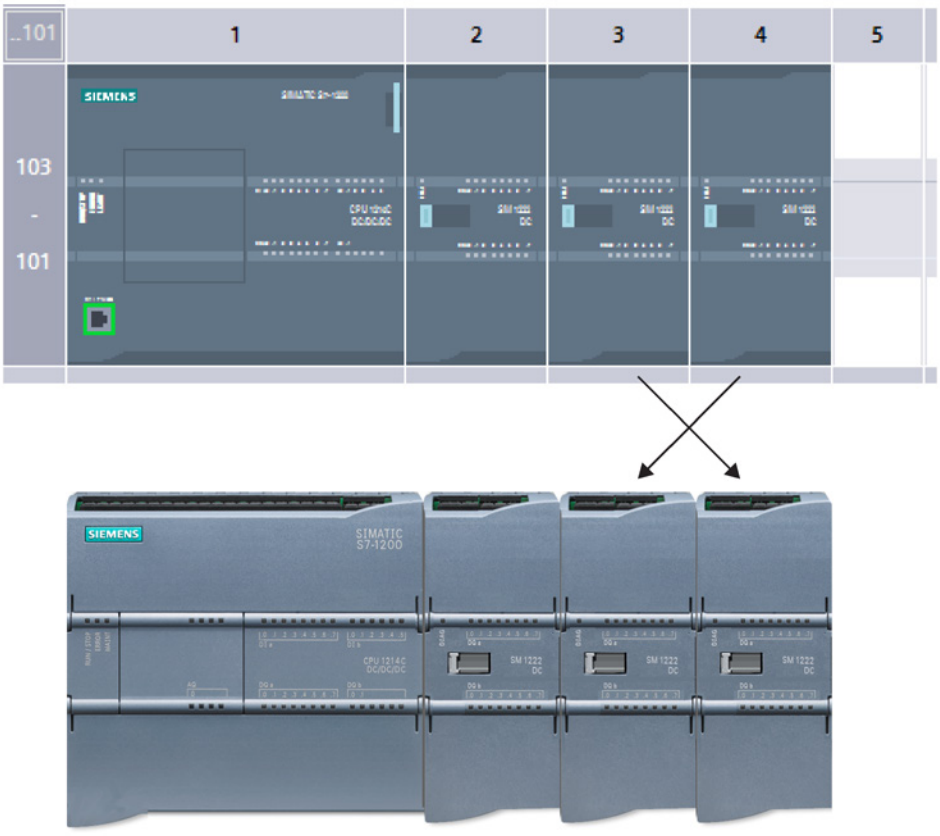


Bild 6-3 Module in Steckplätzen 3 und 4 aus Gerätekonfiguration im tatsächlichen Aufbau vertauscht

Um die Gerätekonfiguration an den tatsächlichen Aufbau anzupassen, sind die Module durch Bearbeiten des Steuerdatensatzes den richtigen Steckplätzen zuzuordnen.

ControlDataRecord

	Name	Datentyp	Startwert	Kommentar
1	Static			
2	ConfigControl	Struct		
3	Block_length	USInt	16	Length of control data record, including header
4	Block_ID	USInt	196	Data record number
5	Version	USInt	5	
6	Subversion	USInt	0	
7	Slot_1	USInt	255	Assignment for CPU annex card/Actual annex ca...
8	Slot_2	USInt	2	Configured slot 2 / Assigned "real" slot
9	Slot_3	USInt	4	Configured slot 3 / Assigned "real" slot
10	Slot_4	USInt	3	Configured slot 4 / Assigned "real" slot
11	Slot_5	USInt	255	Configured slot 5 / Assigned "real" slot
12	Slot_6	USInt	255	Configured slot 6 / Assigned "real" slot
13	Slot_7	USInt	255	Configured slot 7 / Assigned "real" slot
14	Slot_8	USInt	255	Configured slot 8 / Assigned "real" slot
15	Slot_9	USInt	255	Configured slot 9 / Assigned "real" slot
16	Slot_101	USInt	255	Configured slot 101 / Assigned "real" slot
17	Slot_102	USInt	255	Configured slot 102 / Assigned "real" slot
18	Slot_103	USInt	255	Configured slot 103 / Assigned "real" slot

6.5 Ändern eines Geräts

Sie können den Gerätetyp einer konfigurierten CPU oder eines Moduls ändern. Klicken Sie in der Gerätekonfiguration mit der rechten Maustaste auf das Gerät und wählen Sie im Kontextmenü den Befehl "Gerät ändern". Wählen Sie im Dialog die CPU oder das Modul aus, die/das Sie ersetzen möchten. Im Dialog "Gerät ändern" werden Ihnen Kompatibilitätsinformationen für die beiden betroffenen Geräte angezeigt.

Hinweis

Geräteaustausch: Ersetzen einer V3.0-CPU durch eine V4.1-CPU

Sie können ein STEP 7 V12-Projekt in STEP 7 V13 öffnen und V3.0-CPUs durch V4.1-CPUs ersetzen. Sie können keine CPUs mit Vorgängerversionen vor V3.0 ersetzen. Berücksichtigen Sie beim Ersetzen einer V3.0-CPU durch eine V4.1-CPU die Unterschiede (Seite 1349) in den Funktionen und im Verhalten zwischen den beiden Versionen und zwischen den von Ihnen zu ergreifenden Maßnahmen.

Wenn Sie mit einem Projekt für eine CPU-Version vor V3.0 arbeiten, müssen Sie die CPU zächst auf V3.0 hochrüsten und dann auf V4.1.

6.6 Konfigurieren des CPU-Betriebs

6.6.1 Übersicht

Um die Betriebsparameter der CPU zu konfigurieren, wählen Sie die CPU in der Gerätesicht aus (blauer Rahmen um die gesamte CPU) und öffnen dann im Inspektorfenster das Register "Eigenschaften".

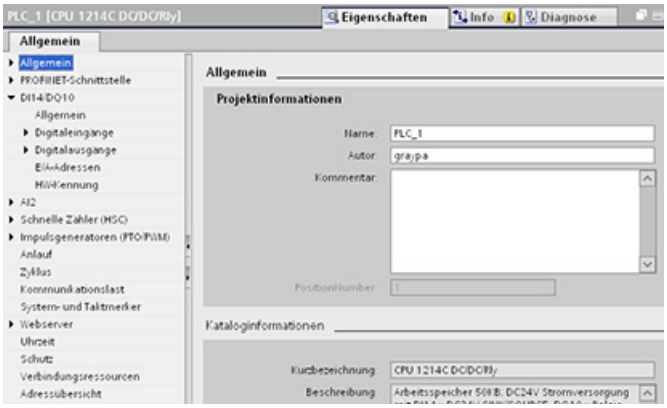


Tabelle 6- 2 CPU-Eigenschaften

Eigenschaft	Beschreibung
PROFINET-Schnittstelle	Einstellen der IP-Adresse für die CPU und die Uhrzeitsynchronisation
DE, DA und AE	Einstellen des Verhaltens der lokalen (integrierten) digitalen und analogen E/A (z.B. Filterzeiten für die Digitaleingänge und die Reaktion von Digitalausgängen auf das Versetzen der CPU in den Betriebszustand STOP).
Schnelle Zähler (Seite 465) und Impulsgeneratoren (Seite 414)	Aktivieren und Einstellen der schnellen Zähler (HSC) und der Impulsgeneratoren für die Impulsfolgen (PTO) und die Impulsdauermodulation (PWM) Wenn Sie die Ausgänge der CPU oder des Signalboards als Impulsgeneratoren (für PWM oder Bewegungssteuerungsanweisungen) konfigurieren, werden die entsprechenden Adressen der Ausgänge aus dem Speicher der Ausgänge entfernt und können nicht für andere Zwecke in Ihrem Anwenderprogramm verwendet werden. Wenn Ihr Anwenderprogramm einen Wert in einen Ausgang schreibt, der als Impulsgenerator genutzt wird, schreibt die CPU diesen Wert nicht in den physischen Ausgang.
Anlauf (Seite 85)	**Anlauf nach NETZ-EIN:** Einstellen des Verhaltens der CPU nach dem Aus- und Wiedereinschalten, z. B. für das Anlaufen im Betriebszustand STOP oder den Wechsel in RUN nach einem Warmstart
	Hardwarekompatibilität: Konfiguration der Ersatzstrategie für alle Systemkomponenten (SM, SB, CM, CP und CPU): • Akzeptablen Ersatz zulassen • Beliebigen Ersatz zulassen (Standard) Jedes Modul enthält interne Anforderungen an die Ersatzkompatibilität basierend auf der Anzahl der E/A, der elektrischen Verträglichkeit und anderen entsprechenden Vergleichspunkten. Beispiel: Ein SM mit 16 Kanälen kann ein akzeptabler Ersatz für ein SM mit 8 Kanälen sein, doch ein SM mit 8 Kanälen kann kein akzeptabler Ersatz für ein SM mit 16 Kanälen sein. Wenn Sie "Akzeptablen Ersatz zulassen" wählen, setzt STEP 7 die Ersetzungsregeln um. Ansonsten gestattet STEP 7 jeden Ersatz.
	Zuweisungszeit der Parameter für dezentrale E/A: Legt einen maximalen Zeitraum (Voreinstellung: 60000 ms) fest, in dem die dezentrale Peripherie online gebracht werden muss. (Die CMs und CPs werden während des Anlaufs von der CPU mit Spannung und Kommunikationsparametern versorgt. Diese Zuweisungszeit gestattet einen Zeitraum, während dessen die an CM oder CP angeschlossenen E/A-Module online gebracht werden müssen.) Die CPU geht in RUN, sobald die dezentrale Peripherie online ist, unabhängig von der Zuweisungszeit. Wurde die dezentrale Peripherie während dieses Zeitraums nicht online gebracht, geht die CPU trotzdem in RUN, ohne die dezentrale Peripherie. **Hinweis:** Wenn Ihre Konfiguration ein CM 1243-5 (PROFIBUS-Master) enthält, legen Sie für diesen Parameter keinen Wert unter 15 Sekunden (15000 ms) fest, um sicherzustellen, dass das Modul online gebracht werden kann.
Zyklus (Seite 104)	Festlegen einer maximalen Zykluszeit oder einer festen Mindestzykluszeit
Kommunikationslast	Zuweisen eines prozentualen Anteils der CPU-Zeit für Kommunikationsaufgaben
System- und Taktmerker (Seite 108)	Setzen eines Bytes für die Systemmerkerfunktionen und Setzen eines Bytes für die Taktmerkerfunktionen (dabei schaltet jedes Bit in einer vorgegebenen Frequenz ein und aus).
Webserver (Seite 817)	Aktiviert und konfiguriert die Webserverfunktion.
Tageszeit	Auswählen der Zeitzone und Einstellen der Sommer-/Winterzeit
Oberflächensprachen	Auswählen einer Sprache für Webserver- und CPU-Anzeige entsprechend der Projektsprache. Für bis zu zwei Projektsprachen können Sie eine entsprechende Oberflächensprache für Webserver- und CPU-Anzeige zuweisen.
Schutz (Seite 205)	Einstellen des Lese-/Schreibschutzes und der Passwörter für den Zugriff auf die CPU

Eigenschaft	Beschreibung
Verbindungsressourcen (Seite 633)	Bietet einen Überblick über die Kommunikationsverbindungsressourcen der CPU und die Anzahl der konfigurierten Verbindungsressourcen.
Übersicht über Adressen	Bietet einen Überblick über die E/A-Adressen, die für die CPU konfiguriert sind.

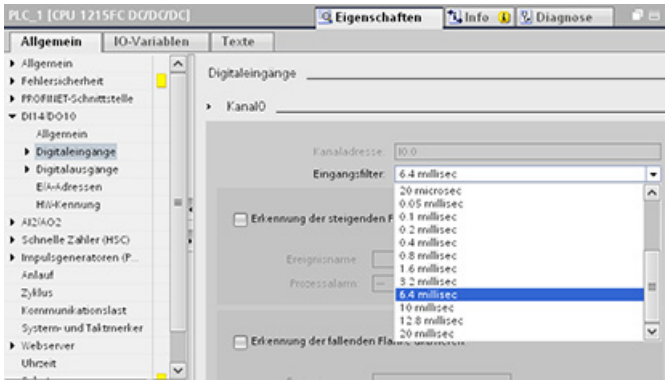
6.6.2 Filterzeiten für Digitaleingänge einrichten

Die Filter der digitalen Eingänge schützen Ihr Programm davor, auf unerwünschte schnelle Veränderungen der Eingangssignale zu reagieren, wie sie z. B. durch Kontaktprellen oder Rauschen verursacht werden können. Die voreingestellte Filterzeit von 6,4 ms verhindert unerwünschte Schalthandlungen durch typische mechanische Kontakte. Bestimmte Funktionen Ihrer Anwendung können kürzere Filterzeiten für die Erkennung und Reaktion auf Eingänge schneller Sensoren erfordern oder längere Filterzeiten für die Ausfilterung langsamer Kontaktpreller oder längeren Impulsrauschens erfordern.

Eine Eingangsfilterzeit von 6,4 ms bedeutet, dass ein einziger Signalwechsel von '0' auf '1' oder von '1' auf '0' etwa 6,4 ms lang anstehen muss, um erkannt zu werden, und dass einzelne hohe oder niedrige Impulse, die kürzer als 6,4 ms anstehen, nicht erkannt werden. Schaltet ein Eingangssignal schneller als die eingestellte Filterzeit zwischen '0' und '1', so kann sich der Wert des Eingangs im Anwenderprogramm ändern, wenn die Gesamtdauer der Impulse mit neuen Werten gegenüber den alten Werten die Filterzeit übersteigt.

Der Filter für Digitaleingänge funktioniert wie folgt:

- Wird "1" eingegeben, wird die Zeit hochgezählt und bei Erreichen der Filterzeit angehalten. Der Punkt im Prozessabbild wechselt von "0" auf "1", wenn die Filterzeit erreicht wird.
- Wird "0" eingegeben, wird die Zeit heruntergezählt und bei Erreichen von "0" angehalten. Der Punkt im Prozessabbild wechselt von "1" auf "0", wenn ein Wert "0" erreicht wird.
- Pendelt der Eingang, wird die Zeit teils hoch-, teils heruntergezählt. Das Prozessabbild ändert sich, wenn die Nettosumme der Zählwerte entweder die Filterzeit oder "0" erreicht.
- Ein schnell veränderliches Signal mit mehr "0" als "1" geht dann möglicherweise auf "0"; sind mehr "1" als "0" vorhanden, wechselt das Prozessabbildregister möglicherweise auf "1".



Jeder Eingang hat eine Filterkonfiguration, die für alle Zwecke gilt: Prozesseingänge, Alarme, Impulsabgriff und HSC-Eingänge. Um die Zeiten für den Eingangsfilter einzurichten, wählen Sie "Digitaleingänge" aus.

Voreingestellt ist eine Filterzeit von 6,4 ms für die Digitaleingänge. Sie können die Filterzeit für die Eingangsfilter aus der Klappliste auswählen. Gültige Filterzeiten liegen zwischen 0,1 µs und 20,0 ms.

⚠ WARNUNG

Gefahren beim Ändern der Filterzeit für digitale Eingangskanäle

Wenn die bisherige Einstellung der Filterzeit eines Digitaleingangskanals verändert wird, muss möglicherweise bis zu 20,0 ms lang ein neuer Eingangswert mit Pegel "0" vorhanden sein, damit der Filter vollständig auf neue Eingänge reagiert. Während dieses Zeitraums werden kurze Impulse mit Pegel "0", die kürzer als 20,0 ms sind, möglicherweise nicht erkannt oder gezählt.

Das Verändern der Filterzeiten kann unerwarteten Betrieb der Maschine bzw. des Prozesses verursachen, was zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen kann.

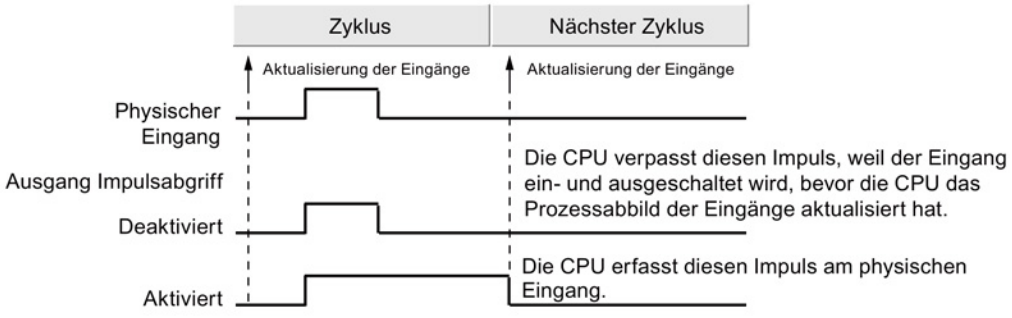
Um sicherzustellen, dass eine neue Filterzeit sofort wirksam wird, müssen Sie die CPU aus- und wieder einschalten.

6.6.3 Impulsabgriff

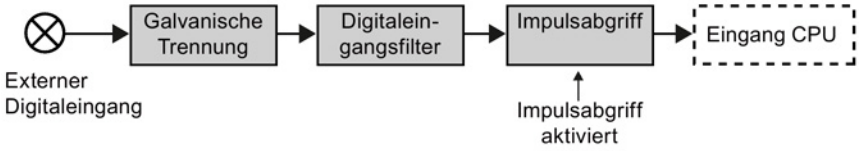
Die S7-1200 CPU bietet den Impulsabgriff für digitale Eingänge. Mit der Funktion "Impulsabgriff" können Sie hohe oder niedrige Impulse erfassen, die eine so kurze Dauer haben, dass sie von der CPU leicht übersehen werden können, wenn die Digitaleingänge zu Beginn eines Zyklus gelesen werden.

Ist die Funktion "Impulsabgriff" für einen Eingang aktiviert, wird ein Signalwechsel an dem Eingang so lange gehalten, bis die Aktualisierung des nächsten Zyklus stattfindet. Dadurch wird sichergestellt, dass ein Impuls, der nur kurze Zeit anhält, erfasst und gehalten wird, bis die CPU die Eingänge liest.

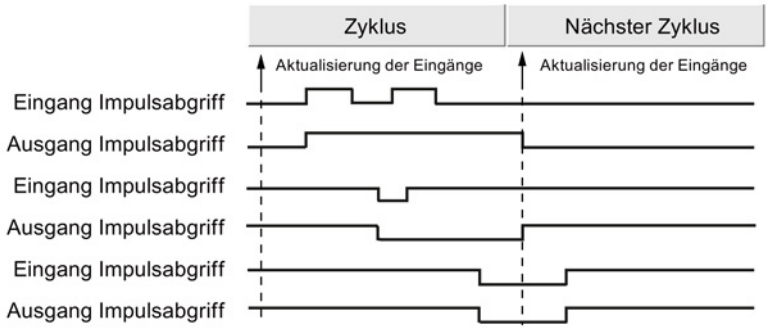
Das folgende Bild zeigt die grundlegende Funktionsweise der S7-1200 CPU mit und ohne aktivierten Impulsabgriff:



Weil sich der Impulsabgriff auf den Eingang auswirkt, nachdem dieser gefiltert wurde, müssen Sie die Zeit für den Eingangsfilter so einstellen, dass der Impuls nicht vom Filter zurückgenommen wird. Das folgende Bild zeigt eine schematische Darstellung des Digitaleingangskreises:



Das folgende Bild zeigt die Reaktion des aktivierten Impulsabgriffs auf verschiedene Eingangsbedingungen. Gibt es mehr als einen Impuls in einem bestimmten Zyklus, wird nur der erste Impuls gelesen. Bei mehreren Impulsen in einem Zyklus sollten Sie die Interruptereignisse für steigende/fallende Flanken einsetzen:



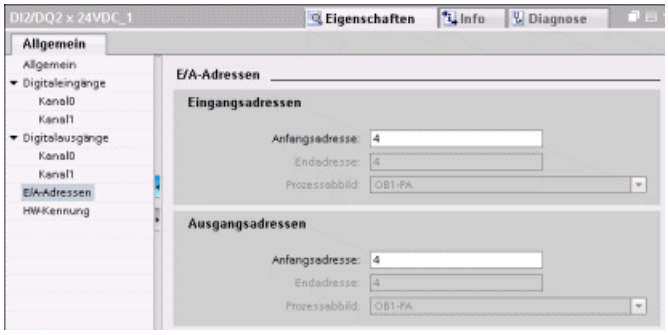
6.7 Modulparameter konfigurieren

Um die Betriebsparameter der Module zu konfigurieren, wählen Sie das Modul in der Gerätesicht aus und öffnen im Inspektorfenster das Register "Eigenschaften", um die Parameter für das Modul einzurichten.

Signalmodul (SM) oder Signalboard (SB) konfigurieren

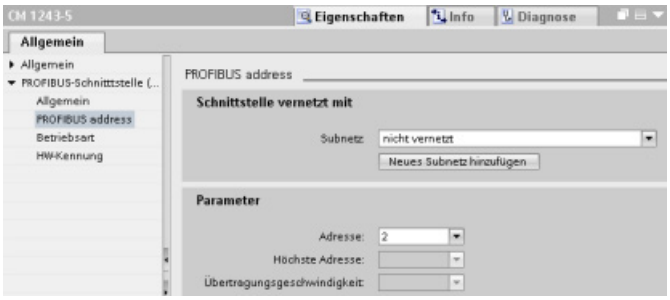
In der Gerätekonfiguration für Signalmodule und Signalboards können Sie Folgendes konfigurieren:

- Digitale E/A: Sie können Eingänge für die Erkennung steigender oder fallender Flanken (jeweils mit Zuweisung eines Ereignisses und eines Prozessalarms) oder für den "Impulsabgriff" (Eingang bleibt nach einem Impuls eingeschaltet) durch die nächste Aktualisierung des Prozessabbilds der Eingänge konfigurieren. Die Ausgänge können eingefroren werden oder es können Ersatzwerte aufgeschaltet werden.
- Analoge E/A: Für einzelne Eingänge konfigurieren Sie Parameter, wie z.B. Messart (Spannung oder Strom), Bereich und Glättung sowie Freigabe der Unter- oder Überlaufdiagnose. Die Analogausgänge bieten Parameter wie Ausgangsart (Spannung oder Strom) und Diagnose, wie z. B. Kurzschluss (bei Spannungsausgängen) oder Diagnose der oberen/unteren Grenzwerte. Die Bereiche von Analogeingängen und Analogausgängen in physikalischen Einheiten konfigurieren Sie nicht im Dialog "Eigenschaften". Dies nehmen Sie in Ihrer Programmlogik vor (siehe "Verarbeitung von Analogwerten (Seite 118)").
- E/A-Adressen: Sie konfigurieren die Startadresse für die Ein- und Ausgänge des Moduls. Sie können die Eingänge und Ausgänge auch einem Teilprozessabbild (TPA0, TPA1, TPA2, TPA3, TPA4), der automatischen Aktualisierung oder der Verwendung keines Teilprozessabbilds zuweisen. Eine Erläuterung von Prozessabbildern und Teilprozessabbildern finden Sie unter "Ausführung des Anwenderprogramms" (Seite 81).



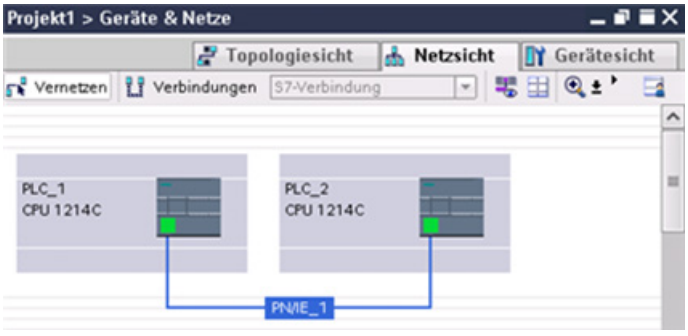
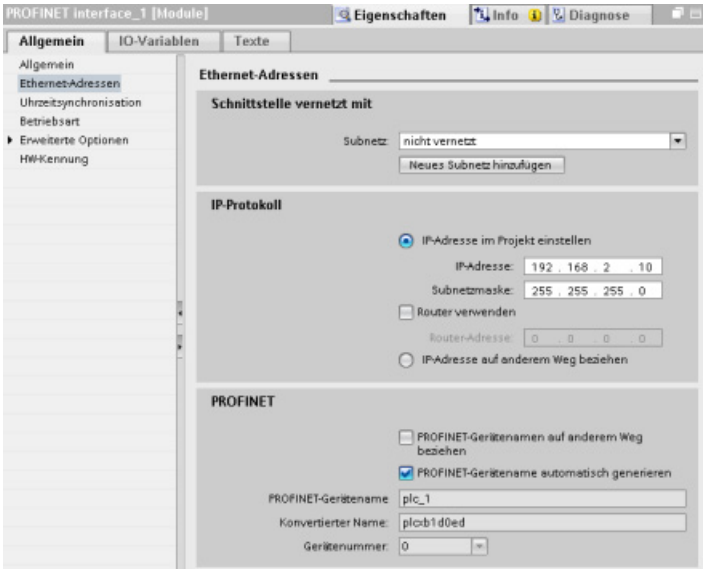
Kommunikationsschnittstelle (CM, CP oder CB) konfigurieren

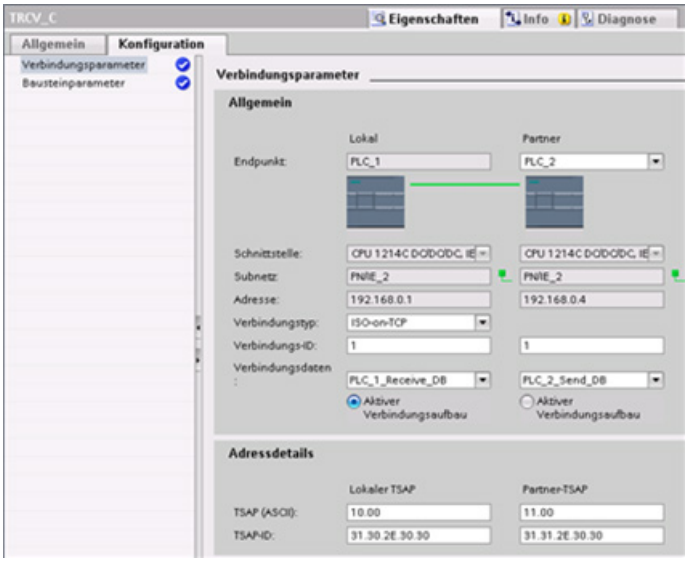
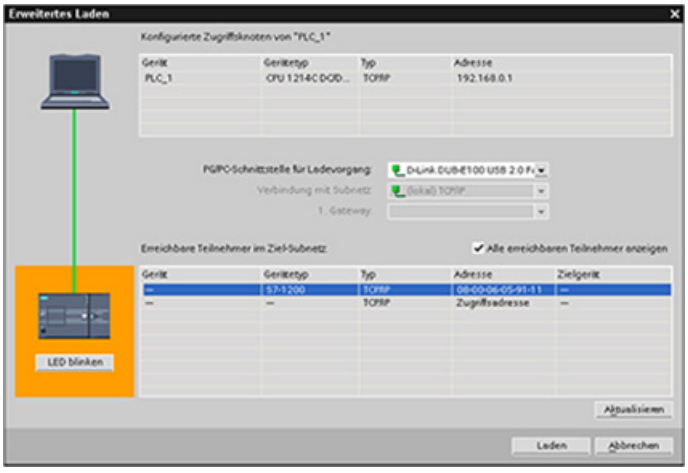
Sie konfigurieren die Parameter für das Netzwerk je nach Typ der Kommunikationsschnittstelle.



6.8 CPU für die Kommunikation konfigurieren

Die S7-1200 erfüllt Ihre Kommunikations- und Vernetzungsanforderungen durch Unterstützung sowohl einfacher als auch komplexer Netze. Die S7-1200 bietet außerdem Werkzeuge für die Kommunikation mit anderen Geräten, z.B. mit Druckern und Waagen, die über eigene Kommunikationsprotokolle verfügen.

	In der Netzsicht der Gerätekonfiguration können Sie die Netzwerkverbindungen zwischen den Geräten in Ihrem Projekt herstellen. Nach dem Herstellen der Netzwerkverbindung können Sie im Register "Eigenschaften" des Inspektorfensters die Netzwerkparameter konfigurieren. Weitere Informationen finden Sie unter "Erstellen einer Netzwerkverbindung" (Seite 637).
	Wählen Sie im Dialog "Eigenschaften" den Eintrag "Ethernet-Adressen". STEP 7 zeigt den Dialog für die Konfiguration der Ethernet-Adresse an, in dem Sie dem Softwareprojekt die IP-Adresse der CPU zuweisen, in die das Projekt geladen wird. Hinweis: Die S7-1200 CPU hat keine vorkonfigurierte IP-Adresse. Sie müssen der CPU daher manuell eine IP-Adresse zuweisen. Weitere Informationen finden Sie unter "Internet Protocol (IP)-Adressen zuweisen" (Seite 642).

	Bei den TCP-, ISO-on-TCP- und UDP-Ethernet-Protokollen konfigurieren Sie die Verbindungen der lokalen und der Partner-CPU in den "Eigenschaften" der Anweisung (TSEND_C, TRCV_C oder TCON). Die Abbildung zeigt die "Verbindungseigenschaften" im Register "Verbindung" einer ISO-on-TCP-Verbindung. Weitere Informationen finden Sie unter "Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren" (Seite 638).
	Nach Abschluss der Konfiguration laden Sie das Projekt in die CPU. Alle IP-Adressen werden beim Laden des Projekts konfiguriert. Weitere Informationen finden Sie unter "Testen des PROFINET-Netzwerks" (Seite 650).

Hinweis

Für die Verbindung zu Ihrer CPU müssen Ihre Netzwerkkarte (NIC) und die CPU der gleichen Netzwerkklasse und dem gleichen Subnetz angehören. Sie können entweder Ihre Netzwerkkarte nach der voreingestellten IP-Adresse der CPU einrichten oder die IP-Adresse der CPU passend zur Netzwerkklasse und zum Subnetz der Netzwerkschnittstellenkarte ändern.

Weitere Informationen finden Sie unter "Internet Protocol (IP)-Adressen zuweisen" (Seite 642).

Programmiergrundlagen 7

7.1 Richtlinien für das Entwerfen einer Automatisierungslösung mit einem PLC-Gerät

Bei der Entwicklung eines PLC-Systems können Sie aus einer Vielzahl von Methoden und Kriterien auswählen. Die folgenden allgemeinen Richtlinien sind auf viele Projekte anwendbar. Dabei sollten Sie sich selbstverständlich an die Verfahrensanweisungen in Ihrem Unternehmen halten und Ihre eigenen Erfahrungen berücksichtigen.

Tabelle 7- 1 Richtlinien für das Entwerfen einer Automatisierungslösung mit einem PLC-Gerät

Empfohlene Schritte	Aufgaben
Gliedern Ihres Prozesses bzw. Ihrer Anlage	Unterteilen Sie Ihren Prozess bzw. Ihre Anlage in Abschnitte, die voneinander unabhängig sind. Diese Abschnitte legen die Grenzen zwischen mehreren Automatisierungssystemen fest und beeinflussen die Beschreibungen der Funktionsbereiche sowie die Zuordnung der Betriebsmittel.
Beschreiben der Funktionsbereiche	Geben Sie die Funktionsbeschreibungen für jeden Abschnitt des Prozesses bzw. der Anlage ein, wie z. B. Ein-/Ausgänge, die funktionale Beschreibung des Ablaufs, die Zustände, die erreicht werden müssen, bevor ein Aktor (z. B. ein Magnetventil, ein Motor oder ein Antrieb) reagieren darf, eine Beschreibung der Benutzeroberfläche und aller Schnittstellen zu anderen Abschnitten des Prozesses oder der Anlage.
Entwerfen der Sicherheitsstromkreise	Bestimmen Sie die Geräte, die aus Sicherheitsgründen festverdrahtete Schaltungen benötigen. Steuerungsgeräte können unsichere Betriebszustände einnehmen, woraus unerwartete Anlaufeigenschaften bzw. geänderte Funktionsabläufe der Anlage resultieren können. Besteht die Gefahr, dass bei unerwartetem bzw. fehlerhaftem Betrieb der Anlage schwere Körperverletzungen oder Sachschäden auftreten, sollten Sie die Einrichtung elektromechanischer Programmeingriffe in Erwägung ziehen, die unabhängig von der Steuerung arbeiten, um unsichere Betriebszustände zu vermeiden. Zum Entwerfen von Sicherheitsstromkreisen gehen Sie folgendermaßen vor: • Definieren Sie falschen bzw. unerwarteten Betrieb von Aktoren, die Gefahrenpotentiale bergen. • Definieren Sie die Bedingungen, unter denen der Betrieb ungefährlich ist, und legen Sie fest, wie diese Bedingungen unabhängig von der SPS erkannt werden. • Definieren Sie, wie die SPS den Prozess beeinflusst, wenn die Spannung eingeschaltet und wieder ausgeschaltet wird und wenn Fehler erkannt werden. Verwenden Sie diese Informationen nur für den Entwurf des normalen und erwarteten unnormalen Betriebs. Verlassen Sie sich aus Sicherheitsgründen nicht auf dieses "Best-Case"-Szenario. • Entwerfen Sie Korrekturen durch Handeingriff bzw. elektromechanische Programmeingriffe, mit denen gefährliche Abläufe unabhängig von der SPS gesperrt werden. • Lassen Sie von den unabhängigen Stromkreisen Statusinformationen an die SPS übermitteln, so dass das Programm und jede Bedienerschnittstelle über die erforderlichen Informationen verfügt. • Definieren Sie weitere Sicherheitsanforderungen, damit der Prozess sicher ablaufen kann.
Systemsicherheit planen	Ermitteln Sie, welche Schutzstufe (Seite 205) Sie für den Zugriff auf Ihren Prozess benötigen. Sie können CPUs und Programmbausteine mittels Passwort vor unberechtigtem Zugriff schützen.

Empfohlene Schritte	Aufgaben
Definieren der OperatorStationen	Erstellen Sie die folgenden Pläne der OperatorStationen anhand der Anforderungen in den Beschreibungen der Funktionsbereiche: • Position aller OperatorStationen in Bezug zum Prozess bzw. zur Anlage • Mechanische Anordnung der Geräte der Operator-Station, z. B. Display, Schalter und Lampen. • Verdrahtungspläne mit den zugehörigen Ein und Ausgängen der SPS und Signalmodule
Zeichnen der Konfigurationspläne	Erstellen Sie die Konfigurationspläne für das Automatisierungssystem anhand der Anforderungen in den Beschreibungen der Funktionsbereiche: • Positionen aller SPS in Bezug zum Prozess bzw. zur Anlage • Mechanische Anordnung aller SPS und E/A-Module (einschließlich Schränke usw.) • Verdrahtungspläne für alle SPS und E/A-Module, einschließlich Gerätenummern, Kommunikationsadressen und Adressen der Ein und Ausgänge
Auflisten der symbolischen Namen	Erstellen Sie eine Liste der symbolischen Namen für die absoluten Adressen. Geben Sie nicht nur die physikalischen Ein und Ausgänge, sondern auch alle anderen Elemente (wie z. B. Variablennamen) an, die Sie in Ihrem Programm verwenden.

7.2 Strukturieren Ihres Anwenderprogramms

Beim Erstellen eines Anwenderprogramms für Automatisierungslösungen fügen Sie die Anweisungen für das Programm in Codebausteine ein:

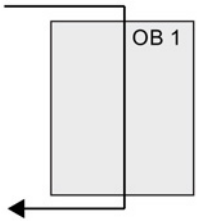
- Ein Organisationsbaustein (OB) reagiert auf ein bestimmtes Ereignis in der CPU und kann die Ausführung des Anwenderprogramms unterbrechen. Der Standardbaustein für die zyklische Ausführung des Anwenderprogramms (OB 1) stellt die Grundstruktur für Ihr Anwenderprogramm bereit. Fügen Sie weitere OBs in Ihr Programm ein, so unterbrechen diese OBs die Ausführung von OB 1. Die anderen OBs führen spezifische Funktionen aus, wie z. B. für Anlauf, Interrupt- und Fehlerbearbeitung, oder für die Ausführung eines spezifischen Programmcodes in bestimmten Zeitabständen.
- Ein Funktionsbaustein (FB) ist ein Unterprogramm, das ausgeführt wird, wenn es aus einem anderen Codebaustein (OB, FB oder FC) heraus aufgerufen wird. Der aufrufende Baustein übergibt dem FB die Parameter und definiert einen bestimmten Datenbaustein (DB), der die Daten für diesen Aufruf oder diese Instanz des FBs speichert. Wird der Instanz-DB geändert, so kann ein allgemeiner FB den Betrieb einer Gerätegruppe steuern. Beispielsweise kann ein FB mehrere Pumpen oder Ventile mit unterschiedlichen Instanz-DBs, die die speziellen Betriebsparameter jeder Pumpe bzw. jedes Ventils enthalten, steuern.
- Eine Funktion (FC) ist eine Unterprogramm, das ausgeführt wird, wenn es aus einem anderen Codebaustein (OB, FB oder FC) heraus aufgerufen wird. Einer FC ist kein Instanz-DB zugeordnet. Der aufrufende Baustein übergibt der FC die Parameter. Die Ausgangswerte der FC müssen in eine Speicheradresse oder in einen globalen DB geschrieben werden.

Art der Struktur für das Anwenderprogramm wählen

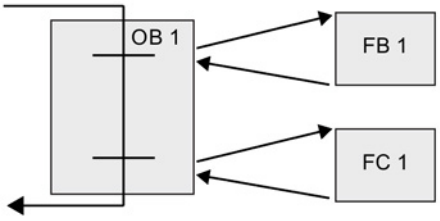
Je nach den Anforderungen Ihrer Anwendung können Sie eine lineare oder eine modulare Struktur für Ihr Anwenderprogramm wählen:

- Ein lineares Programm führt alle Anweisungen für Ihre Automatisierungsaufgaben nacheinander aus. Typischerweise werden bei einem linearen Programm alle Programmanweisungen im Zyklus-OB (OB 1) abgelegt.
- Ein modulares Programm ruft spezielle Codebausteine auf, die spezifische Aufgaben ausführen. Um eine modulare Programmstruktur aufzubauen, gliedern Sie die komplexe Automatisierungsaufgabe in kleinere Teilaufgaben, die den technologischen Funktionen des Prozesses entsprechen. Jeder Codebaustein enthält das Programmsegment für die jeweilige Teilaufgabe. Sie strukturieren Ihr Programm durch den Aufruf eines Codebausteins aus einem anderen Baustein.

Lineare Struktur:



Modulare Struktur:



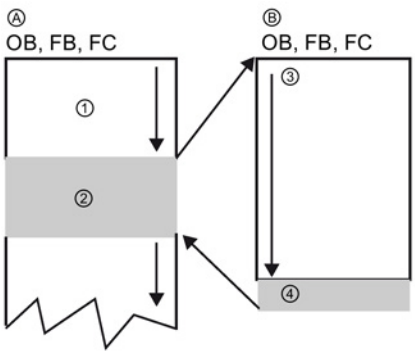
Durch das Anlegen allgemeiner Codebausteine, die im Anwenderprogramm mehrfach genutzt werden können, lassen sich Entwurf und Implementierung des Anwenderprogramms vereinfachen. Die Arbeit mit allgemeinen Codebausteinen hat mehrere Vorteile:

- Sie können wiederverwendbare Codebausteine für Standardaufgaben, wie z. B. für die Steuerung einer Pumpe oder eines Motors, erstellen. Sie können diese allgemeinen Codebausteine ferner in einer Bibliothek speichern, die für verschiedene Anwendungen oder Lösungen verwendet werden kann.
- Durch die modulare Gliederung des Anwenderprogramms in einzelne Komponenten, die sich auf funktionale Aufgaben beziehen, wird der Programmaufbau übersichtlicher und ist einfacher zu handhaben. Die modularen Komponenten ermöglichen nicht nur die Standardisierung des Programmentwurfs, sondern machen auch Anpassungen oder Änderungen des Programmcodes schneller und einfacher.
- Das Erstellen modularer Komponenten vereinfacht das Testen Ihres Programms. Wenn das komplette Programm in eine Folge modularer Programmsegmente gegliedert ist, können Sie die Funktionalität jedes Codebausteins direkt während der Entwicklung testen.
- Mit modularen Komponenten, die sich auf spezifische technologische Funktionen beziehen, wird die Inbetriebnahme der kompletten Anwendung vereinfacht und verkürzt.

7.3 Strukturierung des Programms mit Hilfe von Bausteinen

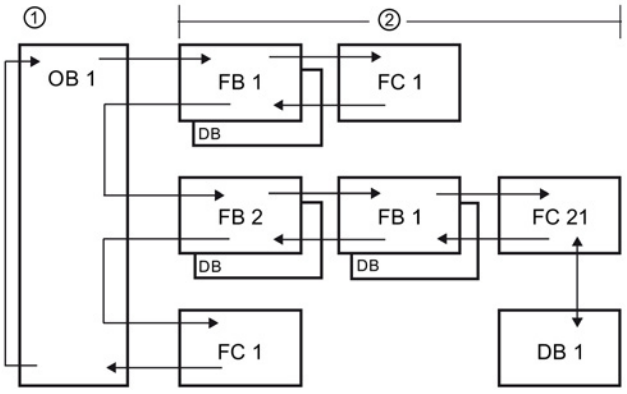
Modulare Codebausteine erstellen Sie durch den Entwurf von FBs und FCs für die Ausführung allgemeiner Aufgaben. Dann strukturieren Sie Ihr Programm, indem andere Codebausteine diese wiederverwendbaren Module aufrufen. Der aufrufende Baustein gibt gerätespezifische Parameter an den aufgerufenen Baustein weiter.

Wird ein Codebaustein von einem anderen Codebaustein aufgerufen, führt die CPU den Programmcode im aufgerufenen Baustein aus. Nachdem der aufgerufene Baustein abgearbeitet ist, setzt die CPU die Ausführung des aufrufenden Bausteins fort. Die Bearbeitung geht weiter mit der Ausführung der nächsten Anweisung nach dem Bausteinauruf.



A Aufrufender Baustein
B Aufgerufener (oder unterbrechender) Baustein
① Programmausführung
② Anweisung oder Ereignis, das die Ausführung eines anderen Bausteins auslöst
③ Programmausführung
④ Bausteinende (Rückkehr zum aufrufenden Baustein)

Die Bausteinaufrufe können verschachtelt werden, um die Struktur noch modularer zu gestalten. Im folgenden Beispiel umfasst die Schachtelungstiefe 3 Ebenen: den Programmzyklus-OB plus 3 Ebenen mit Aufrufen von Codebausteinen.



① Zyklusbeginn
② Schachtelungstiefe

7.3.1 Organisationsbaustein (OB)

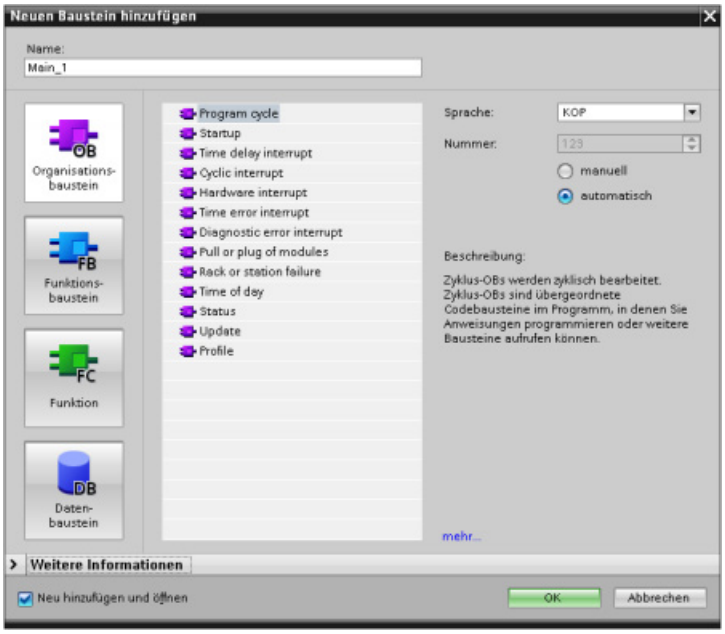
Organisationsbausteine dienen zur Strukturierung Ihres Programms. Sie bilden die Schnittstelle zwischen dem Betriebssystem und dem Anwenderprogramm. OBs sind ereignisgesteuert. Die Ausführung eines OBs durch das Zielsystem wird von einem Ereignis, wie z. B. einem Diagnosealarm oder einem Zeitintervall, angestoßen. Einige OBs haben voreingestellte Startereignisse und Anlaufverhalten.

Der Programmzyklus-OB enthält das Hauptprogramm. Sie können mehrere Programmzyklus-OBs in Ihr Anwenderprogramm aufnehmen. Im Betriebszustand RUN werden die Programmzyklus-OBs mit der niedrigsten Prioritätsstufe ausgeführt und können durch alle anderen Ereignisarten unterbrochen werden. Der Anlauf-OB unterbricht den Programmzyklus-OB nicht, weil die CPU den Anlauf-OB vor dem Wechsel in RUN ausführt.

Nach Abarbeitung des Programmzyklus-OBs führt die CPU die Programmzyklus-OBs sofort erneut aus. Diese zyklische Ausführung ist die "normale" Ablaufart für speicherprogrammierbare Steuerungen. In vielen Anwendungen ist das gesamte Anwenderprogramm in einem einzigen Programmzyklus-OB enthalten.

Sie können andere OBs anlegen, um bestimmte Funktionen auszuführen, z. B. für die Alarm- und Fehlerbearbeitung oder für die Ausführung eines bestimmten Programmcodes in bestimmten Abständen. Diese OBs unterbrechen die Ausführung der Programmzyklus-OBs.

Neue OBs für Ihr Anwenderprogramm legen Sie im Dialog "Neuen Baustein hinzufügen" an.



Die Alarmbearbeitung ist immer ereignisgesteuert. Tritt ein solches Ereignis auf, so unterbricht die CPU die Ausführung des Anwenderprogramms und ruft den OB für die Behandlung dieses Ereignisses auf. Nach der Ausführung des unterbrechenden OBs setzt die CPU die Ausführung des Anwenderprogramms an der Stelle fort, an der es zuvor unterbrochen wurde.

Die CPU ermittelt die Reihenfolge für die Bearbeitung von Alarmereignissen anhand der Prioritäten. Sie können einer Prioritätsklasse mehrere Alarmereignisse zuweisen. Weitere Informationen finden Sie in den Themen zu Organisationsbausteinen (Seite 90) und zur Ausführung des Anwenderprogramms (Seite 81).

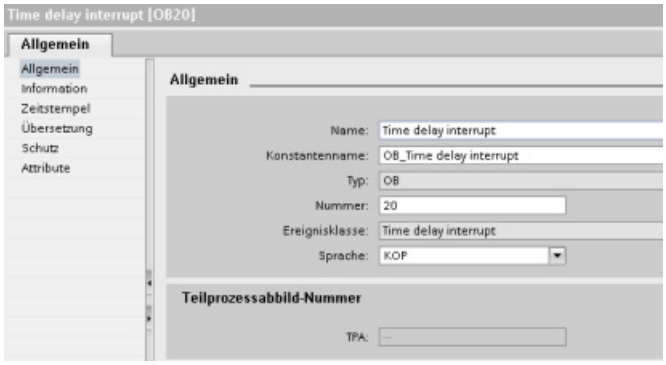
Zusätzliche OBs anlegen

Sie können mehrere OBs für Ihr Anwenderprogramm anlegen. Dies gilt auch für die Programmzyklus- und Anlauf-OB-Ereignisse. Im Dialog "Neuen Baustein hinzufügen" können Sie einen OB anlegen und einen Namen für den neuen OB eingeben.

Wenn Sie mehrere Programmzyklus-OBs für Ihr Anwenderprogramm anlegen, führt die CPU die einzelnen Programmzyklus-OBs in numerischer Reihenfolge aus. Dabei wird mit dem Programmzyklus-OB mit der niedrigsten Nummer begonnen (z. B. OB 1). Beispiel: Nach dem Ende des ersten Programmzyklus-OBs (z. B. OB 1) führt die CPU den Programmzyklus-OB mit der nächsthöheren Nummer aus.

Eigenschaften eines OBs konfigurieren

Sie können die Eigenschaften eines OBs, beispielsweise die OB-Nummer oder die Programmiersprache, konfigurieren.



Hinweis

Beachten Sie, dass Sie einem OB eine Teilprozessabbild-Nummer wie TPA0, TPA1, TPA2, TPA3 oder TPA4 zuweisen können. Wenn Sie für die Teilprozessabbild-Nummer eine Nummer eingeben, erstellt die CPU das Teilprozessabbild. Erläuterungen zu Teilprozessabbildern finden Sie unter "Ausführung des Anwenderprogramms (Seite 81)".

7.3.2 Funktion (FC)

Eine Funktion (FC) ist ein Codebaustein, der typischerweise eine bestimmte Operation mit einer Anzahl von Eingangswerten durchführt. Die FC speichert die Ergebnisse dieser Operation an bestimmten Speicherorten. Verwenden Sie beispielsweise FCs, um Standardoperationen und mehrfach verwendbare Operationen (z. B. mathematische Berechnungen) oder technologische Funktionen (z. B. für einzelne Steuerungsvorgänge über Bitverknüpfungsoperationen) durchzuführen. Eine FC kann auch mehrmals an verschiedenen Stellen eines Programms aufgerufen werden. Diese Wiederverwendung vereinfacht die Programmierung häufig wiederkehrender Aufgaben.

Einer FC ist kein zugehöriger Instanz-Datenbaustein(DB) zugeordnet. Die FC verwendet den lokalen Datenspeicher für die temporären Daten, die für die Berechnung der Operation benötigt werden. Die temporären Daten werden nicht gespeichert. Für die dauerhafte Datenspeicherung muss der Ausgangswert einem globalen Speicherplatz, wie z. B. dem Merkerspeicher, oder einem globalen DB zugewiesen werden.

7.3.3 Funktionsbaustein (FB)

Ein Funktionsbaustein (FB) ist ein Codebaustein, der für seine Parameter und statischen Daten einen Instanz-Datenbaustein nutzt. FBs haben einen variablen Speicher, der sich in einem Datenbaustein (DB) oder einem Instanz-DB befindet. Der Instanz-DB stellt einen Speicherbaustein bereit, der dieser Instanz (oder diesem Aufruf) des FBs zugewiesen ist und die Daten nach Ablauf des FBs speichert. Sie können verschiedenen Aufrufen des FBs verschiedene Instanz-DBs zuordnen. Der Instanz-DB ermöglicht es Ihnen, einen allgemeinen FB für die Steuerung mehrerer Geräte zu verwenden. Sie können Ihr Programm strukturieren, indem ein Codebaustein einen FB und einen Instanz-DB aufruft. Die CPU führt dann den Programmcode in diesem FB aus und speichert die Bausteinparameter und die statischen Lokaldaten im Instanz-DB. Wenn die Ausführung des FBs beendet ist, setzt die CPU die Ausführung mit dem Codebaustein fort, der den FB aufgerufen hatte. Der Instanz-DB speichert die Werte für diese Instanz des FBs. Diese Werte stehen nachfolgenden Aufrufen des Funktionsbausteins entweder in demselben Zyklus oder in anderen Zyklen zur Verfügung.

Wiederverwendbare Codebausteine mit zugewiesenem Speicher

Typischerweise wird ein FB für die Steuerung des Ablaufs von Tätigkeiten oder Geräten verwendet, deren Betrieb nicht innerhalb eines Zyklus endet. Für die Speicherung der Betriebsparameter, damit diese zwischen zwei Zyklen schnell zugänglich sind, hat jeder FB in Ihrem Anwenderprogramm einen oder mehrere Instanz-DBs. Mit dem Aufruf eines FBs geben Sie gleichzeitig einen Instanz-DB an, in dem die Bausteinparameter und die statischen Lokaldaten für diesen Aufruf oder diese "Instanz" des FBs enthalten sind. Der Instanz-DB speichert diese Werte nach der Ausführung des FBs.

Wird der FB für allgemeine Steuerungsaufgaben entworfen, so kann er für mehrere Geräte verwendet werden, indem verschiedene Instanz-DBs für die verschiedenen Aufrufe des FBs ausgewählt werden.

Ein FB speichert die Eingangs-, Ausgangs- und Durchgangsparameter sowie die statischen Parameter in einem Instanz-DB.

Sie können ferner die Funktionsbausteinschnittstelle in Betriebszustand RUN ändern und laden (Seite 1144).

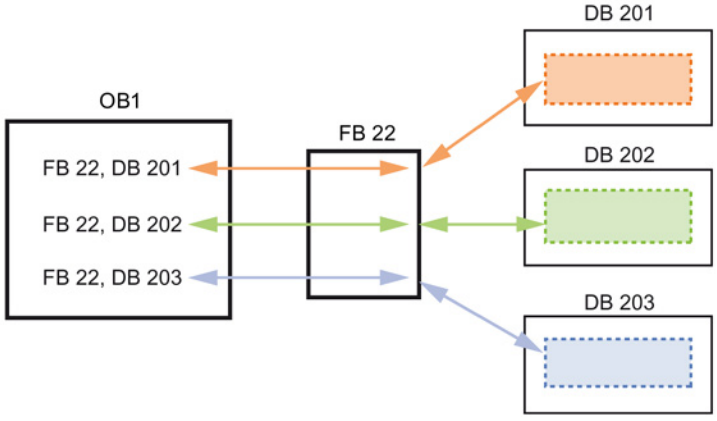
Startwert im Instanz-DB zuweisen

Der Instanz-DB speichert sowohl einen Standardwert als auch einen Startwert für jeden Parameter. Der Startwert gibt den Wert an, der verwendet werden soll, wenn der FB ausgeführt wird. Der Startwert kann dann während der Ausführung Ihres Anwenderprogramms geändert werden.

Die FB-Schnittstelle verfügt auch über eine Spalte für den Standardwert, in der Sie beim Schreiben des Programmcode einen neuen Startwert für den Parameter eingeben können. Dieser Standardwert im FB wird dann in den Startwert des zugehörigen Instanz-DBs übertragen. Wenn Sie in der FB-Schnittstelle einem Parameter keinen neuen Startwert zuweisen, wird der Standardwert aus dem Instanz-DB in den Startwert kopiert.

Einzelnen FB mit DBs verwenden

Die folgende Abbildung zeigt einen OB, der einen FB dreimal aufruft, wobei für jeden Aufruf ein anderer Datenbaustein verwendet wird. Durch diese Struktur kann ein allgemeiner FB für die Steuerung mehrerer gleichartiger Geräte wie z. B. Motoren verwendet werden, indem jedem Aufruf eines Geräts ein anderer Instanzdatenbaustein zugewiesen wird. Jeder Instanz-DB speichert die Daten (wie Drehzahl, Hochlaufzeit und Gesamtbetriebszeit) für ein einziges Gerät.



In diesem Beispiel steuert FB 22 drei verschiedene Geräte, DB 201 speichert die Betriebsdaten für das erste Gerät, DB 202 die Betriebsdaten für das zweite Gerät und DB 203 die Betriebsdaten für das dritte Gerät.

7.3.4 Datenbaustein (DB)

Sie können in Ihrem Anwenderprogramm Datenbausteine (DBs) zum Speichern der Daten für die Codebausteine anlegen. Alle Programmbausteine im Anwenderprogramm können auf die Daten in einem globalen DB zugreifen, doch ein Instanz-DB speichert Daten für einen spezifischen Funktionsbaustein (FB).

Die gespeicherten Daten in einem DB werden nach der Ausführung des zugehörigen Codebausteins nicht gelöscht. Es gibt zwei Arten von Datenbausteinen:

- In einem globalen DB werden die Daten für die Codebausteine in Ihrem Programm gespeichert. Jeder OB, FB oder FC kann auf die Daten in einem globalen DB zugreifen.
- In einem Instanz-DB werden die Daten für einen spezifischen FB gespeichert. Die Datenstruktur in einem Instanz-DB entspricht den Parametern (Input, Output und InOut) und den statischen Daten des FBs. (Der temporäre Speicher des FBs wird im Instanz-DB nicht gespeichert.)

Hinweis

Obwohl der Instanz-DB die Daten für einen bestimmten FB enthält, kann jeder Codebaustein auf die Daten eines Instanz-DBs zugreifen.

Sie können Datenbausteine auch im Betriebszustand RUN ändern und laden (Seite 1144).

Schreibgeschützte Datenbausteine

Ein DB kann so konfiguriert werden, dass er nur gelesen werden kann:

1. Klicken Sie mit der rechten Maustaste in der Projektnavigation auf den DB und wählen Sie im Kontextmenü "Eigenschaften".
2. Wählen Sie Dialog "Eigenschaften" das Element "Attribute".
3. Wählen Sie die Option "Datenbaustein im Gerät schreibgeschützt" und bestätigen Sie mit "OK".

Optimierte und Standard-Datenbausteine

Sie können einen Datenbaustein auch als Standard-Datenbaustein oder als optimierten Datenbaustein konfigurieren. Ein Standard-DB ist mit den Programmierwerkzeugen von STEP 7 Classic und den klassischen S7-300 und S7-400 CPUs kompatibel. Datenbausteine mit optimiertem Zugriff haben keine fest definierte Struktur. Die Datenelemente enthalten nur einen symbolischen Namen in der Deklaration und keine feste Adresse im Baustein. Die CPU speichert die Elemente automatisch im verfügbaren Speicherbereich des Bausteins, so dass keine Lücken im Speicher entstehen. Dadurch wird die Speicherkapazität optimal ausgenutzt.

Um für einen Datenbaustein den optimierten Zugriff festzulegen, gehen Sie wie folgt vor:

1. Erweitern Sie den Ordner der Programmbausteine in der STEP 7 Projektnavigation.
2. Klicken Sie mit der rechten Maustaste auf den Datenbaustein und wählen Sie im Kontextmenü den Befehl "Eigenschaften".
3. Wählen Sie in den Attributen die Option "Optimierter Bausteinzugriff".

Beachten Sie, dass bei neuen Datenbausteinen standardmäßig der optimierte Bausteinzugriff eingestellt wird. Wenn Sie "Optimierter Bausteinzugriff" deaktivieren, wird der Standardzugriff für den Baustein festgelegt.

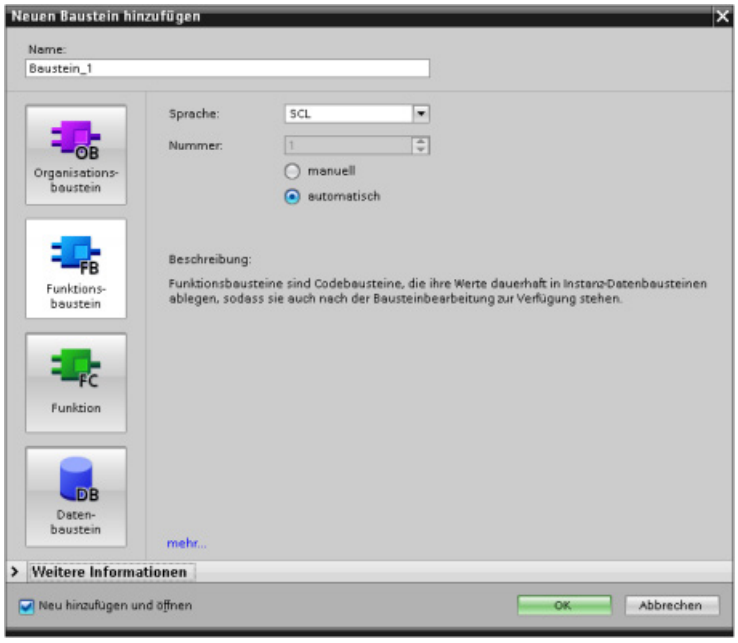
Hinweis

Bausteinzugriffstyp für einen FB und seinen Instanz-DB

Wenn für Ihren FB die Option "Optimierter Bausteinzugriff" eingestellt ist, muss für den Instanz-DB dieses FBs auch die Option "Optimierter Bausteinzugriff" eingestellt sein. Ebenso gilt, dass, wenn für den FB die Option "Optimierter Bausteinzugriff" nicht eingestellt ist und es sich somit um einen FB mit Standardzugriff handelt, auch für den Instanz-DB der Standardzugriff und nicht die Option "Optimierter Bausteinzugriff" eingestellt sein muss.

Wenn die Zugriffstypen nicht kompatibel sind, können während der Ausführung des FBs über ein HMI-Gerät vorgenommene Änderungen an den Eingangs- und Ausgangsparameterwerten des FBs verloren gehen.

7.3.5 Anlegen wiederverwendbarer Codebausteine



OBs, FBs, FCs und globale DBs können Sie in der Projektnavigation unter "Programmbausteine" im Dialog "Neuen Baustein hinzufügen" anlegen.

Beim Anlegen eines Codebausteins müssen Sie die Programmiersprache für den Baustein auswählen. Für einen DB wird keine Sprache ausgewählt, weil er nur Daten speichert.

Bei Auswahl des Kontrollkästchens "Neu hinzufügen und öffnen" (Standardeinstellung) wird der Codebaustein in der Projektansicht geöffnet.

Objekte, die Sie wiederverwenden möchten, können Sie in Bibliotheken speichern. Jedes Projekt besitzt eine Projektbibliothek, die mit dem Projekt verbunden ist. Zusätzlich zur Projektbibliothek können Sie beliebig viele globale Bibliotheken erstellen, die in mehreren Projekten verwendet werden können. Da die Bibliotheken untereinander kompatibel sind, können Bibliothekselemente von einer Bibliothek in die andere kopiert oder verschoben werden.

Bibliotheken dienen beispielsweise dazu, Bausteinvorlagen zu erstellen, die Sie zunächst in die Projektbibliothek kopieren und dann dort weiterentwickeln. Abschließend kopieren Sie die Bausteine von der Projektbibliothek in eine globale Bibliothek. Die globale Bibliothek stellen Sie anderen Kollegen im Projekt zur Verfügung. Ihre Kollegen können die Bausteine verwenden und bei Bedarf weiter an ihre jeweiligen Anforderungen anpassen.

Einzelheiten zu Bibliotheksoperationen finden Sie in den Themen zu Bibliotheken in der STEP 7 Online-Hilfe.

7.3.6 Übergabe von Parametern an Bausteine

Funktionsbausteine (FBs) und Funktionen (FCs) haben drei verschiedene Schnittstellentypen:

- IN
- IN/OUT
- OUT

FBs und FCs empfangen Parameter über die Schnittstellentypen IN und IN/OUT. Die Bausteine verarbeiten die Parameter und geben über die Schnittstellentypen IN/OUT und OUT Werte an den Aufrufer zurück.

Das Anwenderprogramm überträgt Parameter mittels einer von zwei Methoden:

Call-by-Value

Wenn das Anwenderprogramm einen Parameter als "Call-by-Value" an eine Funktion übergibt, kopiert das Anwenderprogramm den tatsächlichen Parameterwert in den Eingangsparameter des Bausteins für den Schnittstellentyp IN. Dieser Vorgang erfordert zusätzlichen Speicher für den kopierten Wert.

Wenn das Anwenderprogramm den Baustein aufruft, kopiert es die Werte.

Call-by-Reference

Wenn das Anwenderprogramm einen Parameter als "Call-by-Reference" an eine Funktion übergibt, verweist das Anwenderprogramm auf die Adresse des tatsächlichen Parameters für den Schnittstellentyp IN/OUT und kopiert den Wert nicht. Dieser Vorgang erfordert keinen zusätzlichen Speicher.

Wenn das Anwenderprogramm den Baustein aufruft, verweist es auf die Adresse der tatsächlichen Parameter.

Hinweis

Der Schnittstellentyp IN/OUT wird im Allgemeinen für strukturierte Variablen (zum Beispiel ARRAY, STRUCT und STRING) verwendet, um eine unnötige Vergrößerung des erforderlichen Datenspeichers zu vermeiden.

Bausteinoptimierung und Parameterübergabe

Bei einfachen Datentypen (zum Beispiel INT, DINT und REAL) übergibt das Anwenderprogramm FC-Parameter als "Call-by-Value". Komplexe Datentypen (zum Beispiel STRUCT, ARRAY und STRING) werden als "Call-by-Reference" übergeben.

Das Anwenderprogramm übergibt FB-Parameter normalerweise in dem zum FB zugehörigen Instanzdatenbaustein (DB):

- Das Anwenderprogramm übergibt einfache Datentypen (zum Beispiel INT, DINT und REAL) als "Call-by-Value", indem die Parameter in den und aus dem Instanz-DB kopiert werden.
- Das Anwenderprogramm kopiert komplexe Datentypen (zum Beispiel STRUCT, ARRAY und STRING) für die Parametertypen IN und OUT in den und aus dem Instanz-DB.
- Das Anwenderprogramm übergibt komplexe Datentypen als "Call-by-Reference" für den Schnittstellentyp IN/OUT.

DBs können mit den Optionen "Optimiert" oder "Standard" (nicht optimiert) angelegt werden. Die optimierten Datenbausteine sind kompakter als die nicht optimierten Datenbausteine. Auch unterscheidet sich die Reihenfolge der Datenelemente innerhalb der DBs bei optimierten und nicht optimierten DBs. Im Abschnitt "Optimierte Bausteine" in der S7-Programmierrichtlinie für S7-1200/1500, STEP 7 (TIA Portal), 03/2014 (http://support.automation.siemens.com/WW/view/en/81318674) finden Sie eine Erläuterung zu optimierten Bausteinen.

Sie erstellen FBs und FCs, um entweder optimierte oder nicht optimierte Daten zu verarbeiten. Sie können das Kontrollkästchen "Optimierter Bausteinzugriff" als eines der Attribute für den Baustein aktivieren. Das Anwenderprogramm optimiert Programmbausteine standardmäßig und die Programmbausteine erwarten die an den Baustein übergebenen Daten im optimierten Format.

Wenn das Anwenderprogramm einen komplexen Parameter (zum Beispiel STRUCT) an eine Funktion übergibt, prüft das System die Optimierungseinstellung des Datenbausteins mit der Struktur und die Optimierungseinstellung des Programmbausteins. Wenn Sie sowohl den Datenbaustein als auch die Funktion optimieren, dann übergibt das Anwenderprogramm die Struktur als "Call-by-Reference". Gleiches gilt, wenn Sie sowohl den Datenbaustein als auch die Funktion nicht optimieren.

Wenn jedoch Funktion und Datenbaustein unterschiedliche Optimierungseinstellungen haben (d. h., ein Baustein ist optimiert und der andere nicht), dann muss die Struktur in das von der Funktion erwartete Format umgewandelt werden. Beispiel: Wenn der Datenbaustein nicht optimiert und die Funktion optimiert ist, dann muss eine Struktur im Datenbaustein in ein optimiertes Format umgewandelt werden, damit die Funktion die Struktur verarbeiten kann. Das System führt diese Umwandlung durch, indem es die Struktur "kopiert" und sie in das von der Funktion erwartete optimierte Format umwandelt.

Zusammenfassend lässt sich sagen, dass, wenn das Anwenderprogramm einen komplexen Datentyp (zum Beispiel STRUCT) als IN/OUT-Parameter an eine Funktion übergibt, die Funktion erwartet, dass das Anwenderprogramm die Struktur als "Call-by-Reference" übergibt:

- Wenn Sie sowohl den Datenbaustein mit der Struktur als auch die Funktion optimieren bzw. nicht optimieren, dann übergibt das Anwenderprogramm die Daten als "Call-by-Reference".
- Wenn Sie für den Datenbaustein und die Funktion nicht die gleichen Optimierungseinstellungen konfigurieren (ein Baustein ist optimiert, der andere ist nicht optimiert), dann muss das System vor der Übergabe an die Funktion eine Kopie der Struktur anlegen. Weil das System diese Kopie der Struktur anlegen muss, wird "Call-by-Reference" dadurch praktisch in "Call-by-Value" umgewandelt.

Auswirkung der Optimierungseinstellungen auf Anwenderprogramme

Das Kopieren der Parameter kann ein Problem in einem Anwenderprogramm verursachen, wenn durch ein HMI-Gerät oder einen Alarm-OB Elemente in der Struktur verändert werden. Beispiel: Es gibt einen IN/OUT-Parameter einer Funktion (der normalerweise als "Call-by-Reference" übergeben wird), doch die Optimierungseinstellungen des Datenbausteins und der Funktion unterscheiden sich:

1. Wenn das Anwenderprogramm die Funktion aufrufen will, muss das System eine "Kopie" der Struktur anlegen, um das Format der Daten an die Funktion anzupassen.
2. Das Anwenderprogramm ruft die Funktion mit einem Verweis auf die "Kopie" der Struktur auf.
3. Ein Alarm-OB tritt während der Ausführung der Funktion auf und der Alarm-OB ändert einen Wert in der ursprünglichen Struktur.
4. Die Funktion wird abgearbeitet und, weil die Struktur ein IN/OUT-Parameter ist, kopiert das System die Werte zurück in die ursprüngliche Struktur im ursprünglichen Format.

Weil eine Kopie der Struktur zum Ändern des Formats angelegt wurde, gehen die vom Alarm-OB geschriebenen Daten verloren. Das Gleiche kann beim Schreiben eines Werts mit einem HMI-Gerät passieren. Das HMI-Gerät kann das Anwenderprogramm unterbrechen und in derselben Weise wie ein Alarm-OB einen Wert schreiben.

Es gibt mehrere Möglichkeiten, dieses Problem zu vermeiden:

- Die beste Lösung ist, bei Verwendung komplexer Datentypen (zum Beispiel STRUCT) die Optimierungseinstellungen des Programmbausteins und des Datenbausteins aneinander anzupassen. Dadurch wird gewährleistet, dass das Anwenderprogramm die Parameter stets als "Call-by-Reference" übergibt.
- Eine andere Lösung ist die, dass ein Alarm-OB bzw. ein HMI-Gerät ein Element in der Struktur nicht direkt ändert. Der OB oder das HMI-Gerät kann eine andere Variable ändern und dann können Sie diese Variable an einem bestimmten Punkt im Anwenderprogramm in die Struktur kopieren.

7.4 Datenkonsistenz

Die CPU erhält die Datenkonsistenz für alle elementaren Datentypen (z.B. Word oder DWord) und alle systemdefinierten Strukturen (z.B. IEC_TIMERS oder DTL). Der Lese- bzw. Schreibvorgang des Werts kann nicht unterbrochen werden. (Die CPU z.B. schützt den Zugriff auf einen Wert vom Datentyp DWord, bis die vier Bytes des DWord gelesen oder geschrieben wurden.) Um sicherzustellen, dass die Programmzyklus-OBs und die Alarm-OBs nicht gleichzeitig in dieselbe Adresse im Speicher schreiben können, führt die CPU einen Alarm-OB erst aus, wenn der Lese- oder Schreibvorgang im Programmzyklus-OB beendet ist.

Wenn in Ihrem Anwenderprogramm mehrere Werte im Speicher von einem Programmzyklus-OB und einem Alarm-OB gemeinsam genutzt werden, muss Ihr Anwenderprogramm auch sicherstellen, dass diese Werte konsistent geändert oder gelesen werden. Mit den Anweisungen DIS_AIRT (Alarmbearbeitung deaktivieren) und EN_AIRT (Alarmbearbeitung aktivieren) können Sie in Ihrem Programmzyklus-OB den Zugriff auf die gemeinsam genutzten Werte schützen.

- Fügen Sie eine Anweisung DIS_AIRT in den Codebaustein ein, um sicherzustellen, dass während eines Lese- oder Schreibvorgangs kein Alarm-OB ausgeführt werden kann.
- Fügen Sie die Anweisungen ein, die die Werte, die von einem Alarm-OB geändert werden könnten, lesen oder schreiben.
- Fügen Sie am Ende der Sequenz eine Anweisung EN_AIRT ein, um die Anweisung DIS_AIRT zu stornieren und die Ausführung eines Alarm-OBs zu gestatten.

Auch eine Kommunikationsanforderung eines HMI-Geräts oder einer anderen CPU kann die Ausführung des Programmzyklus-OBs unterbrechen. Die Kommunikationsanforderungen können auch zu Problemen mit der Datenkonsistenz führen. Die CPU stellt sicher, dass die elementaren Datentypen von den Anweisungen im Anwenderprogramm stets konsistent gelesen und geschrieben werden. Weil das Anwenderprogramm regelmäßig von Kommunikationsanforderungen unterbrochen wird, kann nicht gewährleistet werden, dass mehrere Werte der CPU alle gleichzeitig von der HMI aktualisiert werden. Die in einem HMI-Bild angezeigten Werte beispielsweise können aus unterschiedlichen Zyklen der CPU stammen.

Die PtP-Anweisungen (Punkt-zu-Punkt), PROFINET-Anweisungen (wie TSEND_C und TRCV_C), die PROFINET-Anweisungen für die dezentrale Peripherie (Seite 356) und die PROFIBUS-Anweisungen für die dezentrale Peripherie (Seite 356) übertragen Datenpuffer, die unterbrochen werden können. Stellen Sie die Datenkonsistenz für die Datenpuffer dadurch sicher, dass Sie Lese- und Schreibvorgänge in den Puffern sowohl im Programmzyklus-OB als auch im Alarm-OB verhindern. Sollte es erforderlich sein, die Pufferwerte für diese Anweisungen in einem Alarm-OB zu ändern, verzögern Sie mit der Anweisung DIS_AIRT die Unterbrechung (durch einen Alarm-OB oder eine Kommunikationsunterbrechung von einem HMI-Gerät oder einer anderen CPU), bis eine Anweisung EN_AIRT ausgeführt wird.

Hinweis

Die Verwendung der Anweisung DIS_AIRT verzögert die Verarbeitung von Alarm-OBs, bis die Anweisung EN_AIRT ausgeführt wird, und wirkt sich auf die Alarmlatenz (Zeitraum vom Auftreten eines Ereignisses bis zur Ausführung des Alarm-OBs) Ihres Anwenderprogramms aus.

7.5 Programmiersprache

STEP 7 bietet die folgenden Standardprogrammiersprachen für die S7-1200:

- KOP (Kontaktplan) ist eine grafische Programmiersprache. Die Darstellung beruht auf Schaltplänen (Seite 193).
- FUP (Funktionsplan) ist eine Programmiersprache, die auf den grafischen Logiksymbolen der Booleschen Algebra (Seite 194) basiert.
- SCL (Structured Control Language) ist eine textbasierte, höhere Programmiersprache (Seite 195).

Wenn Sie einen Codebaustein anlegen, müssen Sie die Programmiersprache für den Baustein auswählen.

Ihr Anwenderprogramm kann mit Codebausteinen arbeiten, die in einer dieser Programmiersprachen angelegt wurden.

7.5.1 Kontaktplan (KOP)

Die Elemente eines Schaltplans, wie Öffner- und Schließerkontakte, und Spulen werden zu Netzwerken verknüpft.

Um Verknüpfungen für komplexe Operationen anzulegen, können Sie Verzweigungen für parallele Kreise einfügen. Parallele Verzweigungen sind nach unten geöffnet oder direkt mit der Stromschiene verbunden. Sie beenden die Verzweigungen nach unten.

KOP bietet Box-Anweisungen für eine Vielzahl von Funktionen wie Arithmetik, Zeiten, Zähler und Übertragen.

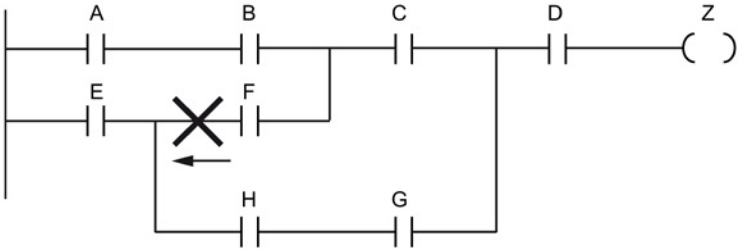
STEP 7 begrenzt die maximale Anzahl von Anweisungen (Zeilen und Spalten) in einem KOP-Netzwerk nicht.

Hinweis

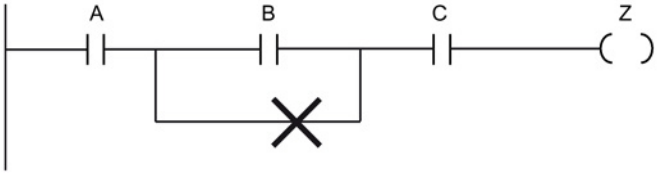
Jedes KOP-Netzwerk muss mit einer Spule oder einer Box abgeschlossen werden.

Beim Anlegen eines KOP-Netzwerks sind die folgenden Regeln zu beachten:

- Sie können keine Verzweigung anlegen, die zu einem Signalfluss in die Gegenrichtung führen könnte.

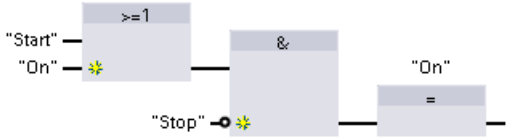


- Sie können keine Verzweigung anlegen, die einen Kurzschluss verursachen würde.



7.5.2 Funktionsplan (FUP)

Ebenso wie KOP ist auch FUP eine grafische Programmiersprache. Die Darstellung der Verknüpfungslogik beruht auf den grafischen Symbolen, die in der booleschen Algebra üblich sind.



Um Verknüpfungen für komplexe Operationen anzulegen, fügen Sie parallele Verzweigungen zwischen den Boxen ein.

Arithmetische Funktionen und andere komplexe Funktionen können direkt in Verbindung mit den Logikboxen dargestellt werden.

STEP 7 begrenzt die maximale Anzahl von Anweisungen (Zeilen und Spalten) in einem FUP-Netzwerk nicht.

7.5.3 SCL

Structured Control Language (SCL) ist eine höhere, PASCAL-basierte Programmiersprache für die SIMATIC S7-CPUs. SCL unterstützt die Bausteinstruktur von STEP 7 (Seite 181). Ihr Projekt kann Programmbausteine in jeder der drei Programmiersprachen: SCL, KOP und FUP enthalten.

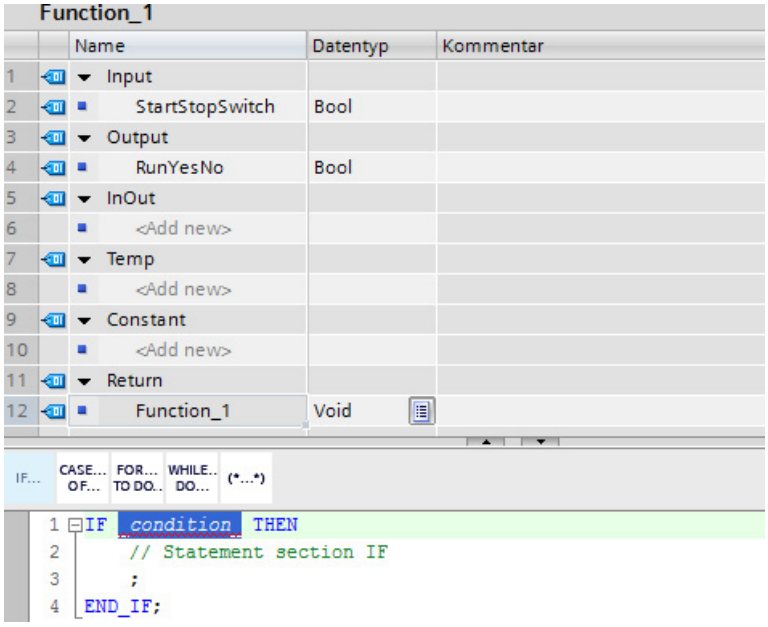
In SCL-Anweisungen werden die Standardoperatoren der Programmierung verwendet, z. B. für Zuweisung (:=), mathematische Funktionen (+ für Addition, - für Subtraktion, * für Multiplikation und / für Division). SCL verwendet auch standardmäßige PASCAL-Operationen für die Programmsteuerung, z. B. IF-THEN-ELSE, CASE, REPEAT-UNTIL, GOTO und RETURN. Für syntaktische Elemente der Programmiersprache SCL können Sie alle PASCAL-Referenzen verwenden. Viele der anderen Anweisungen für SCL wie Zeiten und Zähler entsprechen den Anweisungen in KOP und FUP. Weitere Informationen zu spezifischen Anweisungen finden Sie unter den jeweiligen Anweisungen in den Kapiteln Grundlegende Anweisungen (Seite 217) und Erweiterte Anweisungen (Seite 323).

7.5.3.1 SCL-Programmiereditor

Sie können für alle Bausteintypen (OB, FB oder FC) beim Erstellen des Bausteins angeben, dass er die Programmiersprache SCL verwenden soll. STEP 7 verfügt über einen SCL-Programmiereditor, der die folgenden Elemente enthält:

- Schnittstellenabschnitt zum Definieren der Parameter des Codebausteins
- Codeabschnitt für den Programmcode
- Anweisungsverzeichnis mit den SCL-Anweisungen, die von der CPU unterstützt werden

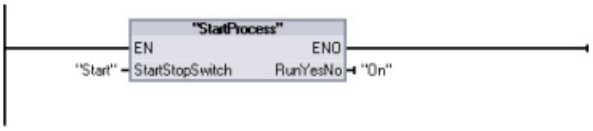
Sie geben den SCL-Code für Ihre Anweisung direkt in den Codeabschnitt ein. Der Editor enthält Schaltflächen für gängige Codekonstruktionen und Kommentare. Um komplexere Anweisungen anzulegen, ziehen Sie die SCL-Anweisungen einfach mit der Maus aus dem Anweisungsverzeichnis in Ihr Programm. Ferner können Sie in jedem Texteditor ein SCL-Programm anlegen und die Datei dann in STEP 7 importieren.



Im Schnittstellenabschnitt des SCL-Codebausteins können Sie die folgenden Arten von Parametern deklarieren:

- Input, Output, InOut und Ret_Val: Diese Parameter definieren die Eingangs- und Ausgangsvariablen sowie den Rückgabewert für den Codebaustein. Der Variablenname, den Sie hier eingeben, wird lokal während der Ausführung des Codebausteins verwendet. Üblicherweise wird der globale Variablenname nicht in der Variablentabelle verwendet.
- Static (nur FBs; die obige Abbildung zeigt einen FC): Der Codebaustein nutzt statische Variablen zum Speichern von statischen Zwischenergebnissen im Instanzdatenbaustein. Der Baustein speichert statische Daten, bis sie überschrieben werden, d. h. zum Teil über mehrere Zyklen. Zusammen mit den statischen Daten werden auch die Namen der Bausteine gespeichert, die in diesem Codebaustein als Multiinstanz-DBs aufgerufen werden.
- Temp: Bei diesen Parametern handelt es sich um temporäre Variablen, die während der Ausführung des Codebausteins verwendet werden.
- Constant: Hierbei handelt es sich um benannte konstante Werte für Ihren Codebaustein.

Wenn Sie den SCL-Codebaustein aus einem anderen Codebaustein aufrufen, treten die Parameter des SCL-Codebausteins als Eingänge oder Ausgänge auf.



In diesem Beispiel entsprechen die Variablen für "Start" und "On" (aus der Variablentabelle des Systems) den Variablen "StartStopSwitch" und "RunYesNo" in der Deklarationstabelle des SCL-Programms.

7.5.3.2 SCL-Ausdrücke und -Operationen

SCL-Ausdruck konstruieren

Ein SCL-Ausdruck ist eine Formel zum Berechnen eines Werts. Der Ausdruck besteht aus Operanden und Operatoren (wie *, /, + oder -). Bei den Operanden kann es sich um Variablen, Konstanten oder Ausdrücke handeln.

Die Auswertung des Ausdrucks erfolgt in einer bestimmten Reihenfolge, die von den folgenden Faktoren festgelegt wird:

- Jeder Operator hat eine vordefinierte Priorität, wobei die Operation mit der höchsten Priorität zuerst ausgeführt wird.
- Bei Operatoren mit gleicher Priorität werden die Operatoren von links nach rechts verarbeitet.
- Mit Hilfe von Klammern kennzeichnen Sie eine Reihe von Operatoren, die gemeinsam ausgewertet werden sollen.

Mit dem Ergebnis eines Ausdrucks kann ein Wert einer von Ihrem Programm verwendeten Variablen zugewiesen werden, das Ergebnis kann als Bedingung für eine Steuerungsanweisung oder als Parameter für eine andere SCL-Anweisung oder zum Aufrufen eines Codebausteins verwendet werden.

Tabelle 7- 2 Operatoren in SCL

Typ	Operation	Operator	Priorität
Klammern	(*Ausdruck*)	(,)	1
Arithmetik	Potenz	**	2
	Vorzeichen (unäres Plus)	+	3
	Vorzeichen (unäres Minus)	-	3
	Multiplikation	*	4
	Division	/	4
	Modulo-Funktion	MOD	4
	Addition	+	5
	Subtraktion	-	5
Vergleich	Kleiner als	<	6
	Kleiner oder gleich	<=	6
	Größer als	>	6
	Größer oder gleich	>=	6
	Gleich	=	7
	Ungleich	<>	7
Bitverknüpfung	Negation (unär)	NOT	3
	Logische UND-Verknüpfung	AND oder &	8
	Logische Exklusiv-ODER-Verknüpfung	XOR	9
	Logische ODER-Verknüpfung	OR	10
Zuweisung	Zuweisung	:=	11

Als höhere Programmiersprache nutzt SCL Standardanweisungen für grundlegende Aufgaben:

- Zuweisungsanweisung: :=
- Mathematische Funktionen: +, -, * und /
- Adressierung von globalen Variablen: "<Variablenname>" (Variablenname oder Datenbausteinname in doppelten Anführungszeichen)
- Adressierung lokaler Variablen: #<Variablenname> (Variablenname mit vorangestelltem Symbol "#")

Die folgenden Beispiele zeigen verschiedene Ausdrücke für verschiedene Einsatzzwecke.

```
"C" := #A+#B;
```
Weist einer Variablen die Summe zweier lokaler Variablen zu

```
"Data_block_1".Tag := #A;
```
Zuweisung zu einer Datenbausteinvariablen

```
IF #A > #B THEN "C" := #A;
```
Bedingung für die IF-THEN-Anweisung

```
"C" := SQRT (SQR (#A) + SQR (#B));
```
Parameter für die SQRT-Anweisung

Arithmetische Operatoren können verschiedene numerische Datentypen verarbeiten. Der Datentyp des Ergebnisses wird vom Datentyp des höchstwertigen Operanden festgelegt. Beispiel: Eine Multiplikationsoperation mit einem Operanden vom Typ INT und einem Operanden vom Typ REAL ergibt einen Wert vom Typ REAL als Ergebnis.

Steuerungsanweisungen

Eine Steuerungsanweisung ist eine besondere Art von SCL-Ausdruck, der die folgenden Aufgaben durchführt:

- Programmverzweigung
- Wiederholung von Abschnitten des SCL-Programmcodes
- Sprung zu anderen Teilen des SCL-Programms
- Bedingte Ausführung

Die SCL-Steuerungsanweisungen umfassen IF-THEN, CASE-OF, FOR-TO-DO, WHILE-DO, REPEAT-UNTIL, CONTINUE, GOTO und RETURN.

Eine einzelne Anweisung belegt üblicherweise eine Codezeile. Sie können mehrere Anweisungen in einer Zeile eingeben oder Sie können die Anweisung auf mehrere Codezeilen verteilen, damit der Code besser lesbar ist. Trennzeichen (wie Tabulatoren, Zeilenumbrüche und zusätzliche Leerzeichen) werden während der Syntaxprüfung ignoriert. Die END-Anweisung beendet die Steuerungsanweisung.

Die folgenden Beispiele zeigen eine FOR-TO-DO-Steuerungsanweisung. (Beide Formen der Codierung sind syntaktisch gültig.)

```
FOR x := 0 TO max DO sum := sum + value(x); END_FOR;
FOR x := 0 TO max DO
     sum := sum + value(x);
END_FOR;
```

Eine Steuerungsanweisung kann auch eine Sprungmarke umfassen. Eine Sprungmarke wird gefolgt von einem Doppelpunkt an den Anfang der Anweisung gesetzt:

```
Bezeichnung: <Anweisung>;
```

Die Online-Hilfe von STEP 7 enthält vollständige Referenzinformationen für die Programmiersprache SCL.

Bedingungen

Eine Bedingung ist ein Vergleichsausdruck oder ein logischer Ausdruck, dessen Ergebnis vom Typ BOOL ist (Wert WAHR oder FALSCH). Die folgenden Beispiele zeigen Bedingungen verschiedener Arten.

`#Temperatur > 50`	Relationaler Ausdruck
`#Zähler <= 100`	
`#CHAR1 < 'S'`	
`(#Alpha <> 12) AND NOT #Beta`	Vergleichsausdruck und logischer Ausdruck
`5 + #Alpha`	Arithmetischer Ausdruck

Eine Bedingung kann arithmetische Ausdrücke verwenden:

- Die Bedingung des Ausdrucks ist WAHR, wenn das Ergebnis ein beliebiger Wert außer Null ist.
- Die Bedingung des Ausdrucks ist FALSCH; wenn das Ergebnis null ist.

Andere Codebausteine aus Ihrem SCL-Programm aufrufen

Um einen anderen Codebaustein in Ihrem Anwenderprogramm aufzurufen, geben Sie einfach den Namen (oder die absolute Adresse) von FB oder FC mit den Parametern ein. Für einen FB müssen Sie den Instanz-DB angeben, der mit dem FB aufgerufen werden soll.

`<DB-Name> (Parameterliste)`	Aufruf als eine Instanz
`<#Instanzname> (Parameterliste)`	Aufruf als Multiinstanz
`"MyDB"(MyInput:=10, MyInOut:="Tag1");`	
`<FC-Name> (Parameterliste)`	Standardaufruf
`<Operand>:=<FC-Name> (Parameterliste)`	Aufruf in einem Ausdruck
`"MyFC"(MyInput:=10, MyInOut:="Tag1");`	

Sie können Bausteine auch mit der Maus aus der Projektnavigation in den SCL-Programmiereditor ziehen und die Parametrierung vervollständigen.

Bausteinkommentare in SCL-Code hinzufügen

Sie können in Ihrem SCL-Code einen Bausteinkommentar hinzufügen, indem Sie Kommentartext zwischen (* und *) einfügen. Sie können zwischen (* und *) eine beliebige Anzahl Kommentarzeilen einfügen. Ihr SCL-Programmbaustein kann viele Bausteinkommentare enthalten. Zur leichteren Programmierung enthält der SCL-Editor neben den üblichen Steuerungsanweisungen eine Schaltfläche für Bausteinkommentare:

Adressierung

Wie bei KOP und FUP können Sie in SCL entweder Variablen (symbolische Adressierung) oder absolute Adressen in Ihrem Anwenderprogramm verwenden. In SCL können Sie eine Variable auch als Array-Index verwenden.

Absolute Adressierung

`%I0.0` `%MB100`	Der absoluten Adresse ist das Symbol "%" voranzustellen. Fehlt das "%", erzeugt STEP 7 beim Übersetzen einen undefinierten Variablenfehler.

Symbolische Adressierung

`"PLC_Tag_1"`	Variable in PLC-Variablentabelle
`"Data_block_1".Tag_1`	Variable in einem Datenbaustein
`"Data_block_1".MyArray[#i]`	Array-Element in einem Datenbaustein-Array

7.5.3.3 Indexierte Adressierung mit den Anweisungen PEEK und POKE

SCL bietet die Anweisungen PEEK und POKE, mit denen Sie aus Datenblöcken, E/A oder dem Speicher lesen oder in diese schreiben können. Sie geben für die Operation Parameter mit einem spezifischen Byte- oder Bit-Versatz an.

Hinweis

Um die Anweisungen PEEK und POKE mit Datenbausteinen zu verwenden, müssen Sie Standarddatenbausteine (keine optimierten Datenbausteine) verwenden. Beachten Sie zudem, dass die Anweisungen PEEK und POKE lediglich Daten übertragen. Die Datentypen an den Adressen werden nicht beachtet.

`PEEK(area:=_in_,`
`dbNumber:=_in_,`
`byteOffset:=_in_);`
	Liest das von byteOffset angegebene Byte aus dem angegebenen Datenbaustein, aus den E/A oder dem Speicherbereich.
	Beispiel für den Verweis auf einen Datenbaustein:
	`%MB100 := PEEK(area:=16#84, dbNumber:=1, byteOffset:=#i);`
	Beispiel für den Verweis auf Eingang EB3:
	`%MB100 := PEEK(area:=16#81, dbNumber:=0, byteOffset:=#i); // when #i = 3`

`PEEK_WORD(area:=_in_,`
`dbNumber:=_in_,`
`byteOffset:=_in_);`
	Liest das von byteOffset angegebene Wort aus dem angegebenen Datenbaustein, aus den E/A oder dem Speicherbereich.
	Beispiel:
	`%MW200 := PEEK_WORD(area:=16#84, dbNumber:=1, byteOffset:=#i);`

```
PEEK_DWORD(area:=_in_,
           dbNumber:=_in_,
           byteOffset:=_in_);
```

Liest das von byteOffset angegebene Doppelwort aus dem angegebenen Datenbaustein, aus den E/A oder dem Speicherbereich.

Beispiel:

```
%MD300 := PEEK_DWORD(area:=16#84,
dbNumber:=1, byteOffset:=#i);
```

```
PEEK_BOOL(area:=_in_,
          dbNumber:=_in_,
          byteOffset:=_in_,
          bitOffset:=_in_);
```

Liest einen von bitOffset und byteOffset angegebenen Booleschen Wert aus dem angegebenen Datenbaustein, aus den E/A oder dem Speicherbereich.

Beispiel:

```
%MB100.0 := PEEK_BOOL(area:=16#84,
dbNumber:=1, byteOffset:=#ii,
bitOffset:=#j);
```

```
POKE(area:=_in_,
     dbNumber:=_in_,
     byteOffset:=_in_,
     value:=_in_);
```

Schreibt den Wert (Byte, Wort oder Doppelwort) in den angegebenen byteOffset des angegebenen Datenbausteins, der E/A oder des Speicherbereichs.

Beispiel für den Verweis auf einen Datenbaustein:

```
POKE(area:=16#84, dbNumber:=2,
byteOffset:=3, value:="Tag_1");
```

Beispiel für den Verweis auf Ausgang AB3:

```
POKE(area:=16#82, dbNumber:=0,
byteOffset:=3, value:="Tag_1");
```

```
POKE_BOOL(area:=_in_,
          dbNumber:=_in_,
          byteOffset:=_in_,
          bitOffset:=_in_,
          value:=_in_);
```

Schreibt den Booleschen Wert in den angegebenen bitOffset und byteOffset des angegebenen Datenbausteins, der E/A oder des Speicherbereichs.

Beispiel:

```
POKE_BOOL(area:=16#84, dbNumber:=2,
byteOffset:=3, bitOffset:=5, val-
ue:=0);
```

```
POKE_BLK(area_src:=_in_,
         dbNumber_src:=_in_,
         byteOffset_src:=_in_,
         area_dest:=_in_,
         dbNumber_dest:=_in_,
         byteOffset_dest:=_in_,
         count:=_in_);
```

Schreibt die unter "count" angegebene Anzahl von Bytes beginnend mit dem angegebenen byteOffset des angegebenen Quelldatenbausteins, der Quell-E/A oder des Quellspeicherbereichs in den angegebenen byteOffset des angegebenen Zieldatenbausteins, der Ziel-E/A oder des Zielspeicherbereichs.

Beispiel:

```
POKE_BLK(area_src:=16#84, dbNum-
ber_src:=#src_db, byteOff-
set_src:=#src_byte, area_dest:=16#84,
dbNumber_dest:=#src_db, byteOff-
set_dest:=#src_byte, count:=10);
```

Bei den Anweisungen PEEK und POKE gelten die folgenden Werte für die Parameter "area", "area_src" und "area_dest". Für andere Bereiche als Datenbausteine muss der Parameter dbNumber 0 sein.

16#81	I
16#82	Q
16#83	M
16#84	DB

7.5.4 EN und ENO in KOP, FUP und SCL

"Signalfluss" (EN und ENO) für eine Anweisung ermitteln

Einige Anweisungen (z. B. mathematische Anweisungen und Übertragungsanweisungen) zeigen Parameter für EN und ENO an. Diese Parameter beziehen sich auf den Signalfluss in KOP oder FUP und legen fest, ob die Anweisung in diesem Zyklus ausgeführt wird. In SCL können Sie den Parameter ENO auch für einen Codebaustein angeben.

- EN (Freigabeeingang) ist ein Boolescher Eingang. An diesem Eingang muss Signalfluss (EN = 1) vorhanden sein, damit die Box ausgeführt werden kann. Wenn der Eingang EN einer KOP-Box direkt an die linke Stromschiene angeschlossen ist, wird die Anweisung immer ausgeführt.
- ENO (Freigabeausgang) ist ein Boolescher Ausgang. Liegt am Eingang EN einer Box ein Signalfluss an und die Box wird fehlerfrei ausgeführt, dann leitet der Ausgang ENO den Signalfluss (ENO = 1) zum nächsten Element weiter. Tritt während der Ausführung der Box ein Fehler auf, dann wird der Signalfluss an der Box-Anweisung, die den Fehler verursacht hat, beendet (ENO = 0).

Tabelle 7- 3 Operanden für EN und ENO

Programm-Editor	Eingän-ge/Ausgänge	Operanden	Datentyp
KOP	EN, ENO	Signalfluss	Bool
FUP	EN	E, E:P, A, M, DB, Temp, Signalfluss	Bool
	ENO	Signalfluss	Bool
SCL	EN[1]	WAHR, FALSCH	Bool
	ENO[2]	WAHR, FALSCH	Bool

[1] Die Verwendung von EN steht nur bei FBs zur Verfügung.

[2] Die Verwendung von ENO mit dem SCL-Codebaustein ist optional. Sie müssen die SCL-Übersetzung so einrichten, dass ENO bei Fertigstellung der Codebausteinbearbeitung gesetzt wird.

Setzen von ENO in SCL konfigurieren

Um die SCL-Übersetzung so einzurichten, dass ENO gesetzt wird, gehen Sie folgendermaßen vor:

1. Wählen Sie im Menü "Extras" den Befehl "Einstellungen".

2. Erweitern Sie die Eigenschaften "PLC-Programmierung" und wählen Sie "SCL (Structured Control Language)".

3. Wählen Sie die Option "ENO automatisch setzen".

Verwendung von ENO im Programmcode

Sie können ENO auch in Ihrem Programmcode verwenden, zum Beispiel durch Zuweisen von ENO zu einer PLC-Variablen oder durch Auswerten von ENO in einem lokalen Baustein.

Beispiele:

```
"MyFunction"
  ( IN1 := … ,
    IN2 := … ,
    OUT1 => #myOut,
    ENO => #statusFlag ); // PLC tag statusFlag holds the value of E
NO

"MyFunction"
  ( IN1 := …
    IN2 := … ,
    OUT1 => #myOut,
    ENO => ENO );  // block status flag of "MyFunction"
                   // is stored in the local block

IF ENO = TRUE THEN
    // execute code only if MyFunction returns true ENO
```

Auswirkung der Parameter Ret_Val oder Status auf ENO

Einige Anweisungen, z. B. die Kommunikationsanweisungen oder die Anweisungen für die Zeichenkettenkonvertierung, verfügen über einen Ausgangsparameter, der Informationen über die Verarbeitung der Anweisung enthält. Einige Anweisungen haben beispielsweise einen Parameter Ret_Val (Rückgabewert), üblicherweise vom Datentyp Int, der Statusinformationen im Bereich von -32768 bis +32767 enthält. Andere Anweisungen haben den Parameter Status, typischerweise vom Datentyp Word, der Statusinformationen im Hexadezimalbereich von 16#0000 bis 16#FFFF speichert. Der numerische Wert in einem der Parameter Ret_Val oder Status gibt den Zustand von ENO der jeweiligen Anweisung an.

- Ret_Val: Ein Wert zwischen 0 und 32767 setzt ENO typischerweise = 1 (bzw. WAHR). Ein Wert zwischen -32768 und -1 setzt ENO typischerweise = 0 (bzw. FALSCH). Um Ret_Val auszuwerten, ändern Sie die Darstellung in hexadezimal.
- Status: Ein Wert zwischen 16#0000 und 16#7FFF setzt ENO typischerweise = 1 (bzw. WAHR). Ein Wert zwischen 16#8000 und 16#FFFF setzt ENO typischerweise = 0 (bzw. FALSCH).

Anweisungen, die über mehrere Zyklen ausgeführt werden, haben häufig den Parameter Busy (Bool), um kenntlich zu machen, dass die Anweisung aktiv ist, die Ausführung jedoch noch nicht beendet ist. Solche Anweisungen haben auch häufig auch den Parameter Done (Bool) und den Parameter Error (Bool). Done zeigt an, dass die Anweisung fehlerfrei ausgeführt wurde, und Error weist darauf hin, dass die Anweisung mit Fehler beendet wurde.

- Wenn Busy = 1 (bzw. WAHR), ist ENO = 1 (bzw. WAHR).
- Wenn Done = 1 (bzw. WAHR), ist ENO = 1 (bzw. WAHR).
- Wenn Error = 1 (bzw. WAHR), ist ENO = 0 (bzw. FALSCH).

Siehe auch

Anweisungen OK (Gültigkeit prüfen) und NOT_OK (Ungültigkeit prüfen) (Seite 244)

7.6 Schutz

7.6.1 Zugriffsschutz für die CPU

Die CPU bietet vier Sicherheitsstufen, um den Zugang zu bestimmten Funktionen einzuschränken. Mit dem Einrichten der Schutzstufe und des Passworts für eine CPU schränken Sie die Funktionen und Speicherbereiche ein, die ohne Eingabe eines Passworts zugänglich sind.

Jede Schutzstufe lässt auch ohne Eingabe eines Passworts den uneingeschränkten Zugriff auf bestimmte Funktionen zu. Die Voreinstellung der CPU ist "ohne Einschränkung" und "ohne Passwortschutz". Um den Zugang zu einer CPU zu schützen, müssen Sie die Eigenschaften der CPU einrichten und das Passwort eingeben.

Wenn Sie ein Netzpasswort eingeben, dann wirkt sich dieses Passwort nicht auf den Passwortschutz der CPU aus. Der Passwortschutz gilt nicht für die Ausführung der Anweisungen des Anwenderprogramms einschließlich Kommunikationsfunktionen. Die Eingabe des richtigen Passworts ermöglicht den Zugriff auf alle Funktionen der jeweiligen Stufe.

Die Kommunikation zwischen CPUs (über die Kommunikationsfunktionen in den Codebausteinen) wird durch die Schutzstufe der CPU nicht eingeschränkt.

Tabelle 7- 4 Schutzstufen der CPU

Schutzstufe	Zugangsbeschränkungen
Vollzugriff (kein Schutz)	Uneingeschränkter Zugriff ohne Passwortschutz.
Lesezugriff	HMI-Zugriff und uneingeschränkte Kommunikation zwischen CPUs ohne Passwortschutz. Ein Passwort ist für Änderungen (Schreibzugriffe) in der CPU und für den Wechsel des Betriebszustands der CPU (RUN/STOP) erforderlich.
HMI-Zugriff	HMI-Zugriff und uneingeschränkte Kommunikation zwischen CPUs ohne Passwortschutz. Ein Passwort ist zum Lesen der Daten in der CPU, für Änderungen (Schreiben) in der CPU und für den Wechsel des Betriebszustands der CPU (RUN/STOP) erforderlich.
Kein Zugriff (kompletter Schutz)	Kein Zugriff ohne Passwortschutz. Ein Passwort ist für den HMI-Zugriff, zum Lesen der Daten in der CPU und für das Ändern (Schreiben) von Daten in der CPU erforderlich.

Bei Passwörtern wird Groß- und Kleinschreibung unterschieden. Um die Schutzstufe und die Passwörter zu konfigurieren, gehen Sie folgendermaßen vor:

1. Wählen Sie in der "Gerätekonfiguration" die CPU aus.
2. Wählen Sie im Inspektorfenster das Register "Eigenschaften".
3. Wählen Sie die Eigenschaft "Schutz", um die Schutzstufe auszuwählen und Passwörter einzugeben.

Wenn Sie diese Konfiguration in die CPU laden, hat der Anwender HMI-Zugriff und kann ohne Passwort auf die HMI-Funktionen zugreifen. Um Daten zu lesen, muss der Anwender das für den "Lesezugriff" konfigurierte Passwort oder das Passwort für "Vollzugriff (kein Schutz)" eingeben. Um Daten zu schreiben, muss der Anwender das für den "Vollzugriff (kein Schutz)" konfigurierte Passwort eingeben.

⚠ WARNUNG

Nicht berechtigter Zugriff auf eine geschützte CPU

Anwender mit vollständigen CPU-Zugriffsrechten können PLC-Variablen lesen und schreiben. Unabhängig von der Zugriffsstufe für die CPU können Webserver-Anwender über Rechte zum Lesen und Schreiben von PLC-Variablen verfügen. Nicht berechtigter Zugriff auf die CPU oder das Einstellen von ungültigen Werten für PLC-Variablen kann den Prozessbetrieb stören und zu tödlichen oder schweren Verletzungen und/oder zu Sachschaden führen.

Berechtigte Anwender sind in der Lage, Änderungen des Betriebszustands vorzunehmen, PLC-Daten zu schreiben und Firmware-Updates durchzuführen. Siemens empfiehlt, die folgenden Sicherheitsvorkehrungen einzuhalten:

- Schützen Sie die CPU-Zugriffsstufen und Webserver-Benutzer-IDs (Seite 821) durch starke Passwörter. Starke Passwörter sind mindestens zehn Zeichen lang, bestehen aus Buchstaben, Zahlen und Sonderzeichen, sind keine Wörter, die in einem Wörterbuch gefunden werden können, und sind keine Namen oder Kennungen, die sich aus persönlichen Daten ableiten lassen. Halten Sie das Passwort geheim und ändern Sie es häufig.
- Aktivieren Sie den Zugriff auf den Webserver nur über das HTTPS-Protokoll.
- Erweitern Sie die standardmäßigen Mindestrechte des Webserver-Benutzers "Jeder" nicht.
- Führen Sie eine Fehlerprüfung und eine Bereichsprüfung für die Variablen in Ihrer Programmlogik durch, weil die Benutzer von Webseiten für PLC-Variablen ungültige Werte festlegen können.

Verbindungsmechanismen

Auch für den Zugriff auf entfernte Verbindungspartner über die Anweisungen PUT/GET benötigt der Anwender die entsprechende Berechtigung.

Standardmäßig ist die Option "Zugriff über PUT/GET-Kommunikation erlauben" nicht aktiviert. In diesem Fall ist der Lese- und Schreibzugriff auf CPU-Daten nur mit Kommunikationsverbindungen möglich, bei denen sowohl die lokale CPU als auch der Kommunikationspartner konfiguriert oder programmiert werden müssen. Beispielsweise ist der Zugriff über die Anweisungen BSEND/BRCV möglich.

Verbindungen, bei denen die lokale CPU lediglich ein Server ist (was bedeutet, dass in der lokalen CPU keine Konfiguration/Programmierung der Kommunikation mit dem Kommunikationspartner vorhanden ist) sind deshalb während des Betriebs der CPU nicht möglich. Dies gilt z. B. für:

- PUT/GET-, FETCH/WRITE- oder FTP-Zugriff über Kommunikationsmodule
- PUT/GET-Zugriff von anderen S7-CPUs
- HMI-Zugriff über PUT/GET-Kommunikation

Wenn Sie den clientseitigen Zugriff auf CPU-Daten erlauben möchten und somit die Kommunikationsdienste der CPU nicht einschränken möchten, gehen Sie wie folgt vor:

1. Konfigurieren Sie für den Zugriffsschutz eine beliebige Schutzstufe außer "Kein Zugriff (kompletter Schutz)".
2. Aktivieren Sie das Kontrollkästchen "Zugriff über PUT/GET-Kommunikation erlauben".

Wenn Sie diese Konfiguration in die CPU laden, erlaubt die CPU die PUT/GET-Kommunikation von entfernten Partnern.

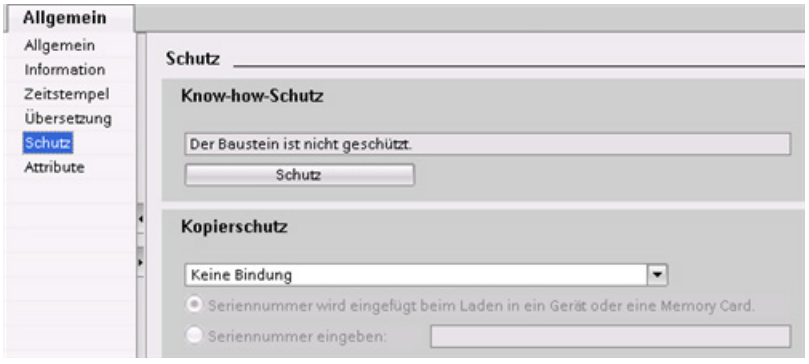
7.6.2 Knowhow-Schutz

Mit dem Knowhow-Schutz können Sie einen oder mehrere Codebausteine (OB, FB, FC oder DB) in Ihrem Programm vor unbefugtem Zugriff schützen. Sie können ein Passwort eingeben, um den Zugriff auf einen Codebaustein einzuschränken. Der Passwortschutz verhindert das unbefugte Lesen oder Ändern des Codebausteins. Ohne Passwort können nur die folgenden Informationen zum Codebaustein gelesen werden:

- Bausteintitel, Kommentar und Bausteineigenschaften
- Übertragungsparameter (IN, OUT, IN_OUT, Rückgabe)
- Aufrufstruktur des Programms
- Globale Variablen in den Querverweisen (ohne Information über die Verwendung), lokale Variablen sind jedoch verborgen

Wenn Sie einen Baustein für den Knowhow-Schutz konfigurieren, so ist der Code in diesem Baustein erst nach Eingabe des Passworts zugänglich.

Den Knowhow-Schutz eines Codebausteins konfigurieren Sie in der Taskcard "Eigenschaften" des jeweiligen Codebausteins. Nach dem Öffnen des Codebausteins wählen Sie unter "Eigenschaften" die Option "Schutz".



1. In den Eigenschaften des Codebausteins klicken Sie auf die Schaltfläche "Eigenschaften", um den Dialog "Knowhow-Schutz" anzuzeigen.
2. Klicken Sie auf die Schaltfläche "Definieren", um das Passwort einzugeben.

Nachdem Sie das Passwort eingegeben und bestätigt haben, klicken Sie auf "OK".

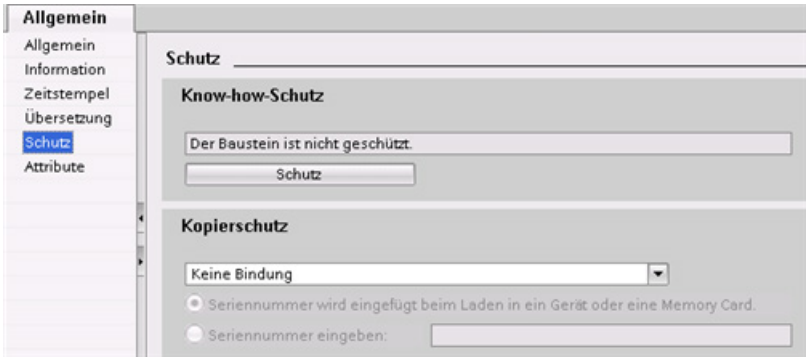
7.6.3 Kopierschutz

Eine weitere Sicherheitsfunktion ermöglicht Ihnen, Programmbausteine mit einer bestimmten Memory Card oder CPU zu verknüpfen. Diese Funktion ist vor allem zum Schutz geistigen Eigentums nützlich. Wird ein Programmbaustein mit einem bestimmten Gerät verknüpft, so ist die Verwendung des Programms oder Codebausteins nur in Verbindung mit einer bestimmten Memory Card oder CPU möglich. Mit dieser Funktion kann ein Programm oder ein Codebaustein elektronisch (zum Beispiel über Internet oder E-Mail) oder durch Versenden eines Speichermoduls verschickt werden. Kopierschutz ist verfügbar für OBs (Seite 182), FBs (Seite 184) und FCs (Seite 184). Die S7-1200 CPU unterstützt drei Arten des Bausteinschutzes:

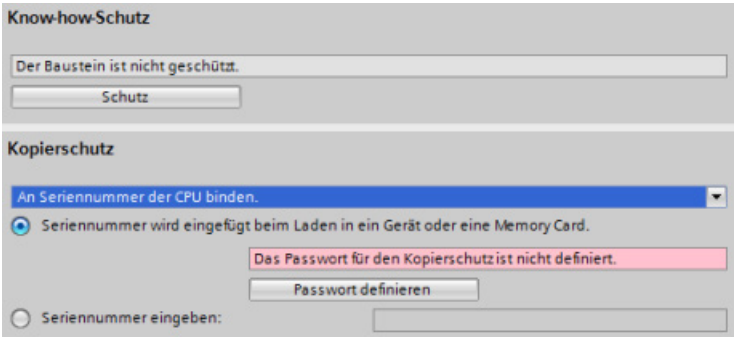
- Verknüpfung mit der Seriennummer einer CPU
- Verknüpfung mit der Seriennummer einer Memory Card
- Dynamische Verknüpfung mit einem obligatorischen Passwort

Um einen Baustein mit einer bestimmten CPU oder Memory Card zu verknüpfen, öffnen Sie die Taskcard "Eigenschaften" des jeweiligen Codebausteins.

1. Nach dem Öffnen des Codebausteins wählen Sie "Schutz".



2. Wählen Sie in der Klappliste "Kopierschutz" die Art des Kopierschutzes aus, die verwendet werden soll.



3. Für die Verknüpfung mit der Seriennummer einer CPU oder Memory Card können Sie wählen, ob die Seriennummer beim Laden eingefügt oder die Seriennummer von der Memory Card oder CPU eingegeben werden soll.

Hinweis

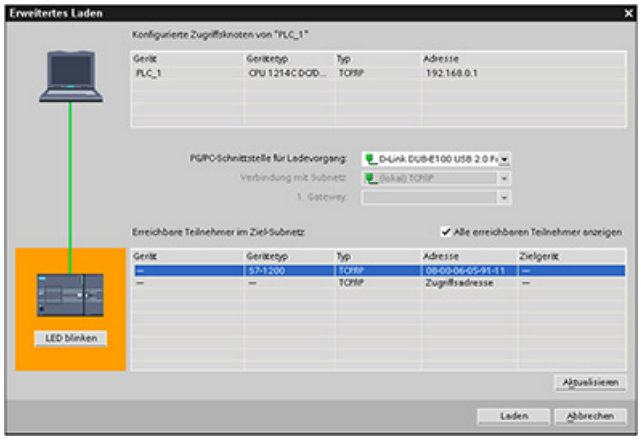
Bei der Seriennummer ist die Groß- und Kleinschreibung zu beachten.

Für die dynamische Verknüpfung mit einem obligatorischen Passwort ist das Passwort festzulegen, mit dem der Baustein geladen oder kopiert werden kann.

Wird dann ein Baustein mit dynamischer Verknüpfung geladen (Seite 211), muss dieses Passwort eingegeben werden, um das Laden zu ermöglichen. Zu beachten ist, dass die Passwörter für den Kopierschutz und für den Knowhow-Schutz (Seite 208) verschieden sind.

7.7 Laden der Programmelemente

Sie können die Elemente Ihres Projekts aus dem Programmiergerät in die CPU laden. Wenn Sie ein Projekt laden, speichert die CPU das Anwenderprogramm (OBs, FCs, FBs und DBs) im nullspannungsfesten Speicher.



Sie können Ihr Projekt aus dem Programmiergerät in die CPU laden, und zwar von einer der folgenden Stellen:

- Projektnavigation: Klicken Sie mit der rechten Maustaste auf das Programmelement und wählen Sie dann "Laden in CPU".
- Onlinemenü: Klicken Sie auf die Option "Laden in Gerät".
- Symbolleiste: Klicken Sie auf das Symbol "Laden in Gerät".

Ist für einen Programmbaustein eine dynamische Verknüpfung mit obligatorischem Passwort (Seite 209) festgelegt, muss zum Laden der geschützten Bausteine das Passwort eingegeben werden. Ist diese Art des Kopierschutzes für mehrere Bausteine eingerichtet, muss für jeden der geschützten Bausteine das Passwort für das Laden des Bausteins eingegeben werden.

Hinweis

Durch das Laden eines Programms werden die Werte im remanenten Speicher weder gelöscht noch verändert. Soll der remanente Speicher vor dem Laden gelöscht werden, ist vor dem Laden des Programms die CPU auf die Werkseinstellungen zurückzusetzen.

Sie können auch einPanel-Projekt für die Basic HMI-Panels (Seite 31) vom TIA Portal auf eine Memory Card in der S7-1200 CPU laden.

7.8 Laden von der Online-CPU

Sie können die Programmbausteine einer Online-CPU oder einer an Ihr Programmiergerät angeschlossenen Memory Card auch kopieren.

Bereiten Sie das Offline-Projekt für die kopierten Programmbausteine vor:

1. Fügen Sie eine CPU hinzu, die der Online-CPU entspricht.
2. Erweitern Sie den CPU-Knoten, so dass der Ordner "Programmbausteine" angezeigt wird.

Um die Programmbausteine aus der Online-CPU in das Offline-Projekt zu laden, gehen Sie folgendermaßen vor:

1. Klicken Sie im Offline-Projekt auf den Ordner "Programmbausteine".
2. Klicken Sie auf die Schaltfläche "Online verbinden".
3. Wählen Sie die Schaltfläche "Laden aus CPU".
4. Bestätigen Sie den Vorgang im Dialog Laden aus CPU (Seite 1119).

Nach dem Laden zeigt STEP 7 alle geladenen Programmbausteine im Projekt.

7.8.1 Vergleich der Online-CPU mit der Offline-CPU

Mit dem Editor "Vergleichen" (Seite 1127) in STEP 7 ermitteln Sie Unterschiede zwischen dem Online- und dem Offline-Projekt. Dies kann vor dem Laden des Programms aus der CPU nützlich sein.

7.9 Debugging und Testen des Programms

7.9.1 Daten in der CPU beobachten und steuern

Sie können Werte in der Online-CPU beobachten und steuern (siehe folgende Tabelle).

Tabelle 7- 5 Daten mit STEP 7 beobachten und steuern

Editor	Beobachten	Steuern	Forcen
Beobachtungstabelle	Ja	Ja	Nein
Forcetabelle	Ja	Nein	Ja
Programmiereditor	Ja	Ja	Nein
Variablentabelle	Ja	Nein	Nein
DB-Editor	Ja	Nein	Nein

In einer Beobachtungstabelle beobachten

Im KOP-Editor beobachten

Im Kapitel "Online und Diagnose" finden Sie weitere Informationen zum Thema Daten in der CPU beobachten und steuern (Seite 1128).

7.9.2 Beobachtungstabellen und Forcetabellen

Mit Hilfe von "Beobachtungstabellen" können Sie die Werte eines Anwenderprogramms, das von der Online-CPU ausgeführt wird, überwachen und ändern. Sie können in Ihrem Projekt unterschiedliche Beobachtungstabellen erstellen und speichern, um eine Vielzahl von Testumgebungen abzudecken. So können Sie Tests zum Beispiel bei der Inbetriebnahme oder für Service- und Wartungszwecke durchführen.

Mit einer Beobachtungstabelle können Sie die Ausführung des Anwenderprogramms durch die CPU überwachen und in die Ausführung eingreifen. Sie können nicht nur für die Variablen der Codebausteine und Datenbausteine, sondern auch für die Speicherbereiche der CPU, einschließlich Eingänge und Ausgänge (E und A), periphere Eingänge (E:P), Merker (M) und Datenbausteine (DB) Werte aufrufen und ändern.

Mit der Beobachtungstabelle können Sie die physischen Ausgänge (A:P) einer CPU, die sich im Betriebszustand STOP befindet, freigeben. Beispielsweise können Sie den Ausgängen bestimmte Werte zuweisen, während Sie die Verdrahtung der CPU testen.

STEP 7 bietet zudem eine Forcetabelle zum Forcen einer Variablen auf einen bestimmten Wert. Weitere Informationen zum Forcen finden Sie im Abschnitt zum Forcen von Werten in der CPU (Seite 1136) im Kapitel "Online und Diagnose".

Hinweis

Die Forcewerte werden in der CPU und nicht in der Beobachtungstabelle gespeichert.

Sie können keinen Eingang forcen (Adresse "E"). Sie können jedoch einen Peripherieeingang forcen. Um einen Peripherieeingang zu forcen, hängen Sie ein ":P" an die Adresse an (Beispiel: "On:P").

Mit STEP 7 ist es ferner möglich, Programmvariablen anhand von Auslösebedingungen zu verfolgen und aufzuzeichnen (Seite 1150).

7.9.3 Querverweis zum Anzeigen der Verwendung

Das Inspektorfenster zeigt Querverweise dazu an, wie ein Objekt innerhalb des gesamten Projekts verwendet wird, z. B. im Anwenderprogramm, in der CPU oder den HMI-Geräten. Im Register "Querverweis" werden die Instanzen angezeigt, wo und von welchen anderen Objekten ein ausgewähltes Objekt verwendet wird. Das Inspektorfenster enthält außerdem Bausteine, die nur online innerhalb der Querverweise verfügbar sind. Um die Querverweise anzuzeigen, wählen Sie den Befehl "Querverweise anzeigen". (In der Projektansicht befindet sich dieser Befehl im Menü "Werkzeuge".)

Hinweis

Zum Anzeigen der Querverweisinformationen muss der Editor nicht geschlossen werden.

Die Einträge der Querverweisliste können verschieden sortiert werden. Die Liste der Querverweise bietet einen Überblick über die Verwendung von Speicheradressen und Variablen im Anwenderprogramm.

- Wenn Sie ein Programm anlegen oder ändern, behalten Sie einen Überblick über die verwendeten Operanden, Variablen und Bausteinaufrufe.
- Aus den Querverweisen können Sie direkt an die Stelle springen, an der die Operanden und Variablen verwendet werden.
- Während eines Programmtests oder einer Fehlerbehebung erhalten Sie Informationen dazu, welche Speicheradresse von welchem Befehl in welchem Baustein verarbeitet wird, welche Variable in welchem Bild verwendet wird und welcher Baustein von welchem anderen Baustein aufgerufen wird.

Tabelle 7- 6 Querverweiselemente

Spalte	Beschreibung
Objekt	Name des Objekts, das die angegebenen unterlagerten Objekte verwendet oder das von den unterlagerten Objekten verwendet wird
Anzahl	Anzahl Verwendungen
Verwendung	Der Ort der Verwendung, z. B. ein Netzwerk
Eigenschaft	Besondere Eigenschaften der referenzierten Objekte, z. B. die Variablennamen in Multiinstanz-Deklarationen
als	Zeigt zusätzliche Informationen zum Objekt an, z. B. ob ein Instanz-DB als Vorlage oder als Multiinstanz verwendet wird
Zugriff	Art des Zugriffs, d. h. ob auf den Operanden Lesezugriff (R) und/oder Schreibzugriff (W) besteht
Adresse	Adresse des Operanden
Typ	Angabe, mit welchem Typ und welcher Sprache das Objekt angelegt wurde
Pfad	Pfad des Objekts in der Projektnavigation

Abhängig von den installierten Produkten werden in der Querverweistabelle zusätzliche oder unterschiedliche Spalten angezeigt.

7.9.4 Aufrufstruktur zur Prüfung der Aufrufhierarchie

Die Aufrufstruktur zeigt die Aufrufhierarchie des Bausteins innerhalb Ihres Anwenderprogramms. Sie bietet einen Überblick über die verwendeten Bausteine, die Aufrufe anderer Bausteine, die Beziehungen zwischen Bausteinen, die Datenanforderungen an jeden Baustein sowie den Status der einzelnen Bausteine. Die Bausteine in der Aufrufstruktur können mit dem Programmiereditor geöffnet und bearbeitet werden.

Durch Anzeigen der Aufrufstruktur erhalten Sie eine Liste der im Anwenderprogramm verwendeten Bausteine. STEP 7 zeigt die erste Ebene der Aufrufstruktur hervorgehoben an und zeigt auch die Bausteine an, die durch keinen anderen Baustein im Programm aufgerufen werden. Die erste Ebene der Aufrufstruktur enthält die OBs sowie diejenigen FCs, FBs und DBs, die nicht durch einen OB aufgerufen werden. Von anderen Bausteinen aufgerufene Codebausteine erscheinen eingerückt unter dem aufrufenden Baustein. In der Aufrufstruktur werden nur die Bausteine angezeigt, die von einem Codebaustein aufgerufen werden.

Sie können selektiv nur jene Bausteine anzeigen, die innerhalb der Aufrufstruktur Konflikte verursachen. Folgende Bedingungen führen zu Konflikten:

- Bausteine, die Aufrufe mit älteren oder neueren Zeitstempeln im Code ausführen
- Bausteine, die einen Baustein mit geänderter Schnittstelle aufrufen
- Bausteine, die eine Variable mit geänderter Adresse und/oder geändertem Datentyp verwenden
- Bausteine, die weder direkt noch indirekt durch einen OB aufgerufen werden
- Bausteine, die einen nicht vorhandenen oder fehlenden Baustein aufrufen

Sie können mehrere Bausteinaufrufe und Datenbausteine zu einer Gruppe zusammenfassen. Über eine Klappliste können Sie die Verknüpfungen mit den verschiedenen Aufrufstellen anzeigen.

Sie können außerdem eine Konsistenzprüfung durchführen, um Zeitstempelkonflikte aufzuzeigen. Zeitstempelkonflikte können durch die Änderung des Zeitstempels eines Bausteins während oder nach der Programmgenererierung verursacht werden. Diese Konflikte führen zu Inkonsistenzen zwischen den aufrufenden und den aufgerufenen Bausteinen.

- Die meisten Zeitstempel- und Schnittstellenkonflikte lassen sich durch erneutes Übersetzen der Codebausteine beheben.
- Wenn durch Übersetzen die Inkonsistenzen nicht beseitigt werden, navigieren Sie mit dem Programmeditor über die Verknüpfung in der Spalte "Details" zur Quelle des Problems. Hier können Sie die Inkonsistenzen manuell beseitigen.
- Sind Bausteine rot markiert, müssen sie erneut übersetzt werden.

Anweisungen

8

8.1 Bitverknüpfungen

8.1.1 Bitverknüpfungsanweisungen

KOP und FUP verarbeiten Boolesche Logik sehr effektiv. SCL ist zwar besonders effektiv bei komplexen mathematischen Berechnungen und bei Projektsteuerstrukturen, doch Sie können SCL auch für Boolesche Logik verwenden.

KOP-Kontakte

Tabelle 8- 1 Schließer- und Öffnerkontakte

KOP	SCL	Beschreibung

"IN"	`IF in THEN`
	`  Statement;`
	`ELSE`
	`  Statement;`
	`END_IF;`
		Schließer- und Öffnerkontakte: Sie können Kontakte untereinander verschalten und so Ihre eigene Verschaltungslogik erstellen. Nutzt das von Ihnen angegebene Eingangsbit die Speicherkennung E (Eingang) oder A (Ausgang), so wird der Bitwert aus dem Prozessabbildregister gelesen. Die physischen Kontaktsignale in Ihrem Steuerungsprozess werden mit Eingangsanschlüssen der PLC-Geräts verschaltet. Die CPU fragt die verschalteten Eingangssignale ab und aktualisiert fortlaufend die entsprechenden Zustandswerte im Prozessabbild der Eingänge.
		Das direkte Lesen eines physischen Eingangs führen Sie mit ":P" nach der E-Adresse durch (Beispiel: "%I3.4:P"). Beim direkten Lesen werden die Bitdatenwerte direkt aus dem physischen Eingang und nicht aus dem Prozessabbild gelesen. Beim direktne Lesen wird das Prozessabbild nicht aktualisiert.

"IN"	`IF NOT (in) THEN`
	`  Statement;`
	`ELSE`
	`  Statement;`
	`END_IF;`	

Tabelle 8- 2 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	Bool	Zugewiesenes Bit

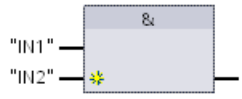
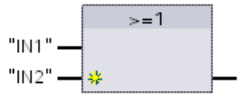
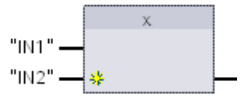
- Der Schließer ist geschlossen (EIN), wenn der zugewiesene Bitwert gleich 1 ist.
- Der Öffner ist geschlossen (EIN), wenn der zugewiesene Bitwert gleich 0 ist.
- In Reihe geschaltete Kontakte bilden logische UND-Verknüpfungen.
- Parallel geschaltete Kontakte bilden logische ODER-Verknüpfungen.

UND-, ODER- und XOR-Boxen in FUP

Bei der FUP-Programmierung werden Netzwerke mit KOP-Kontakten in die Box-Netzwerke UND (&), ODER (>=1) und EXKLUSIV ODER (x) umgewandelt, in denen Sie Bitwerte für die Ein- und Ausgänge der Box angeben können. Sie können ferner Verschaltungen mit anderen Logik-Boxen herstellen und so Ihre eigene Verschaltungslogik erstellen. Nachdem die Box in Ihrem Netzwerk platziert ist, können Sie die Funktion "Eingang einfügen" aus der Funktionsleiste "Favoriten" oder dem Anweisungsverzeichnis zur Eingangsseite der Box ziehen, um weitere Eingänge hinzuzufügen. Sie können auch mit der rechten Maustaste auf den Eingangsanschluss der Box klicken und "Eingang einfügen" auswählen.

Die Ein- und Ausgänge einer Box können mit anderen Boxen verschaltet werden oder Sie können eine Bitadresse oder einen Bitsymbolnamen für einen unverschalteten Eingang eingeben. Bei der Ausführung der Box-Anweisung werden die Eingangszustände auf die binäre Box-Verknüpfung geschaltet und dann der Box-Ausgang, sofern zutreffend, auf Wahr gesetzt.

Tabelle 8- 3 Boxen UND, ODER und EXKLUSIV ODER

FUP	SCL [1]	Beschreibung
	`out := in1 AND in2;`	Damit der Ausgang WAHR ist, müssen alle Eingänge einer UND-Box WAHR sein
	`out := in1 OR in2;`	Damit der Ausgang WAHR ist, muss ein beliebiger Eingang einer ODER-Box WAHR sein
	`out := in1 XOR in2;`	Damit der Ausgang WAHR ist, muss eine ungerade Anzahl der Eingänge einer XOR-Box WAHR sein

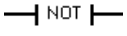
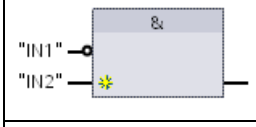
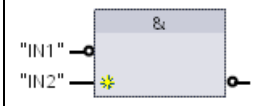
[1] In SCL: Sie müssen das Ergebnis der Operation einer Variable zuweisen, damit es in einer anderen Anweisung verwendet werden kann.

Tabelle 8- 4 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN1, IN2	Bool	Eingangsbit

Logikinvertierer NOT

Tabelle 8- 5 VKE (Verknüpfungsergebnis) invertieren

KOP	FUP	SCL	Beschreibung
	 	`NOT`	Bei der FUP-Programmierung können Sie die Funktion "VKE invertieren" aus der Funktionsleiste "Favoriten" oder dem Anweisungsverzeichnis auf einen Eingang oder einen Ausgang ziehen, um einen Logikinvertierer für diesen Box-Anschluss zu erstellen. Der KOP-Kontakt NOT invertiert den logischen Zustand des Signalflusseingangs. • Ist kein Signalfluss zum NOT-Kontakt vorhanden, so steht ein Signalfluss am Ausgang an. • Ist ein Signalfluss zum NOT-Kontakt vorhanden, so steht kein Signalfluss am Ausgang an.

Ausgangsspule und Zuweisungsbox

Die Anweisung für den Spulenausgang schreibt einen Wert in ein Ausgangsbit. Nutzt das angegebene Ausgangsbit die Speicherkennung A, so schaltet die CPU das Ausgangsbit im Prozessabbildregister ein oder aus und setzt das angegebene Bit jeweils entsprechend dem Signalfluss. Die Ausgangssignale für Ihre Steuerstellglieder werden mit den Ausgangsklemmen der CPU verschaltet. Im Betriebszustand RUN fragt die CPU fortlaufend die Eingangssignale ab, verarbeitet die Eingangszustände gemäß der Programmlogik und reagiert dann, indem sie die neuen Ausgangswerte im Prozessabbild der Ausgänge setzt. Die CPU überträgt die im Prozessabbildregister gespeicherte Reaktion auf den neuen Ausgangszustand zu den verschalteten Ausgangsklemmen.

Tabelle 8- 6 Zuweisung vornehmen und Zuweisung negieren

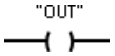
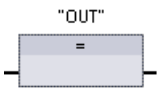
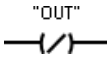
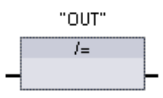
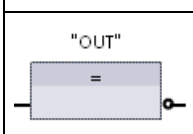
KOP	FUP	SCL	Beschreibung
		`out := <Boolescher Ausdruck>;`	Bei der FUP-Programmierung werden die KOP-Spulen in Zuweisungsboxen (= und /=) umgewandelt, wobei eine Bitadresse für den Boxausgang anzugeben ist. Die Ein- und Ausgänge einer Box können mit anderen Boxverknüpfungen verschaltet werden oder Sie können eine Bitadresse eingeben. Das direkte Schreiben eines physischen Ausgangs geben Sie mit ":P" nach der A-Adresse an (Beispiel: "%Q3.4:P"). Beim direkten Schreiben werden die Bitdatenwerte in den Ausgang im Prozessabbild und direkt in den physischen Ausgang geschrieben.
	 	`out := NOT <Boolescher Ausdruck>;`	

Tabelle 8- 7 Datentypen für die Parameter

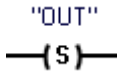
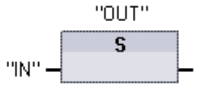
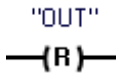
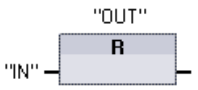
Parameter	Datentyp	Beschreibung
OUT	Bool	Zugewiesenes Bit

- Ist ein Signalfluss durch eine Ausgangsspule vorhanden oder eine FUP-Box "=" aktiviert, so wird das Ausgangsbit auf 1 gesetzt.
- Ist kein Signalfluss durch eine Ausgangsspule vorhanden oder keine FUP-Box "=" aktiviert, so wird das Ausgangsbit auf 0 gesetzt.
- Ist ein Signalfluss durch eine invertierte Ausgangsspule vorhanden oder eine FUP-Box "/=" aktiviert, so wird das Ausgangsbit auf 0 gesetzt.
- Ist kein Signalfluss durch eine invertierte Ausgangsspule vorhanden oder keine FUP-Box "/=" aktiviert, so wird das Ausgangsbit auf 1 gesetzt.

8.1.2 Setz- und Rücksetzoperationen

1 Bit setzen und rücksetzen

Tabelle 8- 8 Anweisungen S und R

KOP	FUP	SCL	Beschreibung
		Nicht verfügbar	Ausgang setzen: Wird S (Setzen) aktiviert, wird der Datenwert an Adresse OUT auf 1 gesetzt. Wird S nicht aktiviert, verändert sich OUT nicht.
		Nicht verfügbar	Ausgang rücksetzen: Wird R (Rücksetzen) aktiviert, wird der Datenwert an Adresse OUT auf 0 gesetzt. Wird R nicht aktiviert, verändert sich OUT nicht.

1 In KOP und FUP: Diese Anweisungen können an jeder beliebigen Stelle im Netzwerk eingefügt werden.

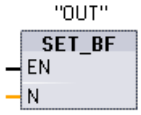
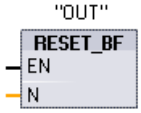
2 In SCL: Sie müssen Code schreiben, um diese Funktion innerhalb Ihrer Anwendung zu replizieren.

Tabelle 8- 9 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN (oder Verschalten mit Kontakt-/Steuerungslogik)	Bool	Bitvariable der zu beobachtenden Adresse
OUT	Bool	Bitvariable der zu setzenden oder rückzusetzenden Adresse

Bitfeld setzen und rücksetzen

Tabelle 8- 10 Anweisungen SET_BF und RESET_BF

KOP[1]	FUP	SCL	Beschreibung
		Nicht verfügbar	Bitfeld setzen: Wird SET_BF aktiviert, so wird der Datenwert 1 in "n" Bits geschrieben, beginnend an Adresse OUT. Wird SET_BF nicht aktiviert, verändert sich OUT nicht.
		Nicht verfügbar	Bitfeld rücksetzen: RESET_BF schreibt den Datenwert 0 in "n" Bits, beginnend an Adresse OUT. Wird RESET_BF nicht aktiviert, verändert sich OUT nicht.

1 In KOP und FUP: Diese Anweisungen müssen ganz rechts in einer Verzweigung angeordnet werden.

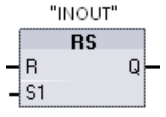
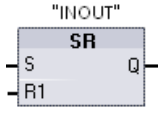
2 In SCL: Sie müssen Code schreiben, um diese Funktion innerhalb Ihrer Anwendung zu replizieren.

Tabelle 8- 11 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
OUT	Bool	Startelement des Bitfelds, das gesetzt oder zurückgesetzt werden soll (Beispiel: #MyArray[3])
n	Konstante (UInt)	Anzahl der zu schreibenden Bits

Flipflop-Schaltungen vorrangig setzen und vorrangig rücksetzen

Tabelle 8- 12 Anweisungen RS und SR

KOP/FUP	SCL	Beschreibung
	Nicht verfügbar	Flipflop rücksetzen/setzen: RS ist ein Flipflop, bei dem das Setzen Vorrang hat. Sind die Signale Setzen (S1) und Rücksetzen (R) beide wahr, so lautet die Ausgangsadresse INOUT 1.
	Nicht verfügbar	Flipflop rücksetzen/setzen: SR ist ein Flipflop, bei dem das Rücksetzen Vorrang hat. Sind die Signale Setzen (S) und Rücksetzen (R1) beide wahr, so lautet die Ausgangsadresse INOUT 0.

1 In KOP und FUP: Diese Anweisungen müssen ganz rechts in einer Verzweigung angeordnet werden.

2 In SCL: Sie müssen Code schreiben, um diese Funktion innerhalb Ihrer Anwendung zu replizieren.

Tabelle 8- 13 Datentypen für die Parameter

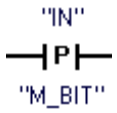
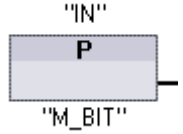
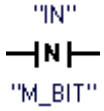
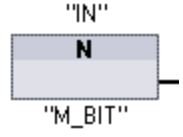
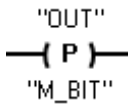
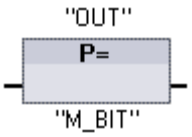
Parameter	Datentyp	Beschreibung
S, S1	Bool	Eingang setzen; 1 weist auf Dominanz hin
R, R1	Bool	Eingang zurücksetzen; 1 weist auf Dominanz hin
INOUT	Bool	Zugewiesene Bitvariable "INOUT"
Q	Bool	Folgt dem Zustand von Bit "INOUT"

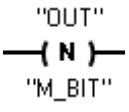
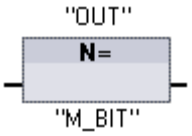
Variable "INOUT" weist die Bitadresse zu, die gesetzt bzw. zurückgesetzt wird. Der optionale Ausgang Q gibt den Signalzustand der Adresse "INOUT" an.

Anweisung	S1	R	Bit "INOUT"
RS	0	0	Vorheriger Zustand
	0	1	0
	1	0	1
	1	1	1
	S	**R1**	
SR	0	0	Vorheriger Zustand
	0	1	0
	1	0	1
	1	1	0

8.1.3 Operationen Steigende Flanke und Fallende Flanke

Tabelle 8- 14 Anweisungen "Steigende Flanke" und "Fallende Flanke"

KOP	FUP	SCL	Beschreibung
		Nicht verfügbar [1]	Operand auf positive Signalflanke abfragen. KOP: Der Zustand dieses Kontakts ist WAHR, wenn eine steigende Flanke (AUS-nach-EIN) am zugewiesenen Bit IN erkannt wird. Der logische Zustand des Kontakts wird dann mit dem Eingangszustand des Signalflusses verknüpft, um den Ausgangszustand des Signalflusses zu setzen. Der Kontakt P kann an einer beliebigen Stelle im Netzwerk angeordnet werden, Ausnahme: am Ende einer Verzweigung. FUP: Der logische Zustand des Ausgangs ist WAHR, wenn eine steigende Flanke (AUS-nach-EIN) am zugewiesenen Eingangsbit erkannt wird. Die Box P darf nur am Anfang einer Verzweigung angeordnet werden.
		Nicht verfügbar [1]	Operand auf negative Signalflanke abfragen. KOP: Der Zustand dieses Kontakts ist WAHR, wenn eine fallende Flanke (EIN-nach-AUS) am zugewiesenen Eingangsbit erkannt wird. Der logische Zustand des Kontakts wird dann mit dem Eingangszustand des Signalflusses verknüpft, um den Ausgangszustand des Signalflusses zu setzen. Der Kontakt N kann an einer beliebigen Stelle im Netzwerk angeordnet werden, Ausnahme: am Ende einer Verzweigung. FUP: Der logische Zustand des Ausgangs ist WAHR, wenn eine fallende Flanke (EIN-nach-AUS) am zugewiesenen Eingangsbit erkannt wird. Die Box N darf nur am Anfang einer Verzweigung angeordnet werden.
		Nicht verfügbar [1]	Operand bei positiver Signalflanke setzen. KOP: Das zugewiesene Bit OUT ist WAHR, wenn eine steigende Flanke (AUS-nach-EIN) am Signalfluss, der in die Spule eintritt, erkannt wird. Der Eingangszustand des Signalflusses durchläuft die Spule immer als Ausgangszustand des Signalflusses. Die Spule P kann an jeder beliebigen Stelle im Netzwerk eingefügt werden. FUP: Das zugewiesene Bit OUT ist WAHR, wenn eine steigende Flanke (AUS-nach-EIN) am logischen Zustand des Eingangsanschlusses der Box oder am zugewiesenen Eingangsbit erkannt wird, sofern sich die Box am Anfang einer Verzweigung befindet. Der logische Zustand des Eingangs durchläuft die Box immer als logischer Zustand des Ausgangs. Die Box P= kann an jeder beliebigen Stelle in der Verzweigung eingefügt werden.

KOP	FUP	SCL	Beschreibung
		Nicht verfügbar [1]	Operand bei negativer Signalflanke setzen. KOP: Das zugewiesene Bit OUT ist WAHR, wenn eine fallende Flanke (EIN-nach-AUS) am Signalfluss, der in die Spule eintritt, erkannt wird. Der Eingangszustand des Signalflusses durchläuft die Spule immer als Ausgangszustand des Signalflusses. Die Spule N kann an jeder beliebigen Stelle im Netzwerk eingefügt werden. FUP: Das zugewiesene Bit OUT ist WAHR, wenn eine fallende Flanke (EIN-nach-AUS) am logischen Zustand des Eingangsanschlusses der Box oder am zugewiesenen Eingangsbit erkannt wird, sofern sich die Box am Anfang einer Verzweigung befindet. Der logische Zustand des Eingangs durchläuft die Box immer als logischer Zustand des Ausgangs. Die Box N= kann an jeder beliebigen Stelle in der Verzweigung eingefügt werden.

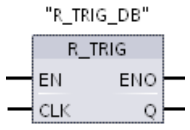
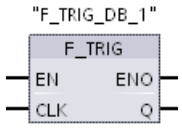
[1] In SCL: Sie müssen Code schreiben, um diese Funktion innerhalb Ihrer Anwendung zu replizieren.

Tabelle 8- 15 P_TRIG und N_TRIG

KOP/FUP	SCL	Beschreibung
	Nicht verfügbar [1]	VKE (Verknüpfungsergebnis) auf positive Signalflanke abfragen. Der Signalfluss oder der logische Zustand des Ausgangs Q ist WAHR, wenn eine steigende Flanke (AUS-nach-EIN) am Eingangszustand CLK (FUP) oder am Signalfluss CLK (KOP) erkannt wird. In KOP darf die Anweisung P_TRIG nicht am Anfang oder Ende eines Netzwerks angeordnet werden. In FUP darf die Anweisung P_TRIG an einer beliebigen Stelle angeordnet werden, Ausnahme: am Ende einer Verzweigung.
	Nicht verfügbar [1]	VKE auf negative Signalflanke abfragen. Der Signalfluss oder der logische Zustand des Ausgangs Q ist WAHR, wenn eine fallende Flanke (EIN-nach-AUS) am Eingangszustand CLK (FUP) oder am Signalfluss CLK (KOP) erkannt wird. In KOP darf die Anweisung N_TRIG nicht am Anfang oder Ende eines Netzwerks angeordnet werden. In FUP darf die Anweisung N_TRIG an einer beliebigen Stelle angeordnet werden, Ausnahme: am Ende einer Verzweigung.

[1] In SCL: Sie müssen Code schreiben, um diese Funktion innerhalb Ihrer Anwendung zu replizieren.

Tabelle 8- 16 Anweisungen R_TRIG und F_TRIG

KOP/FUP	SCL	Beschreibung
	**"R_TRIG_DB"(CLK:=_in_, Q=> _bool_out_);**	Variable bei positiver Signalflanke setzen. Der vorhergehende Zustand des Eingangs CLK wird im zugewiesenen Instanz-DB gespeichert. Der Signalfluss oder der logische Zustand des Ausgangs Q ist WAHR, wenn eine steigende Flanke (AUS-nach-EIN) am Eingangszustand CLK (FUP) oder am Signalfluss CLK (KOP) erkannt wird. In KOP darf die Anweisung R_TRIG nicht am Anfang oder Ende eines Netzwerks angeordnet werden. In FUP darf die Anweisung R_TRIG an einer beliebigen Stelle angeordnet werden, ausgenommen am Ende einer Verzweigung.
	"F_TRIG_DB"(CLK:=_in_, Q=> _bool_out_);	Variable bei negativer Signalflanke setzen. Der vorhergehende Zustand des Eingangs CLK wird im zugewiesenen Instanz-DB gespeichert. Der Signalfluss oder der logische Zustand des Ausgangs Q ist WAHR, wenn eine fallende Flanke (EIN-nach-AUS) am Eingangszustand CLK (FUP) oder am Signalfluss CLK (KOP) erkannt wird. In KOP darf die Anweisung F_TRIG nicht am Anfang oder Ende eines Netzwerks angeordnet werden. In FUP darf die Anweisung F_TRIG an einer beliebigen Stelle angeordnet werden, ausgenommen am Ende einer Verzweigung.

Für R_TRIG und F_TRIG wird automatisch der Dialog "Aufrufoptionen" geöffnet, wenn die Anweisung im Programm eingefügt wird. In diesem Dialog können Sie festlegen, ob der Flanken-Merker in seinem eigenen Datenbaustein gespeichert wird (Einzelinstanz) oder als lokale Variable (Multiinstanz) in der Bausteinschnittstelle gespeichert wird. Wird ein separater Datenbaustein angelegt, so ist dieser in der Projektnavigation im Ordner "Programmressourcen" unter "Programmbausteine > Systembausteine" zu finden.

Tabelle 8- 17 Datentypen für die Parameter (Kontakte/Spulen P und N, P=, N=, P_TRIG and N_TRIG)

Parameter	Datentyp	Beschreibung
M_BIT	Bool	Merker, in dem der vorherige Zustand des Eingangs gespeichert wird
IN	Bool	Eingangsbit, dessen Flanke erkannt wird
OUT	Bool	Ausgangsbit, das angibt, dass eine Flanke erkannt wurde
CLK	Bool	Signalfluss oder Eingangsbit, dessen Flanke erkannt wird
Q	Bool	Ausgang, der angibt, dass eine Flanke erkannt wurde

Alle Flankenanweisungen verwenden einen Merker (M_BIT: P/N Kontakte/Spulen, P_TRIG/N_TRIG) oder (Instanz-DB-Bit: R_TRIG, F_TRIG) zur Speicherung des vorhergehenden Zustands des beobachteten Eingangssignals. Eine Flanke wird durch Vergleichen des Zustands des Eingangs mit dem vorhergehenden Zustand erkannt. Wenn die Zustände am Eingang auf einen Signalwechsel in der gewünschten Richtung hinweisen, wird eine Flanke gemeldet, indem der Ausgang auf WAHR gesetzt wird. Andernfalls wird der Ausgang auf FALSCH gesetzt.

Hinweis

Flankenoperationen werten den Eingang und die Merkerwerte bei jeder Ausführung aus, auch bei der ersten Ausführung. Sie müssen die Ausgangszustände des Eingangs und des Merkers in Ihrem Programm berücksichtigen, um die Flankenerkennung im ersten Zyklus zuzulassen oder nicht.

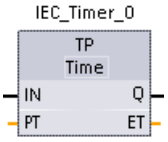
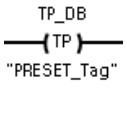
Weil der Merker von einer Ausführung zur nächsten gespeichert werden muss, müssen Sie für jede Flankenoperation ein eindeutiges Bit verwenden. Dieses Bit dürfen Sie nicht an anderen Stellen in Ihrem Programm nutzen. Vermeiden Sie außerdem temporären Speicher und Speicher, der von anderen Systemfunktionen geändert werden kann, z. B. E/A-Aktualisierungen. Verwenden Sie nur Merker (M), globale DBs oder statischen Speicher (in einem Instanz-DB) für M_BIT-Speicherzuweisungen.

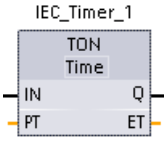
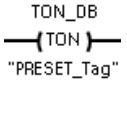
8.2 Funktionsweise der Zeiten

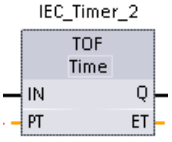
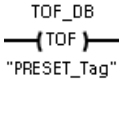
Mit den Zeitanweisungen können Sie programmierte Zeitverzögerungen einrichten. Die Anzahl der Zeiten, die Sie in Ihrem Anwenderprogramm verwenden können, ist lediglich durch den Speicherplatz in der CPU begrenzt. Jede Zeit nutzt eine 16 Byte große DB-Struktur vom Datentyp IEC_Timer, um Zeitdaten zu speichern, die im oberen Bereich der Box- oder Spulenanweisung angegeben werden. STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

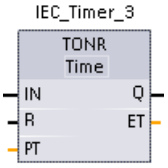
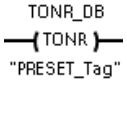
Tabelle 8- 18 Zeitoperationen

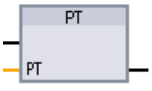
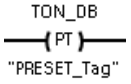
KOP/FUP-Boxen	KOP-Spulen	SCL	Beschreibung

		`"IEC_Timer_0_DB".TP(`
		`IN:=_bool_in_,`
		`PT:=_time_in_,`
		`Q=>_bool_out_,`
		`ET=>_time_out_);`
			Die Zeit TP erzeugt einen Impuls mit einer voreingestellten Dauer.

		`"IEC_Timer_0_DB".TON (`
		`IN:=_bool_in_,`
		`PT:=_time_in_,`
		`Q=>_bool_out_,`
		`ET=>_time_out_);`
			Die Zeit TON setzt den Ausgang Q nach einer voreingestellten Zeit auf EIN.

		`"IEC_Timer_0_DB".TOF (`
		`IN:=_bool_in_,`
		`PT:=_time_in_,`
		`Q=>_bool_out_,`
		`ET=>_time_out_);`
			Die Zeit TOF setzt den Ausgang Q nach einer voreingestellten Zeit auf AUS zurück.

		`"IEC_Timer_0_DB".TONR (`
		`IN:=_bool_in_,`
		`R:=_bool_in_,`
		`PT:=_time_in_,`
		`Q=>_bool_out_,`
		`ET=>_time_out_);`
			Die Zeit TONR setzt den Ausgang Q nach einer voreingestellten Zeit auf EIN. Die abgelaufene Zeit wird über mehrere Zeitintervalle kumuliert, bis Eingang R zum Zurücksetzen der abgelaufenen Zeit angestoßen wird.

Nur FUP:
 
		`PRESET_TIMER(`
		`PT:=_time_in_,`
		
		`TIMER:=_iec_timer_in_);`
			Die Spule PT (Voreingestellte Zeit) lädt einen neuen PRESET-Zeitwert in den angegebenen IEC_Timer.

Nur FUP: 		`RESET_TIMER(` `_iec_timer_in_);`	Die Spule RT (Zeit rücksetzen) setzt den angegebenen IEC_Timer zurück.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 In den SCL-Beispielen ist "IEC_Timer_0_DB" der Name des Instanz-DBs.

Tabelle 8- 19 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
Box: IN Spule: Signalfluss	Bool	TP, TON und TONR: Box: 0 = Zeit deaktivieren, 1 = Zeit aktivieren Spule: Kein Signalfluss = Zeit deaktivieren, Signalfluss = Zeit aktivieren TOF: Box: 0 = Zeit aktivieren, 1 = Zeit deaktivieren Spule: Kein Signalfluss = Zeit aktivieren, Signalfluss = Zeit deaktivieren
R	Bool	Nur TONR-Box: 0 = Nicht zurücksetzen 1= Abgelaufene Zeit und Q-Bit auf 0 zurücksetzen
Box: PT Spule: "PRESET_Tag"	Time	Zeitbox oder -spule: Eingang voreingestellte Zeit
Box: Q Spule: DBdata.Q	Bool	Zeitbox: Q-Boxausgang oder Q-Bit in den DB-Daten der Zeit Zeitspule: Sie können nur das Q-Bit in den DB-Daten der Zeit adressieren
Box: ET Spule: DBdata.ET	Time	Zeitbox: ET-Boxausgang (abgelaufene Zeit) oder ET-Zeitwert in den DB-Daten der Zeit Zeitspule: Sie können nur den ET-Zeitwert in den DB-Daten der Zeit adressieren

Tabelle 8- 20 Auswirkung von Wertänderungen in den Parametern PT und IN

Zeitschaltuhr	Änderungen der Box-Parameter PT und IN und der entsprechenden Spulenparameter
TP	• Ändert sich PT, während die Zeit läuft, hat dies keine Auswirkungen. • Ändert sich IN, während die Zeit läuft, hat dies keine Auswirkungen.
TON	• Ändert sich PT, während die Zeit läuft, hat dies keine Auswirkungen. • Wenn IN nach FALSCH wechselt, während die Zeit läuft, wird die Zeit angehalten und zurückgesetzt.
TOF	• Ändert sich PT, während die Zeit läuft, hat dies keine Auswirkungen. • Wenn IN nach WAHR wechselt, während die Zeit läuft, wird die Zeit angehalten und zurückgesetzt.
TONR	• Ändert sich PT, während die Zeit läuft, hat dies keine Auswirkungen. Es hat dann Auswirkungen, wenn die Zeit fortgesetzt wird. • Wenn IN nach FALSCH wechselt, während die Zeit läuft, wird die Zeit angehalten, jedoch nicht zurückgesetzt. Wenn IN wieder nach WAHR wechselt, beginnt die Zeit ab dem kumulierten Zeitwert zu laufen.

Die Werte von PT (voreingestellte Zeit) und ET (abgelaufene Zeit) werden in den Daten des angegebenen DBs IEC_TIMER als vorzeichenbehaftete doppelte Ganzzahlen gespeichert, die einen Zeitwert in Millisekunden darstellen. Der Datentyp TIME verwendet die Kennung T# und kann als einfache Zeiteinheit (T#200ms oder 200) oder als zusammengesetzte Zeiteinheiten wie T#2s_200ms eingegeben werden.

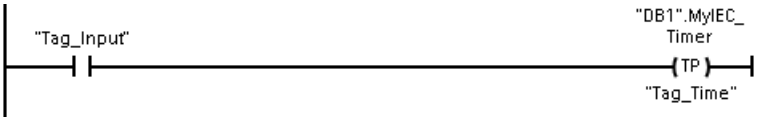
Tabelle 8- 21 Größe und Bereich des Datentyps TIME

Datentyp	Größe	Gültige Zahlenbereiche[1]
TIME	32 Bit, gespeichert alsDInt -Daten	T#-24d_20h_31m_23s_648ms bis T#24d_20h_31m_23s_647ms Gespeichert als -2.147.483.648 ms bis +2.147.483.647 ms

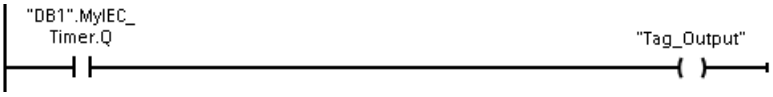
[1] Der negative Bereich des oben dargestellten Datentyps TIME kann für die Zeiten nicht verwendet werden. Negative Werte für PT (voreingestellte Zeit) werden bei Ausführung der Zeitoperation auf Null gesetzt. ET (abgelaufene Zeit) ist immer ein positiver Wert.

Beispiel für eine Zeitspule

Die Zeitspulen -(TP)-, -(TON)-, -(TOF)- und -(TONR)- müssen als letzte Anweisung in einem KOP-Netzwerk angeordnet werden. Wie im Beispiel für eine Zeit gezeigt, wertet eine Kontaktanweisung in einem nachfolgenden Netzwerk das Q-Bit in den Daten des DBs IEC_Timer einer Zeitspule aus. Ebenso müssen Sie das Element ELAPSED in den Daten des DBs IEC_Timer adressieren, wenn Sie den Wert der abgelaufenen Zeit in Ihrem Programm verwenden möchten.



Der Impulszeitgeber wird bei einem Wechsel von 0 nach 1 des Bitwerts von Tag_Input gestartet. Die Zeit läuft für die vom Zeitwert Tag_Time angegebene Zeitdauer.



Solange die Zeit läuft, ist Zustand DB1.MyIEC_Timer.Q=1 und Tag_Output=1. Wenn der Wert für Tag_Time abgelaufen ist, sind DB1.MyIEC_Timer.Q=0 und Tag_Output=0.

Spulen Zeit rücksetzen -(RT)- und Zeitdauer laden -(PT)-

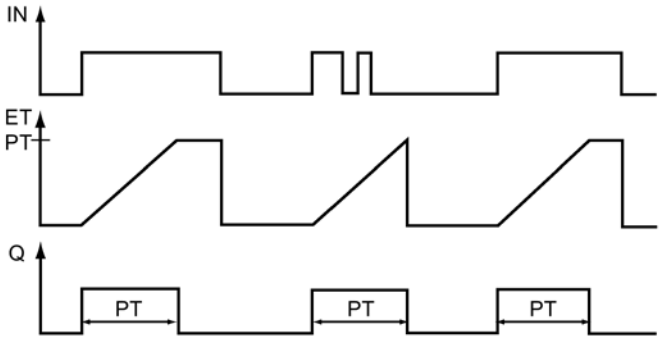
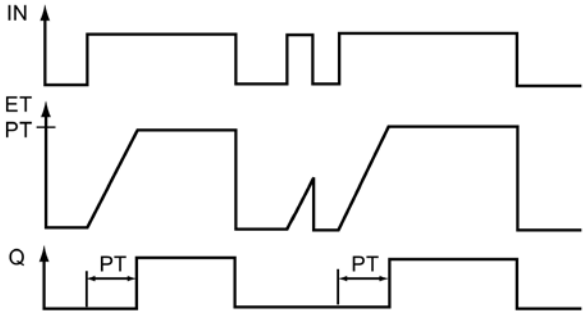
Diese Spulenanweisungen können mit Box- oder Spulenzeiten verwendet werden und in der Mitte eines Strompfads angeordnet werden. Der Signalzustand des Spulenausgangs ist immer der gleiche wie am Spuleneingang. Wenn die Spule -(RT)- aktiviert wird, wird das Zeitelement ELAPSED in den Daten des angegebenen DBs IEC_Timer auf 0 zurückgesetzt. Wenn die Spule -(PT)- aktiviert wird, wird in das Zeitelement PRESET in den Daten des angegebenen DBs IEC_Timer der zugewiesene Zeit-Dauer-Wert geladen.

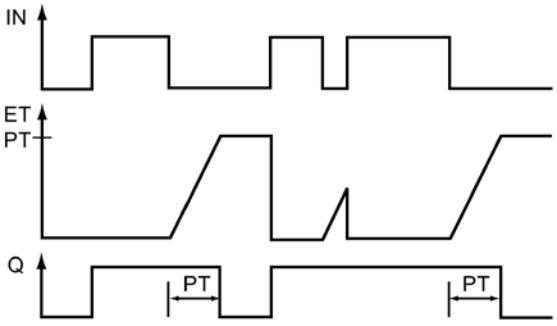
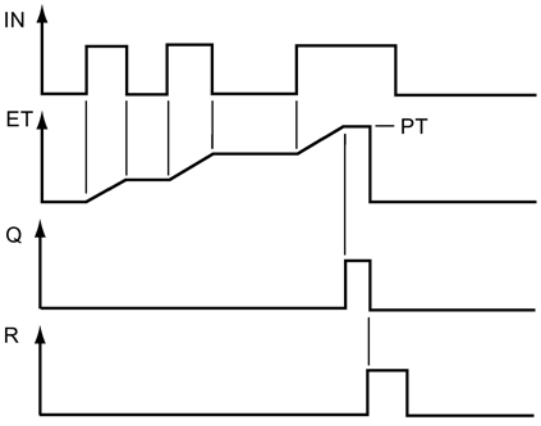
Hinweis

Wenn Sie Zeiten in einem FB platzieren, können Sie die Option "Multiinstanz-Datenbaustein" auswählen. Die Namen der Zeitstrukturen können bei verschiedenen Datenstrukturen unterschiedlich sein, doch die Zeitdaten befinden sich in einem einzigen Datenbaustein, und es ist nicht für jede Zeit ein eigener Datenbaustein erforderlich. Dadurch verringert sich die Verarbeitungszeit und der für die Verwaltung der Zeiten erforderliche Datenspeicher. Zwischen den Datenstrukturen der Zeiten im gemeinsam genutzten Multiinstanz-DB gibt es keine Wechselwirkungen.

Funktionsweise der Zeiten

Tabelle 8- 22 Typen von IEC-Zeiten

Zeit	Zeitdiagramm
TP: Impuls generieren Die Zeit TP erzeugt einen Impuls mit einer voreingestellten Dauer.	
TON: Einschaltverzögerung generieren Die Zeit TON setzt den Ausgang Q nach einer voreingestellten Zeit auf EIN.	

Zeit	Zeitdiagramm
TOF: Ausschaltverzögerung generieren Die Zeit TON setzt den Ausgang Q nach einer voreingestellten Zeit auf AUS zurück.	
TONR: Zeitakkumulator Die Zeit TONR setzt den Ausgang Q nach einer voreingestellten Zeit auf EIN. Die abgelaufene Zeit wird über mehrere Zeitintervalle kumuliert, bis Eingang R zum Zurücksetzen der abgelaufenen Zeit angestoßen wird.	

Hinweis

In der CPU ist keiner spezifischen Zeit eine bestimmte Ressource zugeordnet. Stattdessen nutzt jede Zeit eine eigene Zeitstruktur im DB-Speicher und eine fortlaufend in Betrieb befindliche interne CPU-Zeit, um die Zeitsteuerung durchzuführen.

Wenn eine Zeit aufgrund eines Flankenwechsels am Eingang einer Anweisung TP, TON, TOF oder TONR gestartet wird, wird der Wert der fortlaufend in Betrieb befindlichen internen CPU-Zeit in das START-Element der DB-Struktur kopiert, die dieser Zeit zugeordnet ist. Dieser Startwert bleibt solange unverändert, wie die Zeit läuft, und wird später bei jeder Aktualisierung der Zeit verwendet. Jedesmal, wenn die Zeit gestartet wrid, wird ein neuer Startwert aus der internen CPU-Zeit in die Zeitstruktur geladen.

Wenn eine Zeit aktualisiert wird, wird der oben beschriebene Startwert vom aktuellen Wert der internen CPU-Zeit subtrahiert, um die abgelaufene Zeit zu ermitteln. Die abgelaufene Zeit wird dann mit der Voreinstellung verglichen, um den Zustand des Zeitbits Q zu ermitteln. Die Elemente ELAPSED und Q werden dann in der DB-Struktur, die dieser Zeit zugeordnet ist, aktualisiert. Beachten Sie, dass die abgelaufene Zeit an den voreingestellten Wert geknüpft ist (die Zeit akkumuliert die abgelaufene Zeit nicht weiter, wenn die Voreinstellung erreicht ist).

Die Zeit wird nur dann aktualisiert, wenn:

- Eine Zeitanweisung (TP, TON, TOF oder TONR) ausgeführt wird
- Das Element "ELAPSED" der Zeitstruktur im DB direkt von einer Anweisung referenziert wird
- Das Element "Q" der Zeitstruktur im DB direkt von einer Anweisung referenziert wird

Programmierung von Zeiten

Die folgenden Konsequenzen von Zeiten sind bei der Planung und Erstellung Ihres Anwenderprogramms zu berücksichtigen:

- Sie können mehrere Aktualisierungen einer Zeit im gleichen Zyklus haben. Die Zeit wird bei jeder Ausführung der Zeitanweisung (TP, TON, TOF, TONR) aktualisiert sowie jedesmal, wenn das Element ELAPSED oder Q der Zeitstruktur als Parameter einer anderen ausgeführten Anweisung verwendet wird. Dies ist ein Vorteil, wenn Sie die neuesten Zeitdaten benötigen (praktisch ein direktes Auslesen der Zeit). Wenn Sie jedoch während eines Programmzyklus konsistente Werte nutzen möchten, ordnen Sie Ihre Zeitanweisung vor allen anderen Anweisungen, die diese Werte benötigen, an und verwenden statt der Elemente ELAPSED und Q der Zeitstruktur im DB die Variablen der Ausgänge Q und ET der Zeitanweisung.
- Sie können Zyklen ohne Zeitaktualisierung haben. Es ist möglich, Ihre Zeit in einer Funktion zu starten und diese Funktion dann für einen oder weitere Zyklen nicht mehr aufzurufen. Wenn keine anderen Anweisungen ausgeführt werden, die die Elemente ELAPSED oder Q der Zeitstruktur referenzieren, wird die Zeit nicht mehr aktualisiert. Eine neue Aktualisierung tritt erst auf, wenn entweder die Zeitanweisung erneut ausgeführt wird oder eine andere Anweisung ausgeführt wird, die ELAPSED oder Q aus der Zeitstruktur als Parameter nutzt.
- Es ist zwar nicht gerade üblich, aber Sie können mehreren Zeitanweisungen die gleiche DB-Zeitstruktur zuweisen. In Allgemeinen sollten Sie, um unerwartete Wechselwirkungen zu vermeiden, nur eine Zeit (TP, TON, TOF, TONR) pro DB-Zeitstruktur verwenden.
- Selbstrücksetzende Zeiten sind bei Trigger-Aktionen nützlich, die regelmäßig auftreten sollen. Typischerweise werden selbstrücksetzende Zeiten erstellt, indem ein Öffnerkontakt, der das Zeitbit referenziert, vor der Zeitanweisung angeordnet wird. Dieses Zeitnetzwerk befindet sich typischerweise oberhalb eines oder mehrerer abhängiger Netzwerke, die mit dem Zeitbit Aktionen auslösen. Wenn die Zeit abläuft (die abgelaufene Zeit erreicht den voreingestellten Wert), ist das Zeitbit einen Zyklus lang EIN, so dass die Logik der vom Zeitbit gesteuerten abhängigen Netzwerke ausgeführt werden kann. Bei der nächsten Ausführung des Zeitnetzwerks ist der Öffnerkontakt AUS, wodurch die Zeit zurückgesetzt und das Zeitbit gelöscht wird. Im nächsten Zyklus ist der Öffnerkontakt EIN, weshalb die Zeit neu gestartet wird. Beim Erstellen von selbstrücksetzenden Zeiten wie dieser verwenden Sie das Element "Q" der Zeitstruktur im DB nicht als den Parameter für den Öffnerkontakt vor der Zeitanweisung. Verwenden Sie hierfür stattdessen die an den Ausgang "Q" der Zeitanweisung angeschlossene Variable. Der Grund, weshalb vermieden werden sollte, auf das Element Q der DB-Zeitstruktur zuzugreifen, ist der, dass dadurch eine Aktualisierung der Zeit verursacht wird. Und wenn die Zeit durch den Öffnerkontakt aktualisiert wird, setzt der Kontakt die Zeitanweisung sofort zurück. Der Ausgang Q der Zeitanweisung ist während dieses einen Zyklus nicht EIN, und die abhängigen Netzwerke werden nicht ausgeführt.

Speichern von Zeitdaten nach einem RUN-STOP-RUN-Wechsel oder einem Neustart der CPU

Wenn eine Session im Betriebszustand RUN mit dem Betriebszustand STOP oder einem Neustart der beendet wird und eine neue Session im Betriebszustand RUN gestartet wird, gehen die in der vorherigen RUN-Session gespeicherten Zeitdaten verloren, sofern die Zeitdatenstruktur nicht als remanent definiert ist (Zeiten TP, TON, TOF und TONR).

Wenn Sie beim Einfügen einer Zeitanweisung im Programmiereditor im Dialog der Aufrufoptionen die Standardeinstellung übernehmen, wird automatisch ein Instanz-DB zugewiesen, der **nicht als remanent definiert werden kann**. Um Ihre Zeitdaten als remanent zu definieren, müssen Sie entweder einen globalen DB oder einen Multiinstanz-DB nutzen.

Zeitdaten durch Zuweisen eines globalen DB als remanente Daten speichern

Diese Option funktioniert unabhängig davon, wo die Zeit platziert wird (OB, FC oder FB).

1. Globalen DB erstellen:
   - Doppelklicken Sie in der Projektnavigation auf "Neuen Baustein hinzufügen".
   - Klicken Sie auf das Symbol für Datenbaustein (DB).
   - Wählen Sie globalen DB als Typ.
   - Wenn Sie einzelne Datenelemente im DB als remanent definieren möchten, müssen Sie für den DB-Typ das Kontrollkästchen "Optimiert" aktivieren. Die andere Option für den DB-Typ "Standard - kompatibel mit S7-300/400" gestattet es lediglich, entweder alle Datenelemente des DBs oder keine Datenelemente als remanent zu definieren.
   - Klicken Sie auf "OK".
2. Zeitstruktur(en) zum DB hinzufügen:
   - Fügen Sie im neuen globalen DB eine neue statische Variable vom Datentyp IEC_Timer ein.
   - Aktivieren Sie das Kontrollkästchen in der Spalte "Remanent", damit diese Struktur remanent ist.
   - Wiederholen Sie diesen Vorgang, um für alle Zeiten, die Sie in diesem DB speichern möchten, Strukturen anzulegen. Sie können entweder jede Zeitstruktur in einem eindeutigen globalen DB anordnen, oder Sie können mehrere Zeitstrukturen in demselben globalen DB platzieren. Sie können in diesem globalen DB außer Zeiten auch andere statische Variablen anordnen. Durch Platzieren mehrerer Zeitstrukturen in demselben globalen DB können Sie die Gesamtzahl der Bausteine verringern.
   - Benennen Sie die Zeitstrukturen ggf. um.
3. Öffnen Sie den Programmbaustein (OB, FC oder FB), in den Sie eine remanente Zeit einfügen möchten, für die Bearbeitung.
4. Platzieren Sie die Zeitanweisung an der gewünschten Stelle.
5. Wenn der Dialog mit den Aufrufoptionen angezeigt wird, klicken Sie auf "Abbrechen".
6. Geben Sie oben in der neuen Zeitanweisung den Namen des globalen DB und der Zeitstruktur ein, die Sie oben erstellt haben (wählen Sie keinen Namen aus). Beispiel: "Data_block_3.Static_1".

Zeitdaten durch Zuweisen eines Multiinstanz-DB als remanente Daten speichern

Diese Möglichkeit funktioniert nur, wenn Sie die Zeit in einem FB ablegen.

Diese Variante ist davon abhängig, ob in den FB-Eigenschaften "Optimierter Bausteinzugriff" festgelegt ist (nur die symbolische Adressierung ist möglich). Um zu prüfen, wie das Zugriffsattribut eines vorhandenen FB konfiguriert ist, klicken Sie mit der rechten Maustaste in der Projektnavigation auf den FB, wählen "Eigenschaften" und dann "Attribute".

Wenn für den FB "Optimierter Bausteinzugriff" festgelegt ist (nur die symbolische Adressierung ist möglich):

1. Öffnen Sie den FB zum Bearbeiten.
2. Platzieren Sie die Zeitanweisung an der gewünschten Stelle im FB.
3. Wenn der Dialog mit den Aufrufoptionen angezeigt wird, klicken Sie auf das Multiinstanz-Symbol. Die Multiinstanz-Option ist nur verfügbar, wenn die Anweisung in einen FB eingefügt wird.
4. Geben Sie der Zeit im Dialog der Aufrufoptionen ggf. einen Namen.
5. Klicken Sie auf "OK". Die Zeitanweisung erscheint im Editor und die IEC_TIMER-Struktur erscheint in der FB-Schnittstelle unter "Statisch".
6. Öffnen Sie ggf. den FB-Schnittstelleneditor (Sie müssen möglicherweise auf den kleinen Pfeil klicken, um die Ansicht zu vergrößern).
7. Suchen Sie unter "Statisch" die Zeitstruktur, die gerade für Sie angelegt wurde.
8. Ändern Sie in der Spalte "Remanent" dieser Zeitstruktur die Auswahl in "Remanent". Immer wenn dieser FB später in einem anderen Programmbaustein aufgerufen wird, wird ein Instanz-DB mit dieser Schnittstellendefinition angelegt, bei der die Zeitstruktur als remanent definiert ist.

Ist die Option "Optimierter Bausteinzugriff" für den FB nicht angegeben, wird der Standardbausteinzugriff verwendet, der mit klassischen S7-300/400 Konfigurationen kompatibel ist und symbolischen sowie direkten Zugriff gestattet. Um einem FB mit Standardbausteinzugriff eine Multiinstanz zuzuweisen, gehen Sie wie folgt vor:

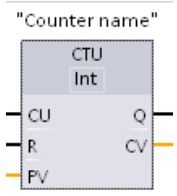
1. Öffnen Sie den FB zum Bearbeiten.
2. Platzieren Sie die Zeitanweisung an der gewünschten Stelle im FB.
3. Wenn der Dialog mit den Aufrufoptionen angezeigt wird, klicken Sie auf das Multiinstanz-Symbol. Die Multiinstanz-Option ist nur verfügbar, wenn die Anweisung in einen FB eingefügt wird.
4. Geben Sie der Zeit im Dialog der Aufrufoptionen ggf. einen Namen.
5. Klicken Sie auf "OK". Die Zeitanweisung erscheint im Editor und die IEC_TIMER-Struktur erscheint in der FB-Schnittstelle unter "Statisch".
6. Öffnen Sie den Baustein, der diesen FB verwenden soll.
7. Platzieren Sie diesen FB an der gewünschten Stelle. Dadurch wird ein Instanz-Datenbaustein für diesen FB angelegt.

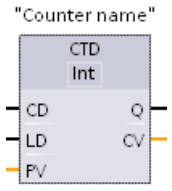
8. Öffnen Sie den Instanz-Datenbaustein, der angelegt wurde, als Sie den FB im Editor platziert haben.
9. Suchen Sie unter "Statisch" die gewünschte Zeitstruktur. Aktivieren Sie in der Spalte "Remanent" dieser Zeitstruktur das Kontrollkästchen, um diese Struktur als remanent zu definieren.

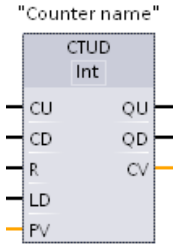
8.3 Funktionsweise der Zähler

Tabelle 8- 23 Zähleranweisungen

KOP/FUP	SCL	Beschreibung

	`"IEC_Counter_0_DB".CTU (`
	`CU:=_bool_in,`
	`R:=_bool_in,`
	`PV:=_in,`
	`Q=>_bool_out,`
	`CV=>_out);`
		Mit den Zähleranweisungen können Sie programminterne Ereignisse und externe Prozessereignisse zählen. Jeder Zähler nutzt eine in einem Datenbaustein abgelegte Struktur, um die Daten des Zählers zu speichern. Sie weisen den Datenbaustein zu, wenn Sie die Zähloperation im Editor einfügen.
		• CTU zählt vorwärts.
		• CTD zählt rückwärts.
		• CTUD zählt vorwärts und rückwärts.

	`"IEC_Counter_0_DB".CTD (`
	`CD:=_bool_in,`
	`LD:=_bool_in,`
	`PV:=_in,`
	`Q=>_bool_out,`
	`CV=>_out);`	

	`"IEC_Counter_0_DB".CTUD(`
	`CU:=_bool_in,`
	`CD:=_bool_in,`
	`R:=_bool_in,`
	`LD:=_bool_in,`
	`PV:=_in_,`
	`QU=>_bool_out,`
	`QD=>_bool_out,`
	`CV=>_out_);`	

[1] In KOP und FUP: Wählen Sie den Datentyp für den Zählwert aus der Klappliste unterhalb des Anweisungsnamens aus.

[2] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

[3] In den SCL-Beispielen ist "IEC_Counter_0_DB" der Name des Instanz-DBs.

Tabelle 8- 24 Datentypen für die Parameter

Parameter	Datentyp[1]	Beschreibung
CU, CD	Bool	Aufwärts- oder Abwärtszählen um jeweils eine Einheit
R (CTU, CTUD)	Bool	Zählwert auf Null zurücksetzen
LD (CTD, CTUD)	Bool	Ladesteuerung für den voreingestellten Wert
PV	SInt, Int, DInt, USInt, UInt, UDInt	Voreingestellter Zählwert
Q, QU	Bool	Wahr, wenn CV >= PV
QD	Bool	Wahr, wenn CV <= 0
CV	SInt, Int, DInt, USInt, UInt, UDInt	Aktueller Zählwert

[1] Der numerische Bereich der Zählwerte hängt vom ausgewählten Datentyp ab. Ist der Zählwert ein ganzzahliger Wert ohne Vorzeichen, können Sie bis Null herunter- oder bis zur Bereichsgrenze hochzählen. Ist der Zählwert ein ganzzahliger Wert mit Vorzeichen, können Sie bis zum unteren Grenzwert herunter- und bis zum oberen Grenzwert hochzählen.

Die Anzahl der Zähler, die Sie in Ihrem Anwenderprogramm verwenden können, ist lediglich durch den Speicherplatz in der CPU begrenzt. Zähler benötigen den folgenden Speicherplatz:

- Bei den Datentypen SInt oder USInt benötigt die Zähleranweisung 3 Byte.
- Bei den Datentypen Int oder UInt benötigt die Zähleranweisung 6 Byte.
- Bei den Datentypen DInt oder UDInt benötigt die Zähleranweisung 12 Byte.

Diese Anweisungen nutzen Softwarezähler, deren maximale Zählgeschwindigkeit durch die Ausführungsrate des OBs, in den sie eingefügt wurden, begrenzt ist. Der OB, in den die Anweisungen eingefügt werden, muss häufig genug ausgeführt werden, um alle Transitionen der Eingänge CU oder CD zu erkennen. Um schneller zu zählen, nutzen Sie die Anweisung CTRL_HSC (Seite 465).

Hinweis

Wenn Sie Zähleranweisungen in einem FB platzieren, können Sie die Option "Multiinstanz-DB" auswählen. Die Namen der Zählerstrukturen können bei verschiedenen Datenstrukturen unterschiedlich sein, doch die Zählerdaten befinden sich in einem einzigen DB, und es ist nicht für jeden Zähler ein eigener DB erforderlich. Dadurch verringert sich die Verarbeitungszeit und der für die Zähler erforderliche Datenspeicher. Zwischen den Datenstrukturen der Zähler im gemeinsam genutzten Multiinstanz-DB gibt es keine Wechselwirkungen.

Funktionsweise der Zähler

Tabelle 8- 25 Funktionsweise von CTU (Vorwärtszählen)

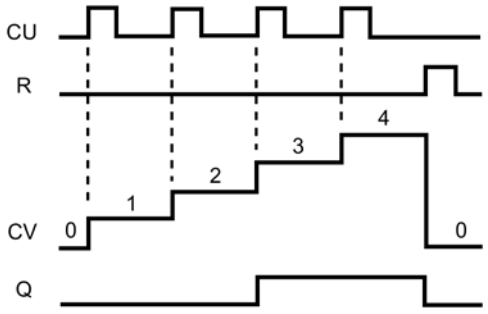
Zähler	Bedienung
Der Zähler CTU zählt um 1 vorwärts, wenn der Wert des Parameters CU von 0 nach 1 wechselt. Das CTU-Zeitdiagramm zeigt die Funktionsweise bei einem vorzeichenlosen ganzzahligen Zählwert (dabei ist PV = 3). • Ist der Wert des Parameters CV (aktueller Zählwert) größer oder gleich dem Wert des Parameters PV (voreingestellter Zählwert), dann lautet der Parameter für den Zählerausgang Q = 1. • Wenn der Wert des Rücksetzparameters R von 0 nach 1 wechselt, wird der aktuelle Zählwert auf 0 zurückgesetzt.	

Tabelle 8- 26 Funktionsweise von CTD (Rückwärtszählen)

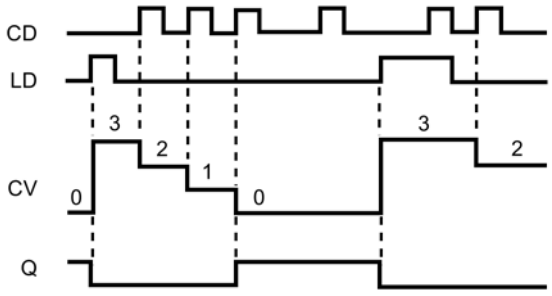
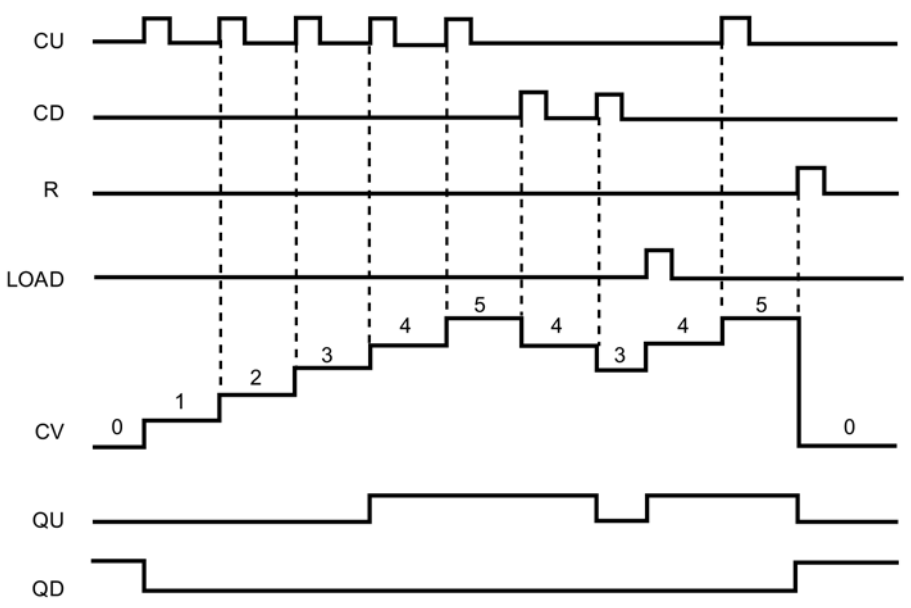
Zähler	Bedienung
Der Zähler CTD zählt um 1 rückwärts, wenn der Wert des Parameters CD von 0 nach 1 wechselt. Das CTD-Zeitdiagramm zeigt die Funktionsweise bei einem vorzeichenlosen ganzzahligen Zählwert (dabei ist PV = 3). • Ist der Wert des Parameters CV (aktueller Zählwert) kleiner oder gleich 0, so lautet der Parameter für den Zählerausgang Q = 1. • Wechselt der Wert von Parameter LOAD von 0 nach 1, wird der Wert an Parameter PV (voreingestellter Wert) als neuer CV (aktueller Zählwert) in den Zähler geladen.	

Tabelle 8- 27 Funktionsweise von CTUD (Vorwärts- und Rückwärtszählen)

Zähler	Bedienung
Der Zähler CTUD zählt um 1 vorwärts oder rückwärts, wenn der Vorwärtszähleingang (CU) oder der Rückwärtszähleingang (CD) von 0 nach 1 wechselt. Das CTUD-Zeitdiagramm zeigt die Funktionsweise bei ganzzahligem Zählwert ohne Vorzeichen (und PV = 4). • Ist der Wert des Parameters CV größer oder gleich dem Wert des Parameters PV, dann lautet der Parameter für den Zählerausgang QU = 1. • Ist der Wert des Parameters CV kleiner oder gleich 0, so lautet der Parameter für den Zählerausgang QD = 1. • Wechselt der Wert von Parameter LOAD von 0 nach 1, wird der Wert an Parameter PV als neuer CV in den Zähler geladen. • Wenn der Wert des Rücksetzparameters R von 0 nach 1 wechselt, wird der aktuelle Zählwert auf 0 zurückgesetzt.	

Speichern von Zählerdaten nach einem RUN-STOP-RUN-Wechsel oder einem Neustart der CPU

Wenn eine Session im Betriebszustand RUN mit dem Betriebszustand STOP oder einem Neustart der CPU beendet wird und eine neue Session im Betriebszustand RUN gestartet wird, gehen die in der vorherigen RUN-Session gespeicherten Zählerdaten verloren, sofern die Zählerdatenstruktur nicht als remanent definiert ist (Zeiten CTU, CTD und CTUD).

Wenn Sie beim Einfügen einer Zähleranweisung im Programmiereditor im Dialog der Aufrufoptionen die Standardeinstellung übernehmen, wird automatisch ein Instanz-DB zugewiesen, der **nicht als remanent definiert werden kann**. Um Ihre Zählerdaten als remanent zu definieren, müssen Sie entweder einen globalen DB oder einen Multiinstanz-DB nutzen.

Zählerdaten durch Zuweisen eines globalen DBs als remanente Daten speichern

Diese Option funktioniert unabhängig davon, wo der Zähler platziert wird (OB, FC oder FB).

1. Globalen DB erstellen:
   - Doppelklicken Sie in der Projektnavigation auf "Neuen Baustein hinzufügen".
   - Klicken Sie auf das Symbol für Datenbaustein (DB).
   - Wählen Sie globalen DB als Typ.
   - Wenn Sie einzelne Elemente im DB als remanent definieren möchten, müssen Sie das Kontrollkästchen "Nur symbolisch adressierbar" aktivieren.
   - Klicken Sie auf "OK".
2. Zählerstruktur(en) zum DB hinzufügen:
   - Fügen Sie im neuen globalen DB eine neue statische Variable vom Datentyp Zähler ein. Berücksichtigen Sie dabei den Typ für die Voreinstellung und den Zählerwert.
   - Aktivieren Sie das Kontrollkästchen in der Spalte "Remanent", damit diese Struktur remanent ist.
   - Wiederholen Sie diesen Vorgang, um für alle Zähler, die Sie in diesem DB speichern möchten, Strukturen anzulegen. Sie können entweder jede Zählerstruktur in einem eindeutigen globalen DB anordnen, oder Sie können mehrere Zählerstrukturen in demselben globalen DB platzieren. Sie können in diesem globalen DB außer Zählern auch andere statische Variablen anordnen. Durch Platzieren mehrerer Zählerstrukturen in demselben globalen DB können Sie die Gesamtzahl der Bausteine verringern.
   - Benennen Sie die Zählerstrukturen ggf. um.
3. Öffnen Sie den Programmbaustein (OB, FC oder FB), in den Sie einen remanenten Zähler einfügen möchten, für die Bearbeitung.
4. Platzieren Sie die Zähleranweisung an der gewünschten Stelle.
5. Wenn der Dialog mit den Aufrufoptionen angezeigt wird, klicken Sie auf "Abbrechen". Nun sollte die neue Zähleranweisung mit dem Platzhalter "???" oberhalb und unterhalb des Anweisungsnamens angezeigt werden.
6. Geben Sie oben in der neuen Zähleranweisung den Namen des globalen DBs und der Zählerstruktur ein, die Sie oben erstellt haben (wählen Sie keinen Namen aus). Beispiel: "Data_block_3.Static_1". Daraufhin wird der entsprechende Typ für die Voreinstellung und den Zählerwert vorgegeben (Beispiel: UInt für eine IEC_UCounter-Struktur).

Zählerdatentyp	Entsprechender Typ für die Voreinstellung und den Zählerwert
IEC_Counter	INT
IEC_SCounter	SINT
IEC_DCounter	DINT
IEC_UCounter	UINT
IEC_USCounter	USINT
IEC_UDCounter	UDINT

Zählerdaten durch Zuweisen eines Multiinstanz-DB als remanente Daten speichern

Diese Möglichkeit funktioniert nur, wenn Sie den Zähler in einem FB ablegen.

Diese Variante ist davon abhängig, ob in den FB-Eigenschaften "Optimierter Bausteinzugriff" festgelegt ist (nur die symbolische Adressierung ist möglich). Um zu prüfen, wie das Zugriffsattribut eines vorhandenen FB konfiguriert ist, klicken Sie mit der rechten Maustaste in der Projektnavigation auf den FB, wählen "Eigenschaften" und dann "Attribute".

Wenn für den FB "Optimierter Bausteinzugriff" festgelegt ist (nur die symbolische Adressierung ist möglich):

1. Öffnen Sie den FB zum Bearbeiten.
2. Platzieren Sie die Zähleranweisung an der gewünschten Stelle im FB.
3. Wenn der Dialog mit den Aufrufoptionen angezeigt wird, klicken Sie auf das Multiinstanz-Symbol. Die Multiinstanz-Option ist nur verfügbar, wenn die Anweisung in einen FB eingefügt wird.
4. Geben Sie dem Zähler im Dialog der Aufrufoptionen ggf. einen Namen.
5. Klicken Sie auf "OK". Die Zähleranweisung erscheint im Editor mit dem Typ INT für die Voreinstellung und den Zählwert, und die IEC_COUNTER-Struktur erscheint in der FB-Schnittstelle unter "Statisch".
6. Ändern Sie ggf. den Typ in der Zähleranweisung von INT in einen anderen Typ. Die Zählerstruktur wird entsprechend verändert.
7. Öffnen Sie ggf. den FB-Schnittstelleneditor (Sie müssen möglicherweise auf den kleinen Pfeil klicken, um die Ansicht zu vergrößern).
8. Suchen Sie unter "Statisch" die Zählerstruktur, die gerade für Sie angelegt wurde.
9. Ändern Sie in der Spalte "Remanent" dieser Zählerstruktur die Auswahl in "Remanent". Immer wenn dieser FB später in einem anderen Programmbaustein aufgerufen wird, wird ein Instanz-DB mit dieser Schnittstellendefinition angelegt, bei der die Zählerstruktur als remanent definiert ist.

Ist die Option "Optimierter Bausteinzugriff" für den FB nicht angegeben, wird der Standardbausteinzugriff verwendet, der mit klassischen S7-300/400 Konfigurationen kompatibel ist und symbolischen sowie direkten Zugriff gestattet. Um einem FB mit Standardbausteinzugriff eine Multiinstanz zuzuweisen, gehen Sie wie folgt vor:

1. Öffnen Sie den FB zum Bearbeiten.
2. Platzieren Sie die Zähleranweisung an der gewünschten Stelle im FB.
3. Wenn der Dialog mit den Aufrufoptionen angezeigt wird, klicken Sie auf das Multiinstanz-Symbol. Die Multiinstanz-Option ist nur verfügbar, wenn die Anweisung in einen FB eingefügt wird.
4. Geben Sie dem Zähler im Dialog der Aufrufoptionen ggf. einen Namen.
5. Klicken Sie auf "OK". Die Zähleranweisung erscheint im Editor mit dem Typ INT für die Voreinstellung und den Zählwert, und die IEC_COUNTER-Struktur erscheint in der FB-Schnittstelle unter "Statisch".
6. Ändern Sie ggf. den Typ in der Zähleranweisung von INT in einen anderen Typ. Die Zählerstruktur wird entsprechend verändert.
7. Öffnen Sie den Baustein, der diesen FB verwenden soll.

8. Platzieren Sie diesen FB an der gewünschten Stelle. Dadurch wird ein Instanz-Datenbaustein für diesen FB angelegt.
9. Öffnen Sie den Instanz-Datenbaustein, der angelegt wurde, als Sie den FB im Editor platziert haben.
10. Suchen Sie unter "Statisch" die gewünschte Zählerstruktur. Aktivieren Sie in der Spalte "Remanent" dieser Zählerstruktur das Kontrollkästchen, um diese Struktur als remanent zu definieren.

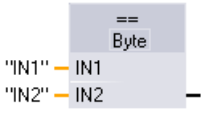
Typ in Zähleranweisung (für Voreinstellung und Zählwert)	Entsprechender Struktur-Typ in FB-Schnittstelle
INT	IEC_Counter
SINT	IEC_SCounter
DINT	IEC_DCounter
UINT	IEC_UCounter
USINT	IEC_USCounter
UDINT	IEC_UDCounter

8.4 Funktionsweise von Vergleichern

8.4.1 Vergleichsoperationen

Tabelle 8- 28 Vergleichsoperationen

KOP	FUP	SCL	Beschreibung

		`out := in1 = in2;`
		or
		`IF in1 = in2`
		`THEN out := 1;`
		`ELSE out := 0;`
		`END_IF;`
			Vergleicht zwei Werte desselben Datentyps. Hat der KOP-Kontaktvergleich das Ergebnis WAHR, wird der Kontakt aktiviert. Ist das Ergebnis des FUP-Boxvergleichs WAHR, dann ist der Box-Ausgang ebenfalls WAHR.

[1] In KOP und FUP: Klicken Sie auf den Anweisungsnamen (wie "=="), um den Vergleichstyp über die Klappliste zu ändern. Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 29 Datentypen für die Parameter

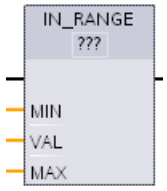
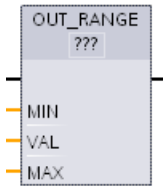
Parameter	Datentyp	Beschreibung
IN1, IN2	Byte, Word, DWord, SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, String, ,WString, Char, Char, Time, Date, TOD, DTL, Konstante	Zu vergleichende Werte

Tabelle 8- 30 Beschreibungen der Vergleiche

Beziehungstyp	Der Vergleich ist wahr, wenn...
=	IN1 gleich IN2 ist
<>	IN1 nicht gleich IN2 ist
>=	IN1 größer oder gleich IN2 ist
<=	IN1 kleiner oder gleich IN2 ist
>	IN1 größer als IN2 ist
<	IN1 kleiner als IN2 ist

8.4.2 Anweisungen IN_Range (Wert innerhalb Bereich) und OUT_Range (Wert außerhalb Bereich)

Tabelle 8- 31 Anweisungen Wert innerhalb eines Bereichs und Wert außerhalb eines Bereichs

KOP/FUP	SCL	Beschreibung
	`out := IN_RANGE(min, val, max);`	Prüft, ob ein Eingabewert innerhalb oder außerhalb eines angegebenen Wertebereichs liegt. Ist das Ergebnis des Vergleichs WAHR, dann ist der Box-Ausgang ebenfalls WAHR.
	`out := OUT_RANGE(min, val, max);`	

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 32 Datentypen für die Parameter

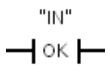
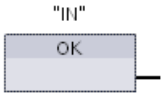
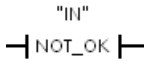
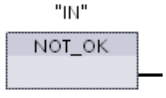
Parameter	Datentyp[1]	Beschreibung
MIN, VAL, MAX	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Konstante	Vergleichereingänge

[1] Die Eingangsparameter MIN, VAL und MAX müssen denselben Datentyp haben.

- Der Vergleich IN_RANGE ist wahr, wenn: MIN <= VAL <= MAX
- Der Vergleich OUT_RANGE ist wahr, wenn: VAL < MIN oder VAL > MAX

8.4.3 Anweisungen OK (Gültigkeit prüfen) und NOT_OK (Ungültigkeit prüfen)

Tabelle 8- 33 Anweisungen OK (Gültigkeit prüfen) und NOT OK (Ungültigkeit prüfen)

KOP	FUP	SCL	Beschreibung
		Nicht verfügbar	Prüft, ob eine Eingabedatenreferenz eine gültige Realzahl nach der IEEE-Spezifikation 754 ist.
		Nicht verfügbar	

[1] In KOP und FUP: Ist der KOP-Kontakt WAHR, wird der Kontakt aktiviert und leitet Signalfluss. Ist die FUP-Box WAHR, dann ist der Box-Ausgang WAHR.

Tabelle 8- 34 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	Real, LReal	Eingangsdaten

Tabelle 8- 35 Funktionsweise

Anweisung	Die Prüfung auf eine reale Zahl ist WAHR, wenn:
OK	Der Eingabewert ist eine gültige Realzahl[1].
NOT_OK	Der Eingabewert ist keine gültige Realzahl[1].

[1] Ein Wert Real oder LReal ist ungültig, wenn er +/- INF (unendlich), NaN (Not a Number, keine Zahl) oder ein denormalisierter Wert ist. Ein denormalisierter Wert ist eine Zahl sehr nah bei Null. In Berechnungen ersetzt die CPU einen denormalisierten Wert durch Null.

8.4.4 Variant- und Array-Vergleichsoperationen

8.4.4.1 Gleich- und Ungleich-Vergleichsoperationen

Die S7-1200 CPU bietet Anweisungen für die Abfrage des Datentyps einer Variablen, auf die ein Variant-Operand zeigt und für den Vergleich, ob diese mit dem Datentyp des anderen Operanden übereinstimmt (gleich) oder nicht (ungleich).

Darüber hinaus bietet die S7-1200 CPU Anweisungen für die Abfrage des Datentyps eines Array-Elements auf Gleichheit oder Ungleichheit mit dem Datentyp des anderen Operanden.

Bei diesen Operationen wird <Operand1> mit <Operand2> verglichen. <Operand1> muss Datentyp Variant aufweisen. <Operand2> kann ein elementarer Datentyp eines PLC-Datentyps sein. In KOP und FUP ist <Operand1> der Operand über der Anweisung. In KOP ist <Operand2> der Operand unter der Anweisung.

Für alle Anweisungen ist das Verknüpfungsergebnis (VKE) 1 (wahr), wenn der Gleich- oder Ungleich-Vergleich erfolgreich war, andernfalls ist es 0 (falsch).

Die Gleich- und Ungleich-Vergleichsoperationen sind nachfolgend beschrieben:

Tabelle 8- 36 EQ_Type (Vergleiche, ob Datentyp GLEICH dem Datentyp einer Variablen ist)
NE_Type (Vergleiche, ob Datentyp UNGLEICH dem Datentyp einer Variablen ist)
EQ_ElemType (Vergleiche, ob Datentyp eines ARRAY-Elements GLEICH dem Datentyp einer Variablen ist)
NE_ElemType (Vergleiche, ob Datentyp eines ARRAY-Elements UNGLEICH dem Datentyp einer Variablen ist)

KOP	FUP	SCL	Beschreibung
#Operand1 EQ_Type "Operand2"	#Operand1 EQ_Type "Operand2" — IN2 OUT	Nicht verfügbar	Prüft, ob die Variable, auf die in Operand1 gezeigt wird, den gleichen Datentyp wie die Variable an Operand2 hat.
#Operand1 NE_Type "Operand2"	#Operand1 NE_Type "Operand2" — IN2 OUT	Nicht verfügbar	Prüft, ob die Variable, auf die in Operand1 gezeigt wird, einen anderen Datentyp als die Variable an Operand2 hat.
#Operand1 EQ_ElemType "Operand2"	#Operand1 EQ_ElemType "Operand2" — IN2 OUT	Nicht verfügbar	Prüft, ob das Array-Element, auf das in Operand1 gezeigt wird, den gleichen Datentyp wie die Variable an Operand2 hat.
#Operand1 NE_ElemType "Operand2"	#Operand1 NE_ElemType "Operand2" — IN2 OUT	Nicht verfügbar	Prüft, ob das Array-Element, auf das in Operand1 gezeigt wird, einen anderen Datentyp als die Variable an Operand2 hat.

Tabelle 8- 37 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
Operand1	Variant	Erster Operand
Operand2	Datentypen Bitzeichenfolgen, Ganzzahlen, Gleitpunktzahlen, Zeiten, Datum und Uhrzeit, Zeichenfolgen, ARRAY, PLC	Zweiter Operand

8.4.4.2 Null-Vergleichsoperationen

Die Anweisungen IS_NULL und NOT_NULL können verwendet werden, um festzustellen, ob der Eingang auf ein Objekt zeigt oder nicht.

Für beide Anweisungen muss <Operand> den Datentyp Variant haben.

Die Null-Vergleichsoperationen sind nachfolgend beschrieben:

Tabelle 8- 38 IS_NULL (Abfrage nach Pointer GLEICH NULL)
NOT_NULL (Abfrage nach Pointer UNGLEICH NULL)

KOP	FUP	SCL	Beschreibung
#Operand IS_NULL	#Operand IS_NULL OUT	Nicht verfügbar	Prüft, ob die Variable, auf die das Element Variant an Operand zeigt, Null und somit kein Objekt ist.
#Operand NOT_NULL	#Operand NOT_NULL OUT	Nicht verfügbar	Prüft, ob die Variable, auf die Element Variant an Operand zeigt, nicht Null und somit ein Objekt ist.

Tabelle 8- 39 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
Operand	Variant	Operand für die Prüfung auf Null oder nicht Null.

8.4.4.3 IS_ARRAY (Auf ARRAY prüfen)

Mit Anweisung "Auf ARRAY prüfen" kann abgefragt werden, ob Element Variant auf eine Variable des Datentyps Array zeigt.

Der <Operand> muss Datentyp Variant haben.

Die Anweisungen geben 1 (wahr) zurück, wenn der Operand ein Array ist.

Tabelle 8- 40 IS_ARRAY (Auf ARRAY prüfen)

KOP	FUP	SCL	Beschreibung
#Operand IS_ARRAY	#Operand IS_ARRAY OUT	IS_ARRAY(_variant_in_)	Prüft, ob die Variable, auf die Element Variant an Operand zeigt, ein Array ist.

Tabelle 8- 41 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
Operand	Variant	Operand zur Prüfung, ob es sich um einen Array handelt.

8.5 Arithmetische Funktionen

8.5.1 Anweisung CALCULATE (Berechnen)

Tabelle 8- 42 Anweisung CALCULATE

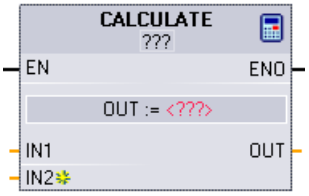
KOP/FUP	SCL	Beschreibung
	Verwenden Sie die herkömmlichen mathematischen SCL-Ausdrücke, um die Gleichung zu erstellen.	Mit der Anweisung CALCULATE können Sie eine arithmetische Funktion erstellen, die an Eingängen ausgeführt wird (IN1, IN2, .. INn) und das Ergebnis an OUT zurückgibt, entsprechend der von Ihnen definierten Gleichung. • Wählen Sie zunächst einen Datentyp aus. Alle Eingänge und der Ausgang müssen denselben Datentyp haben. • Um einen weiteren Eingang hinzuzufügen, klicken Sie auf das Symbol am letzten Eingang.

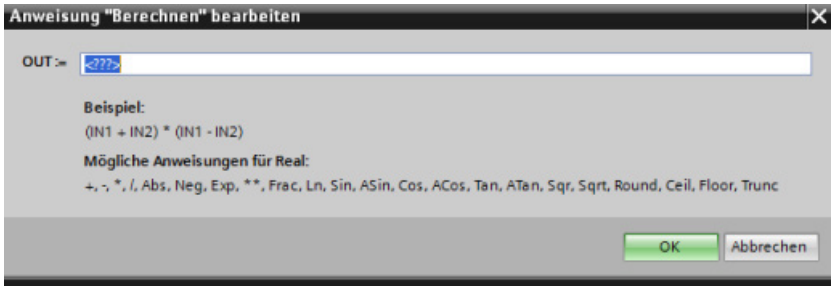
Tabelle 8- 43 Datentypen für die Parameter

Parameter	Datentyp[1]
IN1, IN2, ..INn	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord

[1] Die Parameter IN und OUT müssen denselben Datentyp haben (mit impliziten Umwandlungen der Eingangsparameter). Beispiel: Ein Wert vom Typ SINT für einen Eingang würde in einen Wert vom Typ INT oder REAL umgewandelt werden, wenn OUT vom Typ INT oder REAL ist.

Klicken Sie auf das Taschenrechnersymbol, um den Dialog aufzurufen und ihre mathematische Funktion zu definieren. Sie geben Ihre Gleichung als Eingänge (wie IN1 und IN2) und Operationen ein. Wenn Sie auf "OK" klicken, um die Funktion zu speichern, erstellt der Dialog automatisch die Eingänge für die Anweisung CALCULATE.

Dieser Dialog zeigt ein Beispiel und eine Liste möglicher Anweisungen, die Sie basierend auf dem Datentyp des Parameters OUT aufnehmen können:



Hinweis

Sie müssen außerdem einen Eingang für die Konstanten in Ihrer Funktion anlegen. Der konstante Wert wird dann in den zugewiesenen Eingang der Anweisung CALCULATE eingegeben.

Indem Sie Konstanten als Eingänge eingeben, können Sie die Anweisung CALCULATE an andere Stellen in Ihrem Anwenderprogramm kopieren, ohne die Funktion ändern zu müssen. Sie können dann die Werte oder Variablen der Eingänge für die Anweisung ändern, ohne die Funktion zu verändern.

Wenn die Anweisung CALCULATE ausgeführt wird und alle einzelnen Anweisungen in der Berechnung erfolgreich durchgeführt werden, dann ist ENO = 1. Andernfalls ist ENO = 0.

Ein Beispiel der Anweisung CALCULATE finden Sie in "Erstellen einer komplexen Gleichung mit einer einfachen Anweisung (Seite 41)".

8.5.2 Operationen Addieren, Subtrahieren, Multiplizieren und Dividieren

Tabelle 8- 44 Anweisungen Addieren, Subtrahieren, Multiplizieren und Dividieren

KOP/FUP	SCL	Beschreibung

	`out := in1 + in2;`
	`out := in1 - in2;`
	`out := in1 * in2;`
	`out := in1 / in2;`
		• ADD: Addieren (IN1 + IN2 = OUT)
		• SUB: Subtrahieren (IN1 - IN2 = OUT)
		• MUL: Multiplizieren (IN1 * IN2 = OUT)
		• DIV: Dividieren (IN1 / IN2 = OUT)
		Bei einer ganzzahligen Division werden die Nachkommastellen des Quotienten so verkürzt, dass ein ganzzahliger Ausgangswert entsteht.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 45 Datentypen für die Parameter (KOP und FUP)

Parameter	Datentyp[1]	Beschreibung
IN1, IN2	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Konstante	Eingänge der arithmetischen Operation
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Ausgang der arithmetischen Operation

[1] Die Parameter IN1, IN2 und OUT müssen denselben Datentyp haben.

Um einen Eingang ADD oder MUL hinzuzufügen, klicken Sie auf das Symbol "Erstellen" oder an einem der vorhandenen Parameter IN mit der rechten Maustaste auf den Eingangsanschluss und wählen den Befehl "Eingang einfügen".

Um einen Eingang zu löschen, klicken Sie bei einem der vorhandenen Parameter IN mit der rechten Maustaste auf den Eingangsanschluss (sofern mehr als die zwei ursprünglichen Eingänge vorhanden sind) und wählen den Befehl "Löschen".

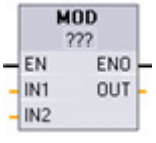
Die arithmetische Anweisung führt, wenn sie aktiviert ist (EN = 1), die angegebene Funktion für die Eingangswerte (IN1 und IN2) aus und speichert das Ergebnis in der vom Ausgangsparameter (OUT) angegebenen Speicheradresse. Nachdem die Operation erfolgreich ausgeführt ist, wird ENO = 1 gesetzt.

Tabelle 8- 46 ENO-Status

ENO	Beschreibung
1	Kein Fehler
0	Der resultierende Wert der arithmetischen Operation liegt außerhalb des gültigen Zahlenbereichs für den ausgewählten Datentyp. Der niederwertigste Teil des Ergebnisses, der in die Zielgröße passt, wird zurückgegeben.
0	Division durch 0 (IN2 = 0): Das Ergebnis ist undefiniert und Null wird zurückgegeben.
0	Real/LReal: Ist einer der Werte NaN (not a number, keine Zahl), so wird der Wert NaN zurückgegeben
0	ADD Real/LReal: Sind beide Eingangswerte INF (unendlich) mit unterschiedlichen Vorzeichen, ist diese Operation ungültig und es wird als Ergebnis NaN zurückgegeben.
0	SUB Real/LReal: Sind beide Eingangswerte INF (unendlich) mit demselben Vorzeichen, ist diese Operation ungültig und es wird als Ergebnis NaN zurückgegeben.
0	MUL Real/LReal: Ist ein Eingangswert Null und der andere INF, ist diese Operation ungültig und es wird als Ergebnis NaN zurückgegeben.
0	DIV Real/LReal: Sind beide Eingangswerte Null oder INF, ist diese Operation ungültig und es wird als Ergebnis NaN zurückgegeben.

8.5.3 Anweisung MOD (Divisionsrest einer Division)

Tabelle 8- 47 Anweisung Modulo (Divisionsrest einer Division)

KOP/FUP	SCL	Beschreibung
	`out := in1 MOD in2;`	Mit der Anweisung MOD können Sie den Divisionsrest einer Ganzzahldivision ausgeben. Der Wert am Eingang IN1 wird durch den Wert am Eingang IN2 dividiert, und der Divisionsrest wird am Ausgang OUT ausgegeben.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 48 Datentypen für die Parameter

Parameter	Datentyp[1]	Beschreibung
IN1 und IN2	SInt, Int, DInt, USInt, UInt, UDInt, Konstante	Modulo-Eingänge
OUT	SInt, Int, DInt, USInt, UInt, UDInt	Modulo-Ausgang

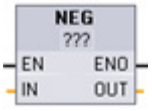
[1] Die Parameter IN1, IN2 und OUT müssen denselben Datentyp haben.

Tabelle 8- 49 ENO-Werte

ENO	Beschreibung
1	Kein Fehler
0	Wert IN2 = 0, OUT wird der Wert Null zugewiesen

8.5.4 Anweisung NEG (Zweierkomplement erzeugen)

Tabelle 8- 50 Anweisung NEG (Zweierkomplement erstellen)

KOP/FUP	SCL	Beschreibung
	`-(in);`	Mit der Anweisung NEG wird das arithmetische Vorzeichen des Werts von Parameter IN umgekehrt und das Ergebnis im Parameter OUT gespeichert.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 51 Datentypen für die Parameter

Parameter	Datentyp[1]	Beschreibung
IN	SInt, Int, DInt, Real, LReal, Konstante	Eingang der arithmetischen Operation
OUT	SInt, Int, DInt, Real, LReal	Ausgang der arithmetischen Operation

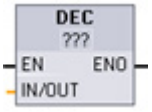
[1] Die Parameter IN und OUT müssen denselben Datentyp haben.

Tabelle 8- 52 ENO-Status

ENO	Beschreibung
1	Kein Fehler
0	Der resultierende Wert liegt außerhalb des gültigen Zahlenbereichs für den ausgewählten Datentyp. Beispiel für SInt: NEG (-128) resultiert in +128, was das Maximum für den Datentyp überschreitet.

8.5.5 Anweisungen INC (Inkrementieren) und DEC (Dekrementieren)

Tabelle 8- 53 Anweisungen INC und DEC

KOP/FUP	SCL	Beschreibung
	`in_out := in_out + 1;`	Erhöht einen ganzzahligen Wert mit oder ohne Vorzeichen: IN_OUT -Wert +1 = IN_OUT -Wert
	`in_out := in_out - 1;`	Verringert einen ganzzahligen Wert mit oder ohne Vorzeichen: IN_OUT -Wert - 1 = IN_OUT -Wert

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 54 Datentypen für die Parameter

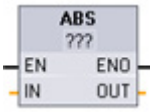
Parameter	Datentyp	Beschreibung
IN/OUT	SInt, Int, DInt, USInt, UInt, UDInt	Ein- und Ausgang der arithmetischen Operation

Tabelle 8- 55 ENO-Status

ENO	Beschreibung
1	Kein Fehler
0	Der resultierende Wert liegt außerhalb des gültigen Zahlenbereichs für den ausgewählten Datentyp. Beispiel für SInt: INC (+127) ergibt +128, was das Maximum für den Datentyp überschreitet.

8.5.6 Anweisung ABS (Absolutwert bilden)

Tabelle 8- 56 Anweisung ABS (Absolutwert bilden)

KOP/FUP	SCL	Beschreibung
	`out := ABS(in);`	Berechnet den Absolutwert einer vorzeichenbehafteten Ganzzahl oder Realzahl an Parameter IN und speichert das Ergebnis in Parameter OUT.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 57 Datentypen für die Parameter

Parameter	Datentyp[1]	Beschreibung
IN	SInt, Int, DInt, Real, LReal	Eingang der arithmetischen Operation
OUT	SInt, Int, DInt, Real, LReal	Ausgang der arithmetischen Operation

[1] Die Parameter IN und OUT müssen denselben Datentyp haben.

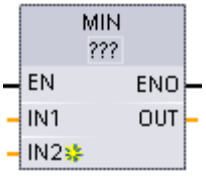
Tabelle 8- 58 ENO-Status

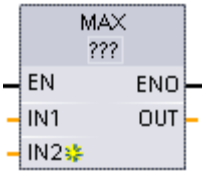
ENO	Beschreibung
1	Kein Fehler
0	Der resultierende Wert der arithmetischen Operation liegt außerhalb des gültigen Zahlenbereichs für den ausgewählten Datentyp. Beispiel für SInt: ABS (-128) resultiert in +128, was das Maximum für den Datentyp überschreitet.

8.5.7 Anweisungen MIN (Minimum ermitteln) und MAX (Maximum ermitteln)

Tabelle 8- 59 Anweisungen MIN (Minimum abrufen) und MAX (Maximum abrufen)

KOP/FUP	SCL	Beschreibung

	`out:= MIN(`
	`in1:=_variant_in_,`
	`in2:=_variant_in_`
	`[,...in32]);`
		Die Anweisung MIN vergleicht den Wert zweier Parameter IN1 und IN2 und weist den kleineren Wert dem Parameter OUT zu.

	`out:= MAX(`
	`in1:=_variant_in_,`
	`in2:=_variant_in_`
	`[,...in32]);`
		Die Anweisung MAX vergleicht den Wert zweier Parameter IN1 und IN2 und weist den größeren Wert dem Parameter OUT zu.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 60 Datentypen für die Parameter

Parameter	Datentyp[1]	Beschreibung
IN1, IN2 [...IN32]	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, Date, TOD, Konstante	Eingänge der arithmetischen Operation (max. 32 Eingänge)
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, Date, TOD	Ausgang der arithmetischen Operation

[1] Die Parameter IN1, IN2 und OUT müssen denselben Datentyp haben.

Um einen Eingang hinzuzufügen, klicken Sie auf das Symbol "Erstellen" oder an einem der vorhandenen Parameter IN mit der rechten Maustaste auf den Eingangsanschluss und wählen den Befehl "Eingang einfügen".

Um einen Eingang zu löschen, klicken Sie bei einem der vorhandenen Parameter IN mit der rechten Maustaste auf den Eingangsanschluss (sofern mehr als die zwei ursprünglichen Eingänge vorhanden sind) und wählen den Befehl "Löschen".

Tabelle 8- 61 ENO-Status

ENO	Beschreibung
1	Kein Fehler
0	Nur beim Datentyp Real: • Mindestens ein Eingang ist keine Realzahl (NaN). • Der resultierende Ausgang OUT ist +/- INF (unendlich).

8.5.8 Anweisung LIMIT (Limitieren)

Tabelle 8- 62 Anweisung LIMIT (Grenzwert setzen)

KOP/FUP	SCL	Beschreibung

	`LIMIT(MN:=_variant_in_,`
	`IN:=_variant_in_,`
	`MX:=_variant_in_,`
	`OUT:=_variant_out_);`
		Mit der Anweisung Limit können Sie prüfen, ob der Wert von Parameter IN innerhalb des mit Parameter MIN und MAX and if not, clamps the value at MIN or MAX. vorgegebenen Wertebereichs liegt.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 63 Datentypen für die Parameter

Parameter	Datentyp[1]	Beschreibung
MN, IN und MX	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, Date, TOD· Konstante	Eingänge der arithmetischen Operation
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, Date, TOD	Ausgang der arithmetischen Operation

[1] Die Parameter MN, IN, MX und OUT müssen denselben Datentyp haben.

Liegt der Wert von Parameter IN innerhalb des angegebenen Bereichs, so wird der Wert für IN in Parameter OUT gespeichert. Liegt der Wert von Parameter IN außerhalb des angegebenen Bereichs, wird in OUT der Wert von Parameter MIN (wenn der Wert IN kleiner als der Wert MIN ist) oder der Wert von Parameter MAX (wenn der Wert IN größer als der Wert MAX ist) ausgegeben.

Tabelle 8- 64 ENO-Status

ENO	Beschreibung
1	Kein Fehler
0	Real: Ist einer der Werte für MIN, IN oder MAX keine Zahl (NaN), wird NaN zurückgegeben.
0	Ist MIN größer als MAX, wird der Wert IN dem Ausgang OUT zugewiesen.

SCL-Beispiele:

- MyVal := LIMIT(MN:=10,IN:=53, MX:=40); //Ergebnis: MyVal = 40
- MyVal := LIMIT(MN:=10,IN:=37, MX:=40); //Ergebnis: MyVal = 37
- MyVal := LIMIT(MN:=10,IN:=8, MX:=40); //Ergebnis: MyVal = 10

8.5.9 Exponential-, Logarithmus- und Trigonometrie-Anweisungen

Mit den Gleitpunktanweisungen können Sie arithmetische Funktionen mit dem Datentyp Real oder LReal programmieren:

- SQR: Quadrat bilden ($IN^2 = OUT$)
- SQRT: Quadratwurzel bilden ($\sqrt{IN} = OUT$)
- LN: Natürlichen Logarithmus bilden (LN(IN = OUT)
- EXP: Exponentialwert bilden ($e^{IN} = OUT$), mit der Basis e = 2,71828182845904523536
- EXPT: Potenzieren ($IN1^{IN2} = OUT$)

  Die EXPT-Parameter IN1 und OUT haben immer den gleichen Datentyp, für den Sie Real oder LReal auswählen müssen. Für den Exponentialparameter IN2 können Sie den Datentyp aus vielen Datentypen wählen.
- FRAC: Nachkommastellen zurückgeben (Nachkommastelle der Gleitpunktzahl IN = OUT)
- SIN: Sinuswert bilden (sin(IN radians) = OUT)
- ASIN: Arcussinuswert bilden (arcsine(IN) = OUT radians), wobei sin(OUT radians) = IN
- COS: Cosinuswert bilden (cos(IN radians) = OUT)
- ACOS: Arcuscosinuswert bilden (arcsine(IN) = OUT radians), wobei cos(OUT radians) = IN
- TAN: Tangenswert bilden (tan(IN radians) = OUT)
- ATAN: Arcustangenswert bilden (arctan(IN) = OUT radians), wobei tan(OUT radians) = IN

Tabelle 8- 65 Beispiele für arithmetische Gleitpunktanweisungen

KOP/FUP	SCL	Beschreibung

	`out := SQR(in);`
	oder
	`out := in * in;`
		Quadrat: IN ^{2} = OUT
		Beispiel: Wenn IN = 9, dann OUT = 81.

	`out := in1 ** in2;`	Potenzieren: IN1 ^{IN2} = OUT Beispiel: Wenn IN1 = 3 und IN2 = 2, dann OUT = 9.

[1] In KOP und FUP: Klicken Sie auf "???" (beim Anweisungsnamen) und wählen Sie den Datentyp in der Klappliste aus.

[2] In SCL: Sie können die mathematischen Ausdrücke in SCL auch mit den grundlegenden mathematischen Operatoren erstellen.

Tabelle 8- 66 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN, IN1	Real, LReal, Konstante	Eingänge
IN2	SInt, Int, DInt, USInt, UInt,UDInt, Real, LReal, Konstante	Eingang EXPT-Exponent
OUT	Real, LReal	Ausgänge

Tabelle 8- 67 ENO-Status

ENO	Anweisung	Bedingung	Ergebnis (OUT)
1	Alle	Kein Fehler	Gültiges Ergebnis
0	SQR	Ergebnis überschreitet den gültigen Bereich für Real/LReal	+INF
		IN ist +/- NaN (keine Zahl)	+NaN
	SQRT	IN ist negativ	-NaN
		IN ist +/- INF (unendlich) oder +/- NaN	+/- INF oder +/- NaN
	LN	IN ist 0,0, negativ, -INF oder -NaN	-NaN
		IN ist +INF oder +NaN	+INF oder +NaN
	EXP	Ergebnis überschreitet den gültigen Bereich für Real/LReal	+INF
		IN ist +/- NaN	+/- NaN
	SIN, COS, TAN	IN ist +/- INF oder +/- NaN	+/- INF oder +/- NaN
	ASIN, ACOS	IN ist außerhalb des gültigen Bereichs von -1,0 bis +1,0	+NaN
		IN ist +/- NaN	+/- NaN
	ATAN	IN ist +/- NaN	+/- NaN
	FRAC	IN ist +/- INF oder +/- NaN	+NaN
	EXPT	IN1 ist +INF und IN2 ist nicht -INF	+INF
		IN1 ist negativ oder -INF	+NaN, wenn IN2 = Real/LReal, ansonsten -INF
		IN1 oder IN2 ist +/- NaN	+NaN
		IN1 ist 0,0 und IN2 ist Real/LReal (nur)	+NaN

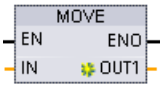
8.6 Anweisungen zum Übertragen von Daten

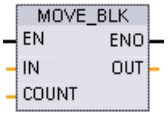
8.6.1 Anweisungen MOVE (Wert kopieren), MOVE_BLK (Bereich kopieren), UMOVE_BLK (Bereich ununterbrechbar kopieren) und MOVE_BLK_VARIANT (Bereich kopieren)

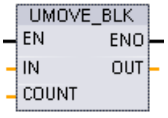
Mit den Übertragungsanweisungen kopieren Sie Datenelemente in eine neue Adresse im Speicher und wandeln die Daten von einem Datentyp in einen anderen um. Die Quelldaten werden dadurch nicht verändert.

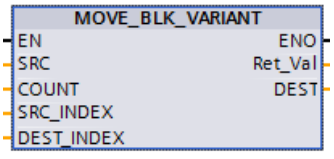
- Mit der Anweisung MOVE wird ein einzelnes Datenelement von der mit Parameter IN angegebenen Quelladresse in die mit Parameter OUT angegebenen Zieladressen kopiert.
- Die Anweisungen MOVE_BLK und UMOVE_BLK verfügen zusätzlich über einen Parameter COUNT. Mit COUNT wird festgelegt, wie viele Datenelemente kopiert werden sollen. Die Anzahl der Bytes pro kopiertem Element hängt davon ab, welcher Datentyp den Variablennamen der Parameter IN und OUT in der PLC-Variablentabelle zugewiesen ist.

Tabelle 8- 68 Anweisungen MOVE, MOVE_BLK, UMOVE_BLK und MOVE_BLK_VARIANT

KOP/FUP	SCL	Beschreibung
	`out1 := in;`	Kopiert ein unter einer bestimmten Adresse gespeichertes Datenelement in eine neue Adresse oder in mehrere Adressen.[1]

	`MOVE_BLK(`
	`  in:=_variant_in,`
	`  count:=_uint_in,`
	`  out=>_variant_out);`
		Unterbrechbare Übertragung, die einen Bereich mit Datenelementen in eine neue Adresse kopiert.

	`UMOVE_BLK(`
	`  in:=_variant_in,`
	`  count:=_uint_in,`
	`  out=>_variant_out);`
		Ununterbrechbare Übertragung, die einen Bereich mit Datenelementen in eine neue Adresse kopiert.

	`MOVE_BLK(`
	`  SRC:=_variant_in,`
	`  COUNT:=_udint_in,`
	`  SRC_INDEX:=_dint_in,`
	
	`DEST_INDEX:=_dint_in,`
	`  DEST=>_variant_out);`
		Kopiert den Inhalt eines Quellspeicherbereichs in einen Zielspeicherbereich.
		Es kann ein komplettes Feld oder Elemente eines Felds in ein anderes Feld mit dem gleichen Datentyp kopiert werden. Die Größen (Anzahl der Elemente) von Quell- und Zielfeld können unterschiedlich sein. Es können mehrere oder einzelne Elemente eines Felds kopiert werden. Es werden Variant-Datentypen verwendet, um auf Quell- und Zielfelder zu zeigen.

[1] Anweisung MOVE: Um einen weiteren Ausgang in KOP oder FUP hinzuzufügen, klicken Sie auf das Symbol "Erstellen" neben dem Ausgangsparameter. In SCL verwenden Sie mehrere Zuweisungsanweisungen. Sie können auch eine der Schleifenkonstruktionen verwenden.

Tabelle 8- 69 Datentypen für die Anweisung MOVE

Parameter	Datentyp	Beschreibung
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Char, WChar, Array, Struct, DTL, Time, Date, TOD, IEC Datentypen, PLC-Datentypen	Quelladresse
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Char, WChar, Array, Struct, DTL, Time, Date, TOD, IEC Datentypen, PLC-Datentypen	Zieladresse

Um Ausgänge MOVE hinzuzufügen, klicken Sie auf das Symbol "Erstellen" oder an einem der vorhandenen Parameter OUT mit der rechten Maustaste auf den Ausgangsanschluss und wählen den Befehl "Ausgang einfügen".

Um einen Ausgang zu löschen, klicken Sie bei einem der vorhandenen Parameter OUT mit der rechten Maustaste auf den Ausgangsanschluss (sofern mehr als die zwei ursprünglichen Ausgänge vorhanden sind) und wählen den Befehl "Löschen".

Tabelle 8- 70 Datentypen für die Anweisungen MOVE_BLK und UMOVE_BLK

Parameter	Datentyp	Beschreibung
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal Byte, Word, DWord, Time, Date, TOD, WChar	Anfangsadresse der Quelle
COUNT	UInt	Anzahl der zu kopierenden Datenelemente
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, WChar	Anfangsadresse des Ziels

Tabelle 8- 71 Datentypen für die Anweisung MOVE_BLK_VARIANT

Parameter	Datentyp	Beschreibung
SRC	Variant (zeigt auf ein Feld oder ein einzelnes Feldelement)	Quellbereich für das Kopieren
COUNT	UDInt	Anzahl der zu kopierenden Datenelemente
SRC_INDEX	DInt	Nullbasierter Index in das SRC-Feld
DEST_INDEX	DInt	Nullbasierter Index in das DEST-Feld
RET_VAL	Int	Fehlerinformationen
DEST	Variant (zeigt auf ein Feld oder ein einzelnes Feldelement)	Zielbereich, in den die Inhalte des Quellbereichs kopiert werden

Hinweis

Regeln für die Anweisungen zum Übertragen von Daten

- Um Daten vom Datentyp Bool zu kopieren, verwenden Sie SET_BF, RESET_BF, R, S oder eineAusgangsspule (KOP) (Seite 220).
- Um Daten eines einzelnen elementaren Datentyps zu kopieren, verwenden Sie MOVE.
- Um ein Feld eines elementaren Datentyps zu kopieren, verwenden Sie MOVE_BLK oder UMOVE_BLK.
- Um eine Struktur zu kopieren, verwenden Sie MOVE.
- Um eine Zeichenkette zu kopieren, verwenden Sie S_MOVE (Seite 333).
- Um ein einzelnes Zeichen in einer Zeichenkette zu kopieren, verwenden Sie MOVE.
- Die Operationen MOVE_BLK und UMOVE_BLK können Sie nicht verwenden, um Felder oder Strukturen in die Speicherbereiche E, A oder M zu kopieren.

Die Operationen MOVE_BLK und UMOVE_BLK unterscheiden sich in der Verarbeitung von Alarmen:

- Alarmereignisse werden **in die Warteschlange gestellt** und während der Ausführung von MOVE_BLK verarbeitet. Die Operation MOVE_BLK nutzen Sie, wenn die Daten an der Zieladresse der Übertragung nicht in einem Unterprogramm eines Alarm-OBs verwendet werden, oder, sofern sie verwendet werden, die Zieldaten nicht konsistent sein müssen. Wenn eine Operation MOVE_BLK unterbrochen wird, dann ist das zuletzt übertragene Datenelement an der Zieladresse vollständig und konsistent. Die Operation MOVE_BLK wird nach Ausführung des Alarm-OBs fortgesetzt.
- Alarmereignisse werden **in die Warteschlange gestellt, aber erst verarbeitet,** wenn die Ausführung von UMOVE_BLK beendet ist. Die Operation UMOVE_BLK nutzen Sie, wenn die Übertragungsoperation beendet und die Zieldaten konsistent sein müssen, bevor das Unterprogramm eines Alarm-OBs ausgeführt wird. Beachten Sie für weitere Informationen den Abschnitt zur Datenkonsistenz (Seite 191).

ENO ist nach der Ausführung der Operation MOVE immer wahr.

Tabelle 8- 72 ENO-Status

ENO	Bedingung	Ergebnis
1	Kein Fehler	Alle COUNT-Elemente wurden erfolgreich kopiert.
0	Der Quellbereich (IN) oder der Zielbereich (OUT) überschreitet den verfügbaren Speicherbereich.	Elemente, die passen, werden kopiert. Es werden keine Teilelemente kopiert.

Tabelle 8- 73 Bedingungscodes für die Anweisung MOVE_BLK_VARIANT

RET_VAL (W#16#...)	Beschreibung
0000	Kein Fehler
80B4	Datentypen passen nicht.
8151	Zugriff auf Parameter SRC nicht möglich.
8152	Der Operand von Parameter SRC hat einen ungültigen Typ.
8153	Fehler bei der Codegenerierung an Parameter SRC
8154	Der Operand von Parameter SRC hat Datentyp Bool.
8281	Parameter COUNT hat einen ungültigen Wert.
8382	Der Wert von Parameter SRC_INDEX liegt außerhalb der Grenzwerte für Variant.
8383	Der Wert von Parameter SRC_INDEX übersteigt den oberen Grenzwerts für das Feld.
8482	Der Wert von Parameter DEST_INDEX liegt außerhalb der Grenzwerte für Variant.
8483	Der Wert von Parameter DEST_INDEX übersteigt den oberen Grenzwert für das Feld.
8534	Parameter DEST ist schreibgeschützt.
8551	Zugriff auf Parameter DEST nicht möglich.
8552	Der Operand von Parameter DEST hat einen ungültigen Typ.
8553	Fehler bei der Codegenerierung an Parameter DEST
8554	Der Operand von Parameter DEST hat Datentyp Bool.
*Feldercodes können im Programmeditor als Ganzzahlen oder Hexadezimalwerte angezeigt werden.	

8.6.2 Deserialize

Mit Anweisung "Deserialize" kann die sequentielle Darstellung eines PLC-Datentyps (UDT) zurück in einen PLC-Datentyp verwandelt und sein gesamter Inhalt aufgefüllt werden. Ist der Vergleich WAHR, ist der Box-Ausgang WAHR.

Der Speicherbereich mit der sequentiellen Darstellung eines PLC-Datentyps muss den Datentyp "Array of Byte" aufweisen und der Datenbaustein muss für Standardzugriff (nicht optimierten Zugriff) deklariert sein. Vor der Umwandlung ist zu prüfen, ob genug Speicherplatz zur Verfügung steht.

Mit dieser Anweisung können mehrere sequentielle Darstellungen umgewandelter PLC-Datentypen in ihre ursprünglichen Datentypen zurückgewandelt werden.

Hinweis

Soll nur eine einzelne sequentielle Darstellung eines PLC-Datentyps (UDT) zurückgewandelt werden, kann Anweisung "TRCV: Daten über Kommunikationsverbindung empfangen" verwendet werden.

Tabelle 8- 74 Anweisung DESERIALIZE

KOP/FUP	SCL	Beschreibung

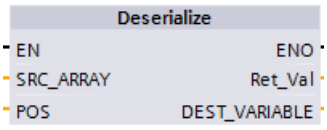
	`ret_val := Deserialize(`
	`  SRC_ARRAY:=_variant_in_,`
	`  DEST_VARIABLE=>_variant_out`
	`_',`
	`  POS:=_dint_inout_);`
		Wandelt die sequentielle Darstellung eines PLC-Datentyps (UDT) zurück in einen PLC-Datentyp und füllt den gesamten Inhalt auf.

Tabelle 8- 75 Parameter für die DESERIALIZE-Anweisung

Parameter	Typ	Datentyp	Beschreibung
SRC_ARRAY	IN	Variant	Globaler Datenbaustein mit dem Datenstrom
DEST_VARIABLE	INOUT	Variant	Variable, in der der umgewandelte PLC-Datentyp (UDT) gespeichert werden soll
POS	INOUT	DInt	Anzahl Bytes, die der umgewandelte PLC-Datentyp verwendet
RET_VAL	OUT	Int	Fehlerinformationen

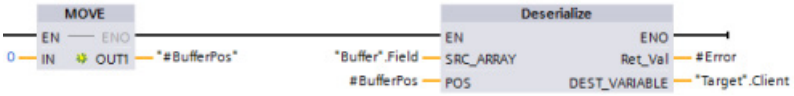
Tabelle 8- 76 Parameter RET_VAL

RET_VAL* (W#16#...)	Beschreibung
0000	Kein Fehler
80B0	Die Speicherbereiche für die Parameter SRC_ARRAY und DEST_VARIABLE überlappen sich.
8136	Der Datenbaustein an Parameter DEST_VARIABLE ist kein Baustein mit Standardzugriff.
8150	Datentyp Variant von Parameter SRC_ARRAY enthält keinen Wert.
8151	Fehler bei der Codegenerierung an Parameter SRC_ARRAY.
8153	Nicht genug freier Speicherplatz für Parameter SRC_ARRAY.
8250	Datentyp Variant von Parameter DEST_VARIABLE enthält keinen Wert.
8251	Fehler bei der Codegenerierung an Parameter DEST_VARIABLE.
8254	Ungültiger Datentyp für Parameter DEST_VARIABLE.
8382	Der Wert von Parameter POS liegt außerhalb der Grenzwerte für das Feld.
*Feldercodes können im Programmeditor als Ganzzahlen oder Hexadezimalwerte angezeigt werden.	

Beispiel: Anweisung "Deserialize"

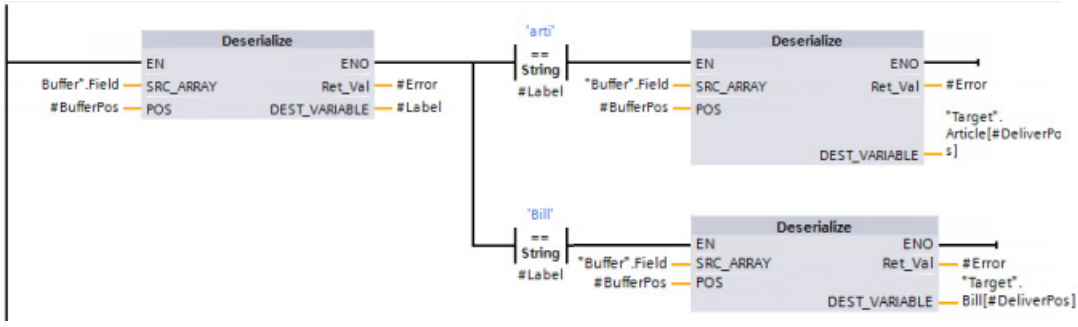
Das folgende Beispiel zeigt, wie die Anweisung funktioniert:

Netzwerk 1:



Mit Anweisung "MOVE" wird Wert "0" in die Variable des Datenbausteins "#BufferPos" kopiert. Mit Anweisung Deserialize wird dann die sequentielle Darstellung der Kundendaten aus dem Datenbaustein "Buffer" deserialisiert und in Datenbaustein "Target" geschrieben. Mit Anweisung Deserialize wird die Anzahl Bytes berechnet, die von den umgewandelten Daten benutzt werden, und in der Variablen des Datenbausteins "#BufferPos" gespeichert.

Netzwerk 2:



Mit Anweisung "Deserialize" wird die sequentielle Darstellung des Datenstroms, auf den mit "Buffer" gezeigt wird, deserialisiert und die Zeichen in Operand "#Label" geschrieben. Die Logik vergleicht die Zeichen mit Hilfe der Vergleichsoperationen "arti" und "Bill". Ist der Vergleich für "arti" = WAHR, sind die Daten Artikeldaten, die deserialisiert und in Datenstruktur "Article" des Datenbausteins "Target" geschrieben werden. Ist der Vergleich für "Bill" = WAHR, sind die Daten Abrechnungsdaten, die deserialisiert und in Datenstruktur "Bill" des Datenbausteins "Target" geschrieben werden.

Funktionsbausteinschnittstelle (oder Funktionsschnittstelle):

		Name	Datentyp
1		Input	
2		DeliverPos	Int
3		Output	
4		InOut	
5		Static	
6		Temp	
7		BufferPos	DInt
8		Error	Int
9		Label	String[4]

Benutzerdefinierte PLC-Datentypen:

Die Strukturen der beiden PLC-Datentypen (UDT) für dieses Beispiel sind nachstehend dargestellt:

Article

	Name	Datentyp
1	Number	DInt
2	Declaration	String
3	Colli	Int

Client

	Name	Datentyp
1	Title	Int
2	Firstname	String[10]
3	Surname	String[10]

Datenbausteine:

Die beiden Datenbausteine für dieses Beispiel sind nachfolgend dargestellt:

Target

	Name	Datentyp
1	Static	
2	Client	"Client"
3	Article	Array[0..10] of "Article"
4	Bill	Array[0..10] of Int

Buffer

	Name	Datentyp
1	Static	
2	Field	Array[0..294] of Byte

8.6.3 Serialize

Mit Anweisung "Serialize" können mehrere PLC-Datentypen (UDT) ohne jeglichen Strukturverlust in eine sequentielle Darstellung umgewandelt werden.

Die Anweisung kann verwendet werden, um mehrere strukturierte Datenelemente aus Ihrem Programm vorübergehend in einem Puffer abzulegen, zum Beispiel einem globalen Datenbaustein, und sie dann in eine andere CPU zu übertragen. Der Speicherbereich, in dem die umgewandelten PLC-Datentypen abgelegt werden, muss Datentyp "ARRAY of BYTE" aufweisen und mit Standardzugriff deklariert sein. Vor der Umwandlung ist zu prüfen, ob genug Speicherplatz zur Verfügung steht.

Der POS-Parameter enthält Informationen über die Anzahl Bytes, die die umgewandelten PLC-Datentypen verwenden.

Hinweis

Soll nur ein einzelner PLC-Datentyp (UDT) übertragen werden, kann dafür Anweisung "TSEND: Daten über Kommunikationsverbindung senden" verwendet werden.

Tabelle 8- 77 Anweisung SERIALIZE

KOP/FUP	SCL	Beschreibung

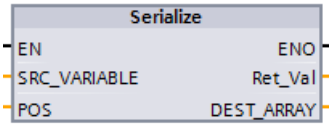
	`ret_val := Serialize(`
	`  SRC_VARIABLE=>_variant_in_,`
	`  DEST_ARRAY:=_variant_out_,`
	`  POS:=_dint_inout_);`
		Wandelt einen PLC-Datentyp (UDT) in eine sequentielle Darstellung um.

Tabelle 8- 78 Parameter für die SERIALIZE-Anweisung

Parameter	Typ	Datentyp	Beschreibung
SRC_VARIABLE	IN	Variant	PLC-Datentyp (UDT), der in eine serielle Darstellung umgewandelt werden soll
DEST_ARRAY	INOUT	Variant	Datenbaustein, in dem der generierte Datenstrom gespeichert werden soll
POS	INOUT	DInt	Anzahl Bytes, die die umgewandelten PLC-Datentypen verwenden. Der berechnete POS-Parameter ist nullbasiert.
RET_VAL	OUT	Int	Fehlerinformationen

Tabelle 8- 79 Parameter RET_VAL

RET_VAL* (W#16#...)	Beschreibung
0000	Kein Fehler
80B0	Die Speicherbereiche für die Parameter SRC_VARIABLE und DEST_ARRAY überlappen sich.
8150	Datentyp Variant von Parameter SRC_VARIABLE enthält keinen Wert.
8152	Fehler bei der Codegenerierung an Parameter SRC_VARIABLE.
8236	Der Datenbaustein an Parameter DEST_ARRAY ist kein Baustein mit Standardzugriff.
8250	Datentyp Variant von Parameter DEST_ARRAY enthält keinen Wert.
8252	Fehler bei der Codegenerierung an Parameter DEST_ARRAY.
8253	Nicht genug freier Speicherplatz für Parameter DEST_ARRAY.
8254	Ungültiger Datentyp für Parameter DEST_VARIABLE.
8382	Der Wert von Parameter POS liegt außerhalb der Grenzwerte für das Feld.
*Feldercodes können im Programmeditor als Ganzzahlen oder Hexadezimalwerte angezeigt werden.		

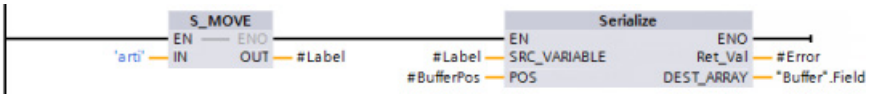
Beispiel: Anweisung "Serialize"

Das folgende Beispiel zeigt, wie die Anweisung funktioniert:

Netzwerk 1:

Mit Anweisung "MOVE" wird Wert "0" in Parameter "#BufferPos" kopiert. Mit Anweisung "Serialize" werden die Kundendaten aus dem Datenbaustein "Source" serialisiert und in sequentieller Darstellung in Datenbaustein "Buffer" geschrieben. Die Anweisung speichert die Anzahl der von der sequentiellen Darstellung verwendeten Bytes in Parameter "#BufferPos".

Netzwerk 2:



Die Logik fügt dann einen Trenntext ein, damit die sequentielle Darstellung später einfacher deserialisiert werden kann. Mit Anweisung "S_MOVE" wird die Zeichenfolge "arti in Parameter "#Label" kopiert. Anweisung "Serialize" schreibt diese Zeichen nach den Quelldaten in Datenbaustein "Buffer". Die Anweisung fügt die Anzahl Bytes in der Textzeichenfolge "arti" zu der in Parameter "#BufferPos" bereits gespeicherten Anzahl hinzu.

Netzwerk 3:

Anweisung "Serialize" serialisiert die Daten eines bestimmten Artikels, die während der Laufzeit berechnet werden, aus Datenbaustein "Source" und schreibt sie in sequentieller Darstellung in Datenbaustein "Buffer" nach den Zeichen "arti".

Bausteinschnittstelle:

		Name	Datentyp
1		Input	
2		DeliverPos	Int
3		Output	
4		InOut	
5		Static	
6		Temp	
7		BufferPos	DInt
8		Error	Int
9		Label	String[4]

Benutzerdefinierte PLC-Datentypen:

Die Strukturen der beiden PLC-Datentypen (UDT) für dieses Beispiel sind nachstehend dargestellt:

Article

	Name	Datentyp
1	Number	DInt
2	Declaration	String
3	Colli	Int

Client

	Name	Datentyp
1	Title	Int
2	Firstname	String[10]
3	Surname	String[10]

Datenbausteine:

Die beiden Datenbausteine für diese Beispiel sind nachfolgend dargestellt:

Source

	Name	Datentyp
1	Static	
2	Client	"Client"
3	Article	Array[0..10] of "Article"

Buffer

	Name	Datentyp
1	Static	
2	Field	Array[0..294] of Byte

8.6.4 FILL_BLK (Bereich befüllen) und UFILL_BLK (Bereich ununterbrechbar befüllen)

Tabelle 8- 80 Anweisungen FILL_BLK und UFILL_BLK

KOP/FUP	SCL	Beschreibung

	`FILL_BLK(`
	`  in:=_variant_in,`
	`  count:=int,`
	`  out=>_variant_out);`
		Anweisung Unterbrechbar befüllen: Befüllt einen Adressbereich mit Kopien eines angegebenen Datenelements.

	`UFILL_BLK(`
	`  in:=_variant_in,`
	`  count:=int`
	`  out=>_variant_out);`
		Anweisung Ununterbrechbar befüllen: Befüllt einen Adressbereich mit Kopien eines angegebenen Datenelements.

Tabelle 8- 81 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Datenquelladresse
COUNT	UDint, USInt, UInt	Anzahl der zu kopierenden Datenelemente
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Datenzieladresse

Hinweis

Regeln für die Anweisungen Speicher mit Bitmuster belegen

- Um Daten vom Datentyp BOOL zu kopieren, verwenden Sie SET_BF, RESET_BF, R, S oder eine Ausgangsspule (KOP).
- Um den Speicher mit einem einzelnen elementaren Datentyp zu belegen, verwenden Sie MOVE.
- Um ein Feld mit einem elementaren Datentyp zu belegen, verwenden Sie FILL_BLK oder UFILL_BLK.
- Um ein einzelnes Zeichen in einer Zeichenkette zu kopieren, verwenden Sie MOVE.
- Die Anweisungen FILL_BLK und UFILL_BLK können Sie nicht verwenden, um Felder in den Speicherbereichen E, A oder M zu belegen.

Mit den Anweisungen FILL_BLK und UFILL_BLK wird das Quelldatenelement IN in die Zieladresse kopiert, wobei die Anfangsadresse im Parameter OUT festgelegt ist. Das Kopieren läuft so lange in den jeweils benachbarten Adressbereichen weiter, bis die Anzahl der Kopien mit dem Wert des Parameters COUNT übereinstimmt.

Die Anweisungen FILL_BLK und UFILL_BLK unterscheiden sich in der Verarbeitung von Alarmen:

- Alarmereignisse werden **in die Warteschlange gestellt** und während der Ausführung von FILL_BLK verarbeitet. Die Anweisung FILL_BLK nutzen Sie, wenn die Daten an der Zieladresse der Übertragung nicht in einem Unterprogramm eines Alarm-OBs verwendet werden, oder, sofern sie verwendet werden, die Zieldaten nicht konsistent sein müssen.
- Alarmereignisse werden **in die Warteschlange gestellt, aber erst verarbeitet,** wenn die Ausführung von UFILL_BLK beendet ist. Die Anweisung UFILL_BLK nutzen Sie, wenn die Übertragungsanweisung beendet und die Zieldaten konsistent sein müssen, bevor das Unterprogramm eines Alarm-OBs ausgeführt wird.

Tabelle 8- 82 ENO-Status

ENO	Bedingung	Ergebnis
1	Kein Fehler	Das Element IN wurde erfolgreich in alle COUNT-Ziele kopiert.
0	Der Zielbereich (OUT) überschreitet den verfügbaren Speicherbereich	Elemente, die passen, werden kopiert. Es werden keine Teilelemente kopiert.

8.6.5 Anweisung SWAP (Anordnung ändern)

Tabelle 8- 83 Anweisung SWAP

KOP/FUP	SCL	Beschreibung
	`out := SWAP(in);`	Kehrt die Bytereihenfolge bei Zwei-Byte- und Vier-Byte-Datenelementen um. Die Bitfolge innerhalb eines Bytes wird nicht geändert. ENO ist nach der Ausführung der Anweisung SWAP immer WAHR.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 84 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	Word, DWord	Normal angeordnete Datenbytes in Parameter IN
OUT	Word, DWord	Vertauschte Reihenfolge der Datenbytes in Parameter OUT

Beispiel 1	Parameter IN = MB0 (vor Ausführung)		Parameter OUT = MB4, (nach Ausführung)	
Adresse	MW0	MB1	MW4	MB5
W#16#1234	12	34	34	12
WORD	MSB	LSB	MSB	LSB

Beispiel 2	Parameter IN = MB0 (vor Ausführung)				Parameter OUT = MB4, (nach Ausführung)			
Adresse	MD0	MB1	MB2	MB3	MD4	MB5	MB6	MB7
DW#16# 12345678	12	34	56	78	78	56	34	12
DWORD	MSB			LSB	MSB			LSB

8.6.6 Anweisungen Speicher lesen / in Speicher schreiben

8.6.6.1 Anweisungen PEEK und POKE (nur SCL)

SCL bietet die Anweisungen PEEK und POKE, mit denen Sie aus Datenblöcken, E/A oder dem Speicher lesen oder in diese schreiben können. Sie geben für die Operation Parameter mit einem spezifischen Byte- oder Bit-Versatz an.

Hinweis

Um die Anweisungen PEEK und POKE mit Datenbausteinen zu verwenden, müssen Sie Standarddatenbausteine (keine optimierten Datenbausteine) verwenden. Beachten Sie zudem, dass die Anweisungen PEEK und POKE lediglich Daten übertragen. Die Datentypen an den Adressen werden nicht beachtet.

```
PEEK(area:=_in_,
     dbNumber:=_in_,
     byteOffset:=_in_);
```

Liest das von byteOffset angegebene Byte aus dem angegebenen Datenbaustein, aus den E/A oder dem Speicherbereich.

Beispiel für den Verweis auf einen Datenbaustein:

```
%MB100 := PEEK(area:=16#84,
dbNumber:=1, byteOffset:=#i);
```

Beispiel für den Verweis auf Eingang EB3:

```
%MB100 := PEEK(area:=16#81,
dbNumber:=0, byteOffset:=#i); // when
#i = 3
```

```
PEEK_WORD(area:=_in_,
     dbNumber:=_in_,
     byteOffset:=_in_);
```

Liest das von byteOffset angegebene Wort aus dem angegebenen Datenbaustein, aus den E/A oder dem Speicherbereich.

Beispiel:

```
%MW200 := PEEK_WORD(area:=16#84,
dbNumber:=1, byteOffset:=#i);
```

```
PEEK_DWORD(area:=_in_,
     dbNumber:=_in_,
     byteOffset:=_in_);
```

Liest das von byteOffset angegebene Doppelwort aus dem angegebenen Datenbaustein, aus den E/A oder dem Speicherbereich.

Beispiel:

```
%MD300 := PEEK_DWORD(area:=16#84,
dbNumber:=1, byteOffset:=#i);
```

```
PEEK_BOOL(area:=_in_,
     dbNumber:=_in_,
     byteOffset:=_in_,
     bitOffset:=_in_);
```

Liest einen von bitOffset und byteOffset angegebenen Booleschen Wert aus dem angegebenen Datenbaustein, aus den E/A oder dem Speicherbereich.

Beispiel:

```
%MB100.0 := PEEK_BOOL(area:=16#84,
dbNumber:=1, byteOffset:=#ii,
bitOffset:=#j);
```

```
POKE(area:=_in_,
     dbNumber:=_in_,
     byteOffset:=_in_,
     value:=_in_);
```

Schreibt den Wert (Byte, Wort oder Doppelwort) in den angegebenen byteOffset des angegebenen Datenbausteins, der E/A oder des Speicherbereichs.

Beispiel für den Verweis auf einen Datenbaustein:

```
POKE(area:=16#84, dbNumber:=2,
byteOffset:=3, value:="Tag_1");
```

Beispiel für den Verweis auf Ausgang AB3:

```
POKE(area:=16#82, dbNumber:=0,
byteOffset:=3, value:="Tag_1");
```

```
POKE_BOOL(area:=_in_,
     dbNumber:=_in_,
     byteOffset:=_in_,
     bitOffset:=_in_,
     value:=_in_);
```

Schreibt den Booleschen Wert in den angegebenen bitOffset und byteOffset des angegebenen Datenbausteins, der E/A oder des Speicherbereichs.

Beispiel:

```
POKE_BOOL(area:=16#84, dbNumber:=2,
byteOffset:=3, bitOffset:=5, value:=0);
```

```
POKE_BLK(area_src:=_in_,
     dbNumber_src:=_in_,
     byteOffset_src:=_in_,
     area_dest:=_in_,
     dbNumber_dest:=_in_,
     byteOffset_dest:=_in_,
     count:=_in_);
```

Schreibt die unter "count" angegebene Anzahl von Bytes beginnend mit dem angegebenen byteOffset des angegebenen Quelldatenbausteins, der Quell-E/A oder des Quellspeicherbereichs in den angegebenen byteOffset des angegebenen Zieldatenbausteins, der Ziel-E/A oder des Zielspeicherbereichs.

Beispiel:

```
POKE_BLK(area_src:=16#84, dbNumber_src:=#src_db, byteOffset_src:=#src_byte, area_dest:=16#84, dbNumber_dest:=#src_db, byteOffset_dest:=#src_byte, count:=10);
```

Bei den Anweisungen PEEK und POKE gelten die folgenden Werte für die Parameter "area", "area_src" und "area_dest". Für andere Bereiche als Datenbausteine muss der Parameter dbNumber 0 sein.

16#81	I
16#82	Q
16#83	M
16#84	DB

8.6.6.2 Anweisungen Big- und Little-Endian-Format lesen und schreiben (SCL)

Die S7-1200 CPU stellt SCL-Anweisungen für das Lesen und Schreiben von Daten in Little-Endian- und Big-Endian-Format bereit. Little-Endian-Format bedeutet, dass das Byte mit dem niederwertigsten Bit an der kleinsten Speicheradresse gespeichert wird. Big-Endian-Format bedeutet, dass das Byte mit dem höchstwertigen Bit an der kleinsten Speicheradresse gespeichert wird.

Die vier SCL-Anweisungen für das Lesen und Schreiben von Daten im Little- und Big-Endian-Format sind nachstehend beschrieben:

- READ_LITTLE (Daten in Little-Endian-Format lesen)
- WRITE_LITTLE (Daten in Little-Endian-Format schreiben)
- READ_BIG (Daten in Big-Endian-Format lesen)
- WRITE_BIG (Daten in Big-Endian-Format schreiben)

Tabelle 8- 85 Anweisungen Big- und Little-Endian-Format lesen und schreiben

KOP/FUP	SCL	Beschreibung

Nicht verfügbar	`READ_LITTLE(`
	`  src_array:=_variant_in_,`
	`  dest_Variable =>_out_,`
	`  pos:=_dint_inout)`
		Liest Daten aus einem Speicherbereich und schreibt sie in eine einzelne Variable im Little-Endian-Byte-Format.

Nicht verfügbar	`WRITE_LITTLE(`
	`  src_variable:=_in_,`
	`  dest_array =>_variant_inout_,`
	`  pos:=_dint_inout)`
		Schreibt Daten aus einer einzelnen Variablen in einen Speicherbereich im Little-Endian-Byte-Format.

Nicht verfügbar	`READ_BIG(`
	`  src_array:=_variant_in_,`
	`  dest_Variable =>_out_,`
	`  pos:=_dint_inout)`
		Liest Daten aus einem Speicherbereich und schreibt sie in eine einzelne Variable im Big-Endian-Byte-Format.

Nicht verfügbar	`WRITE_BIG(`
	`  src_variable:=_in_,`
	`  dest_array =>_variant_inout_,`
	`  pos:=_dint_inout)`
		Schreibt Daten aus einer einzelnen Variablen in einen Speicherbereich im Big-Endian-Byte-Format.

Tabelle 8- 86 Parameter für die READ_LITTLE and READ_BIG -Anweisung

Parameter	Datentyp	Beschreibung
src_array	Array of Byte	Speicherbereich, aus dem Daten gelesen werden sollen
dest_Variable	Bitzeichenfolgen, Ganzzahlen, Gleitpunktzahlen, Zeiten, Datum und Uhrzeit, Zeichenfolgen	Zielvariable, in die Daten zu schreiben sind
pos	DINT	Nullbasierte Position, ab der Daten aus Eingang src_array gelesen werden sollen.

Tabelle 8- 87 Parameter für die WRITE_LITTLE and WRITE_BIG -Anweisung

Parameter	Datentyp	Beschreibung
src_variable	Bitzeichenfolgen, Ganzzahlen, Gleitpunktzahlen, LDT, TOD, LTOD, DATA, Char, WChar	Quelldaten aus Variable
dest_array	Array of Byte	Speicherbereich, in den Daten geschrieben werden sollen
pos	DINT	Nullbasierte Position, ab der Daten in Ausgang dest_array geschrieben werden sollen.

Tabelle 8- 88 Parameter RET_VAL

RET_VAL* (W#16#...)	Beschreibung
0000	Kein Fehler
80B4	SRC_ARRAY oder DEST_ARRAY ist kein Array of Byte
8382	Der Wert von Parameter POS liegt außerhalb der Grenzwerte für das Feld.
8383	Der Wert von Parameter POS liegt innerhalb der Grenzwerte des Felds, aber die Größe des Speicherbereichs überschreitet die Obergrenze des Felds.
*Feldercodes können im Programmeditor als Ganzzahlen oder Hexadezimalwerte angezeigt werden.	

8.6.7 Variant-Anweisungen

8.6.7.1 VariantGet (VARIANT Variablenwert lesen)

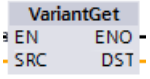
Mit der Anweisung "Variant Variablenwert lesen" kann der Wert der Variablen ausgelesen werden, auf die der Pointer Variant am Parameter SRC zeigt, und in die Variable von Parameter DST geschrieben werden.

Parameter SRC hat Datentyp Variant. Für Parameter DST kann jeder Datentyp außer Variant programmiert werden.

Der Datentyp der Variablen in Parameter DST muss mit dem Datentyp übereinstimmen, auf den Variant zeigt.

Tabelle 8- 89 Anweisung VariantGet

KOP/FUP	SCL	Beschreibung

	`VariantGet(`
	`  SRC:=_variant_in_,`
	`  DST=>_variant_out_);`
		Liest die Variable, auf die Parameter SRC zeigt und schreibt sie in die Variable in Parameter DST.

Hinweis

Um Strukturen und Felder zu kopieren, kann Anweisung "MOVE_BLK_VARIANT: Bereich kopieren" verwendet werden.

Tabelle 8- 90 Parameter für die VariantGet-Anweisung

Parameter	Datentyp	Beschreibung
SRC	Variant	Pointer auf Quelldaten
DST	Datentypen Bitzeichenfolgen, Ganzzahlen, Gleitpunktzahlen, Zeiten, Datum und Uhrzeit, Zeichenfolgen, ARRAY-Elemente, PLC	Ziel, in das die Daten zu schreiben sind

Tabelle 8- 91 ENO-Status

ENO	Bedingung	Ergebnis
1	Kein Fehler	Die Anweisung hat die Variablendaten, auf die SRC zeigt, in Variable DST kopiert.
0	Freigabeeingang EN hat Signalzustand "0" oder Datentypen passen nicht.	Die Anweisung hat keine Daten kopiert.

8.6.7.2 Anweisung VariantPut (VARIANT Variablenwert schreiben)

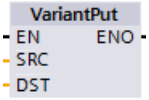
Mit Anweisung "VARIANT Variablenwert schreiben" kann der Wert der Variablen in Parameter SRC in die Variable in Parameter DST geschrieben werden, auf die VARIANT zeigt.

Parameter DST hat Datentyp VARIANT. Für Parameter SRC kann jeder Datentyp außer VARIANT programmiert werden.

Der Datentyp der Variablen in Parameter SRC muss mit dem Datentyp übereinstimmen, auf den VARIANT zeigt.

Tabelle 8- 92 Anweisung VariantPut

KOP/FUP	SCL	Beschreibung

	`VariantPut(`
	`  SRC:=_variant_in_,`
	`  DST=>_variant_in_);`
		Schreibt die mit Parameter SRC referenzierte Variable in das Variant-Element, auf das Parameter DST zeigt.

Hinweis

Um Strukturen und Felder zu kopieren, kann Anweisung "MOVE_BLK_VARIANT: Bereich kopieren" verwendet werden.

Tabelle 8- 93 Parameter für die VariantPut-Anweisung

Parameter	Datentyp	Beschreibung
SRC	Bit strings, integers, floating-point numbers, timers, date and time, character strings, ARRAY elements, PLC data types	Pointer auf Quelldaten
DST	Variant	Ziel, in das die Daten zu schreiben sind

Tabelle 8- 94 ENO-Status

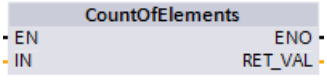
ENO	Bedingung	Ergebnis
1	Kein Fehler	Die Anweisung hat die SRC-Variablendaten in Variable DST kopiert.
0	Freigabeeingang EN hat Signalzustand "0" oder Datentypen passen nicht.	Die Anweisung hat keine Daten kopiert.

8.6.7.3 Anweisung CountOfElements (Anzahl ARRAY-Elemente abfragen)

Mit Anweisung "Anzahl ARRAY-Elemente abfragen" kann abgefragt werden, wie viele Array-Elemente in einer Variablen vorhanden sind, auf die ein Variant-Element zeigt.

Bei einem eindimensionalen ARRAY gibt die Anweisung die Differenz zwischen dem oberen und unteren Grenzwert +1 aus. Bei einem multidimensionalen ARRAY gibt die Anweisung das Produkt aller Dimensionen zurück.

Tabelle 8- 95 Anweisung CountOfElements

KOP/FUP	SCL	Beschreibung
	`Result := CountOfElements( _variant_in_);`	Zählt die Anzahl der Array-Elemente im Array, auf das Parameter IN zeigt.

Hinweis

Zeigt Variant auf ein Array of Bool, zählt die Anweisung die Füllelemente bis zur nächsten Bytegrenze. Für Array[0..1] of Bool wird beispielsweise ein Wert 8 zurückgegeben.

Tabelle 8- 96 Parameter für die CountOfElements-Anweisung

Parameter	Datentyp	Beschreibung
IN	Variant	Variable mit den Array-Elementen, die gezählt werden sollen
RET_VAL	UDint	Anweisungsergebnis

Tabelle 8- 97 ENO-Status

ENO	Bedingung	Ergebnis
1	Kein Fehler	Die Anweisung gibt die Anzahl der Array-Elemente zurück.
0	Freigabeeingang EN hat Signalzustand "0" oder Variant zeigt nicht auf ein Array.	Die Anweisung gibt 0 zurück.

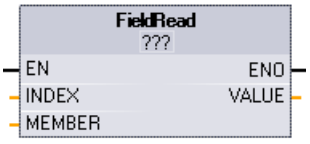
8.6.8 Anweisungen in älteren Systemen

8.6.8.1 Anweisungen FieldRead (Feld lesen) und FieldWrite (Feld schreiben)

Hinweis

In STEP 7 V10.5 wurden eine Variablenreferenz als Array-Index oder multidimensionale Arrays **nicht unterstützt**. Mit den Anweisungen FeldLesen und FeldSchreiben wurden variable Array-Index-Anweisungen für ein eindimensionales Array bereitgestellt. Ab STEP 7 V11 **werden** eine Variablenreferenz als Array-Index und multidimensionale Arrays **unterstützt**. FeldLesen und FeldSchreiben sind ab STEP 7 V11 enthalten und sind abwärts kompatibel für Programme, die diese Anweisungen umfassen.

Tabelle 8- 98 Anweisungen FeldLesen und FeldSchreiben

KOP/FUP	SCL	Beschreibung
	`value := member[index];`	FeldLesen liest das Array-Element mit dem Indexwert INDEX aus dem Array, dessen erstes Element vom Parameter MEMBER angegeben wird. Der Wert des Array-Elements wird an die vom Parameter VALUE angegebene Adresse übertragen.
	`member[index] := value;`	FeldSchreiben überträgt den Wert an der vom Parameter VALUE angegebenen Adresse zu dem Array, dessen erstes Element vom Parameter MEMBER angegeben wird. Der Wert wird zu dem Array-Element übertragen, dessen Array-Index vom Parameter INDEX angegeben wird.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 99 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
Index	Eingang	DInt	Die Indexnummer des Array-Elements, das gelesen oder in das geschrieben werden soll.
Member [1]	Eingang	Binärzahlen, Ganzzahlen, Gleitpunktzahlen, Zeiten, DATE, TOD, CHAR und WCHAR als Komponenten einer ARRAY-Variablen	Adresse des ersten Elements in einem eindimensionalen Array, definiert in einem globalen Datenbaustein oder in einer Bausteinschnittstelle. Beispiel: Wird der Array-Index angegeben als [-2..4], dann ist der Index des ersten Elements -2 und nicht 0.
Value [1]	Ausgang	Binärzahlen, Ganzzahlen, Gleitpunktzahlen, Zeiten, DATE, TOD, CHAR, WCHAR	Adresse, an die das angegebene Array-Element kopiert wird (FeldLesen). Adresse des Werts, der in das angegebene Array-Element kopiert wird (FeldSchreiben).

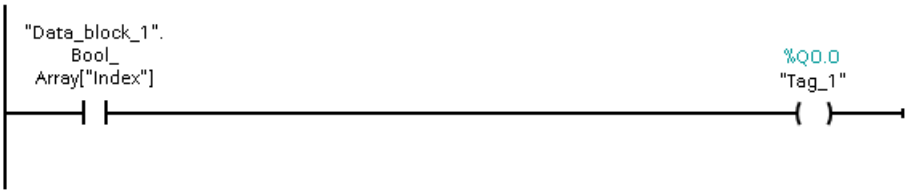
[1] Der Datentyp des vom Parameter MEMBER angegebenen Array-Elements und der Parameter VALUE müssen denselben Datentyp haben.

Der Freigabeausgang ENO ist gleich 0, wenn eine der folgenden Bedingungen zutrifft:

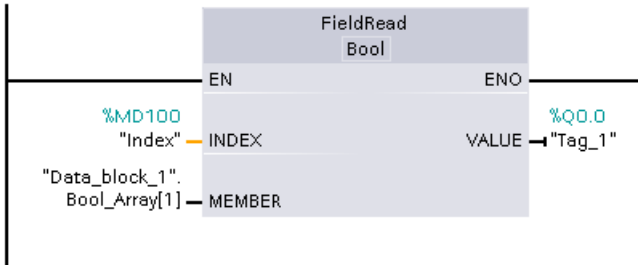
- Der Eingang EN ist im Signalzustand 0.
- Das vom Parameter INDEX angegebene Array-Element ist im vom Parameter MEMBER angegebenen Array nicht definiert.
- Während der Verarbeitung treten Fehler auf, z. B. Überlauf.

Beispiel: Zugriff auf Daten über Array-Indexierung

Um mit einer Variable auf Elemente eines Arrays zuzugreifen, verwenden Sie einfach die Variable als Array-Index in Ihrer Programmlogik. Im folgenden Netzwerk beispielsweise wird basierend auf dem Booleschen Wert eines Arrays aus Booleschen Werten ein Ausgang in "Datenbaustein_1", der von der PLC-Variable "Index" angegeben wird, gesetzt.



Die Logik mit dem Variablen-Array-Index entspricht dem bisherigen Verfahren über die Anweisung FeldLesen:



Die Anweisungen FeldLesen und FeldSchreiben können durch Indexierungslogik mit Variablen-Arrays ersetzt werden.

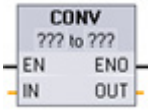
In SCL gibt es keine Anweisungen FeldLesen oder FeldSchreiben, es wird jedoch die indirekte Adressierung eines Arrays mit einer Variablen unterstützt:

```
#Tag_1 := "Data_block_1".Bool_Array[#Index];
```

8.7 Umwandlungsoperationen

8.7.1 Anweisung CONV (Wert konvertieren)

Tabelle 8- 100 Anweisung Umwandeln (CONV)

KOP/FUP	SCL	Beschreibung
	`out := <data type in>_TO_<data type out>(in);`	Konvertiert ein Datenelement von einem Datentyp in einen anderen Datentyp.

1 In KOP und FUP: Klicken Sie auf die "???" und wählen Sie die Datentypen in der Klappliste aus.

2 In SCL: Konstruieren Sie die Umwandlungsanweisung durch Angeben des Datentyps für den Eingangsparameter (in) und den Ausgangsparameter (out). Beispiel: DWORD_TO_REAL konvertiert einen Doppelwortwert in einen Realzahlenwert.

Tabelle 8- 101 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	Bitzeichenkette[1], SInt, USInt, Int, UInt, DInt, UDInt, Real, LReal, BCD16, BCD32, Char, WChar	Eingangswert
OUT	Bitzeichenkette[1], SInt, USInt, Int, UInt, DInt, UDInt, Real, LReal, BCD16, BCD32, Char, WChar	Eingangswert, umgewandelt in einen neuen Datentyp

1 Für die Anweisung dürfen keine Bitzeichenketten (Byte, Word, DWord) ausgewählt werden. Um für einen Parameter der Anweisung einen Operanden vom Datentyp Byte, Word oder DWord einzugeben, wählen Sie eine vorzeichenlose Ganzzahl mit der gleichen Bitlänge aus. Beispiel: Für ein Byte wählen Sie USInt, für ein Word wählen Sie UInt und für DWord wählen Sie UDInt.

Nach Auswahl des Quelldatentyps (Konvertieren aus) wird in der Klappliste eine Reihe möglicher Umwandlungen (Konvertieren in) angezeigt. Umwandlungen von und in BCD16 sind auf den Datentyp Int begrenzt. Umwandlungen von und in BCD32 sind auf den Datentyp DInt begrenzt.

Tabelle 8- 102 ENO-Status

ENO	Beschreibung	Ergebnis OUT
1	Kein Fehler	Gültiges Ergebnis
0	IN ist +/- INF oder +/- NaN	+/- INF oder +/- NaN
0	Ergebnis überschreitet den gültigen Bereich für den Datentyp von OUT	OUT wird auf den Wert von IN gesetzt

8.7.2 Umwandlungsanweisungen in SCL

Umwandlungsanweisungen in SCL

Tabelle 8- 103 Umwandlung von Bool, Byte, Word oder DWord

Datentyp	Anweisung	Ergebnis
Bool	`BOOL_TO_BYTE, BOOL_TO_WORD, BOOL_TO_DWORD, BOOL_TO_INT, BOOL_TO_DINT`	Der Wert wird in das niederwertigste Bit des Zieldatentyps übertragen.
Byte	`BYTE_TO_BOOL`	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	`BYTE_TO_WORD, BYTE_TO_DWORD`	Der Wert wird in das niederwertigste Byte des Zieldatentyps übertragen.
	`BYTE_TO_SINT, BYTE_TO_USINT`	Der Wert wird in den Zieldatentyp übertragen.
	`BYTE_TO_INT, BYTE_TO_UINT, BYTE_TO_DINT, BYTE_TO_UDINT`	Der Wert wird in das niederwertigste Byte des Zieldatentyps übertragen.
Word	`WORD_TO_BOOL`	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	`WORD_TO_BYTE`	Das niederwertigste Byte des Quellwerts wird in den Zieldatentyp übertragen.
	`WORD_TO_DWORD`	Der Wert wird in das niederwertigste Wort des Zieldatentyps übertragen.
	`WORD_TO_SINT, WORD_TO_USINT`	Das niederwertigste Byte des Quellwerts wird in den Zieldatentyp übertragen.
	`WORD_TO_INT, WORD_TO_UINT`	Der Wert wird in den Zieldatentyp übertragen.
	`WORD_TO_DINT, WORD_TO_UDINT`	Der Wert wird in das niederwertigste Wort des Zieldatentyps übertragen.
DWord	`DWORD_TO_BOOL`	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	`DWORD_TO_BYTE, DWORD_TO_WORD, DWORD_TO_SINT`	Das niederwertigste Byte des Quellwerts wird in den Zieldatentyp übertragen.
	`DWORD_TO_USINT, DWORD_TO_INT, DWORD_TO_UINT`	Das niederwertigste Wort des Quellwerts wird in den Zieldatentyp übertragen.
	`DWORD_TO_DINT, DWORD_TO_UDINT, DWORD_TO_REAL`	Der Wert wird in den Zieldatentyp übertragen.

Tabelle 8- 104 Umwandlung einer kurzen Ganzzahl (SInt oder USInt)

Datentyp	Anweisung	Ergebnis
SInt	**SINT_TO_BOOL**	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	SINT_TO_BYTE	Der Wert wird in den Zieldatentyp übertragen.
	SINT_TO_WORD, SINT_TO_DWORD	Der Wert wird in das niederwertigste Byte des Zieldatentyps übertragen.
	SINT_TO_INT, SINT_TO_DINT, SINT_TO_USINT, SINT_TO_UINT, SINT_TO_UDINT, SINT_TO_REAL, SINT_TO_LREAL, SINT_TO_CHAR, SINT_TO_STRING	Der Wert wird umgewandelt.
USInt	**USINT_TO_BOOL**	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	USINT_TO_BYTE	Der Wert wird in den Zieldatentyp übertragen.
	USINT_TO_WORD, USINT_TO_DWORD, USINT_TO_INT, USINT_TO_UINT, USINT_TO_DINT, USINT_TO_UDINT	Der Wert wird in das niederwertigste Byte des Zieldatentyps übertragen.
	USINT_TO_SINT, USINT_TO_REAL, USINT_TO_LREAL, USINT_TO_CHAR, USINT_TO_STRING	Der Wert wird umgewandelt.

Tabelle 8- 105 Umwandlung einer Ganzzahl (Int oder UInt)

Datentyp	Anweisung	Ergebnis
Int	**INT_TO_BOOL**	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	INT_TO_BYTE, INT_TO_DWORD, INT_TO_SINT, INT_TO_USINT, INT_TO_UINT, INT_TO_UDINT, INT_TO_REAL, INT_TO_LREAL, INT_TO_CHAR, INT_TO_STRING	Der Wert wird umgewandelt.
	INT_TO_WORD	Der Wert wird in den Zieldatentyp übertragen.
	INT_TO_DINT	Der Wert wird in das niederwertigste Byte des Zieldatentyps übertragen.
UInt	**UINT_TO_BOOL**	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	UINT_TO_BYTE, UINT_TO_SINT, UINT_TO_USINT, UINT_TO_INT, UINT_TO_REAL, UINT_TO_LREAL, UINT_TO_CHAR, UINT_TO_STRING	Der Wert wird umgewandelt.
	UINT_TO_WORD, UINT_TO_DATE	Der Wert wird in den Zieldatentyp übertragen.
	UINT_TO_DWORD, UINT_TO_DINT, UINT_TO_UDINT	Der Wert wird in das niederwertigste Byte des Zieldatentyps übertragen.

Tabelle 8- 106 Umwandlung einer doppelten Ganzzahl (Dint oder UDInt)

Datentyp	Anweisung	Ergebnis
DInt	**DINT_TO_BOOL**	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	DINT_TO_BYTE, DINT_TO_WORD, DINT_TO_SINT, DINT_TO_USINT, DINT_TO_INT, DINT_TO_UINT, DINT_TO_UDINT, DINT_TO_REAL, DINT_TO_LREAL, DINT_TO_CHAR, DINT_TO_STRING	Der Wert wird umgewandelt.
	DINT_TO_DWORD, DINT_TO_TIME	Der Wert wird in den Zieldatentyp übertragen.
UDInt	**UDINT_TO_BOOL**	Das niederwertigste Bit wird in den Zieldatentyp übertragen.
	UDINT_TO_BYTE, UDINT_TO_WORD, UDINT_TO_SINT, UDINT_TO_USINT, UDINT_TO_INT, UDINT_TO_UINT, UDINT_TO_DINT, UDINT_TO_REAL, UDINT_TO_LREAL, UDINT_TO_CHAR, UDINT_TO_STRING	Der Wert wird umgewandelt.
	UDINT_TO_DWORD, UDINT_TO_TOD	Der Wert wird in den Zieldatentyp übertragen.

Tabelle 8- 107 Umwandlung einer Realzahl (Real oder LReal)

Datentyp	Anweisung	Ergebnis
Real	**REAL_TO_DWORD, REAL_TO_LREAL**	Der Wert wird in den Zieldatentyp übertragen.
	REAL_TO_SINT, REAL_TO_USINT, REAL_TO_INT, REAL_TO_UINT, REAL_TO_DINT, REAL_TO_UDINT, REAL_TO_STRING	Der Wert wird umgewandelt.
LReal	**LREAL_TO_SINT, LREAL_TO_USINT, LREAL_TO_INT, LREAL_TO_UINT, LREAL_TO_DINT, LREAL_TO_UDINT, LREAL_TO_REAL, LREAL_TO_STRING**	Der Wert wird umgewandelt.

Tabelle 8- 108 Umwandlung von Time, DTL, TOD oder Date

Datentyp	Anweisung	Ergebnis
Time	**TIME_TO_DINT**	Der Wert wird in den Zieldatentyp übertragen.
DTL	**DTL_TO_DATE, DTL_TO_TOD**	Der Wert wird umgewandelt.
TOD	**TOD_TO_UDINT**	Der Wert wird umgewandelt.
Date	**DATE_TO_UINT**	Der Wert wird umgewandelt.

Tabelle 8- 109 Umwandlung von Char oder String

Datentyp	Anweisung	Ergebnis
Char	`CHAR_TO_SINT, CHAR_TO_USINT, CHAR_TO_INT, CHAR_TO_UINT, CHAR_TO_DINT, CHAR_TO_UDINT`	Der Wert wird umgewandelt.
	`CHAR_TO_STRING`	Der Wert wird zum ersten Zeichen der Zeichenkette übertragen.
String	`STRING_TO_SINT, STRING_TO_USINT, STRING_TO_INT, STRING_TO_UINT, STRING_TO_DINT, STRING_TO_UDINT, STRING_TO_REAL, STRING_TO_LREAL`	Der Wert wird umgewandelt.
	`STRING_TO_CHAR`	Das erste Zeichen der Zeichenkette wird in Char kopiert.

8.7.3 Anweisungen ROUND (Zahl runden) und TRUNC (Ganzzahl erzeugen)

Tabelle 8- 110 Anweisungen ROUND und TRUNC

KOP/FUP	SCL	Beschreibung
	`out := ROUND (in);`	konvertiert eine Realzahl in eine Ganzzahl. Klicken Sie für KOP/FUP auf das "???" im Anweisungsfeld, um den Datentyp für den Ausgang auszuwählen, zum Beispiel "DInt". Für SCL ist der Datentyp für den Ausgang der Anweisung ROUND standardmäßig DINT. Für die Rundung auf einen anderen Ausgangsdatentyp ist der Name der Anweisung mit dem expliziten Namen des Datentyps einzugeben, zum Beispiel ROUND_REAL oder ROUND_LREAL. Die Nachkommastellen der Realzahl werden auf den nächsten ganzzahligen Wert gerundet (IEEE - runden). Wenn die Zahl genau die Hälfte der Spanne zwischen zwei Ganzzahlen ist (z. B. 10,5), wird die Zahl auf die gerade Ganzzahl gerundet. Beispiel: • ROUND (10.5) = 10 • ROUND (11.5) = 12
	`out := TRUNC(in);`	TRUNC konvertiert eine Realzahl in eine Ganzzahl. Die Nackommastellen der Realzahl werden auf Null verkürzt (IEEE - runden auf Null).

[1] In KOP und FUP: Klicken Sie auf "???" (beim Anweisungsnamen) und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 111 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	Real, LReal	Eingang Gleitpunktzahl
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Gerundeter oder ganzzahliger Ausgang

Tabelle 8- 112 ENO-Status

ENO	Beschreibung	Ergebnis OUT
1	Kein Fehler	Gültiges Ergebnis
0	IN ist +/- INF oder +/- NaN	+/- INF oder +/- NaN

8.7.4 Anweisungen CEIL und FLOOR (Aus Gleitpunktzahl nächsthöhere und nächstniedere Ganzzahl erzeugen)

Tabelle 8- 113 Anweisungen CEIL und FLOOR

KOP/FUP	SCL	Beschreibung
	`out := CEIL(in);`	Konvertiert eine Realzahl (Real oder LReal) in die nächste Ganzzahl, die größer oder gleich der ausgewählten Realzahl ist (IEEE - Runden auf +unendlich).
	`out := FLOOR(in);`	Konvertiert eine Realzahl (Real oder LReal) in die nächste Ganzzahl, die kleiner oder gleich der ausgewählten Realzahl ist (IEEE - Runden auf -unendlich).

[1] In KOP und FUP: Klicken Sie auf "???" (beim Anweisungsnamen) und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 114 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	Real, LReal	Eingang Gleitpunktzahl
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Umgewandelter Ausgang

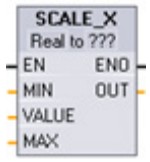
Tabelle 8- 115 ENO-Status

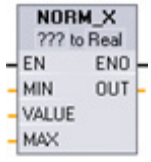
ENO	Beschreibung	Ergebnis OUT
1	Kein Fehler	Gültiges Ergebnis
0	IN ist +/- INF oder +/- NaN	+/- INF oder +/- NaN

8.7.5 Anweisungen SCALE_X (Skalieren) und NORM_X (Normalisieren)

Tabelle 8- 116 Anweisungen SCALE_X und NORM_X

KOP/FUP	SCL	Beschreibung

	`out :=SCALE_X(min:=_in_,`
	`value:=_in_,`
	`max:=_in_);`
		Skaliert den normalisierten Realparameter VALUE (0,0 <= VALUE <= 1,0) in den mit den Parametern MIN und MAX vorgegebenen Datentyp und Wertebereich:
		OUT = VALUE (MAX - MIN) + MIN

	`out :=NORM_X(min:=_in_,`
	`value:=_in_,`
	`max:=_in_);`
		Normalisiert den Parameter VALUE innerhalb des von den Parametern MIN und MAX angegebenen Wertebereichs:
		OUT = (VALUE - MIN) / (MAX - MIN),
		dabei ist (0,0 <= OUT <= 1,0)

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 117 Datentypen für die Parameter

Parameter	Datentyp[1]	Beschreibung
MIN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Eingang Mindestwert des Bereichs
VALUE	SCALE_X: Real, LReal NORM_X: SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Eingangswert für Skalierung oder Normierung
MAX	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Eingang Höchstwert des Bereichs
OUT	SCALE_X: SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal NORM_X: Real, LReal	Skalierter oder normierter Ausgangswert

[1] Bei SCALE_X: Die Parameter MIN, MAX und OUT müssen denselben Datentyp haben.
Bei NORM_X: Die Parameter MIN, VALUE und MAX müssen denselben Datentyp haben.

Hinweis

Der Parameter VALUE von SCALE_X muss im Bereich (0,0 <= VALUE <= 1,0) liegen.

Falls der Parameter VALUE kleiner als 0,0 oder größer als 1,0 ist:

- Die lineare Skalierungsoperation kann Ausgabewerte OUT erzeugen, die kleiner als der Wert des Parameters MIN oder größer als der Wert des Parameters MAX sind, sofern es sich um OUT-Werte handelt, die im Wertebereich des Datentyps von OUT liegen. Bei der Ausführung von SCALE_X wird in diesen Fällen ENO = WAHR gesetzt.
- Es ist möglich, skalierte Zahlen zu erzeugen, die nicht im Bereich des Datentyps von OUT liegen. In diesem Fall wird der Wert des Parameters OUT auf einen Zwischenwert gesetzt, der dem niederwertigsten Anteil der skalierten Realzahl vor der endgültigen Umwandlung in den Datentyp OUT entspricht. Bei der Ausführung von SCALE_X wird in diesen Fällen ENO = FALSCH gesetzt.

Der Parameter VALUE von NORM_X muss im Bereich (MIN <= VALUE <= MAX) liegen.

Wenn der Parameter VALUE kleiner als MIN oder größer als MAX ist, kann die lineare Skalierungsanweisung normalisierte Ausgabewerte OUT erzeugen, die kleiner als 0,0 oder größer als 1,0 sind. Bei der Ausführung von NORM_X wird dann ENO = WAHR gesetzt.

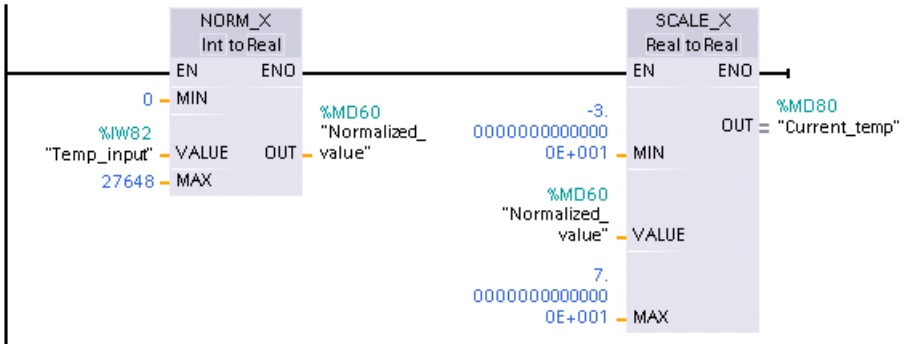
Tabelle 8- 118 ENO-Status

ENO	Bedingung	Ergebnis OUT
1	Kein Fehler	Gültiges Ergebnis
0	Das Ergebnis überschreitet den gültigen Bereich für den Datentyp von OUT.	Zwischenergebnis: Der niederwertigste Anteil einer Realzahl vor der endgültigen Umwandlung in den Datentyp von OUT.
0	ParameterMAX <= MIN	SCALE_X: Der niederwertigste Anteil der Realzahl VALUE, mit dem die Größe von OUT belegt wird. NORM_X: VALUE im Datentyp VALUE, zur Belegung der Doppelwortgröße erweitert.
0	Parameter VALUE = +/- INF oder +/- NaN	VALUE wird geschrieben in OUT

Beispiel (KOP): Normalisieren und Skalieren eines Analogeingangswerts

Ein Analogeingang eines analogen Signalmoduls oder Signalboards mit Stromeingang liegt im Bereich von 0 bis 27648 der gültigen Werte. Angenommen, ein Analogeingang stellt eine Temperatur dar, wobei der Wert 0 des Analogeingangs -30,0 Grad C und der Wert 27648 die Temperatur 70,0 Grad C darstellt.

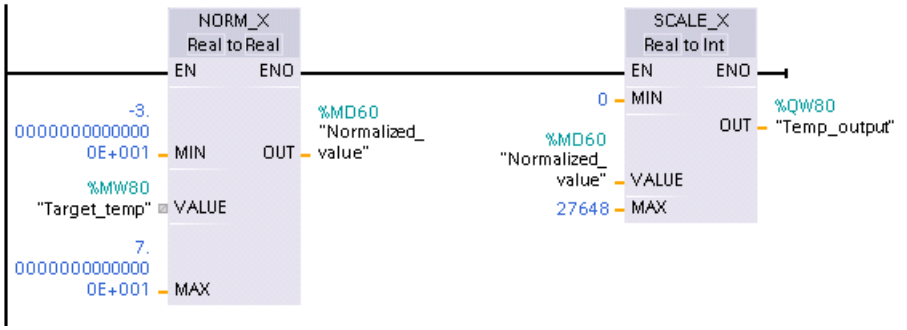
Um den Analogwert in die entsprechenden physikalischen Einheiten umzuwandeln, normalisieren Sie den Eingang in einen Wert zwischen 0,0 und 1,0 und skalieren ihn dann zwischen -30,0 und 70,0. Der resultierende Wert ist die vom Analogeingang dargestellte Temperatur in Grad C:



Beachten Sie, dass, wenn der Analogeingang von einem analogen Signalmodul oder Signalboard mit Spannungseingang kommen würde, der MIN-Wert für die Anweisung NORM_X -27648 und nicht 0 wäre.

Beispiel (KOP): Normalisieren und Skalieren eines Analogausgangswerts

Ein in einem analogen Signalmodul oder Signalboard mit Stromausgang zu setzender Analogausgang muss im Bereich von 0 bis 27648 der gültigen Werte liegen. Angenommen, ein Analogausgang stellt eine Temperatureinstellung dar, wobei der Wert 0 des Analogeingangs -30,0 Grad C und der Wert 27648 die Temperatur 70,0 Grad C darstellt. Um im Speicher einen Temperaturwert, der zwischen -30,0 und 70,0 liegt, in einen Wert für den Analogausgang im Bereich zwischen 0 und 27648 umzuwandeln, müssen Sie den Wert in physikalischen Einheiten in einen Wert zwischen 0,0 und 1,0 normalisieren und ihn dann in den Bereich des Analogausgangs zwischen 0 und 27648 skalieren:



Beachten Sie, dass, wenn der Analogausgang für ein analoges Signalmodul oder Signalboard mit Spannungseingang vorgesehen wäre, der MIN-Wert für die Anweisung SCALE_X -27648 und nicht 0 wäre.

Weitere Informationen zur Darstellung von Analogeingängen (Seite 1254) und zur Darstellung von Analogausgängen (Seite 1255) für Spannung und Strom finden Sie in den technischen Daten.

8.7.6 Variant-Umwandlungsanweisungen

8.7.6.1 Anweisung VARIANT_TO_DB_ANY (VARIANT und DB_ANY umwandeln)

Mit der SCL-Anweisung "VARIANT in DB_ANY umwandeln" wird der Operand von Parameter IN gelesen und in den Datentyp DB_ANY umgewandelt. Parameter IN hat Datentyp Variant und stellt entweder einen Instanzdatenbaustein oder einen ARRAY-Datenbaustein dar. Beim Anlegen des Programms muss nicht bekannt sein, welcher Datenbaustein dem IN-Parameter entspricht. Die Anweisung liest die Datenbausteinnummer während der Laufzeit und schreibt sie in den Operanden von Parameter RET_VAL.

Tabelle 8- 119 Anweisung VARIANT_TO_DB_ANY

KOP/FUP	SCL	Beschreibung
Nicht verfügbar	`RET_VAL := VARIANT_TO_DB_ANY( in := _variant_in_, err => _int_out_);`	Liest den Operanden von Parameter Variant IN und speichert ihn im Funktionsergebnis des Typs DB_ANY.

Tabelle 8- 120 Parameter für die VARIANT_TO_DB_ANY-Anweisung

Parameter	Datentyp	Beschreibung
IN	Variant	Variant-Element, das einen Instanz- oder Array-Datenbaustein darstellt.
RET_VAL	DB_ANY	Ausgegebener Datentyp DB_ANY, der die umgewandelte Datenbausteinnummer enthält
ERR	Int	Fehlerinformationen

Tabelle 8- 121 ENO-Status

ENO	Bedingung	Ergebnis
1	Kein Fehler	Die Anweisung wandelt den Variant-Eingangswert um und speichert ihn im Funktionsausgang DB_ANY.
0	Freigabeeingang EN hat Signalzustand "0" oder IN-Parameter ist ungültig.	Die Anweisung tut nichts.

Tabelle 8- 122 Fehlercodes für Anweisung VARIANT_TO_DB_ANY

Err (W#16#...)	Beschreibung
0000	Kein Fehler
252C	Datentyp Variant von Parameter IN hat den Wert 0. Die CPU wechselt in Betriebszustand STOP.
8131	Der Datenbaustein existiert nicht oder ist zu kurz (erster Zugriff).
8132	Der Datenbaustein ist zu kurz und kein Array-Datenbaustein (zweiter Zugriff).
8134	Der Datenbaustein ist schreibgeschützt
8150	Datentyp Variant von Parameter IN liefert den Wert "0". Für diese Fehlermeldung muss die Bausteineigenschaft "Fehler in Baustein bearbeiten" aktiviert sein. Sonst wechselt die CPU in Betriebszustand STOP und sendet Fehlercode 16#252C
8154	Der Datenbaustein hat den falschen Datentyp.
*Feldercodes können im Programmeditor als Ganzzahlen oder Hexadezimalwerte angezeigt werden.	

8.7.6.2 Anweisung DB_ANY_TO_VARIANT (DB_ANY in VARIANT umwandeln)

Mit der SCL-Anweisung "DB_ANY in VARIANT umwandeln" kann die Nummer des Datenbausteins gelesen werden, der die unten stehenden Voraussetzungen erfüllt. Der Operand von Parameter IN hat Datentyp DB_ANY, beim Anlegen des Programms muss daher nicht bekannt sein, welcher Datenbaustein angegeben wird, dessen Nummer gelesen werden soll. Die Datenbausteinnummer wird während der Laufzeit gelesen und mit einem VARIANT-Pointer in den mit Parameter RET_VAL angegebenen Operanden geschrieben.

Tabelle 8- 123 Anweisung DB_ANY_TO_VARIANT

KOP/FUP	SCL	Beschreibung
Nicht verfügbar	`RET_VAL := DB_ANY_TO_VARIANT( in := _db_any_in_, err => _int_out_);`	Liest die Datenbausteinnummer von Parameter Variant IN und speichert sie im Funktionsergebnis des Typs VARIANT.

Tabelle 8- 124 Parameter für die Anweisung DB_ANY_TO_VARIANT

Parameter	Datentyp	Beschreibung
IN	DB_ANY	Variant-Element, das die Datenbausteinnummer enthält
RET_VAL	Variant	Ausgegebener Datentyp DB_ANY, der die umgewandelte Datenbausteinnummer enthält
ERR	Int	Fehlerinformationen

Tabelle 8- 125 ENO-Status

ENO	Bedingung	Ergebnis
1	Kein Fehler	Die Anweisung wandelt die Datenbaustein-nummer in das Variant-Format und speichert sie im Funktionsausgang DB_ANY
0	Freigabeeingang EN hat Signalzustand "0" oder IN-Parameter ist ungültig.	Die Anweisung tut nichts.

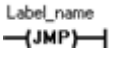
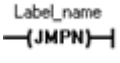
Tabelle 8- 126 Fehlercodes für Anweisung DB_ANY_TO_VARIANT

Err (W#16#...)	Beschreibung
0000	Kein Fehler
8130	Nummer des Datenbausteins ist 0.
8131	Der Datenbaustein existiert nicht oder ist zu kurz.
8132	Der Datenbaustein ist zu kurz und kein Array-Datenbaustein.
8134	Der Datenbaustein ist schreibgeschützt.
8154	Der Datenbaustein hat den falschen Datentyp.
8155	Unbekannter Typcode
*Feldercodes können im Programmeditor als Ganzzahlen oder Hexadezimalwerte angezeigt werden.	

8.8 Programmsteuerungsoperationen

8.8.1 Anweisungen JMP (Springen bei VKE = 1), JMPN (Springen bei VKE = 0) und Label (Sprungmarke)

Tabelle 8- 127 Anweisungen JMP, JMPN und LABEL

KOP	FUP	SCL	Beschreibung
		Siehe GOTO (Seite 314)-Anweisung.	Springen bei VKE (Verknüpfungsergebnis) = 1: Bei Signalfluss zu einer JMP-Spule (KOP) oder wenn der JMP-Boxeingang wahr ist (FUP), wird die Programmausführung mit der ersten Anweisung nach der angegebenen Sprungmarke fortgesetzt.
			Springen bei VKE = 0: Ist kein Signalfluss zu einer JMPN-Spule (KOP) vorhanden oder ist der JMPN-Boxeingang falsch (FUP), wird die Programmausführung mit der ersten Anweisung nach der angegebenen Sprungmarke fortgesetzt.
			Sprungziel für eine Sprunganweisung JMP oder JMPN

[1] Sie erstellen die Namen für die Sprungmarken, indem Sie die Anweisung LABEL direkt eingeben. Über das Parametersymbol können Sie die verfügbaren Sprungmarken für die Anweisung JMP oder JMPN auswählen. Sie können den Namen der Sprungmarke auch direkt in die Anweisung JMP oder JMPN eingeben.

Tabelle 8- 128 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
Label_name	Kennung der Sprungmarke	Kennzeichnung für Sprunganweisungen und die entsprechende Sprungmarke für das Sprungziel

- Jede Sprungmarke muss innerhalb eines Codebausteins eindeutig sein.
- Sie können innerhalb eines Codebausteins springen, aber Sie können nicht von einem Codebaustein in einen anderen Codebaustein springen.
- Sie können vorwärts oder rückwärts springen.
- Sie können von mehreren Stellen eines Codebausteins zu derselben Sprungmarke springen.

8.8.2 Anweisung JMP_LIST (Sprungliste definieren)

Tabelle 8- 129 Anweisung JMP_LIST

KOP/FUP	SCL	Beschreibung

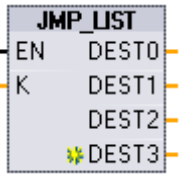
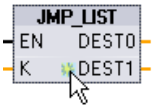
	`CASE k OF`
	`  0: GOTO dest0;`
	`  1: GOTO dest1;`
	`  2: GOTO dest2;`
	`  [n: GOTO destn;]`
	`END_CASE;`
		Die Anweisung JMP_LIST verteilt die Programmsprünge, um die Ausführung von Programmabschnitten zu steuern. Je nach Wert des Eingangs K erfolgt ein Sprung zur entsprechenden Sprungmarke im Programm. Die Programmausführung wird mit den Programmanweisungen fortgesetzt, die auf das Sprungziel folgen. Wenn der Wert des Eingangs K die Anzahl der Sprungmarken - 1 überschreitet, wird kein Sprung durchgeführt und die Verarbeitung wird mit dem nächsten Netzwerk im Programm fortgesetzt.

Tabelle 8- 130 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
K	UInt	Steuerwert für die Sprungverteilung
DEST0, DEST1, .., DESTn.	Sprungmarken im Programm	Sprungmarken der Sprungziele entsprechend spezifischen Werten des Parameters K: Wenn der Wert von K gleich 0 ist, wird ein Sprung zu der Sprungmarke durchgeführt, die dem Ausgang DEST0 zugewiesen ist. Wenn der Wert von K gleich 1 ist, wird ein Sprung zu der Sprungmarke durchgeführt, die dem Ausgang DEST1 zugewiesen ist usw. Wenn der Wert des Eingangs K (die Anzahl der Sprungmarken - 1) überschreitet, wird kein Sprung durchgeführt und die Verarbeitung wird mit dem nächsten Netzwerk im Programm fortgesetzt.

In KOP und FUP: Beim Einfügen der Box JMP_LIST in Ihr Programm sind zwei Ausgänge für Sprungmarken vorhanden. Sie können Sprungziele hinzufügen oder löschen.

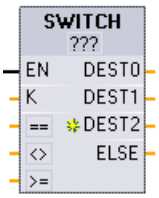


Klicken Sie in der Box auf das Symbol zum Erstellen (links vom letzten Parameter DEST), um neue Ausgänge für Sprungmarken hinzuzufügen.

- Klicken Sie mit der rechten Maustaste auf einen Ausgangsanschluss und wählen Sie "Ausgang einfügen".
- Klicken Sie mit der rechten Maustaste auf einen Ausgangsanschluss und wählen Sie "Löschen".

8.8.3 Anweisung SWITCH (Sprungverteiler)

Tabelle 8- 131 Anweisung SWITCH

KOP/FUP	SCL	Beschreibung
	`Nicht verfügbar`	Die Anweisung SWITCH verteilt die Programmsprünge, um die Ausführung von Programmabschnitten zu steuern. Je nach Ergebnis des Vergleichs zwischen dem Wert des Eingangs K und den Werten der angegebenen Vergleichseingänge wird ein Sprung zu der Sprungmarke durchgeführt, die dem ersten Vergleichstest mit dem Ergebnis WAHR entspricht. Ist keiner der Vergleiche wahr, erfolgt ein Sprung zu der Sprungmarke, die ELSE zugewiesen ist. Die Programmausführung wird mit den Programmanweisungen fortgesetzt, die auf das Sprungziel folgen.

1 In KOP und FUP: Klicken Sie unterhalb des Box-Namens und wählen Sie in der Klappliste einen Datentyp aus.

2 In SCL: Verwenden Sie Vergleiche über IF-THEN.

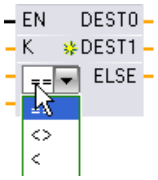
Tabelle 8- 132 Datentypen für die Parameter

Parameter	Datentyp[1]	Beschreibung
K	UInt	Eingang für den allgemeinen Vergleichswert
==, <>, <, <=, >. >=	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, TOD, Date	Eingänge für einzelne Vergleichswerte für spezifische Vergleichsarten
DEST0, DEST1, .., DESTn, ELSE	Sprungmarken im Programm	Sprungmarken der Sprungziele entsprechend spezifischen Vergleichen: Der Vergleichseingang unter und neben dem Eingang K wird zuerst verarbeitet und bewirkt einen Sprung zu der Sprungmarke, die DEST0 zugewiesen ist, wenn der Vergleich zwischen dem Wert K und diesem Eingang wahr ist. Der nächste Vergleichstest verwendet den nächsten Eingang darunter und bewirkt einen Sprung zu der Sprungmarke, die DEST1 zugewiesen ist, wenn der Vergleich wahr ist. Die übrigen Vergleiche werden auf ähnliche Weise verarbeitet und wenn keiner der Vergleiche wahr ist, erfolgt ein Sprung zu der Sprungmarke, die dem Ausgang ELSE zugewiesen ist.

1 Der Eingang K und die Vergleichseingänge (==, <>, <, <=, >, >=) müssen denselben Datentyp haben.

Eingänge hinzufügen, Eingänge löschen und Vergleichsarten angeben

Beim Einfügen der Box SWITCH in KOP oder FUP in Ihrem Programm sind zwei Vergleichseingänge vorhanden. Sie können Vergleichsarten zuweisen und Eingänge bzw. Sprungziele hinzufügen oder löschen (siehe unten).



Klicken Sie in der Box auf einen Vergleichsoperator und wählen Sie in der Klappliste einen neuen Operator aus.

Klicken Sie in der Box auf das Symbol zum Erstellen (links vom letzten Parameter DEST), um neue Parameter für Vergleichsziele hinzuzufügen.

- Klicken Sie mit der rechten Maustaste auf einen Eingangsanschluss und wählen Sie "Eingang einfügen".
- Klicken Sie mit der rechten Maustaste auf einen Eingangsanschluss und wählen Sie "Löschen".

Tabelle 8- 133 Auswahl des Datentyps für die Box SWITCH und zulässige Vergleichsanweisungen

Datentyp	Vergleich	Syntax des Operators
Byte, Word, DWord	Gleich	==
	Ungleich	<>
SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Time, TOD, Date	Gleich	==
	Ungleich	<>
	Größer oder gleich	>=
	Kleiner oder gleich	<=
	Größer als	>
	Kleiner als	<

Regeln für die Platzierung der Box SWITCH

- Vor dem Vergleichseingang darf keine KOP/FUP-Anweisung angeschlossen werden.
- Es gibt keinen ENO-Ausgang, deshalb ist nur eine SWITCH-Anweisung in einem Netzwerk zulässig, und die SWITCH-Anweisung muss die letzte Anweisung im Netzwerk sein.

8.8.4 Anweisung RET (Zurück springen)

Mit der optionalen Anweisung RET wird die Ausführung des aktuellen Bausteins beendet. Nur bei Signalfluss zur Spule RET (KOP) oder wenn die Box RET wahr ist (FUP), wird die Programmausführung des aktuellen Bausteins an diesem Punkt beendet; nach der Anweisung RET werden keine weiteren Anweisungen ausgeführt. Wenn es sich bei dem aktuellen Baustein um einen OB handelt, wird der Parameter "Return_Value" ignoriert. Wenn es sich bei dem aktuellen Baustein um eine FC oder einen FB handelt, wird der Wert des Parameters "Return_Value " als ENO-Wert der aufgerufenen Box wieder an die aufrufende Routine übergeben.

Sie müssen die Anweisung RET nicht als letzte Anweisung in einen Baustein einfügen, dies geschieht automatisch. Sie können mehrere Anweisungen RET in einen einzigen Baustein einfügen.

Für SCL siehe RETURN (Seite 314)-Anweisung.

Tabelle 8- 134 Programmsteuerungsanweisung Rückgabewert (RET)

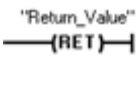
KOP	FUP	SCL	Beschreibung
		`RETURN;`	Beendet die Ausführung des aktuellen Bausteins.

Tabelle 8- 135 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
Return_Value	Bool	Der Parameter "Return_value" der Anweisung RET ist dem Ausgang ENO der Bausteinaufruf-Box im aufrufenden Baustein zugewiesen.

Beispiel für die Verwendung einer Anweisung RET in einem Codebaustein FC:

1. Erstellen Sie ein neues Projekt und fügen Sie eine FC ein:
2. Bearbeiten Sie die FC:
   - Fügen Sie Anweisungen aus dem Anweisungsverzeichnis ein.
   - Fügen Sie eine Anweisung RET einschließlich einem der folgenden Elemente für den Parameter "Return_Value" ein:

     WAHR, FALSCH oder eine Adresse, die den erforderlichen Rückgabewert angibt.
   - Fügen Sie weitere Anweisungen ein.
3. Rufen Sie die FC aus dem Hauptprogramm MAIN [OB1] auf.

Der Eingang EN der FC-Box im Codebaustein MAIN muss wahr sein, damit die FC ausgeführt wird.

Der von der Anweisung RET in der FC angegebene Wert liegt nach der Ausführung der FC, an deren Anweisung RET Signalfluss anliegt, am Ausgang ENO der FC-Box im Codebaustein MAIN an.

8.8.5 Anweisung ENDIS_PW (Passwort-Legitimierung einschränken und freigeben)

Tabelle 8- 136 Anweisung ENDIS_PW

KOP/FUP	SCL	Beschreibung

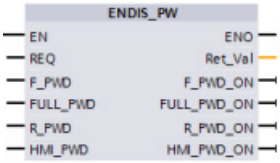
	`ENDIS_PW(`
	`  req:=_bool_in_,`
	`  f_pwd:=_bool_in_,`
	`  full_pwd:=_bool_in_,`
	`  r_pwd:=_bool_in_,`
	`  hmi_pwd:=_bool_in_,`
	`  f_pwd_on=>_bool_out_,`
	`  full_pwd_on=>_bool_out_,`
	`  r_pwd_on=>_bool_out_,`
	`  hmi_pwd_on=>_bool_out_);`
		Mit der Anweisung ENDIS_PW kann die Verbindung eines Clients mit einer S7-1200 CPU freigegeben oder gesperrt werden, auch wenn der Client das richtige Passwort bereitstellt.
		Diese Anweisung sperrt keine Webserver-Passwörter.

Tabelle 8- 137 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Funktion ausführen, wenn REQ=1
F_PWD	IN	Bool	Passwort für fehlersicheren Betrieb: Freigeben (=1) oder sperren (=0)
FULL_PWD	IN	Bool	Passwort für Vollzugriff: Passwort für Vollzugriff freigeben (=1) oder sperren (=0)
R_PWD	IN	Bool	Passwort für Lesezugriff: Freigeben (=1) oder sperren (=0)
HMI_PWD	IN	Bool	HMI-Passwort: Freigeben (=1) oder sperren (=0)
F_PWD_ON	OUT	Bool	Status des Passworts für fehlersicheren Betrieb: Freigegeben (=1) oder gesperrt (=0)
FULL_PWD_ON	OUT	Bool	Status des Passworts für Vollzugriff: Freigegeben (=1) oder gesperrt (=0)
R_PWD_ON	OUT	Bool	Status des Passworts für Lesezugriff: Freigegeben (=1) oder gesperrt (=0)
HMI_PWD_ON	OUT	Bool	Status des HMI-Passworts: Freigegeben (=1) oder gesperrt (=0)
Ret_Val	OUT	Word	Funktionsergebnis

Wird ENDIS_PW mit REQ=1 aufgerufen, so werden die Passworttypen gesperrt, für die der entsprechende Passwort-Eingangsparameter FALSCH ist. Jeder Passworttyp kann einzeln freigegeben oder gesperrt werden. Wird zum Beispiel nur das Passwort für den fehlersicheren Betrieb freigegeben und werden alle anderen Passwörter gesperrt, kann damit der Zugriff auf die CPU auf eine kleine Benutzergruppe beschränkt werden.

ENDIS_PW wird synchron in einem Programmzyklus ausgeführt und die Passwort-Ausgangsparameter zeigen immer den aktuellen Stand der Passwortfreigabe, unabhängig vom Eingangsparameter REQ. Passwörter, die Sie freigeben, müssen grundsätzlich auch gesperrt werden können. Andernfalls wird ein Fehler ausgegeben und es werden alle Passwörter freigegeben, die vor der Ausführung von ENDIS_PW freigegeben waren. Das bedeutet, dass bei einer Standard-CPU (bei der das Passwort für fehlersicheren Betrieb nicht konfiguriert ist) F_PWD stets auf 1 gesetzt sein muss, um einen Rückgabewert von 0 zu erreichen. In diesem Fall ist F_PWD_ON stets 1.

Hinweis

- Die Ausführung von ENDIS_PW kann den Zugriff auf HMI-Geräte blockieren, wenn das HMI-Passwort gesperrt ist.
- Client-Sitzungen, die vor der Ausführung von ENDIS_PW freigegeben waren, werden durch die Ausführung von ENDIS_PW nicht verändert.

Nach dem Anlauf ist der CPU-Zugriff durch die Passwörter eingeschränkt, die zuvor in der regulären Konfiguration des CPU-Schutzes definiert wurden. Die Fähigkeit, ein gültiges Passwort zu sperren, muss mit der Ausführung von ENDIS_PW erneut eingerichtet werden. Wird ENDIS_PW jedoch sofort ausgeführt und werden dadurch erforderliche Passwörter gesperrt, kann der Zugriff auf das TIA Portal gesperrt werden. Mithilfe einer Zeitoperation können Sie die Ausführung von ENDIS_PW verzögern und einen Zeitraum zur Eingabe von Passwörtern gestatten, bevor die Passwörter gesperrt werden.

Hinweis

Wiederherstellen einer CPU, deren Kommunikation mit dem TIA Portal gesperrt wurde

Weitere Informationen zum Löschen des internen Ladespeichers eines PLC mithilfe einer Memory Card finden Sie unter "Wiederherstellung eines verlorenen Passworts (Seite 149)".

Ein Wechsel des Betriebszustands nach STOP auf Grund von Fehlern, der STP-Ausführung oder von STEP 7 beeinflusst den Schutz nicht. Der Schutz gilt, bis die CPU aus- und wieder eingeschaltet wird. In der folgenden Tabelle finden Sie weitere Details.

Aktion	Betriebszustand	ENDIS_PW-Passwortschutz
Nach dem Urlöschen durch STEP 7	STOP	Aktiv: Gesperrte Passwörter bleiben gesperrt.
Nach dem Einschalten oder Wechseln einer Memory Card	STOP	Aus: Keine Passwörter sind gesperrt.
Nach Ausführung von ENDIS_PW in einem Zyklus- oder Anlauf-OB	STARTUP, RUN	Aktiv: Passwörter sind entsprechend den ENDIS_PW-Parametern gesperrt
Nach einem Wechsel des Betriebszustands von RUN oder STARTUP nach STOP durch STP-Anweisung, Fehler oder STEP 7	STOP	Aktiv: Gesperrte Passwörter bleiben gesperrt

Hinweis

Schützen Sie die CPU-Zugriffsstufen durch starke Passwörter. Starke Passwörter sind mindestens zehn Zeichen lang, bestehen aus Buchstaben, Zahlen und Sonderzeichen, sind keine Wörter, die in einem Wörterbuch gefunden werden können, und sind keine Namen oder Kennungen, die sich aus persönlichen Daten ableiten lassen. Halten Sie das Passwort geheim und ändern Sie es häufig.

Tabelle 8- 138 Bedingungscodes

RET_VAL (W#16#...)	Beschreibung
0000	Kein Fehler
8090	Die Anweisung wird nicht unterstützt.
80D0	Das Passwort für fehlersicheren Betrieb ist nicht konfiguriert.
80D1	Das Passwort für Lese-/Schreibzugriff ist nicht konfiguriert.
80D2	Das Passwort für Lesezugriff ist nicht konfiguriert.
80D3	Das Passwort für HMI-Zugriff ist nicht konfiguriert.

8.8.6 Anweisung RE_TRIGR (Zyklusüberwachungszeit neu starten)

Tabelle 8- 139 Anweisung RE_TRIGR

KOP/FUP	SCL	Beschreibung
	`RE_TRIGR();`	Mit RE_TRIGR (Zykluszeitüberwachung neu starten) kann die maximal zulässige Zeit bis zur Erzeugung einer Fehlermeldung durch die Zykluszeitüberwachung verlängert werden.

Mit der Anweisung RE_TRIGR starten Sie die Zykluszeitüberwachung während eines Zyklusablaufs neu. Dadurch wird die maximal zulässige Zykluszeit mit der letzten Ausführung der Funktion RE_TRIGR um eine maximale Zykluszeit verlängert.

Hinweis

Vor der Firmware-Version 2.2 der S7-1200 CPU musste RE_TRIGR von einem Programmzyklus-OB ausgeführt werden und konnte dazu verwendet werden, die Zykluszeit des Zielsystems unbegrenzt zu verlängern. ENO = FALSCH und die Zykluszeitüberwachung wird nicht zurückgesetzt, wenn RE_TRIGR aus einem Anlauf-OB, einem Alarm-OB oder einem Fehler-OB aufgerufen wird.

Ab Firmware-Version 2.2 kann RE_TRIGR von jedem OB ausgeführt werden (auch Anlauf-, Alarm- und Fehler-OBs). Die Zykluszeit des Zielsystems kann jedoch nur um das 10-Fache der konfigurierten maximalen Zykluszeit verlängert werden.

Maximale Zykluszeit des PLC-Geräts einstellen

Konfigurieren Sie den Wert für die maximale Zykluszeit in der Gerätekonfiguration unter "Zykluszeit".

Tabelle 8- 140 Zykluszeitwerte

Zykluszeitüberwachung	Mindestwert	Höchstwert	Voreinstellung
Maximale Zykluszeit	1 ms	6000 ms	150 ms

Laufzeitüberwachung

Endet die maximale Zykluszeit, bevor der Zyklus vollständig ist, wird ein Fehler erzeugt. Wenn das Anwenderprogramm einen Zeitfehleralarm-OB (OB 80) enthält, führt die CPU den Zeitfehleralarm-OB aus. Dieser kann Logik enthalten, um eine besondere Reaktion zu bewirken.

Wenn das Anwenderprogramm keinen Zeitfehleralarm-OB enthält, wird die erste Timeout-Bedingung ignoriert und die CPU bleibt im Betriebszustand RUN. Wenn in demselben Programmzyklus eine zweite Überschreitung der maximalen Zykluszeit auftritt (2 x maximaler Zykluszeitwert), wird ein Fehler ausgelöst, der bewirkt, dass das Zielsystem in den Betriebszustand STOP geht.

Im Betriebszustand STOP wird die Ausführung des Programms angehalten, die CPU-Systemkommunikation und die Systemdiagnose laufen jedoch weiter.

8.8.7 Anweisung STP (Programm beenden)

Tabelle 8- 141 Anweisung STP

KOP/FUP	SCL	Beschreibung
	`STP();`	STP versetzt die CPU in den Betriebszustand STOP. Ist die CPU im Betriebszustand STOP, so werden die Ausführung des Programms und die Aktualisierung der Ein- und Ausgänge im Prozessbild angehalten.

Weitere Informationen finden Sie unter: Konfigurieren der Ausgänge für den Wechsel von RUN in STOP (Seite 111).

Ist EN = WAHR, so geht die CPU in den Betriebszustand STOP, die Programmausführung endet und der Zustand von ENO ist bedeutungslos. Sonst ist EN = ENO = 0.

8.8.8 Anweisungen GET_ERROR und GET_ERROR_ID (Fehler lokal abrufen und Fehler-ID lokal abrufen)

Die Anweisungen GET_ERROR liefern Informationen über Fehler in der Ausführung von Programmbausteinen. Wenn Sie eine Anweisung GET_ERROR oder GET_ERROR_ID in Ihren Codebaustein aufnehmen, können Sie Programmfehler in Ihrem Programmbaustein behandeln.

GET_ERROR

Tabelle 8- 142 Anweisung GET_ERROR

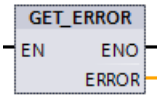
KOP/FUP	SCL	Beschreibung
	`GET_ERROR(_out_);`	Zeigt an, wenn ein Fehler in der Ausführung eines lokalen Programmbausteins aufgetreten ist und belegt eine vordefinierte Fehlerdatenstruktur mit ausführlichen Fehlerinformationen.

Tabelle 8- 143 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
ERROR	ErrorStruct	Fehlerdatenstruktur: Sie können die Struktur, nicht jedoch die Elemente innerhalb der Struktur umbenennen.

Tabelle 8- 144 Elemente der ErrorStruct-Datenstruktur

Strukturkomponenten		Datentyp	Beschreibung
ERROR_ID		Word	Fehler-ID
FLAGS		Byte	Zeigt an, ob während eines Bausteinaufrufs ein Fehler aufgetreten ist. • 16#01: Fehler während eines Bausteinaufrufs. • 16#00: Kein Fehler während eines Bausteinaufrufs.
REACTION		Byte	Standardreaktion: • 0: Ignorieren (Schreibfehler), • 1: Mit Ersatzwert "0" fortfahren (Lesefehler), • 2: Anweisung überspringen (Systemfehler)
CODE_ADDRESS		CREF	Informationen zur Adresse und zum Bausteintyp
	BLOCK_TYPE	Byte	Typ des Bausteins, in dem der Fehler aufgetreten ist: • 1: OB • 2: FC • 3: FB
	CB_NUMBER	UInt	Nummer des Codebausteins
	OFFSET	UDInt	Verweis auf den internen Speicher

<table>
<tr><th>Strukturkomponenten</th><th>Datentyp</th><th>Beschreibung</th></tr>
<tr><td>MODE</td><td>Byte</td><td>Zugriffsart: Je nach Art des Zugriffs können die folgenden Informationen ausgegeben werden:
<table>
<tr><th>Betriebsart</th><th>(A)</th><th>(B)</th><th>(C)</th><th>(D)</th><th>(E)</th></tr>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>1</td><td></td><td></td><td></td><td></td><td>Offset</td></tr>
<tr><td>2</td><td></td><td></td><td>Area</td><td></td><td></td></tr>
<tr><td>3</td><td>Adresse</td><td>Umfang</td><td></td><td>Number (Num-mer)</td><td></td></tr>
<tr><td>4</td><td></td><td></td><td>Area</td><td></td><td>Offset</td></tr>
<tr><td>5</td><td></td><td></td><td>Area</td><td>DB-Nr.</td><td>Offset</td></tr>
<tr><td>6</td><td>PtrNr. /Zugr.</td><td></td><td>Area</td><td>DB-Nr.</td><td>Offset</td></tr>
<tr><td>7</td><td>PtrNr. /Zugr.</td><td>Stpl.-Nr./Umfang</td><td>Area</td><td>DB-Nr.</td><td>Offset</td></tr>
</table></td></tr>
<tr><td>OPERAND_NUMBER</td><td>UInt</td><td>Operandennummer des Maschinenbefehls</td></tr>
<tr><td>POINTER_NUMBER_ LOCATION</td><td>UInt</td><td>(A) Interner Pointer</td></tr>
<tr><td>SLOT_NUMBER_SCOPE</td><td>UInt</td><td>(B) Speicherbereich im internen Speicher</td></tr>
<tr><td>DATA_ADDRESS</td><td>NREF</td><td>Informationen zur Adresse eines Operanden</td></tr>
<tr><td>AREA</td><td>Byte</td><td>(C) Speicherbereich:
L: 16#40 – 4E, 86, 87, 8E, 8F, C0 – CE
I: 16#81
A: 16#82
M: 16#83
DB: 16#84, 85, 8A, 8B</td></tr>
<tr><td>DB_NUMBER</td><td>UInt</td><td>(D) Nummer des Datenbausteins</td></tr>
<tr><td>OFFSET</td><td>UDInt</td><td>(E) Relative Adresse des Operanden</td></tr>
</table>

GET_ERROR_ID

Tabelle 8- 145 Anweisung GetErrorID

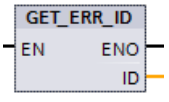
KOP/FUP	SCL	Beschreibung
	`GET_ERR_ID();`	Zeigt an, wenn ein Fehler in der Ausführung eines Programmbausteins aufgetreten ist und meldet die ID (Kennung) des Fehlers.

Tabelle 8- 146 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
ID	Word	Fehlerkennwerte für das ErrorStruct-Element ERROR_ID

Tabelle 8- 147 Error_ID-Werte

ERROR_ID hexa-dezimal	ERROR_ID dezimal	Ausführungsfehler Programmbaustein
0	0	Kein Fehler
2520	9504	Zeichenkette beschädigt
2522	9506	Lesefehler Operand außerhalb des Bereichs
2523	9507	Schreibfehler Operand außerhalb des Bereichs
2524	9508	Lesefehler ungültiger Bereich
2525	9509	Schreibfehler ungültiger Bereich
2528	9512	Lesefehler Datenausrichtung (fehlerhafte Bitausrichtung)
2529	9513	Schreibfehler Datenausrichtung (fehlerhafte Bitausrichtung)
252C	9516	Fehler nicht initialisierter Pointer
2530	9520	DB schreibgeschützt
2533	9523	Ungültiger Pointer
2538	9528	Zugriffsfehler: DB ist nicht vorhanden
2539	9529	Zugriffsfehler: Falscher DB
253A	9530	Globaler DB ist nicht vorhanden
253C	9532	Falsche Version oder FC ist nicht vorhanden
253D	9533	Anweisung ist nicht vorhanden
253E	9534	Falsche Version oder FB ist nicht vorhanden
253F	9535	Anweisung ist nicht vorhanden
2550	9552	Zugriffsfehler: DB ist nicht vorhanden
2575	9589	Fehler in der Schachtelungstiefe des Programms
2576	9590	Zuordnungsfehler Lokaldaten
2942	10562	Physischer Eingang ist nicht vorhanden
2943	10563	Physischer Ausgang ist nicht vorhanden

Funktionsweise

Standardmäßig reagiert die CPU auf einen Bausteinausführungsfehler durch Eintragen des Fehlers in den Diagnosepuffer. Wenn Sie jedoch eine oder mehrere Anweisungen GET_ERROR oder GET_ERROR_ID in einem Codebaustein anordnen, kann dieser Baustein die Fehler im Baustein behandeln. Dann meldet die CPU keinen Fehler im Diagnosepuffer. Stattdessen werden die Fehlerinformationen im Ausgang der Anweisung GET_ERROR oder GET_ERROR_ID gemeldet. Ausführliche Fehlerinformationen können Sie mit der Anweisung GET_ERROR auslesen, mit der Anweisung GET_ERROR_ID wird nur die Fehler-ID gelesen. Normalerweise ist der erste Fehler der wichtigste und die weiteren Fehler lediglich Folgen des ersten Fehlers.

Die erste Ausführung einer Anweisung GET_ERROR oder GET_ERROR_ID innerhalb eines Bausteins gibt den ersten während der Bausteinausführung erkannten Fehler aus. Dieser Fehler kann an einer beliebigen Stelle zwischen dem Start des Bausteins und der Ausführung von GET_ERROR oder GET_ERROR_ID aufgetreten sein. Nachfolgende Ausführungen von GET_ERROR oder GET_ERROR_ID geben den ersten Fehler nach der vorherigen Ausführung von GET_ERROR oder GET_ERROR_ID aus. Der Verlauf der Fehler wird nicht gespeichert, und die Ausführung der Anweisung initiiert das Zielsystem neu, um den nächsten Fehler zu erfassen.

Der Datentyp ErrorStruct der Anweisung GET_ERROR kann im Datenbausteineditor und in den Bausteinschnittstellen eingefügt werden, damit Ihre Programmlogik auf diese Werte zugreifen kann. Wählen Sie den Datentyp ErrorStruct in der Klappliste aus, um diese Struktur zu ergänzen. Mithilfe von eindeutigen Namen können Sie mehrere ErrorStruct-Elemente erstellen. Die Elemente eines ErrorStruct können nicht umbenannt werden.

Von ENO angezeigte Fehlerbedingung

Wenn gilt: EN = WAHR und GET_ERROR oder GET_ERROR_ID wird ausgeführt, dann:

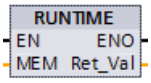
- ENO = WAHR gibt an, dass während der Ausführung eines Codebausteins ein Fehler aufgetreten ist und Fehlerdaten vorliegen
- ENO = FALSCH gibt an, dass während der Ausführung eines Codebausteins kein Fehler aufgetreten ist

Sie können die Logik des Fehlerverarbeitungsprogramms mit dem Eingang ENO verknüpfen, der nach dem Auftreten eines Fehlers gesetzt wird. Bei einem Fehler werden die Fehlerdaten im Ausgangsparameter gespeichert, wo Ihr Programm darauf zugreifen kann.

Mit GET_ERROR und GET_ERROR_ID können Fehlerinformationen des aktuellen Bausteins (des aufgerufenen Bausteins) an einen aufrufenden Baustein gesendet werden. Platzieren Sie die Anweisung im letzten Netzwerk des aufgerufenen Bausteinprogramms, um den endgültigen Ausführungsstatus des aufgerufenen Bausteins zu melden.

8.8.9 Anweisung RUNTIME (Programmlaufzeit messen)

Tabelle 8- 148 Anweisung RUNTIME

KOP/FUP	SCL	Beschreibung
	`Ret_Val := RUNTIME(` `  _lread_inout_);`	Misst die Laufzeit des gesamten Programms, einzelner Bausteine oder von Befehlssequenzen.

Soll die Laufzeit des gesamten Programms gemessen werden, ist Anweisung "Programmlaufzeit messen" in OB1 aufzurufen. Die Laufzeitmessung beginnt mit dem ersten Aufruf und Ausgang RET_VAL gibt die gemessene Programmlaufzeit nach dem zweiten Aufruf zurück. Die gemessene Laufzeit umfasst alle CPU-Prozesse, die während der Programmausführung auftreten können, zum Beispiel Unterbrechungen durch übergeordnete Ereignisse oder Kommunikation. Anweisung "Programmlaufzeit messen" liest einen internen Zähler der CPU und schreibt den Wert in den IN-OUT-Parameter MEM. Die Anweisung berechnet die aktuelle Programmlaufzeit anhand der internen Zählerfrequenz und schreibt sie in Ausgang RET_VAL.

Soll die Laufzeit einzelner Bausteine oder Befehlssequenzen gemessen werden, werden drei separate Netzwerke benötigt. Rufen Sie Anweisung "Programmlaufzeit messen" in einem einzelnen Netzwerk Ihres Programms auf. Sie setzen den Startpunkt der Laufzeitmessung mit dem ersten Aufruf der Anweisung. Rufen Sie dann den benötigten Programmbaustein oder die Befehlssequenz im nächsten Netzwerk auf. Rufen Sie Anweisung "Programmlaufzeit messen" in einem anderen Netzwerk ein zweites Mal auf und weisen Sie dem IN-OUT-Parameter MEM den gleichen Speicher wie beim ersten Aufruf der Anweisung zu. Die Anweisung "Programmlaufzeit messen" im dritten Netzwerk liest einen CPU-internen Zähler, berechnet die aktuelle Laufzeit der Programmbausteine oder der Befehlssequenz anhand der internen Zählerfrequenz und schreibt diese in Ausgang RET_VAL.

Anweisung "Programmlaufzeit messen" nutzt einen internen Hochfrequenzzähler für die Berechnung der Zeit. Bei Zählerüberlauf gibt die Anweisung Werte <= 0.0 zurück. Diese Laufzeitwerte können ignoriert werden.

Hinweis

Die CPU kann die Laufzeit einer Befehlssequenz nicht genau bestimmen, weil sich die Anweisungsfolge innerhalb einer Befehlssequenz während der optimierten Programmkompilierung ändert.

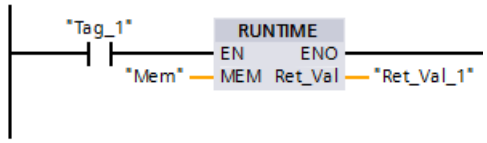
Tabelle 8- 149 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
MEM	LReal	Startpunkt für die Laufzeitmessung
RET_VAL	LReal	Gemessene Laufzeit in Sekunden

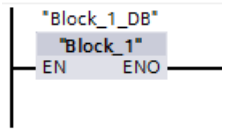
Beispiel: Anweisung RUNTIME

Das folgende Beispiel zeigt, wie Anweisung RUNTIME für die Messung der Ausführungszeit eines Funktionsbausteins verwendet wird:

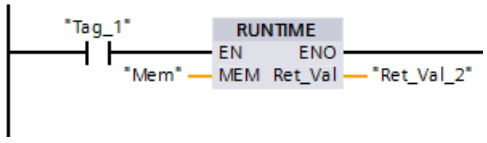
Netzwerk 1:



Netzwerk 2:



Netzwerk 3:



Wenn der Operand "Tag_1" in Netzwerk 1 Signalzustand "1" hat, wird Anweisung RUNTIME ausgeführt. Der Startpunkt für die Laufzeitmessung wird mit dem ersten Aufruf der Anweisung gesetzt und als Referenz für den zweiten Aufruf der Anweisung in Operand "Mem" gepuffert.

Funktionsbaustein FB1 wird in Netzwerk 2 ausgeführt.

Wenn Programmbaustein FN1 ausgeführt ist und Operand "Tag_1" Signalzustand "1" hat, wird Anweisung RUNTIME in Netzwerk 3 ausgeführt. Der zweite Aufruf der Anweisung berechnet die Laufzeit des Programmbausteins und schreibt das Ergebnis in Ausgang RET_VAL_2.

8.8.10 Programmsteuerungsanweisungen in SCL

8.8.10.1 Übersicht über die Programmsteuerungsanweisungen in SCL

Structured Control Language (SCL) bietet drei Arten von Programmsteuerungsanweisungen für die Strukturierung Ihres Anwenderprogramms:

- Auswahlanweisungen: Mit Hilfe einer Auswahlanweisung können Sie den Programmfluss in alternative Anweisungsfolgen steuern.
- Schleifen: Sie können die Schleifenausführung über Wiederholungsanweisungen steuern. Eine Wiederholungsanweisung gibt an, welche Teile eines Programm abhängig von bestimmten Bedingungen wiederholt werden sollen.
- Programmsprünge: Ein Programmsprung bewirkt einen direkten Sprung zu einem angegebenen Sprungziel und damit zu einer anderen Anweisung in demselben Baustein.

Die Programmsteuerungsanweisungen verwenden die Syntax der Programmiersprache PASCAL.

Tabelle 8- 150 Arten von Programmsteuerungsanweisungen in SCL

Programmsteuerungsanweisung		Beschreibung
Auswahl	IF-THEN-Anweisung (Seite 306)	Ermöglicht Ihnen, den Programmfluss in Abhängigkeit von einer Bedingung, die entweder WAHR oder FALSCH ist, in eine von zwei Alternativen zu verzweigen.
	CASE-Anweisung (Seite 307)	Ermöglicht die selektive Ausführung einer von *n* alternativen Verzweigungen, basierend auf dem Wert einer Variablen.
Schleife	FOR-Anweisung (Seite 308)	Dient zur Wiederholung einer Folge von Anweisungen, solange die Steuervariable innerhalb des angegebenen Wertebereichs liegt.
	WHILE-DO-Anweisung (Seite 310)	Dient zur Wiederholung einer Folge von Anweisungen, solange eine Durchführungsbedingung erfüllt ist.
	REPEAT-UNTIL-Anweisung (Seite 311)	Dient zur Wiederholung einer Folge von Anweisungen, bis eine Abbruchbedingung erfüllt ist.
Programmsprung	CONTINUE-Anweisung (Seite 312)	Dient zum Abbruch der Ausführung des momentanen Schleifendurchlaufs.
	EXIT-Anweisung (Seite 313)	Dient zum Verlassen einer Schleife an beliebiger Stelle und unabhängig vom Erfülltsein der Abbruchbedingung.
	GOTO-Anweisung (Seite 314)	Bewirkt den sofortigen Sprung zu einer angegebenen Sprungmarke.
	RETURN-Anweisung (Seite 314)	Bewirkt das Verlassen des gerade ausgeführten Codebausteins und die Rückkehr zum aufrufenden Codebaustein.

8.8.10.2 IF-THEN-Anweisung

Die IF-THEN-Anweisung ist eine bedingte Anweisung, die den Programmfluss steuert, indem eine Gruppe von Anweisungen basierend auf der Auswertung eines Bool-Werts eines logischen Ausdrucks ausgeführt wird. Die Ausführung mehrerer IF-THEN-Anweisungen können Sie auch mit Hilfe von Klammern verschachteln bzw. strukturieren.

Tabelle 8- 151 Elemente der IF-THEN-Anweisung

SCL	Beschreibung

`IF "Bedingung" THEN`
`  Anweisung_A;`
`  Anweisung_B;`
`  Anweisung_C;`
`  ;`
	Falls "Bedingung" WAHR oder 1 ist, dann die folgenden Anweisungen bis zur Anweisung END_IF ausführen.
	Falls "Bedingung" FALSCH oder 0 ist, dann bis zur Anweisung END_IF überspringen (es sei denn, das Programm enthält optionale ELSIF- oder ELSE-Anweisungen).

`[ELSIF "Bedingung-n" THEN`
`  Anweisung_N;`
`  ;]`
	Die optionale ELSEIF[1]-Anweisung bietet weitere auszuwertende Bedingungen. Beispiel: Falls "Bedingung" in der IF-THEN-Anweisung FALSCH ist, wertet das Programm "Bedingung-n" aus. Falls "Bedingung-n" WAHR ist, dann "Anweisung_N" ausführen.

`[ELSE`
`  Anweisung_X;`
`  ;]`
	Die optionale ELSE-Anweisung bietet Anweisungen, die auszuführen sind, wenn die "Bedingung" der IF-THEN-Anweisung FALSCH ist.

`END_IF;`	Die END_IF-Anweisung beendet die IF-THEN-Anweisung.

[1] Sie können in eine IF-THEN-Anweisung mehrere ELSIF-Anweisungen einfügen.

Tabelle 8- 152 Variablen der IF-THEN-Anweisung

Variablen	Beschreibung
"Bedingung"	Erforderlich. Der logische Ausdruck ist entweder WAHR (1) oder FALSCH (0).
"Anweisung_A"	Optional. Eine oder mehrere Anweisungen, die auszuführen sind, wenn "Bedingung" WAHR ist.
"Bedingung-n"	Optional. Der logische Ausdruck, der von der optionalen ELSIF-Anweisung auszuwerten ist.
"Anweisung_N"	Optional. Eine oder mehrere Anweisungen, die auszuführen sind, wenn "Bedingung-n" der ELSIF-Anweisung WAHR ist.
"Anweisung_X"	Optional. Eine oder mehrere Anweisungen, die auszuführen sind, wenn "Bedingung" der IF-THEN-Anweisung FALSCH ist.

Eine IF-Anweisung wird entsprechend den folgenden Regeln ausgeführt:

- Die erste Anweisungsfolge, deren logischer Ausdruck = WAHR ist, wird ausgeführt. Die übrigen Anweisungsfolgen werden nicht ausgeführt.
- Wenn kein Boolescher Ausdruck = WAHR ist, wird die von ELSE eingeführte Anweisungsfolge ausgeführt (oder keine Anweisungsfolge, falls keine ELSE-Verzweigung vorhanden ist).
- Die Anzahl der ELSIF-Anweisungen ist nicht begrenzt.

Hinweis

Die Verwendung einer oder mehrerer ELSIF-Verzweigungen hat gegenüber einer Folge von IF-Anweisungen den Vorteil, dass die logischen Ausdrücke, die auf einen gültigen Ausdruck folgen, nicht mehr ausgewertet werden. Die Laufzeit eines Programms kann dadurch verringert werden.

8.8.10.3 CASE-Anweisung

Tabelle 8- 153 Elemente der CASE-Anweisung

SCL	Beschreibung

`CASE "Testwert" OF`
`  "Werteliste": Anweisung[; Anweisung, ...]`
`  "Werteliste": Anweisung[; Anweisung, ...]`
`[ELSE`
`Else-Anweisung[; Else-Anweisung, ...]]`
`END_CASE;`
	Die Anweisung CASE führt eine von mehreren Anweisungsgruppen abhängig vom Wert eines Ausdrucks aus.

Tabelle 8- 154 Parameter

Parameter	Beschreibung
"Testwert"	Erforderlich. Beliebiger numerischer Ausdruck vom Datentyp Int.
"Werteliste"	Erforderlich. Ein einzelner Wert oder eine Liste mit durch Komma getrennten Werten oder Wertebereichen. (Einen Wertebereich geben Sie mittels zwei Punkten an: 2..8). Das folgende Beispiel zeigt die verschiedenen Varianten von Wertelisten: 1: Anweisung_A; 2, 4: Anweisung _B; 3, 5..7,9: Anweisung _C;
Anweisung	Erforderlich. Eine oder mehrere Anweisungen, die ausgeführt werden, wenn "Testwert" einem Wert in der Werteliste entspricht.
Else-Anweisung	Optional. Eine oder mehrere Anweisungen, die ausgeführt werden, wenn kein Wert der "Werteliste" eine Übereinstimmung ergibt.

Eine CASE-Anweisung wird entsprechend den folgenden Regeln ausgeführt:

- Der Testwert-Ausdruck muss einen Wert vom Typ Int zurückgeben.
- Bei der Verarbeitung einer Anweisung CASE prüft das Programm, ob der Wert des Testwert-Ausdrucks in einer angegebenen Werteliste enthalten ist. Wird eine Übereinstimmung gefunden, wird die der Liste zugewiesene Anweisungskomponente ausgeführt.
- Wird keine Übereinstimmung gefunden, wird der auf ELSE folgende Programmabschnitt ausgeführt. Falls keine ELSE-Verzweigung vorhanden ist, wird keine Anweisung ausgeführt.

Beispiel: Verschachtelte CASE-Anweisungen

CASE-Anweisungen können verschachtelt werden. Jede verschachtelte CASE-Anweisung benötigt eine zugehörige END_CASE-Anweisung.

```
CASE "var1" OF
       1 : #var2 := 'A';
       2 : #var2 := 'B';
ELSE
       CASE "var3" OF

          65..90: #var2 := 'Großbuchstabe';
          97..122: #var2 := 'Kleinbuchstabe';
       ELSE

          #var2:= 'Sonderzeichen';
       END_CASE;
END_CASE;
```

8.8.10.4 FOR-Anweisung

Tabelle 8- 155 Elemente der FOR-Anweisung

SCL	Beschreibung
`FOR "Steuervariable" := "Anfang" TO "Ende" [BY "Inkrement"] DO Anweisung; ; END_FOR;`	Eine FOR-Anweisung dient zur Wiederholung einer Folge von Anweisungen, solange eine Steuervariable innerhalb des angegebenen Wertebereichs liegt. Die Definition einer Schleife mit FOR umfasst die Angabe eines Anfangs- und eines Endwerts. Beide Werte müssen denselben Typ haben wie die Steuervariable. FOR-Schleifen können Sie verschachteln. Die END_FOR-Anweisung bezieht sich auf die zuletzt ausgeführte FOR-Anweisung.

Tabelle 8- 156 Parameter

Parameter	Beschreibung
"Steuervariable"	Erforderlich. Eine Ganzzahl (Int oder DInt), die als Schleifenzähler dient.
"Anfang"	Erforderlich. Einfacher Ausdruck, der den Anfangswert der Steuervariablen angibt.
"Ende"	Erforderlich. Einfacher Ausdruck, der den Abschlusswert der Steuervariablen angibt.
"Inkrement"	Optional. Betrag, um den eine "Steuervariable" nach jeder Schleife geändert wird. Das "Inkrement" hat denselben Datentyp wie "Steuervariable". Wenn der Wert von "Inkrement" nicht angegeben ist, wird der Wert der Variablen nach jeder Schleife um 1 erhöht. Sie können "Inkrement" nicht während der Ausführung der FOR-Anweisung ändern.

Die FOR-Anweisung wird wie folgt ausgeführt:

- Beim Start der Schleife wird die Steuervariable auf den Anfangswert gesetzt (Anfangszuweisung) und bei jedem Schleifendurchlauf um das angegebene Inkrement erhöht (positives Inkrement) oder verringert (negatives Inkrement), bis der Endwert erreicht ist.
- Nach jedem Schleifendurchlauf wird geprüft, ob die Bedingung (Endwert erreicht) erfüllt ist oder nicht. Ist die Endbedingung nicht erfüllt, so wird die Anweisungssequenz erneut ausgeführt; andernfalls endet die Schleife, und die Ausführung wird mit der unmittelbar auf die Schleife folgenden Anweisung fortgesetzt.

Regeln für die Formulierung von FOR-Anweisungen:

- Die Steuervariable darf nur vom Datentyp Int oder DInt sein.
- Sie können die Anweisung BY [Inkrement] weglassen. Wenn kein Inkrement angegeben wird, wird automatisch ein Inkrement von +1 angenommen.

Um die Schleife unabhängig vom Zustand des Ausdrucks "Bedingung" zu beenden, verwenden Sie die Anweisung EXIT-Anweisung (Seite 313). Die EXIT-Anweisung führt die Anweisung aus, die unmittelbar auf die END_FOR-Anweisung folgt.

Mit der Anweisung CONTINUE-Anweisung (Seite 312) überspringen Sie die nachfolgenden Anweisungen einer FOR-Schleife und setzen die Schleife mit der Überprüfung fort, ob die Bedingung für die Beendigung erfüllt ist.

8.8.10.5 WHILE-DO-Anweisung

Tabelle 8- 157 WHILE-Anweisung

SCL	Beschreibung
WHILE "Bedingung" DO **Anweisung;** **Anweisung;** **...;** **END_WHILE;**	Die WHILE-Anweisung führt eine Anweisungsfolge aus, bis eine vorgegebene Bedingung WAHR ist. WHILE-Schleifen können Sie verschachteln. Die END_WHILE-Anweisung bezieht sich auf die zuletzt ausgeführte WHILE-Anweisung.

Tabelle 8- 158 Parameter

Parameter	Beschreibung
"Bedingung"	Erforderlich. Ein logischer Ausdruck, dessen Auswertung WAHR oder FALSCH ergibt. (Die Bedingung "Null" wird als FALSCH ausgewertet.)
Anweisung	Optional. Eine oder mehrere Anweisungen, die ausgeführt werden, bis die Bedingung WAHR ist.

Hinweis

Die WHILE-Anweisung wertet den Zustand der "Bedingung" aus, bevor die Anweisungen ausgeführt werden. Um die Anweisungen unabhängig vom Zustand der "Bedingung" mindestens einmal auszuführen, verwenden Sie die REPEAT-Anweisung (Seite 311).

Die WHILE-Anweisung wird entsprechend den folgenden Regeln ausgeführt:

- Vor jeder Ausführung des Schleifeninhalts, wird die Ausführungsbedingung ausgewertet.
- Der auf DO folgende Schleifeninhalt wird solange wiederholt, bis die Ausführungsbedingung den Wert WAHR annimmt.
- Tritt der Wert FALSCH auf, wird die Schleife übersprungen und die auf die Schleife folgende Anweisung wird ausgeführt.

Um die Schleife unabhängig vom Zustand des Ausdrucks "Bedingung" zu beenden, verwenden Sie die EXIT-Anweisung (Seite 313). Die EXIT-Anweisung führt die Anweisung aus, die unmittelbar auf die END_WHILE-Anweisung folgt.

Mit der CONTINUE-Anweisung überspringen Sie die nachfolgenden Anweisungen einer WHILE-Schleife und setzen die Schleife mit der Überprüfung fort, ob die Bedingung für die Beendigung erfüllt ist.

8.8.10.6 REPEAT-UNTIL-Anweisung

Tabelle 8- 159 REPEAT-Anweisung

SCL	Beschreibung
REPEAT **Anweisung;** **;** **UNTIL "Bedingung"** **END_REPEAT;**	Die REPEAT-Anweisung führt eine Gruppe von Anweisungen aus, bis eine vorgegebene Bedingung WAHR ist. REPEAT-Schleifen können Sie verschachteln. Die END_REPEAT-Anweisung bezieht sich immer auf die zuletzt ausgeführte REPEAT-Anweisung.

Tabelle 8- 160 Parameter

Parameter	Beschreibung
Anweisung	Optional. Eine oder mehrere Anweisungen, die ausgeführt werden, bis die Bedingung WAHR ist.
"Bedingung"	Erforderlich. Einer oder mehrere Ausdrücke der zwei folgenden Arten: Ein numerischer Ausdruck oder ein Zeichenkettenausdruck, dessen Auswertung WAHR oder FALSCH ergibt. Die Bedingung "Null" wird als FALSCH ausgewertet.

Hinweis

Vor der Auswertung des Zustands der "Bedingung" führt die REPEAT-Anweisung die Anweisungen während der ersten Durchführung der Schleife aus (auch wenn die "Bedingung" FALSCH ist). Um den Zustand der "Bedingung" vor der Ausführung der Anweisungen zu prüfen, verwenden Sie die WHILE-Anweisung (Seite 310).

Um die Schleife unabhängig vom Zustand des Ausdrucks "Bedingung" zu beenden, verwenden Sie die EXIT-Anweisung (Seite 313). Die EXIT-Anweisung führt die Anweisung aus, die unmittelbar auf die END_REPEAT-Anweisung folgt.

Mit der CONTINUE-Anweisung (Seite 312) überspringen Sie die nachfolgenden Anweisungen einer REPEAT-Schleife und setzen die Schleife mit der Überprüfung fort, ob die Bedingung für die Beendigung erfüllt ist.

8.8.10.7 CONTINUE-Anweisung

Tabelle 8- 161 CONTINUE-Anweisung

SCL	Beschreibung
CONTINUE **Anweisung;** **;**	Die CONTINUE-Anweisung überspringt die nachfolgenden Anweisungen einer Programmschleife (FOR, WHILE, REPEAT) und setzt die Schleife mit der Überprüfung fort, ob die Bedingung für die Beendigung erfüllt ist. Ist dies nicht der Fall, wird die Schleife fortgesetzt.

Die CONTINUE-Anweisung wird entsprechend den folgenden Regeln ausgeführt:

- Diese Anweisung beendet die Ausführung des Schleifeninhalts sofort.
- Abhängig davon, ob die Bedingung für die Wiederholung der Schleife erfüllt ist oder nicht, wird der Schleifeninhalt erneut ausgeführt oder die Wiederholungsanweisung beendet und die unmittelbar nachfolgende Anweisung ausgeführt.
- In einer FOR-Anweisung wird die Steuervariable direkt nach einer CONTINUE-Anweisung um das angegebene Inkrement erhöht.

Verwenden Sie die CONTINUE-Anweisung nur innerhalb einer Schleife. In verschachtelten Schleifen bezieht sich die CONTINUE-Anweisung immer auf die Schleife, in der sich die Anweisung befindet. CONTINUE wird üblicherweise in Verbindung mit einer IF-Anweisung verwendet.

Wenn die Schleife unabhängig von der Beendigungsprüfung beendet werden soll, verwenden Sie die EXIT-Anweisung.

Beispiel: CONTINUE -Anweisung

Das folgende Beispiel zeigt die Verwendung der CONTINUE-Anweisung, mit der beim Berechnen eines Prozentwerts der Fehler "Division durch 0" verhindert werden soll:

```
FOR i := 0 TO 10 DO
IF wert[i] = 0 THEN CONTINUE; END_IF;
     p := part / value[i] * 100;
     s := INT_TO_STRING(p);
     percent := CONCAT(IN1:=s, IN2:="%");
END_FOR;
```

8.8.10.8 EXIT-Anweisung

Tabelle 8- 162 Anweisung EXIT

SCL	Beschreibung
EXIT;	Die EXIT-Anweisung dient zum Beenden einer Schleife (FOR, WHILE oder REPEAT) an beliebiger Stelle und unabhängig davon, ob die Beendigungsbedingung erfüllt ist.

Die EXIT-Anweisung wird entsprechend den folgenden Regeln ausgeführt:

- Diese Anweisung bewirkt, dass die Wiederholungsanweisung, die die EXIT-Anweisung unmittelbar umgibt, sofort beendet wird.
- Die Ausführung des Programm wird nach dem Ende der Schleife fortgesetzt (z. B. nach END_FOR).

Verwenden Sie die EXIT-Anweisung innerhalb einer Schleife. In verschachtelten Schleifen gibt die EXIT-Anweisung die Verarbeitung an die nächsthöhere Schachtelungebene zurück.

Beispiel: EXIT -Anweisung

```
FOR i := 0 TO 10 DO
CASE Wert[i, 0] OF
 1..10: Wert [i, 1]:="A";
 11..40: Wert [i, 1]:="B";
 41..100: Wert [i, 1]:="C";
ELSE
EXIT;
END_CASE;
END_FOR;
```

8.8.10.9 GOTO-Anweisung

Tabelle 8- 163 GOTO-Anweisung

SCL	Beschreibung

`GOTO JumpLabel;`
`Anweisung;`
`... ;`
`JumpLabel: Anweisung;`
	Die GOTO-Anweisung überspringt Anweisungen, indem sie zu einer Sprungmarke in demselben Baustein springt.
	Die Sprungmarke ("JumpLabel") und die GOTO-Anweisung müssen sich in demselben Baustein befinden. Der Name einer Sprungmarke darf innerhalb eines Bausteins nur einmal vergeben werden. Jede Sprungmarke kann Ziel verschiedener GOTO-Anweisungen sein.

Es ist nicht möglich, in einen Schleifenabschnitt zu springen (FOR, WHILE oder REPEAT). Es ist möglich, aus einer Schleife herauszuspringen.

Beispiel: GOTO -Anweisung

Im folgenden Beispiel gilt: Je nach Wert des Operanden "Variablenwert" wird die Ausführung des Programms an dem Punkt wieder aufgenommen, der von der entsprechenden Sprungmarke vorgegeben wird. Ist "Variablenwert" gleich 2, nimmt das Programm die Ausführung an Sprungmarke "MyLabel2" wieder auf und überspringt "MyLabel1".

```
CASE "Variablenwert" OF
1 : GOTO MeineBeschriftung1;
2 : GOTO MeineBeschriftung2;
ELSE GOTO MeineBeschriftung3;
END_CASE;
MeineBeschriftung1: "Variable_1" := 1;
MeineBeschriftung2: "Variable_2" := 1;
MeineBeschriftung3: "Variable_4" := 1;
```

8.8.10.10 RETURN-Anweisung

Tabelle 8- 164 Anweisung RETURN

SCL	Beschreibung
`RETURN;`	Die RETURN-Anweisung beendet den Codebaustein, der gerade ausgeführt wird, ohne Bedingungen. Die Programmausführung kehrt zum aufrufenden Baustein oder zum Betriebssystem zurück (beim Beenden eines OB).

Beispiel: RETURN -Anweisung:

```
IF "Fehler" <> 0 THEN
RETURN;
END_IF;
```

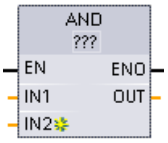
Hinweis

Nach der Ausführung der letzten Anweisung kehrt der Codebaustein automatisch zum aufrufenden Baustein zurück. Fügen Sie eine RETURN-Anweisung nicht am Ende des Codebausteins ein.

8.9 Wortverknüpfung

8.9.1 Verknüpfungsoperationen AND (UND), OR (ODER) und XOR (EXKLUSIV ODER)

Tabelle 8- 165 Verknüpfungsoperationen AND (UND), OR (ODER) und XOR (EXKLUSIV ODER)

KOP/FUP	SCL	Beschreibung
	`out := in1 AND in2;`	AND: UND-Verknüpfung
	`out := in1 OR in2;`	OR: ODER-Verknüpfung
	`out := in1 XOR in2;`	XOR: EXKLUSIV-ODER-Verknüpfung

1 In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Um einen Eingang hinzuzufügen, klicken Sie auf das Symbol "Erstellen" oder an einem der vorhandenen Parameter IN mit der rechten Maustaste auf den Eingangsanschluss und wählen den Befehl "Eingang einfügen".

Um einen Eingang zu löschen, klicken Sie bei einem der vorhandenen Parameter IN mit der rechten Maustaste auf den Eingangsanschluss (sofern mehr als die zwei ursprünglichen Eingänge vorhanden sind) und wählen den Befehl "Löschen".

Tabelle 8- 166 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN1, IN2	Byte, Word, DWord	Verknüpfungseingänge
OUT	Byte, Word, DWord	Verknüpfungsausgang

1 Durch die Auswahl des Datentyps werden die Parameter IN1, IN2 und OUT auf den gleichen Datentyp gesetzt.

Die entsprechenden Bitwerte von IN1 und IN2 werden zu einem binären Ergebnis in Parameter OUT verknüpft. ENO ist nach der Ausführung dieser Anweisungen immer WAHR.

8.9.2 Anweisung INV (Einerkomplement erzeugen)

Tabelle 8- 167 Anweisung INV

KOP/FUP	SCL	Beschreibung
	`Nicht verfügbar`	Berechnet das binäre Einerkomplement des Parameters IN. Das Einerkomplement wird gebildet durch Invertieren jedes Bitwerts des Parameters IN (Ändern von 0 in 1 und 1 in 0). ENO ist nach der Ausführung dieser Anweisung immer WAHR.

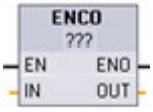
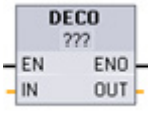
[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 168 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	SInt, Int, DInt, USInt, UInt, UDInt, Byte, Word, DWord	Datenelement für die Invertierung
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Byte, Word, DWord	Invertierter Ausgangswert

8.9.3 Anweisungen DECO (Decodieren) and ENCO (Encodieren)

Tabelle 8- 169 Anweisungen ENCO und DECO

KOP/FUP	SCL	Beschreibung
	`out := ENCO(_in_);`	Wandelt ein Bitmuster in eine Binärzahl um. Die Anweisung ENCO wandelt den Parameter IN in die Binärzahl um, die der Bitposition des niederwertigsten Bits von Parameter IN entspricht, und gibt das Ergebnis in Parameter OUT aus. Ist Parameter IN entweder 0000 0001 oder 0000 0000, so wird der Wert 0 an Parameter OUT zurückgegeben. Ist der Wert des Parameters IN gleich 0000 0000, dann wird ENO auf FALSCH gesetzt.
	`out := DECO(_in_);`	Wandelt eine Binärzahl in ein Bitmuster um. Die Anweisung DECO wandelt eine Binärzahl aus Parameter IN um, indem die entsprechende Bitposition in Parameter OUT auf 1 gesetzt wird (alle anderen Bits werden auf 0 gesetzt). ENO ist nach der Ausführung der Anweisung DECO immer WAHR. Hinweis: Der Standarddatentyp für die Anweisung DECO ist DWORD. In SCL ändern Sie den Anweisungsnamen in DECO_BYTE oder DECO_WORD, um einen Byte- oder Wortwert zu decodieren, und Sie weisen eine Byte- oder Wortvariable oder -adresse zu.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 170 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	ENCO: Byte, Word, DWord DECO: UInt	ENCO: Bitmuster für die Umwandlung DECO: Binärwert für die Umwandlung
OUT	ENCO: Int DECO: Byte, Word, DWord	ENCO: Umgewandelter Wert DECO: Umgewandeltes Bitmuster

Tabelle 8- 171 ENO-Status

ENO	Bedingung	Ergebnis (OUT)
1	Kein Fehler	Gültige Bitnummer
0	IN ist Null	OUT wird auf Null gesetzt

Die möglichen Datentypen Byte, Word oder DWord für den DECO-Parameter OUT begrenzen den nützlichen Bereich des Parameters IN. Wenn der Wert des Parameters IN den nützlichen Bereich überschreitet, wird eine Modulo-Operation durchgeführt, um die niederwertigsten Bits zu extrahieren (siehe unten).

Bereich für den DECO-Parameter IN:

- 3 Bits (Werte 0-7) IN werden verwendet, um 1 Bitposition in einem Byte OUT zu setzen
- 4 Bits (Werte 0-15) IN werden verwendet, um 1 Bitposition in einem Word OUT zu setzen
- 5 Bits (Werte 0-31) IN werden verwendet, um 1 Bitposition in einem DWord OUT zu setzen

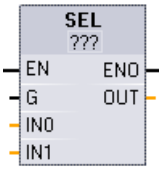
Tabelle 8- 172 Beispiele

Wert IN für DECO			Wert OUT für DECO (Einzelne Bitposition umwandeln)
Byte OUT 8 Bits	Min. IN	0	00000001
	Max. IN	7	10000000
Word OUT 16 Bits	Min. IN	0	0000000000000001
	Max. IN	15	1000000000000000
DWord OUT 32 Bits	Min. IN	0	00000000000000000000000000000001
	Max. IN	31	10000000000000000000000000000000

8.9.4 Anweisungen SEL (Selektieren), MUX (Multiplexen) und DEMUX (Demultiplexen)

Tabelle 8- 173 Anweisung SEL (Selektieren)

KOP/FUP	SCL	Beschreibung

	`out := SEL(`
	`g:=_bool_in,`
	`in0:-_variant_in,`
	`in1:=_variant_in);`
		Die Anweisung SEL weist einen von zwei Eingangswerten, abhängig vom Wert des Parameters G, dem Parameter OUT zu.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 8- 174 Datentypen für die Anweisung SEL

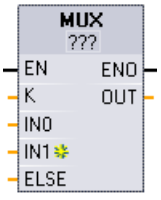
Parameter	Datentyp[1]	Beschreibung
G	Bool	• 0 wählt IN0 • 1 wählt IN1
IN0, IN1	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Eingänge
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Ausgang

[1] Die Eingangsvariablen und die Ausgangsvariable müssen vom gleichen Datentyp sein.

Bedingungscodes: ENO ist nach der Ausführung der Anweisung SEL immer WAHR.

Tabelle 8- 175 Anweisung MUX (Multiplexen)

KOP/FUP	SCL	Beschreibung

	`out := MUX(`
	`k:=_unit_in,`
	`in1:=variant_in,`
	`in2:=variant_in,`
	
	`[...in32:=variant_in,]`
	`inelse:=variant_in);`
		Die Anweisung MUX kopiert einen von vielen Eingangswerten, abhängig vom Wert des Parameters K, in den Parameter OUT. Überschreitet der Wert von Parameter K (IN*n* - 1), so wird der Wert des Parameters ELSE in den Parameter OUT kopiert.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Um einen Eingang hinzuzufügen, klicken Sie auf das Symbol "Erstellen" oder an einem der vorhandenen Parameter IN mit der rechten Maustaste auf den Eingangsanschluss und wählen den Befehl "Eingang einfügen".

Um einen Eingang zu löschen, klicken Sie bei einem der vorhandenen Parameter IN mit der rechten Maustaste auf den Eingangsanschluss (sofern mehr als die zwei ursprünglichen Eingänge vorhanden sind) und wählen den Befehl "Löschen".

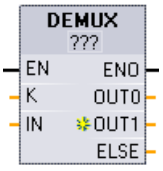
Tabelle 8- 176 Datentypen für die Anweisung MUX

Parameter	Datentyp	Beschreibung
K	UInt	• 0 wählt IN1 • 1 wählt IN2 • *n* wählt IN*n*
IN0, IN1, .. INn	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Eingänge
ELSE	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Eingang Ersatzwert (optional)
OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Ausgang

[1] Die Eingangsvariablen und die Ausgangsvariable müssen vom gleichen Datentyp sein.

Tabelle 8- 177 Anweisung DEMUX (Demultiplexen)

KOP/FUP	SCL	Beschreibung

	`DEMUX(`
	`k:=_unit_in,`
	`in:=variant_in,`
	`out1:=variant_in,`
	`out2:=variant_in,`
	
	`[...out32:=variant_in,]`
	
	`outelse:=variant_in);`
		DEMUX kopiert den Wert der Adresse, die dem Parameter IN zugewiesen ist, in einen von vielen Ausgängen. Der Wert des Parameters K gibt an, welcher Ausgang als Ziel des Werts IN ausgewählt wird. Ist der Wert von K größer als die Zahl (OUT*n* - 1), dann wird der Wert IN in die Adresse kopiert, die dem Parameter ELSE zugewiesen ist.

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Um einen Ausgang hinzuzufügen, klicken Sie auf das Symbol "Erstellen" oder an einem der vorhandenen Parameter OUT mit der rechten Maustaste auf den Ausgangsanschluss und wählen den Befehl "Ausgang einfügen".

Um einen Ausgang zu löschen, klicken Sie bei einem der vorhandenen Parameter OUT mit der rechten Maustaste auf den Ausgangsanschluss (sofern mehr als die zwei ursprünglichen Ausgänge vorhanden sind) und wählen den Befehl "Löschen".

Tabelle 8- 178 Datentypen für die Anweisung DEMUX

Parameter	Datentyp[1]	Beschreibung
K	UInt	Auswahlwert: • 0 wählt OUT1 • 1 wählt OUT2 • n wählt OUTn
IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Eingang
OUT0, OUT1, .. OUTn	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Ausgänge
ELSE	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal, Byte, Word, DWord, Time, Date, TOD, Char, WChar	Ausgang ersetzen, wenn K größer als (OUTn - 1) ist

[1] Die Eingangsvariable und die Ausgangsvariablen müssen vom gleichen Datentyp sein.

Tabelle 8- 179 ENO-Zustand für die Anweisungen MUX und DEMUX

ENO	Bedingung	Ergebnis OUT
1	Kein Fehler	MUX: Ausgewählter IN-Wert wird in OUT kopiert DEMUX: IN-Wert wird in ausgewählten OUT kopiert
0	MUX: K ist größer als die Anzahl der Eingänge - 1	• Kein ELSE angegeben: OUT wird nicht verändert, • ELSE angegeben, ELSE-Wert wird OUT zugewiesen
	DEMUX: K ist größer als die Anzahl der Ausgänge -1	• Kein ELSE angegeben: Ausgänge werden nicht verändert, • ELSE angegeben, IN-Wert wird in ELSE kopiert

8.10 Schieben und Rotieren

8.10.1 Anweisungen SHL (Rechts schieben) und SHL (Links schieben)

Tabelle 8- 180 Anweisungen SHR und SHL

KOP/FUP	SCL	Beschreibung

	`out := SHR(`
	`  in:=_variant_in_,`
	`  n:=_uint_in);`
	`out := SHL(`
	`  in:=_variant_in_,`
	`  n:=_uint_in);`
		Mit den Schiebeanweisungen (SHL und SHR) schieben Sie das Bitmuster von Parameter IN. Das Ergebnis wird dem Parameter OUT zugewiesen. Parameter N gibt die Anzahl der geschobenen Bitpositionen an:
		• SHR: Bitmuster nach rechts schieben
		• SHL: Bitmuster nach links schieben

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie die Datentypen in der Klappliste aus.

Tabelle 8- 181 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	Ganzzahlen	Bitmuster, das verschoben werden soll
N	USInt, UDint	Anzahl der Bitpositionen, die verschoben werden sollen
OUT	Ganzzahlen	Bitmuster nach dem Schieben

- Bei N=0 wird nicht geschoben. Der Wert von IN wird OUT zugewiesen.
- In die Bitpositionen, die von der Schiebeanweisung geleert wurden, werden Nullen geschoben.
- Falls die Anzahl der zu schiebenden Positionen (N) die Anzahl der Bits im Zielwert überschreitet (8 bei Byte, 16 bei Word, 32 bei DWord), dann werden alle ursprünglichen Bitwerte hinausgeschoben und durch Nullen ersetzt (OUT wird Null zugewiesen).
- ENO ist bei den Schiebeanweisungen immer WAHR.

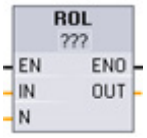
Tabelle 8- 182 Beispiel: SHL mit Word-Daten

Schiebt die Bits eines Worts nach links, indem von rechts Nullen eingefügt werden (N = 1)			
IN	1110 0010 1010 1101	Wert von OUT vor dem ersten Schieben:	1110 0010 1010 1101
		Nach dem ersten Schieben links:	1100 0101 0101 1010
		Nach dem zweiten Schieben links:	1000 1010 1011 0100
		Nach dem dritten Schieben links:	0001 0101 0110 1000

8.10.2 Anweisungen ROR (Rechts rotieren) und ROL (Links rotieren)

Tabelle 8- 183 Anweisungen ROR und ROL

KOP/FUP	SCL	Beschreibung

	`out := ROL(`
	`  in:=_variant_in_,`
	`  n:=_uint_in);`
	`out := ROR(`
	`  in:=_variant_in_,`
	`  n:=_uint_in);`
		Mit den Rotieranweisungen (ROR und ROL) können Sie das Bitmuster von Parameter IN rotieren. Das Ergebnis wird Parameter OUT zugewiesen. Parameter N legt die Zahl der rotierten Bitpositionen fest.
		• ROR: Bitmuster nach rechts rotieren
		• ROL: Bitmuster nach links rotieren

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie die Datentypen in der Klappliste aus.

Tabelle 8- 184 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	Ganzzahlen	Bitmuster, das rotiert werden soll
N	USInt, UDint	Anzahl der Bitpositionen, die rotiert werden sollen
OUT	Ganzzahlen	Bitmuster nach dem Rotieren

- Bei N=0 wird nicht rotiert. Der Wert von IN wird OUT zugewiesen.
- Bitdaten, die auf einer Seite des Zielwerts hinausrotiert werden, werden auf der anderen Seite des Zielwerts hineinrotiert, so dass keine ursprünglichen Bitwerte verloren gehen.
- Falls die Anzahl der zu rotierenden Bitpositionen (N) die Anzahl der Bits im Zielwert überschreitet (8 bei Byte, 16 bei Word, 32 bei DWord), dann wird trotzdem rotiert.
- ENO ist nach der Ausführung der Rotieranweisungen immer WAHR.

Tabelle 8- 185 Beispiel: ROR mit Word-Daten

Bits rechts hinausrotieren und links hineinrotieren (N = 1)			
IN	0100 0000 0000 0001	Wert von OUT vor dem ersten Rotieren:	0100 0000 0000 0001
		Nach dem ersten Rotieren rechts:	1010 0000 0000 0000
		Nach dem zweiten Rotieren rechts:	0101 0000 0000 0000

Erweiterte Anweisungen

9

9.1 Datums-, Uhrzeit- und Uhrfunktionen

9.1.1 Datums- und Uhrzeitanweisungen

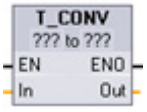
Verwenden Sie die Datum- und Uhrzeitanweisungen für Kalender- und Zeitberechnungen.

- T_CONV konvertiert einen Wert in oder aus (Datum- und Zeit-Datentypen) und (Byte-, Wort- und Doppelwort-Datentypen).
- T_ADD addiert Time- und DTL-Werte: (Time + Time = Time) oder (DTL + Time = DTL)
- T_SUB subtrahiert Time- und DTL-Werte: (Time - Time = Time) oder (DTL - Time = DTL)
- T_DIFF gibt die Differenz zwischen zwei DTL-Werten als Time-Wert aus: DTL - DTL = Time
- T_COMBINE verknüpft einen Date-Wert und einen Time_and_Date-Wert, um einen DTL-Wert zu erhalten

Informationen zum Format von DTL- und Time-Daten finden Sie im Abschnitt zu den Uhrzeit- und Datums-Datentypen (Seite 123).

Tabelle 9- 1 Anweisung T_CONV (Zeiten umwandeln und extrahieren)

KOP/FUP	SCL-Beispiel	Beschreibung

	`out := DINT_TO_TIME( in:=_variant_in);`
	
	`out := TIME_TO_DINT( in:=_variant_in);`
		T_CONV konvertiert einen Wert in oder aus (Datum- und Zeit-Datentypen) und (Byte-, Wort- und Doppelwort-Datentypen).

[1] In KOP- und FUP-Boxen: Klicken Sie auf die "???" und wählen Sie die Datentypen für Quelle und Ziel aus der Klappliste aus.

[2] In SCL: Ziehen Sie die Anweisung T_CONV aus dem Anweisungsverzeichnis in den Programmiereditor. Wählen Sie dann die Datentypen für Quelle und Ziel aus.

Tabelle 9- 2 Gültige Datentypen für T_CONV-Umwandlungen

Datentyp IN (oder OUT)	Datentypen OUT (oder IN)
TIME (Millisekunden)	DInt, Int, SInt, UDInt, UInt, USInt, TOD Nur in SCL: Byte, Word, Dword
DATE (Anzahl der Tage seit 1. Januar 1990)	DInt, Int, SInt, UDInt, UInt, USInt, DTL Nur in SCL: Byte, Word, Dword
TOD (Millisekunden seit Mitternacht - 24:00:00.000)	DInt, Int, SInt, UDInt, UInt, USInt, TIME, DTL Nur in SCL: Byte, Word, Dword

Hinweis

Mit T_CONV eine größere Datengröße in eine kleinere Datengröße umwandeln

Wenn Sie einen größeren Datentyp mit mehr Bytes in einen kleineren Datentyp mit weniger Bytes umwandeln, können Datenwerte abgeschnitten werden. Tritt dieser Fehler auf, wird ENO auf 0 gesetzt.

Umwandlung aus dem/in den Datentyp DTL

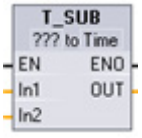
Der Datentyp DTL (Date and Time Long) enthält Daten für Jahr, Monat, Datum und Uhrzeit. DTL-Daten können in/aus Datentypen DATE und TOD umgewandelt werden.
Die DTL-Umwandlung nur mit Datentyp DATE betrifft jedoch nur die Werte für Jahr, Monat und Tag. Eine DTL-Umwandlung mit TOD-Daten betrifft nur die Werte für Stunden, Minuten und Sekunden.

Wenn T_CONV in DTL umwandelt, werden die nicht betroffenen Datenelemente im DTL-Format nicht verändert.

Tabelle 9- 3 Anweisungen T_ADD (Zeiten addieren) und T_SUB (Zeiten subtrahieren)

KOP/FUP	SCL	Beschreibung

	`out := T_ADD(`
	`  in1:=_variant_in,`
	`  in2:=_time_in);`
		T_ADD addiert den Wert aus Eingang IN1 (Datentyp DTL oder Time) zum Wert in Eingang IN2 (Datentyp Time). Der Parameter OUT liefert das Ergebnis als Wert vom Datentyp DTL oder Time. Zwei Datentyp-Anweisungen sind möglich:
		• Time + Time = Time
		• DTL + Time = DTL

	`out := T_SUB(`
	`  in1:=_variant_in,`
	`  in2:=_time_in);`
		T_SUB subtrahiert den Wert aus Eingang IN2 (Datentyp Time) vom Wert von IN1 (Datentyp DTL oder Time). Der Parameter OUT liefert den Differenzwert als Datentyp DTL oder Time. Zwei Datentyp-Anweisungen sind möglich:
		• Time - Time = Time
		• DTL - Time = DTL

[1] In KOP und FUP: Klicken Sie auf die "???" und wählen Sie die Datentypen in der Klappliste aus.

Tabelle 9- 4 Datentypen für die Parameter von T_ADD und T_SUB

Parameter und Datentyp		Datentyp	Beschreibung
IN1[1]	IN	DTL, Time	DTL- oder Time-Wert
IN2	IN	Time	Zu addierender oder subtrahierender Time-Wert
OUT	OUT	DTL, Time	DTL- oder Time-Summe oder Differenz

[1] Wählen Sie den Datentyp für IN1 aus der Klappliste unter dem Namen der Anweisung aus. Durch die Auswahl des Datentyps für IN1 wird auch der Datentyp für den Parameter OUT gesetzt.

Tabelle 9- 5 Anweisung T_DIFF (Zeitdifferenz)

KOP/FUP	SCL	Beschreibung

	`out := T_DIFF(`
	`in1:=_DTL_in,`
	`in2:=_DTL_in);`
		T_DIFF subtrahiert den DTL-Wert (IN2) vom DTL-Wert (IN1). Der Parameter OUT liefert den Differenzwert als Datentyp Time.
		• DTL - DTL = Time

Tabelle 9- 6 Datentypen für die Parameter von T_DIFF

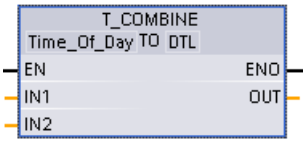
Parameter und Datentyp		Datentyp	Beschreibung
IN1	IN	DTL	DTL-Wert
IN2	IN	DTL	Zu subtrahierender DTL-Wert
OUT	OUT	Time	Time-Differenz

Bedingungscodes: ENO = 1 bedeutet, dass kein Fehler aufgetreten ist. ENO = 0 und Parameter OUT = 0 Fehler:

- Ungültiger DTL-Wert
- Ungültiger Time-Wert

Tabelle 9- 7 Anweisung T_COMBINE (Zeiten verknüpfen)

KOP/FUP	SCL	Beschreibung

	`out :=`
	`CONCAT_DATE_TOD(`
	`In1 := _date_in,`
	`In2 := _tod_in);`
		T_COMBINE verknüpft einen Date-Wert und einen Time_of_Day-Wert, um einen DTL-Wert zu erhalten.

[1] Beachten Sie, dass die Anweisung T_COMBINE in den erweiterten Anweisungen der Funktion CONCAT_DATE_TOD in SCL entspricht.

Tabelle 9- 8 Datentypen für die Parameter von T_COMBINE

Parameter und Datentyp		Datentyp	Beschreibung
IN1	IN	Date	Der zu verknüpfende Date-Wert muss zwischen DATE#1990-01-01 und DATE#2089-12-31 liegen
IN2	IN	Time_of_Day	Zu verknüpfende Time_of_Day-Werte
OUT	OUT	DTL	DTL-Wert

9.1.2 Uhrzeitfunktionen

⚠ WARNUNG

Wenn ein Angreifer über die NTP-Synchronisierung (Network Time Protocol) auf Ihre Netzwerke zugreifen kann, kann er möglicherweise durch Veränderung der CPU-Systemzeit in gewissem Rahmen Ihren Prozess kontrollieren.

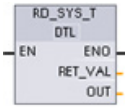
Die NTP-Client-Funktion der S7-1200 CPU ist standardmäßig deaktiviert und ermöglicht im aktivierten Zustand lediglich konfigurierten IP-Adressen, als NTP-Server zu fungieren. Die CPU deaktiviert diese Funktion standardmäßig, und Sie müssen die Funktion konfigurieren, um ferngesteuerte Korrekturen der CPU-Systemzeit zu gestatten.

Die S7-1200 CPU unterstützt Uhrzeitalarme und Uhrzeitoperationen, die von einer korrekten CPU-Systemzeit abhängig sind. Wenn Sie NTP konfigurieren und die Uhrzeitsynchronisierung von einem Server akzeptieren, müssen Sie sicherstellen, dass es sich bei dem Server um eine vertrauenswürdige Quelle handelt. Ist dies nicht der Fall, kann es zu einer Sicherheitsverletzung kommen, bei der ein unbekannter Benutzer die CPU-Systemzeit verändern und dadurch in gewissem Rahmen Ihren Prozess kontrollieren kann.

Informationen und Empfehlungen bezüglich der Sicherheit finden Sie in unseren Operational Guidelines für Industrial Security (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf) auf der Website "Service & Support" von Siemens.

Mit den Uhrzeitanweisungen können Sie die Systemuhr der CPU einstellen und lesen. Der Datentyp DTL (Seite 123) stellt Datum- und Uhrzeitwerte bereit.

Tabelle 9- 9 Anweisungen für die Systemzeit

KOP/FUP	SCL	Beschreibung
	`ret_val := WR_SYS_T( in:= DTL_in );`	WR_SYS_T (Uhrzeit einstellen) stellt die Echtzeituhr in der CPU anhand des DTL-Werts von Parameter IN ein. Dieser Zeitwert berücksichtigt weder die jeweilige Ortszeit noch die Sommer-/Winterzeitumschaltung.
	`ret_val := RD_SYS_T( out=>DTL_out);`	Mit RD_SYS_T (Uhrzeit lesen) wird die aktuelle Systemzeit der CPU ausgelesen. Dieser Zeitwert berücksichtigt weder die jeweilige Ortszeit noch die Sommer-/Winterzeitumschaltung.

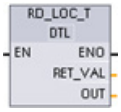
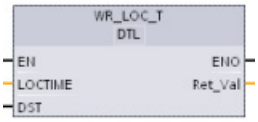
KOP/FUP	SCL	Beschreibung
	`ret_val := RD_LOC_T( out=>_DTL_out);`	RD_LOC_T (Lokalzeit lesen) stellt die aktuelle Lokalzeit der CPU als Datentyp DTL bereit. Dieser Uhrzeitwert entspricht der lokalen Zeitzone sowie der Anpassung an Sommer-/Winterzeit (sofern konfiguriert).
	`ret_val := WR_LOC_T( LOCTIME:=DTL_in_, DST:_in_;`	WR_LOC_T (Lokalzeit schreiben) stellt das Datum und die Uhrzeit der CPU-Uhr ein. Die Datums- und Uhrzeitangaben weisen Sie über den Parameter LOCTIME mit dem Datentyp DTL als Ortszeit zu. Die Anweisung berechnet die Systemzeit mithilfe der DB-Struktur "TimeTransformationRule (Seite 329)". Die Detailgenauigkeit der Zeitangaben für die Ortszeit und die Systemzeit ist produktspezifisch und beträgt mindestens eine Millisekunde. Sind Eingangswerte am Parameter LOCTIME kleiner als von der CPU unterstützt, werden diese Werte während der Berechnung der Systemzeit aufgerundet. **Hinweis**: Sie müssen die Eigenschaften "Uhrzeit" in der CPU-Gerätekonfiguration einstellen (Zeitzone, Aktivierung der Sommerzeit, Beginn und Ende der Sommerzeit). Andernfalls kann WR_LOC_T den Wechsel zur Sommerzeit nicht interpretieren.

Tabelle 9- 10 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	DTL	Einzustellende Uhrzeit für die Systemuhr der CPU
OUT	OUT	DTL	RD_SYS_T: Aktuelle Systemzeit der CPU RD_LOC_T: Aktuelle Ortszeit, einschließlich Anpassung an Sommer-/Winterzeit (sofern konfiguriert)
LOCTIME	IN	DTL	WR_LOC_T: Ortszeit
DST	IN	BOOL	WR_LOC_T: **D**aylight **S**aving **T**ime (Sommerzeit) wird nur während der "doppelten Stunde" ausgewertet, wenn die Uhr zur Sommerzeit wechselt. • TRUE = Sommerzeit (erste Stunde) • FALSE = Winterzeit (zweite Stunde)
RET_VAL	OUT	Int	Ausführungsbedingung

- Zur Berechnung der Lokalzeit werden die Zeitzone und die Sommer-/Winterzeitumschaltungen, die Sie im Register "Allgemein" der Gerätekonfiguration unter "Uhrzeit" eingegeben haben, verwendet.
- Die Zeitzone ist bezogen auf die UTC- oder GMT-Systemzeit.
- Für die Sommerzeitumschaltung sind Monat, Woche, Tag und Stunde der Umstellung auf Sommerzeit einzugeben.
- Für die Winterzeitumschaltung sind ebenfalls Monat, Woche, Tag und Stunde der Umstellung auf Winterzeit einzugeben.
- Die Zeitzonendifferenz gilt immer bezogen auf die Systemzeit. Die Sommerzeitverschiebung gilt nur, wenn eine Umstellung auf Sommerzeit stattfindet.

Hinweis

Anfangszeit von Sommer- und Winterzeit einstellen

Für die Eigenschaften "Uhrzeit" unter "Anfangszeit von Sommer- und Winterzeit" in der CPU-Gerätekonfiguration müssen Sie Ihre lokale Uhrzeit angeben.

Bedingungscodes: ENO = 1 bedeutet, dass kein Fehler aufgetreten ist. ENO = 0 bedeutet, dass ein Ausführungsfehler aufgetreten ist und ein Bedingungscode am Ausgang RET_VAL ansteht.

RET_VAL (W#16#....)	Beschreibung
0000	Die aktuelle Lokalzeit wird als Winterzeit angegeben.
0001	Es wurde die Sommerzeit wurde konfiguriert und die aktuelle Lokalzeit wird als Sommerzeit angegeben.
8080	Ortszeit nicht verfügbar oder LOCTIME-Wert ist ungültig.
8081	Vom Parameter LOCTIME zugewiesener unzulässiger Jahreswert oder Zeitwert ist ungültig
8082	Unzulässiger Monatswert (Byte 2 im DTL-Format)
8083	Unzulässiger Tageswert (Byte 3 im DTL-Format)
8084	Unzulässiger Stundenwert (Byte 5 im DTL-Format)
8085	Unzulässiger Minutenwert (Byte 6 im DTL-Format)
8086	Unzulässiger Sekundenwert (Byte 7 im DTL-Format)
8087	Unzulässiger Nanosekundenwert (Byte 8 bis 11 im DTL-Format)
8089	Zeitwert nicht vorhanden (Stunde bei Wechsel zur Sommerzeit bereits vergangen)
80B0	Die Echtzeituhr ist ausgefallen
80B1	Die Struktur "TimeTransformationRule" wurde nicht definiert.

9.1.3 Datenstruktur TimeTransformationRule

Beschreibung

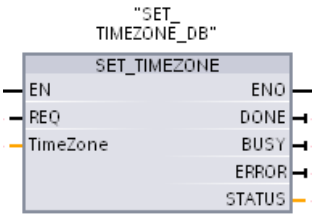
Die Regeln für den Wechsel zwischen Sommer- und Winterzeit sind in der Struktur TimeTransformationRule festgelegt. Die Struktur ist wie folgt:

Name		Datentyp	Beschreibung
TimeTransformationRule		STRUCT	
	Bias	INT	Zeitunterschied zwischen Ortszeit und UTC [Minuten] Bereich: -1439 bis 1439
	DaylightBias	INT	Zeitunterschied zwischen Sommerzeit und Winterzeit [Minuten] Bereich: 0 bis 60
	DaylightStartMonth	USINT	Monat der Umstellung auf Sommerzeit Bereich: 1 bis 12
	DaylightStartWeek	USINT	Woche der Umstellung auf Sommerzeit 1 = Erstes Auftreten des Wochentags im Monat, ..., 5 = Letztes Auftreten des Wochentags im Monat
	DaylightStartWeekday	USINT	Wochentag der Umstellung auf Sommerzeit: 1 = Sonntag
	DaylightStartHour	USINT	Stunde der Umstellung auf Sommerzeit Bereich: 0 bis 23
	DaylightStartMinute	USINT	Minute der Umstellung auf Sommerzeit Bereich: 0 bis 59
	StandardStartMonth	USINT	Monat der Umstellung auf Winterzeit Bereich: 1 bis 12
	StandardStartWeek	USINT	Woche der Umstellung auf Winterzeit 1 = Erstes Auftreten des Wochentags im Monat, ..., 5 = Letztes Auftreten des Wochentags im Monat
	StandardStartWeekday	USINT	Wochentag der Umstellung auf Winterzeit: 1 = Sonntag
	StandardStartHour	USINT	Stunde der Umstellung auf Winterzeit Bereich: 0 bis 23
	StandardStartMinute	USINT	Minute der Umstellung auf Winterzeit Bereich: 0 bis 59
	TimeZoneName	STRING[80]	Name der Zeitzone: "(GMT +01:00) Amsterdam, Berlin, Bern, Rom, Stockholm, Wien "

9.1.4 Anweisung SET_TIMEZONE (Zeitzone setzen)

Tabelle 9- 11 Anweisung SET_TIMEZONE

KOP/FUP	SCL	Beschreibung

	`"SET_TIMEZONE_DB"(`
	`REQ:=_bool_in,`
	`Timezone:=_struct_in,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Stellt die Parameter für lokale Zeitzone und Sommer-/Winterzeitumschaltung ein, anhand deren die CPU-Systemzeit in die jeweilige Ortszeit umgewandelt wird.

[1] Im SCL-Beispiel ist "SET_TIMEZONE_DB" der Name des Instanz-DBs.

Tabelle 9- 12 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	REQ = 1: Funktion ausführen
Timezone	IN	TimeTransformationRule	Regeln für die Umwandlung der Systemzeit in Lokalzeit
DONE	OUT	Bool	Funktion beendet
BUSY	OUT	Bool	Funktion besetzt
ERROR	OUT	Bool	Fehler erkannt
STATUS	OUT	Word	Funktionsergebnis / Fehlermeldung

Um die Zeitzonenparameter für die CPU manuell zu konfigurieren, verwenden Sie die Eigenschaften "Uhrzeit" im Register "Allgemein" der Gerätekonfiguration.

Mit der Anweisung SET_TIMEZONE stellen Sie die Ortszeit programmatisch ein. Die Parameter der Struktur "TimeTransformationRule (Seite 329)" weisen die lokale Zeitzone sowie die Zeiteinstellungen für die automatische Umschaltung zwischen Sommer- und Winterzeit zu.

Bedingungscodes: ENO = 1 bedeutet, dass kein Fehler aufgetreten ist. ENO = 0 bedeutet, dass ein Ausführungsfehler aufgetreten ist und ein Bedingungscode am Ausgang STATUS ansteht.

STATUS (W#16#....)	Beschreibung
0	Kein Fehler
7000	Keine laufende Auftragsbearbeitung
7001	Start der Auftragsbearbeitung. Parameter BUSY = 1, DONE = 0
7002	Zwischenzeitlicher Aufruf (REQ irrelevant): Anweisung bereits aktiv; BUSY auf Wert "1".
808x	Fehler an der x.ten Komponente: Zum Beispiel zeigt 8084 an, dass DaylightStartWeekif kein Wert von 1 bis 5 ist.

9.1.5 Anweisung RTM (Betriebsstundenzähler)

Tabelle 9- 13 Anweisung RTM

KOP/FUP	SCL	Beschreibung

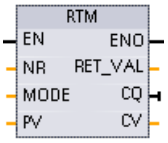
	`RTM(NR:=_uint_in_,`
	`MODE:=_byte_in_,`
	`PV:=_dint_in_,`
	`CQ=>_bool_out_,`
	`CV=>_dint_out_);`
		Die Anweisung RTM (Betriebsstundenzähler) kann die Betriebsstundenzähler in der CPU setzen, starten, stoppen und lesen.

Tabelle 9- 14 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
NR	IN	UInt	Nummer des Betriebsstundenzählers: (mögliche Werte: 0..9)
MODE	IN	Byte	Nummer des RTM-Ausführungsmodus: • 0 = Werte abrufen (der Zustand wird dann in CQ und der aktuelle Wert in CV geschrieben) • 1 = Starten (ab dem letzten Zählerwert) • 2 = Stoppen • 4 = Setzen (auf den in PV angegebenen Wert) • 5 = Setzen (auf den in PV angegebenen Wert) und dann starten • 6 = Setzen (auf den in PV angegebenen Wert) und dann stoppen • 7 = Alle RTM-Werte der CPU auf der MC (Memory Card) speichern
PV	IN	DInt	Stundenwert für den angegebenen Betriebsstundenzähler voreinstellen
RET_VAL	OUT	Int	Funktionsergebnis / Fehlermeldung
CQ	OUT	Bool	Zustand Betriebsstundenzähler (1 = läuft)
CV	OUT	DInt	Aktueller Betriebsstundenwert für den angegebenen Betriebsstundenzähler

Die CPU betreibt bis zu 10 Betriebsstundenzähler, um die Betriebsstunden von kritischen Untersystemen der Steuerung zu verfolgen. Sie müssen die einzelnen Betriebsstundenzähler mit einer RTM-Anweisung je Zähler starten. Wenn die CPU von RUN in STOP wechselt, werden alle Betriebsstundenzähler gestoppt. Einzelne Zähler können Sie auch mit Ausführungsmodus 2 der RTM-Anweisung stoppen.

Wenn eine CPU von STOP in RUN wechselt, müssen Sie die Betriebsstundenzähler mit einer RTM-Anweisung je Zähler neu starten. Ist der Wert eines Betriebsstundenzählers größer als 2.147.483.647 Stunden, wird der Zählbetrieb beendet und ein Überlauffehler gemeldet. Um einen Zähler zurückzusetzen oder zu ändern, müssen Sie die Anweisung RTM einmal pro Zähler ausführen.

Ein Spannungsausfall oder Neustart der CPU bewirkt einen Abschaltvorgang, bei dem die aktuellen Werte der Betriebsstundenzähler im remanenten Speicher abgelegt werden. Beim Anlauf der CPU werden die gespeicherten Werte der Betriebsstundenzähler erneut in die Zähler geladen, so dass die vorherigen Betriebsstunden nicht verloren sind. Die Betriebsstundenzähler müssen neu gestartet werden, um weitere Betriebsstunden zu zählen.

Ihr Programm kann zudem über Ausführungsmodus 7 der RTM-Anweisung die Werte der Betriebsstundenzähler auf einer Memory Card speichern. Zum Zeitpunkt der Ausführung von Modus 7 der RTM-Anweisung werden die Zustände aller Betriebsstundenzähler auf der Memory Card gespeichert. Diese gespeicherten Werte können im Lauf der Zeit durch Start- und Stoppvorgänge der Betriebsstundenzähler im Programmverlauf inkorrekt werden. Sie müssen die Werte auf der Memory Card regelmäßig aktualisieren, um wichtige Laufzeitereignisse zu erfassen. Der Vorteil dabei, die RTM-Werte auf der Memory Card zu speichern, ist der, dass Sie die Memory Card in eine Ersatz-CPU einschieben können, wo Ihr Programm und die gespeicherten RTM-Werte dann zur Verfügung stehen. Wenn Sie die RTM-Werte nicht auf einer Memory Card gespeichert haben, sind die Betriebsstundenwerte verloren (bei Nutzung einer Ersatz-CPU).

Hinweis

Vermeiden Sie übermäßige Programmaufrufe bei Schreibvorgängen auf der Memory Card

Halten Sie die Anzahl der Schreibvorgänge auf der Memory Card möglichst gering, um die Lebensdauer der Memory Card zu verlängern.

Tabelle 9- 15 Bedingungscodes

RET_VAL (W#16#....)	Beschreibung
0	Kein Fehler
8080	Falsche Nummer des Betriebsstundenzählers
8081	Ein negativer Wert wurde an den Parameter PV übergeben
8082	Überlauf des Betriebsstundenzählers
8091	Der Eingangsparameter MODE enthält einen unzulässigen Wert
80B1	Wert kann nicht auf MC gespeichert werden (MODUS = 7)

9.2 Zeichenketten- und Zeichenanweisungen

9.2.1 Datentyp String

Datentyp STRING

Daten des Typs String werden mit einem Header aus 2 Bytes gefolgt von bis zu 254 Zeichenbytes im ASCII-Code gespeichert. Ein Header für String-Daten hat zwei Längenbytes. Das erste Byte enthält die maximale Länge, die bei der Initialisierung einer Zeichenkette in eckigen Klammern angegeben bzw. auf 254 voreingestellt ist. Das zweite Header-Byte enthält die tatsächliche Länge, die der Anzahl gültiger Zeichen in der Zeichenkette entspricht. Die tatsächliche Länge muss kleiner oder gleich der maximalen Länge sein. Die Anzahl gespeicherter Bytes für das String-Format ist um 2 Bytes größer als die maximale Länge.

String-Daten initialisieren

Die Ein- und Ausgangsdaten des Datentyps String müssen vor der Ausführung von Anweisungen als gültige Zeichenketten im Speicher initialisiert werden.

Gültige String-Daten

Gültige String-Daten haben eine maximale Länge, die größer als null und kleiner als 255 sein muss. Die tatsächliche Länge muss kleiner oder gleich der maximalen Länge sein.

String-Daten können den Speicherbereichen E oder A nicht zugewiesen werden.

Weitere Informationen finden Sie unter: Format des Datentyps String (Seite 125).

9.2.2 Anweisung S_MOVE (Zeichenkette verschieben)

Tabelle 9- 16 Anweisung Zeichenkette verschieben

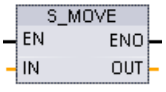
KOP/FUP	SCL	Beschreibung
	`out := in;`	Zeichenkette der Quelle IN in die Adresse OUT kopieren. Die Ausführung S_MOVE wirkt sich nicht auf den Inhalt der Quellzeichenkette aus.

Tabelle 9- 17 Datentypen für die Parameter

Parameter	Datentyp	Beschreibung
IN	String	Quellzeichenkette
OUT	String	Zieladresse

Wenn die tatsächliche Länge der Zeichenkette am Eingang IN die maximale Länge einer am Ausgang OUT gespeicherten Zeichenkette überschreitet, dann wird der Teil der Zeichenkette IN, der in OUT passt, kopiert.

9.2.3 Zeichenkettenkonvertierungsoperationen

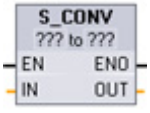
9.2.3.1 Anweisungen S_CONV, STRG_VAL und VAL_STRG (In/von Zeichenkette und Zahlenwert umwandeln)

Mit den folgenden Anweisungen können Sie Zahlenzeichenketten in Zahlenwerte und Zahlenwerte in Zahlenzeichenketten umwandeln:

- Mit S_CONV wird eine Zahlenzeichenkette in einen Zahlenwert oder ein Zahlenwert in eine Zahlenzeichenkette umgewandelt.
- Mit STRG_VAL wird eine Zahlenzeichenkette in einen Zahlenwert mit Formatoptionen umgewandelt.
- Mit VAL_STRG wird ein Zahlenwert in eine Zahlenzeichenkette mit Formatoptionen umgewandelt.

S_CONV (Zeichenkette umwandeln)

Tabelle 9- 18 Anweisungen für die Zeichenkettenkonvertierung

KOP/FUP	SCL	Beschreibung
	`out := <Type>_TO_<Type>(in);`	Konvertiert eine Zeichenkette in den entsprechenden Wert bzw. einen Wert in die entsprechende Zeichenkette. Die Anweisung S_CONV verfügt über keine Optionen zur Formatierung des Ausgangswerts. Daher ist S_CONV einfacher, aber auch weniger flexibel als die Anweisungen STRG_VAL und VAL_STRG.

[1] In KOP/FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

[2] In SCL: Wählen Sie S_CONV aus den erweiterten Anweisungen und beantworten Sie die Abfragen der Datentypen für die Umwandlung. STEP 7 stellt daraufhin die entsprechende Umwandlungsanweisung zur Verfügung.

Tabelle 9- 19 Datentypen (Zeichenkette in Zahlenwert umwandeln)

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	String, WString	Eingegebene Zeichenkette
OUT	OUT	String, WString, Char, WChar, SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Ausgegebener Zahlenwert

Die Umwandlung des Zeichenkettenparameters IN beginnt mit dem ersten Zeichen und wird bis zum Ende der Zeichenkette oder bis zum ersten Zeichen, das nicht "0" bis "9", "+", "-" oder "." ist, fortgesetzt. Das Ergebnis steht an der mit Parameter OUT angegebenen Adresse bereit. Passt der ausgegebene Zahlenwert nicht in den Bereich des Datentyps von OUT, so wird der Parameter OUT auf 0 und ENO auf FALSCH gesetzt. Andernfalls enthält der Parameter OUT ein gültiges Ergebnis und ENO wird auf WAHR gesetzt.

Formatregeln für die Zeichenketteneingabe:

- Enthält die Zeichenkette in IN ein Dezimalzeichen, so muss das Zeichen "." verwendet werden.
- Kommazeichen "," als Tausendertrennzeichen links vom Dezimalpunkt sind zulässig, werden aber ignoriert.
- Führende Leerzeichen werden ignoriert.

S_CONV (Umwandlung von Wert in Zeichenkette)

Tabelle 9- 20 Datentypen (Zahlenwert in Zeichenkette umwandeln)

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	String, WString, Char, WChar, SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Eingegebener Zahlenwert
OUT	OUT	String, WString	Ausgegebene Zeichenkette

Ein ganzzahliger Wert mit oder ohne Vorzeichen oder ein Gleitpunktwert IN wird in die entsprechende Zeichenkette umgewandelt und steht in Parameter OUT bereit. Der Parameter OUT muss eine gültige Zeichenkette referenzieren, bevor die Umwandlung erfolgt. Eine gültige Zeichenkette besteht aus der maximalen Zeichenkettenlänge im ersten Byte, der tatsächlichen Zeichenkettenlänge im zweiten Byte und den tatsächlichen Zeichen in den nächsten Bytes. Die umgewandelte Zeichenkette ersetzt Zeichen in der Zeichenkette OUT, beginnend mit dem ersten Zeichen, und passt das Byte mit der tatsächlichen Länge der ausgegebenen Zeichenkette entsprechend an. Das Byte mit der maximalen Länge der ausgegebenen Zeichenkette wird nicht verändert.

Wie viele Zeichen ersetzt werden, hängt von Datentyp und Zahlenwert des Parameters IN ab. Die ersetzte Zeichenzahl muss in die in Parameter OUT festgelegte Zeichenkettenlänge passen. Die maximale Zeichenkettenlänge (erstes Byte) der Zeichenkette in Parameter OUT muss größer oder gleich der maximal erwarteten Zahl umgewandelter Zeichen sein. Die folgende Tabelle zeigt Beispiele für die Umwandlung von Werten in Zeichenketten mit S_CONV.

Formatregeln für die Zeichenkettenausgabe:

- In Parameter OUT geschriebene Werte verwenden kein führendes "+" Zeichen.
- Es wird die Festpunktdarstellung verwendet (keine Exponentialdarstellung).
- Mit dem Punktzeichen "." wird der Dezimalpunkt dargestellt, wenn der Parameter IN vom Datentyp Real ist.
- Die Werte werden in der ausgegebenen Zeichenkette rechtsbündig angeordnet und ihnen stehen Leerzeichen voran, die leere Zeichenpositionen belegen.

Tabelle 9- 21 Maximale Zeichenkettenlänge für jeden Datentyp

Datentyp IN	Von S_CONV zugeordnete Zeichenpositionen	Beispiel für eine umgewandelte Zeichenkette[1]	Gesamtlänge der Zeichenkette mit den Bytes für maximale und tatsächliche Länge
USInt	4	"x255"	6
SInt	4	"-128"	6
UInt	6	"x65535"	8
Int	6	"-32768"	8
UDInt	11	"x4294967295"	13
DInt	11	"-2147483648"	13
Real	14	"x-3.402823E+38" "x-1.175495E-38" "x+1.175495E-38" "x+3.402823E+38"	16
LReal	21	"-1.7976931348623E+308" "-2.2250738585072E-308" "+2.2250738585072E-308" "+1.7976931348623E+308"	23

[1] Die Zeichen "x" stellen Leerzeichen dar, die in dem umgewandelten Wert zugeordneten rechtsbündigen Feld die leeren Positionen belegen.

STRG_VAL (Zeichenkette in Zahlenwert umwandeln)

Tabelle 9- 22 Anweisung Zeichenkette in Zahlenwert wandeln

KOP/FUP	SCL	Beschreibung

	`"STRG_VAL"(`
	`in:=_string_in,`
	`format:=_word_in,`
	`p:=uint_in,`
	`out=>_variant_out);`
		Konvertiert eine Zahlenzeichenkette in die entsprechende ganzzahlige oder Gleitpunktdarstellung.

[1] In KOP/FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 9- 23 Datentypen für die Anweisung STRG_VAL

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	String, WString	ASCII-Zeichenkette, die umgewandelt werden soll
FORMAT	IN	Word	Optionen für Ausgangsformate
P	IN	UInt, Byte, USInt	IN: Verweis zum ersten Zeichen, das umgewandelt werden soll (erstes Zeichen = 1)
OUT	OUT	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Umgewandelter Zahlenwert

Die Umwandlung beginnt in der eingegebenen Zeichenkette an Zeichenversatz P und läuft bis zum Ende der Zeichenkette oder bis zum ersten Zeichen, das nicht "+", "-", ".", ",", "e", "E" oder eines der Zeichen von "0" bis "9" ist. Das Ergebnis wird an der in Parameter OUT angegebenen Adresse ausgegeben.

Die Daten vom Datentyp String müssen vor der Ausführung als gültige Zeichenkette im Speicher initialisiert werden.

Der Parameter FORMAT der Anweisung STRG_VAL ist nachstehend definiert. Die freien Bitpositionen müssen auf Null gesetzt werden.

Tabelle 9- 24 Format der Anweisung STRG_VAL

Bit 16							**Bit 8**	**Bit 7**							**Bit 0**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	f	r

f = Darstellungsformat 1= Exponentielle Darstellung
0 = Festpunktdarstellung

r = Dezimalpunktformat 1 = "," (Kommazeichen)
0 = "." (Punktzeichen)

Tabelle 9- 25 Werte des Parameters FORMAT

FORMAT (W#16#)	**Darstellungsformat**	**Dezimalpunktdarstellung**
0000 (Voreinstellung)	Festpunkt	"."
0001		","
0002	Exponentiell	"."
0003		","
0004 bis FFFF	Ungültige Werte	

Regeln für die STRG_VAL-Umwandlung:

- Wird das Punktzeichen "." für den Dezimalpunkt benutzt, so werden Kommazeichen "," links vom Dezimalpunkt als Tausendertrennzeichen interpretiert. Kommazeichen sind zulässig und werden ignoriert.
- Wird das Kommazeichen "," als Dezimaltrennzeichen benutzt, so werden Punktzeichen "." links vom Dezimalzeichen als Tausendertrennzeichen interpretiert. Diese Punktzeichen sind zulässig und werden ignoriert.
- Führende Leerzeichen werden ignoriert.

VAL_STRG (Zahlenwert in Zeichenkette umwandeln)

Tabelle 9- 26 Anweisung Zahlenwert in Zeichenkette wandeln

KOP/FUP	SCL	Beschreibung

	`"VAL_STRG"(`
	`in:=_variant_in,`
	`size:=_usint_in,`
	`prec:=_usint_in,`
	`format:=_word_in,`
	`p:=uint_in,`
	`out=>_string_out);`
		Konvertiert einen ganzzahligen Wert mit oder ohne Vorzeichen oder einen Gleitpunktwert in die entsprechende Zeichenkettendarstellung.

[1] In KOP/FUP: Klicken Sie auf die "???" und wählen Sie den Datentyp in der Klappliste aus.

Tabelle 9- 27 Datentypen für die Anweisung VAL_STRG

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	SInt, Int, DInt, USInt, UInt, UDInt, Real, LReal	Wert, der umzuwandeln ist
SIZE	IN	USInt	Anzahl der Zeichen, die in die Zeichenkette OUT geschrieben werden sollen
PREC	IN	USInt	Genauigkeit oder Größe der Nachkommastellen. Dies schließt nicht das Dezimaltrennzeichen ein.
FORMAT	IN	Word	Optionen für Ausgangsformate
P	IN	UInt, Byte, USInt	IN: Verweis zum ersten Zeichen der Zeichenkette OUT, das ersetzt werden soll (erstes Zeichen = 1)
OUT	OUT	String, WString	Umgewandelte Zeichenkette

Diese Anweisung wandelt den in Parameter IN dargestellten Wert in eine Zeichenkette um, die in Parameter OUT referenziert wird. Der Parameter OUT muss eine gültige Zeichenkette sein, bevor die Umwandlung erfolgt.

Die umgewandelte Zeichenkette ersetzt die Zeichen der Zeichenkette OUT beginnend mit dem in Parameter P festgelegten Zeichenversatz bis zu der mit Parameter SIZE festgelegten Zeichenzahl. Die Anzahl der Zeichen in SIZE muss in die Länge der Zeichenkette OUT passen, gezählt ab Zeichenposition P. Ist Parameter SIZE Null, werden die Zeichen an Position P in Zeichenkette OUT ohne Begrenzung überschrieben. Diese Anweisung ist nützlich, um Ziffernzeichen in eine aus Text bestehende Zeichenkette einzufügen. So können beispielsweise die Zahlen "120" in die Zeichenkette "Pumpendruck = 120 psi" eingefügt werden.

Der Parameter PREC gibt die Genauigkeit oder die Anzahl der Nachkommastellen der Zeichenkette an. Ist Parameter IN ein ganzzahliger Wert, so gibt PREC die Stelle des Dezimaltrennzeichens an. Ist beispielsweise der Datenwert 123 und PREC = 1, so ist das Ergebnis "12.3". Die maximal unterstützte Genauigkeit für den Datentyp Real ist 7-stellig.

Ist Parameter P größer als die tatsächliche Größe der Zeichenkette OUT, so werden Leerzeichen bis zu Position P eingefügt und das Ergebnis wird am Ende der Zeichenkette angehängt. Die Umwandlung endet, sobald die maximale Länge der Zeichenkette OUT erreicht ist.

Der Parameter FORMAT der Anweisung VAL_STRG ist nachstehend definiert. Die freien Bitpositionen müssen auf Null gesetzt werden.

Tabelle 9- 28 Format der Anweisung VAL_STRG

Bit 16							**Bit 8**	**Bit 7**							**Bit 0**
0	0	0	0	0	0	0	0	0	0	0	0	0	s	f	r

s = Zeichen für das Vorzeichen — 1= Vorzeichen "+" und "-"
0 = nur Vorzeichen "-"

f = Darstellungsformat — 1= Exponentielle Darstellung
0 = Festpunktdarstellung

r = Dezimalpunktformat — 1 = "," (Kommazeichen)
0 = "." (Punktzeichen)

Tabelle 9- 29 Werte des Parameters FORMAT

FORMAT (WORD)	Vorzeichen	Darstellungsformat	Dezimalpunktdarstellung
W#16#0000	nur "-"	Festpunkt	"."
W#16#0001			","
W#16#0002		Exponentiell	"."
W#16#0003			","
W#16#0004	"+" und "-"	Festpunkt	"."
W#16#0005			","
W#16#0006		Exponentiell	"."
W#16#0007			","
W#16#0008 bis W#16#FFFF	Ungültige Werte		

Formatregeln für Zeichenketten in Parameter OUT:

- Führende Leerzeichen werden ganz links an die Zeichenkette angefügt, wenn die umgewandelte Zeichenkette kleiner als die angegebene Größe ist.
- Ist das Vorzeichenbit von Parameter FORMAT FALSCH, so werden ganzzahlige Werte mit und ohne Vorzeichen ohne führendes Vorzeichen "+" in den Ausgangspuffer geschrieben. Vorzeichen "-" wird bei Bedarf verwendet.
  <führende Leerzeichen><Ziffern ohne führende Nullen>'.'<PREC-Ziffern>
- Ist das Vorzeichenbit von Parameter WAHR, so werden ganzzahlige Werte mit und ohne Vorzeichen immer mit einem führenden Vorzeichen in den Ausgangspuffer geschrieben.

  <führende Leerzeichen><Vorzeichen><Ziffern ohne führende Nullen>'.'<PREC-Ziffern>
- Ist als FORMAT die exponentielle Darstellung eingestellt, so werden Werte des Datentyps Real wie folgt in den Ausgangspuffer geschrieben:

  <führende Leerzeichen><Vorzeichen><Ziffer> '.' <PREC-Ziffern>'E'
  <Vorzeichen><Ziffern ohne führende Null>

- Ist als FORMAT die Festpunktdarstellung eingestellt, so werden ganzzahlige Werte mit und ohne Vorzeichen und Werte des Datentyps Real wie folgt in den Ausgangspuffer geschrieben:

  <führende Leerzeichen><Vorzeichen><Ziffern ohne führende Nullen>'.'<PREC-Ziffern>
- Führende Nullen links vom Dezimalpunkt (mit Ausnahme der Ziffer direkt neben dem Dezimalpunkt) werden unterdrückt.
- Der Wert rechts vom Dezimalpunkt wird gerundet, damit er der mit Parameter PREC eingestellten Anzahl der Ziffern entspricht, die rechts vom Dezimalpunkt angegeben werden sollen.
- Die Größe der ausgegebenen Zeichenkette muss mindestens drei Bytes mehr umfassen als die Anzahl der Ziffern rechts vom Dezimalpunkt.
- Die Werte werden rechtsbündig in der ausgegebenen Zeichenkette angeordnet.

Mit ENO gemeldete Bedingungen

Tritt bei der Umwandlung ein Fehler auf, gibt die Anweisung die folgenden Ergebnisse aus:

- ENO wird auf 0 gesetzt.
- OUT wird auf 0 gesetzt oder auf den Wert, der in den Beispielen für die Umwandlung von Zeichenketten in Werte angegeben ist.
- OUT bleibt unverändert oder auf dem in den Beispielen gezeigten Wert, wenn OUT eine Zeichenkette ist.

Tabelle 9- 30 ENO-Zustand

ENO	Beschreibung
1	Kein Fehler
0	Unzulässiger oder ungültiger Parameter, zum Beispiel ein Zugriff auf einen DB, der nicht existiert
0	Unzulässige Zeichenkette mit maximaler Länge von 0 oder 255
0	Unzulässige Zeichenkette, deren Länge größer als die maximale Länge ist
0	Der umgewandelte Zahlenwert ist zu groß für den für OUT angegebenen Datentyp.
0	Die maximale Zeichenkettengröße für Parameter OUT muss groß genug sein für die Zahl der mit Parameter SIZE angegebenen Zeichen, beginnend mit der Zeichenposition in Parameter P.
0	Unzulässiger Wert P, wobei P=0 oder P größer als die aktuelle Zeichenkettenlänge ist
0	Parameter SIZE muss größer als Parameter PREC sein.

Tabelle 9- 31 Beispiel für die Umwandlung von Zeichenketten in Werte mit S_CONV

IN-Zeichenkette	Datentyp OUT	OUT-Wert	ENO
"123"	Int oder DInt	123	WAHR
"-00456"	Int oder DInt	-456	WAHR
"123.45"	Int oder DInt	123	WAHR
"+2345"	Int oder DInt	2345	WAHR
"00123AB"	Int oder DInt	123	WAHR
"123"	Real	123.0	WAHR
"123.45"	Real	123.45	WAHR
"1.23e-4"	Real	1.23	WAHR
"1.23E-4"	Real	1.23	WAHR
"12,345.67"	Real	12345.67	WAHR
"3.4e39"	Real	3.4	WAHR
"-3.4e39"	Real	-3.4	WAHR
"1.17549e-38"	Real	1.17549	WAHR
"12345"	SInt	0	FALSCH
"A123"	-/-	0	FALSCH
""	-/-	0	FALSCH
"++123"	-/-	0	FALSCH
"+-123"	-/-	0	FALSCH

Tabelle 9- 32 Beispiele für die Umwandlung von Werten in Zeichenketten mit S_CONV

Datentyp	IN-Wert	OUT-Zeichenkette [1]	ENO
UInt	123	"xxx123"	WAHR
UInt	0	"xxxxx0"	WAHR
UDInt	12345678	"xxx12345678"	WAHR
Real	+9123.456	"xx+9.123456E+3"	WAHR
LReal	+9123.4567890123	"xx+9.1234567890123 E+3"	WAHR
Real	-INF	"xxxxxxxxxxINF"	FALSCH
Real	+INF	"xxxxxxxxxxINF"	FALSCH
Real	NaN	"xxxxxxxxxxNaN"	FALSCH

[1] Die Zeichen "x" stellen Leerzeichen dar, die in dem umgewandelten Wert zugeordneten rechtsbündigen Feld die leeren Positionen belegen.

Tabelle 9- 33 Beispiel: STRG_VAL-Umwandlung

IN-Zeichenkette	FORMAT (W#16#....)	Datentyp OUT	OUT-Wert	ENO
"123"	0000	Int oder DInt	123	WAHR
"-00456"	0000	Int oder DInt	-456	WAHR
"123.45"	0000	Int oder DInt	123	WAHR
"+2345"	0000	Int oder DInt	2345	WAHR
"00123AB"	0000	Int oder DInt	123	WAHR
"123"	0000	Real	123.0	WAHR
"-00456"	0001	Real	-456.0	WAHR
"+00456"	0001	Real	456.0	WAHR
"123.45"	0000	Real	123.45	WAHR
"123.45"	0001	Real	12345.0	WAHR
"123.45"	0000	Real	12345.0	WAHR
"123.45"	0001	Real	123.45	WAHR
".00123AB"	0001	Real	123.0	WAHR
"1.23e-4"	0000	Real	1.23	WAHR
"1.23E-4"	0000	Real	1.23	WAHR
"1.23E-4"	0002	Real	1.23E-4	WAHR
"12,345.67"	0000	Real	12345.67	WAHR
"12,345.67"	0001	Real	12.345	WAHR
"3.4e39"	0002	Real	+INF	WAHR
"-3.4e39"	0002	Real	-INF	WAHR
"1.1754943e-38" (und kleiner)	0002	Real	0.0	WAHR
"12345"	-/-	SInt	0	FALSCH
"A123"	-/-	-/-	0	FALSCH
""	-/-	-/-	0	FALSCH
"++123"	-/-	-/-	0	FALSCH
"+-123"	-/-	-/-	0	FALSCH

Die folgenden Beispiele für VAL_STRG-Umwandlungen beruhen auf einer Zeichenkette OUT, die wie folgt initialisiert wurde:

"`Current Temp = xxxxxxxxxx C`"
Dabei steht das Zeichen "`x`" für Leerzeichen, die dem umgewandelten Wert zugeordnet sind.

Tabelle 9- 34 Beispiel: VAL_STRG-Umwandlung

Datentyp	IN-Wert	P	SIZE	FORMAT (W#16#....)	PREC	OUT-Zeichenkette	ENO
UInt	123	16	10	0000	0	`Current Temp = xxxxxxx123 C`	WAHR
UInt	0	16	10	0000	2	`Current Temp = xxxxxx0.00 C`	WAHR
UDInt	12345678	16	10	0000	3	`Current Temp = x12345.678 C`	WAHR
UDInt	12345678	16	10	0001	3	`Current Temp = x12345,678 C`	WAHR
Int	123	16	10	0004	0	`Current Temp = xxxxxx+123 C`	WAHR
Int	-123	16	10	0004	0	`Current Temp = xxxxxx-123 C`	WAHR
Real	-0.00123	16	10	0004	4	`Current Temp = xxx-0.0012 C`	WAHR
Real	-0.00123	16	10	0006	4	`Current Temp = -1.2300E-3 C`	WAHR
Real	-INF	16	10	-/-	4	`Current Temp = xxxxxx-INF C`	FALSCH
Real	+INF	16	10	-/-	4	`Current Temp = xxxxxx+INF C`	FALSCH
Real	NaN	16	10	-/-	4	`Current Temp = xxxxxxxNaN C`	FALSCH
UDInt	12345678	16	6	-/-	3	`Current Temp = xxxxxxxxxx C`	FALSCH

9.2.3.2 Anweisungen Strg_TO_Chars und Chars_TO_Strg (In/aus Zeichenkette und Array aus CHAR umwandeln)

Strg_TO_Chars kopiert eine ASCII-Zeichenkette in ein Array aus Zeichenbytes.

Chars_TO_Strg kopiert ein Array aus ASCII-Zeichen-Bytes in eine Zeichenkette.

Hinweis

Nur die nullbasierten Array-Typen (Array [0..n] of Char) oder (Array [0..n] of Byte) sind als Eingangsparameter Chars bei der Anweisung Chars_TO_Strg zulässig, oder als IN_OUT-Parameter Chars bei der Anweisung Strg_TO_Chars .

Tabelle 9- 35 Anweisung Strg_TO_Chars

KOP/FUP	SCL	Beschreibung

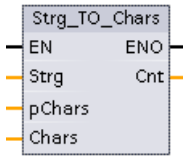
	`Strg_TO_Chars(`
	`  Strg:=_string_in_,`
	`  pChars:=_dint_in_,`
	`  Cnt=>_uint_out_,`
	
	`Chars:=_variant_inout_);`
		Die vollständige Eingangszeichenkette Strg wird in ein Array aus Zeichen am IN_OUT-Parameter Chars. kopiert.
		Die Anweisung überschreibt Bytes beginnend mit der Array-Elementnummer, die vom Parameter pChars angegeben wird.
		Zeichenketten aller unterstützten maximalen Längen (1 .. 254) können verwendet werden.
		Es wird kein Abschlusstrennzeichen geschrieben, hierfür sind Sie selbst verantwortlich. Wenn Sie das Abschlusstrennzeichen direkt nach dem letzten geschriebenen Array-Zeichen setzen möchten, verwenden Sie die nächste Array-Elementnummer [pChars+Cnt].

Tabelle 9- 36 Datentypen für die Parameter (Strg_TO_Chars)

Parameter und Datentyp		Datentyp	Beschreibung
Strg	IN	String, WString	Quellzeichenfolge
pChars	IN	DInt	Array-Elementnummer des ersten Zeichens der Zeichenkette, das in das Ziel-Array geschrieben wird
Chars	IN_OUT	Variante	Der Parameter Chars ist ein Pointer auf ein nullbasiertes Array [0..n] aus Zeichen, das aus der Eingangszeichenkette kopiert wird. Das Array kann in einem DB oder als lokale Variablen in der Bausteinschnittstelle deklariert werden. Beispiel: "DB1".MyArray verweist auf die Elementwerte MyArray [0..10] of Char in DB1.
Cnt	OUT	UInt	Anzahl der kopierten Zeichen

Tabelle 9- 37 Anweisung Chars_TO_Strg

KOP/FUP	SCL	Beschreibung

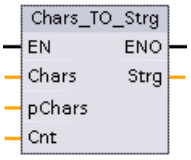
	`Chars_TO_Strg(`
	`  Chars:=_variant_in_,`
	`  pChars:=_dint_in_,`
	`  Cnt:=_uint_in_,`
	`  Strg=>_string_out_);`
		Das gesamte Array oder ein Teil eines Arrays aus Zeichen wird in eine Zeichenkette kopiert.
		Die Ausgangszeichenkette muss deklariert sein, bevor Chars_TO_Strg ausgeführt wird. Die Zeichenkette wird dann von der Anweisung Chars_TO_Strg überschrieben.
		Zeichenketten aller unterstützten maximalen Längen (1 ... 254) können verwendet werden.
		Der Wert für die maximale Länge der Zeichenkette wird von der Anweisung Chars_TO_Strg nicht verändert. Das Kopieren aus dem Array in die Zeichenkette endet, wenn die maximale Länge der Zeichenkette erreicht ist.
		Das nul-Zeichen "$00" oder der Wert 16#00 im Zeichen-Array fungiert als Trennzeichen und beendet das Kopieren von Zeichen in die Zeichenkette.

Tabelle 9- 38 Datentypen für die Parameter (Chars_TO_Strg)

Parameter und Datentyp		Datentyp	Beschreibung
Chars	IN	Variante	Der Parameter Chars ist ein Pointer auf das nullbasierte Array [0..n] aus Zeichen, das in eine Zeichenkette umgewandelt werden soll. Das Array kann in einem DB oder als lokale Variablen in der Bausteinschnittstelle deklariert werden. Beispiel: "DB1".MyArray verweist auf die Elementwerte MyArray [0..10] of Char in DB1.
pChars	IN	Dint	Elementnummer des ersten zu kopierenden Zeichens im Array. Array-Element [0] ist der Standardwert.
Cnt	IN	UInt	Anzahl der zu kopierenden Zeichen: 0 bedeutet alle
Strg	OUT	String, WString	Zielzeichenkette

Tabelle 9- 39 ENO-Status

ENO	Beschreibung
1	Kein Fehler
0	Chars_TO_Strg: Versuch, mehr Zeichenbytes in die Ausgangszeichenkette zu kopieren als entsprechend dem Byte für die maximale Länge in der Zeichenkettendeklaration zulässig sind
0	Chars_TO_Strg: Der Nullzeichenwert (16#00) wurde im Byte-Array des Eingangszeichens gefunden.
0	Strg_TO_Chars: Versuch, mehr Zeichenbytes in das Ausgangs-Array zu kopieren als entsprechend dem Grenzwert für die Elementanzahl zulässig sind

9.2.3.3 ATH und HTA (In/aus ASCII-Zeichenkette und Hexadezimalzahl umwandeln)

Mit den Anweisungen ATH (ASCII-Zeichenkette in Hexadezimalzahl wandeln) und HTA (Hexadezimalzahl in ASCII-Zeichenkette wandeln) können Sie ASCII-Zeichen (0 bis 9 und Großbuchstaben A bis F) in die entsprechenden 4-Bit-Hexadezimaldarstellungen umwandeln und umgekehrt.

Tabelle 9- 40 Anweisung ATH

KOP/FUP	SCL	Beschreibung

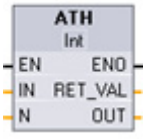
	`ret_val := ATH(`
	`  in:=_variant_in_,`
	`  n:=_int_in_,`
	
	`out=>_variant_out_);`
		Konvertiert ASCII-Zeichen in gepackte Hexadezimalziffern.

Tabelle 9- 41 Datentypen für die Anweisung ATH

Parametertyp		Datentyp	Beschreibung
IN	IN	Variante	Pointer zum ASCII-Zeichenbyte-Datenfeld
N	IN	UInt	Anzahl der ASCII-Zeichenbytes, die umzuwandeln sind
RET_VAL	OUT	Word	Ausführungsbedingung
OUT	OUT	Variante	Zeigt zum umgewandelten Hexadezimalbyte-Datenfeld

Die Umwandlung beginnt an der von Parameter IN festgelegten Stelle und umfasst N Bytes. Das Ergebnis wird an der in Parameter OUT angegebenen Adresse ausgegeben. Es können nur gültige ASCII-Zeichen von 0 bis 9, Kleinbuchstaben a bis f und Großbuchstaben A bis F umgewandelt werden. Alle anderen Zeichen werden in Null umgewandelt.

8-Bit ASCII-codierte Zeichen werden in 4-Bit Hexadezimalzeichen umgewandelt. Zwei ASCII-Zeichen können in ein einzelnes Byte mit zwei 4-Bit Hexadezimalzeichen umgewandelt werden.

Die Parameter IN und OUT geben Byte-Arrays, keine hexadezimalen String-Daten an. Die Umwandlung der ASCII-Zeichen und ihre Ausgabe in Hexadezimalzeichen erfolgt in der Reihenfolge, in der sie gelesen werden. Bei einer ungeraden Anzahl ASCII-Zeichen werden Nullen in das äußerste rechte Halbbyte der zuletzt umgewandelten Hexadezimalstelle geschrieben.

Tabelle 9- 42 Beispiele: Umwandlung von ASCII-Zeichenketten in Hexadezimalzahlen (ATH)

IN-Zeichenbytes	N	OUT-Wert	ENO
'0a23'	4	W#16#0A23	WAHR
'123AFx1a23'	10	16#123AF01023	FALSCH
'a23'	3	W#16#A230	WAHR

Tabelle 9- 43 Anweisung HTA

KOP/FUP	SCL	Beschreibung

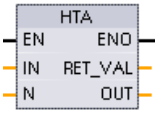
	`ret_val := HTA(`
	`in:=_variant_in_,`
	`n:=_uint_in_,`
	`out=>_variant_out_);`
		Konvertiert gepackte Hexadezimalziffern in die entsprechenden ASCII-Zeichenbytes.

Tabelle 9- 44 Datentypen für die Anweisung HTA

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	Variante	Zeigt zum Eingangsbyte-Datenfeld
N	IN	UInt	Anzahl der Bytes, die umgewandelt werden sollen (jedes Eingangsbyte hat zwei 4-Bit-Halbbytes und erzeugt 2N ASCII-Zeichen)
RET_VAL	OUT	Word	Ausführungsbedingung
OUT	OUT	Variante	Pointer zum ASCII-Zeichenbyte-Datenfeld

Die Umwandlung beginnt an der von Parameter IN festgelegten Stelle und umfasst N Bytes. Jedes 4-Bit-Halbbyte wird in ein einzelnes 8-Bit-ASCII-Zeichen umgewandelt und erzeugt 2N-ASCII-Zeichenbytes am Ausgang. Alle 2N Bytes des Ausgangs werden als ASCII-Zeichen 0 bis 9 bzw. Großbuchstaben A bis F geschrieben. Der Parameter OUT gibt ein Byte-Array, keine Zeichenkette, an.

Jedes Halbbyte des Hexadezimalbytes wird in ein Zeichen in der Reihenfolge des Einlesens umgewandelt (das linke Halbbyte einer Hexadezimalziffer wird zuerst umgewandelt, dann das rechte Halbbyte des gleichen Bytes).

Tabelle 9- 45 Beispiele: Umwandlung von Hexadezimalzahlen in ASCII-Zeichenketten (HTA)

IN-Wert	N	OUT-Zeichenbytes	ENO (ENO immer WAHR nach Ausführung von HTA)
W#16#0123	2	'0123'	WAHR
DW#16#123AF012	4	'123AF012'	WAHR

Tabelle 9- 46 ATH and HTA-Bedingungscodes

RET_VAL (W#16#....)	Beschreibung	ENO
0000	Kein Fehler	WAHR
0007	Ungültiges ATH-Eingangszeichen: Es wurde ein Zeichen gefunden, bei dem es sich nicht um ein ASCII-Zeichen 0-9, einen Kleinbuchstaben a-f oder um einen Großbuchstaben A-F handelte	FALSCH
8101	Unzulässiger oder ungültiger Eingangspointer, zum Beispiel ein Zugriff auf einen DB, der nicht existiert.	FALSCH
8120	Eingangszeichenkette hat ein ungültiges Format, d. h. Max. = 0, Max. = 255, aktuell > Max. oder Länge im Pointer < Max.	FALSCH
8182	Eingangspuffer ist zu klein für N	FALSCH
8151	Datentyp für Eingangspuffer nicht zulässig	FALSCH
8301	Unzulässiger oder ungültiger Ausgangspointer, zum Beispiel ein Zugriff auf einen DB, der nicht existiert.	FALSCH
8320	Ausgangszeichenkette hat ein ungültiges Format, d. h. Max. = 0, Max. = 255, aktuell > Max. oder Länge im Pointer < Max.	FALSCH
8382	Ausgangspuffer ist zu klein für N	FALSCH
8351	Datentyp für Ausgangspuffer nicht zulässig	FALSCH

9.2.4 Zeichenkettenanweisungen

Ihr Steuerungsprogramm kann mit den folgenden Zeichenketten- und Zeichenanweisungen Meldungen für Bedieneranzeigen und Prozessprotokolle erzeugen.

9.2.4.1 Anweisung MAX_LEN (Maximale Länge einer Zeichenkette)

Tabelle 9- 47 Anweisung Maximale Länge

KOP/FUP	SCL	Beschreibung
	`out := MAX_LEN(in);`	MAX_LEN (Maximale Länge der Zeichenkette) gibt die der Zeichenkette IN am Ausgang OUT zugewiesene maximale Länge an. Wenn während der Verarbeitung der Anweisung Fehler auftreten, wird eine leere Zeichenkettenlänge ausgegeben. Die Datentypen String und WString enthalten zwei Längen: Das erste Byte (oder Wort) gibt die maximale Länge an und das zweite Byte (oder Wort) gibt die aktuelle Länge an (dies ist die derzeitige Anzahl gültiger Zeichen). • Die maximale Länge der Zeichenkette wird für jede String- oder WString-Deklaration in eckigen Klammern angegeben. Die Anzahl der von einem String belegten Bytes ist um 2 Bytes größer als die maximale Länge. Die Anzahl der von einem WString belegten Wörter ist um 2 Wörter größer als die maximale Länge. • Die aktuelle Länge stellt die Anzahl der tatsächlich verwendeten Zeichen dar. Die tatsächliche Länge muss kleiner oder gleich der maximalen Länge sein. Die aktuelle Länge wird für einen String in Bytes und für einen WString in Wörtern angegeben. Mit der Anweisung MAX_LEN i rufen Sie die maximale Länge der Zeichenkette ab und mit der Anweisung LEN die aktuelle Länge einer Zeichenkette.

Tabelle 9- 48 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	String, WString	Eingegebene Zeichenkette
OUT	OUT	DInt	Maximal zulässige Anzahl Zeichen in der Zeichenkette IN

9.2.4.2 Anweisung LEN (Länge einer Zeichenkette ermitteln)

Tabelle 9- 49 Anweisung LEN

KOP/FUP	SCL	Beschreibung
	`out := LEN(in);`	LEN (Länge) gibt die aktuelle Länge der Zeichenkette IN am Ausgang OUT aus. Eine leere Zeichenkette hat die Länge Null.

Tabelle 9- 50 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	String, WString	Eingegebene Zeichenkette
OUT	OUT	Int, DInt, Real, LReal	Anzahl gültiger Zeichen in Zeichenkette IN

Tabelle 9- 51 ENO-Zustand

ENO	Bedingung	OUT
1	Keine gültige Zeichenkette	Gültige Zeichenkettenlänge
0	Aktuelle Länge von IN überschreitet die maximale Länge von IN	Aktuelle Länge wird auf 0 gesetzt
	Maximale Länge von IN passt nicht in den zugewiesenen Speicherbereich	
	Maximale Länge von IN ist 255 (unzulässige Länge)	

9.2.4.3 Anweisung CONCAT (Zeichenketten verketten)

Tabelle 9- 52 Anweisung Zeichenketten verketten

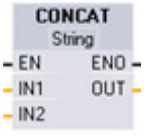
KOP/FUP	SCL	Beschreibung
	`out := CONCAT(in1, in2);`	CONCAT (Zeichenketten verketten) verknüpft die String-Parameter IN1 und IN2 zu einer Zeichenkette, die am Ausgang OUT bereitsteht. Nach der Verknüpfung ist Zeichenkette IN1 der linke und Zeichenkette IN2 der rechte Teil der kombinierten Zeichenkette.

Tabelle 9- 53 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN1	IN	String, WString	Eingegebene Zeichenkette 1
IN2	IN	String, WString	Eingegebene Zeichenkette 2
OUT	OUT	String, WString	Kombinierte Zeichenkette (Zeichenkette 1 + Zeichenkette 2)

Tabelle 9- 54 ENO-Zustand

ENO	Bedingung	OUT
1	Keine Fehler	Gültige Zeichen
0	Resultierende Zeichenkette nach Verknüpfung ist größer als die maximale Länge von Zeichenkette OUT	Die resultierenden Zeichen werden kopiert, bis die maximale Länge von OUT erreicht ist
	Die aktuelle Länge von IN1 überschreitet die maximale Länge von IN1, die aktuelle Länge von IN2 überschreitet die maximale Länge von IN2 oder die aktuelle Länge von OUT überschreitet die maximale Länge von OUT (ungültige Zeichenkette)	Aktuelle Länge wird auf 0 gesetzt
	Maximale Länge von IN1, IN2 oder OUT passt nicht in den zugewiesenen Speicherbereich	
	Die maximale Länge von IN1 oder IN2 ist 255, die maximale Länge von OUT ist 0 oder 255 (Datentyp String)	

ENO	Bedingung	OUT
	Die maximale Länge von IN1 oder IN2 ist 65534, die maximale Länge von OUT ist 0 oder 65534 (Datentyp WString)	

9.2.4.4 Anweisungen LEFT, RIGHT und MID (Teilzeichenketten in einer Zeichenkette lesen)

Tabelle 9- 55 Anweisungen LEFT, RIGHT und MID

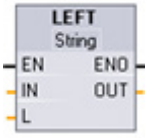
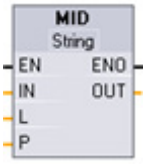
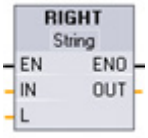
KOP/FUP	SCL	Beschreibung
	`out := LEFT(in, L);`	LEFT (Linke Teilzeichenkette) gibt eine Teilzeichenkette aus, die aus den ersten L Zeichen von Parameter IN besteht. • Ist L größer als die aktuelle Länge der Zeichenkette IN, so wird die gesamte Zeichenkette IN an Ausgang OUT ausgegeben. • Wird eine leere Zeichenkette eingegeben, so wird auch eine leere Zeichenkette an OUT ausgegeben.
	`out := MID(in, L, p);`	MID (Mittlere Teilzeichenkette) gibt den mittleren Teil einer Zeichenkette aus. Die mittlere Teilzeichenkette ist L Zeichen lang und beginnt an Zeichenposition P (einschließlich). Wenn die Summe aus L und P die aktuelle Länge des String-Parameters IN überschreitet, wird eine Teilzeichenkette von Zeichenposition P bis zum Ende von Zeichenkette IN ausgegeben.
	`out := RIGHT(in, L);`	RIGHT (Rechte Teilzeichenkette) gibt die letzten L Zeichen einer Zeichenkette aus. • Ist L größer als die aktuelle Länge der Zeichenkette IN, so wird die gesamte Zeichenkette IN an Ausgang OUT ausgegeben. • Wird eine leere Zeichenkette eingegeben, so wird auch eine leere Zeichenkette an OUT ausgegeben.

Tabelle 9- 56 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	String, WString	Eingegebene Zeichenkette
L	IN	Int	Länge der zu erstellenden Teilzeichenkette: • LEFT verwendet die Anzahl der Zeichen ganz links in der Zeichenkette • RIGHT verwendet die Anzahl der Zeichen ganz rechts in der Zeichenkette • MID verwendet die Anzahl der Zeichen ab Position P in der Zeichenkette
P	IN	Int	Nur MID: Position des ersten Zeichens in der Teilzeichenkette, das kopiert werden soll P = 1, für die Position des ersten Zeichens der Zeichenkette IN
OUT	OUT	String, WString	Ausgegebene Zeichenkette

Tabelle 9- 57 ENO-Zustand

ENO	Bedingung	OUT
1	Keine Fehler	Gültige Zeichen
0	• L oder P ist kleiner oder gleich 0 • P ist größer als die maximale Länge von IN • Aktuelle Länge von IN überschreitet maximale Länge von IN, oder aktuelle Länge von OUT überschreitet maximale Länge von OUT • Maximale Länge von IN oder OUT passt nicht in den zugewiesenen Speicher • Die maximale Länge von IN oder OUT ist 0 oder 255 (Datentyp String) oder 0 oder 65534 (Datentyp WString)	Aktuelle Länge wird auf 0 gesetzt
	Länge (L) der Teilzeichenkette, die kopiert werden soll, ist größer als die maximale Länge von Zeichenkette OUT	Es werden so viele Zeichen kopiert, bis die maximale Länge von OUT erreicht ist
	Nur MID: L oder P ist kleiner oder gleich 0	Aktuelle Länge wird auf 0 gesetzt
	Nur MID: P ist größer als die maximale Länge von IN	
	Aktuelle Länge von IN1 überschreitet maximale Länge von IN1 oder aktuelle Länge von IN2 überschreitet maximale Länge von IN2 (ungültige Zeichenkette)	Aktuelle Länge wird auf 0 gesetzt
	Maximale Länge von IN1, IN2 oder OUT passt nicht in den zugewiesenen Speicherbereich	
	Die maximale Länge von IN1, IN2 oder OUT ist unzulässig: 0 oder 255 (Datentyp String) oder 0 oder 65534 (Datentyp WString)	

9.2.4.5 Anweisung DELETE (Zeichen in einer Zeichenkette löschen)

Tabelle 9- 58 Anweisung Teilzeichenkette löschen

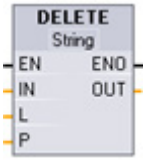
KOP/FUP	SCL	Beschreibung
	`out := DELETE(in, L, p);`	Löscht L Zeichen aus der Zeichenkette IN. Mit dem Löschen der Zeichen wird an Zeichenposition P (einschließlich) begonnen, und die restliche Teilzeichenkette wird an Parameter OUT ausgegeben. • Ist L gleich Null, so wird die eingegebene Zeichenkette an OUT ausgegeben. • Ist die Summe aus L und P größer als die Länge der eingegebenen Zeichenkette, so wird die Zeichenkette bis zum Ende gelöscht.

Tabelle 9- 59 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN	IN	String, WString	Eingegebene Zeichenkette
L	IN	Int	Anzahl der zu löschenden Zeichen
P	IN	Int	Position des ersten Zeichens, das gelöscht werden soll: Das erste Zeichen der Zeichenkette IN hat Position 1.
OUT	OUT	String, WString	Ausgegebene Zeichenkette

Tabelle 9- 60 ENO-Zustand

ENO	Bedingung	OUT
1	Keine Fehler	Gültige Zeichen
0	P ist größer als die aktuelle Länge von IN	IN wird in OUT kopiert, es werden keine Zeichen gelöscht
	Resultierende Zeichenkette nach Löschen ist größer als die maximale Länge von Zeichenkette OUT	Die resultierenden Zeichen werden kopiert, bis die maximale Länge von OUT erreicht ist
	L ist kleiner als 0 oder P ist kleiner oder gleich 0	Aktuelle Länge wird auf 0 gesetzt
	Aktuelle Länge von IN überschreitet maximale Länge von IN, oder aktuelle Länge von OUT überschreitet maximale Länge von OUT	
	Maximale Länge von IN oder OUT passt nicht in den zugewiesenen Speicher	
	Maximale Länge von IN oder OUT ist 0 oder 255	

9.2.4.6 Anweisung INSERT (Zeichen in einer Zeichenkette einfügen)

Tabelle 9- 61 Anweisung Teilzeichenkette einfügen

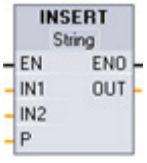
KOP/FUP	SCL	Beschreibung
	`out := INSERT(in1, in2, p);`	Fügt die Zeichenkette IN2 in die Zeichenkette IN1 ein. Das Einfügen beginnt nach dem Zeichen an Position P.

Tabelle 9- 62 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN1	IN	String, WString	Eingegebene Zeichenkette 1
IN2	IN	String, WString	Eingegebene Zeichenkette 2
P	IN	Int	Letzte Zeichenposition in Zeichenkette IN1 vor dem Einfügungspunkt für Zeichenkette IN2. Das erste Zeichen der Zeichenkette IN1 hat Position 1.
OUT	OUT	String, WString	Resultierende Zeichenkette

Tabelle 9- 63 ENO-Zustand

ENO	Bedingung	OUT
1	Keine Fehler	Gültige Zeichen
0	P ist größer als die Länge von IN1	IN2 wird unmittelbar nach dem letzten Zeichen von IN1 mit IN1 verkettet
	P ist kleiner als 0	Aktuelle Länge wird auf 0 gesetzt
	Resultierende Zeichenkette nach Einfügen ist größer als die maximale Länge von Zeichenkette OUT	Die resultierenden Zeichen werden kopiert, bis die maximale Länge von OUT erreicht ist
	Die aktuelle Länge von IN1 überschreitet die maximale Länge von IN1, die aktuelle Länge von IN2 überschreitet die maximale Länge von IN2 oder die aktuelle Länge von OUT überschreitet die maximale Länge von OUT (ungültige Zeichenkette)	Aktuelle Länge wird auf 0 gesetzt
	Maximale Länge von IN1, IN2 oder OUT passt nicht in den zugewiesenen Speicherbereich	
	Die maximale Länge von IN1 oder IN2 ist 255 oder maximale Länge von OUT ist 0 oder 255 (Datentyp String)	
	Die maximale Länge von IN1 oder IN2 ist 65534 oder maximale Länge von OUT ist 0 oder 65534 (Datentyp WString)	

9.2.4.7 Anweisung REPLACE (Zeichen in einer Zeichenkette ersetzen)

Tabelle 9- 64 Anweisung Teilzeichenkette ersetzen

KOP/FUP	SCL	Beschreibung

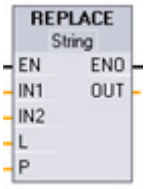
	`out := REPLACE(`
	`in1:=_string_in_,`
	`in2:=_string_in_,`
	`L:=_int_in_,`
	`p:=_int_in);`
		Ersetzt L Zeichen in der Zeichenkette im Parameter IN1. Das Ersetzen beginnt an Zeichenposition P (einschließlich) der Zeichenkette IN1, die Ersetzungszeichen werden aus Parameter IN2 übernommen.

Tabelle 9- 65 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN1	IN	String, WString	Eingegebene Zeichenkette
IN2	IN	String, WString	Zeichenkette mit Ersatzzeichen
L	IN	Int	Anzahl der Zeichen, die ersetzt werden sollen
P	IN	Int	Position des ersten Zeichens, das ersetzt werden soll
OUT	OUT	String, WString	Resultierende Zeichenkette

Ist Parameter L gleich Null, so wird Zeichenkette IN2 an Position P von Zeichenkette IN1 eingefügt, ohne dass Zeichen in Zeichenkette IN1 gelöscht werden.

Ist P gleich 1, werden die ersten L Zeichen von Zeichenkette IN1 durch die Zeichen der Zeichenkette IN2 ersetzt.

Tabelle 9- 66 ENO-Zustand

ENO	Bedingung	OUT
1	Keine Fehler	Gültige Zeichen
0	P ist größer als die Länge von IN1	IN2 wird unmittelbar nach dem letzten Zeichen von IN1 mit IN1 verkettet
	P Punkte in IN1, aber weniger als L Zeichen in IN1	IN2 ersetzt das Endzeichen von IN1 ab Position P
	Resultierende Zeichenkette nach Ersetzen ist größer als die maximale Länge von Zeichenkette OUT	Die resultierenden Zeichen werden kopiert, bis die maximale Länge von OUT erreicht ist
	Maximale Länge von IN1 ist 0	IN2 Zeichen werden kopiert in OUT
	L ist kleiner als 0 oder P ist kleiner oder gleich 0	Aktuelle Länge wird auf 0 gesetzt
	Die aktuelle Länge von IN1 überschreitet die maximale Länge von IN1, die aktuelle Länge von IN2 überschreitet die maximale Länge von IN2 oder die aktuelle Länge von OUT überschreitet die maximale Länge von OUT	
	Maximale Länge von IN1, IN2 oder OUT passt nicht in den zugewiesenen Speicherbereich	
	Die maximale Länge von IN1 oder IN2 ist 255 oder maximale Länge von OUT ist 0 oder 255 (Datentyp String)	

ENO	Bedingung	OUT
	Die maximale Länge von IN1 oder IN2 ist 65534 oder maximale Länge von OUT ist 0 oder 65534 (Datentyp WString)	

9.2.4.8 Anweisung FIND (Zeichen in einer Zeichenkette finden)

Tabelle 9- 67 Anweisung Teilzeichenkette finden

KOP/FUP	SCL	Beschreibung

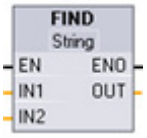
	`out := FIND(`
	`in1:=_string_in_,`
	`in2:=_string_in);`
		Liefert die Zeichenposition der von IN2 angegebenen Zeichenkette in der Zeichenkette IN1. Die Suche beginnt links. Die Zeichenposition des ersten Vorkommens von IN2 wird an OUT ausgegeben. Wird die Zeichenkette IN2 nicht in der Zeichenkette IN1 gefunden, so wird Null ausgegeben.

Tabelle 9- 68 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
IN1	IN	String, WString	In dieser Zeichenkette suchen
IN2	IN	String, WString	Diese Zeichenkette suchen
OUT	OUT	Int	Zeichenposition in Zeichenkette IN1, erstes positives Suchergebnis

Tabelle 9- 69 ENO-Zustand

ENO	Bedingung	OUT
1	Keine Fehler	Gültige Zeichenposition
0	IN2 ist größer als IN1	Zeichenposition wird auf 0 gesetzt
	Aktuelle Länge von IN1 überschreitet maximale Länge von IN1 oder aktuelle Länge von IN2 überschreitet maximale Länge von IN2 (ungültige Zeichenkette)	
	Maximale Länge von IN1 oder IN2 passt nicht in den zugewiesenen Speicherbereich	
	Die maximale Länge von IN1 oder IN2 ist 255 (Datentyp String) oder 65535 (Datentyp WString)	

9.3 Dezentrale E/A (PROFINET, PROFIBUS oder AS-i)

9.3.1 Anweisungen für die dezentrale E/A

Die folgenden Anweisungen für die dezentrale Peripherie können mit PROFINET, PROFIBUS oder AS-i verwendet werden:

- Anweisung RDREC (Seite 357): Mit der Nummer INDEX aus einem Modul oder Gerät können Sie einen Datensatz lesen.
- Anweisung WRREC (Seite 357): Sie können einen Datensatz mit der Nummer INDEX in ein durch ID angegebenes Modul oder Gerät übertragen.
- Anweisung RALRM (Seite 360): Sie können einen Alarm mit allen entsprechenden Informationen von einem Modul oder Gerät empfangen und diese Informationen an die Ausgangsparameter übergeben.
- Anweisung DPRD_DAT (Seite 368): Konsistente Datenbereiche größer als 64 Byte müssen Sie über die Anweisung DPRD_DAT aus einem Modul oder Gerät auslesen.
- Anweisung DPWR_DAT (Seite 368): Konsistente Datenbereiche größer als 64 Byte müssen Sie über die Anweisung DPWR_DAT in ein Modul oder Gerät schreiben.

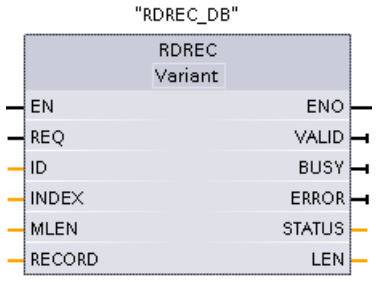
Die Anweisung DPNRM_DG (Seite 371) kann nur mit PROFIBUS verwendet werden. Die aktuellen Diagnosedaten eines DP-Slaves können Sie in dem Format auslesen, das in EN 50 170 Band 2, PROFIBUS, angegeben ist.

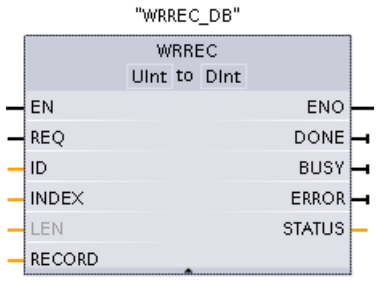
9.3.2 Anweisungen RDREC und WRREC (Datensatz lesen/schreiben)

Die Anweisungen RDREC (Datensatz lesen) und WRREC (Datensatz schreiben) können Sie für PROFINET, PROFIBUS und AS-i einsetzen.

Tabelle 9- 70 Anweisungen RDREC und WRREC

KOP/FUP	SCL	Beschreibung

	`"RDREC_DB"(`
	`req:=_bool_in_,`
	`ID:=_word_in_,`
	`index:=_dint_in_,`
	`mlen:=_uint_in_,`
	`valid=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_dword_out_,`
	`len=>_uint_out_,`
	`rec-`
	`ord:=_variant_inout_);`
		Mit der Anweisung RDREC lesen Sie einen Datensatz mit der Nummer INDEX aus der Komponente, die von ID angegeben wird, z. B. aus einem zentralen Baugruppenträger oder einer dezentralen Komponente (PROFIBUS DP oder PROFINET IO). Geben Sie die maximale Anzahl der zu lesenden Bytes in MLEN an. Die ausgewählte Länge des Zielbereichs RECORD muss mindestens die Länge der von MLEN angegebenen Bytes haben.

	`"WRREC_DB"(`
	`req:=_bool_in_,`
	`ID:=_word_in_,`
	`index:=_dint_in_,`
	`len:=_uint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_dword_out_,`
	`rec-`
	`ord:=_variant_inout_);`
		Mit der Anweisung WRREC übertragen Sie einen Datensatz (RECORD) mit der Nummer INDEX an einen DP-Slave bzw. eine PROFINET IO-Gerätekomponente, der/die von ID angegeben wird, z. B. ein Modul im zentralen Baugruppenträger oder an eine dezentrale Komponente (PROFIBUS DP oder PROFINET IO).
		Geben Sie die Byte-Länge des zu übertragenden Datensatzes an. Die ausgewählte Länge des Quellbereichs RECORD muss deshalb mindestens die Länge der von LEN angegebenen Bytes haben.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 In den SCL-Beispielen sind "RDREC_DB" und "WRREC_DB" die Namen der Instanz-DBs.

Tabelle 9- 71 Datentypen für die Parameter von RDREC und WRREC

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	REQ = 1: Datensatz übertragen
ID	IN	HW_IO (Word)	Logische Adresse der DP-Slave/PROFINET IO-Komponente (Modul oder Submodul): • Bei einem Ausgabemodul muss Bit 15 gesetzt werden (zum Beispiel für Adresse 5: ID:=DW#16#8005). • Bei einem Kombimodul muss die kleinere der beiden Adressen angegeben werden. **Hinweis**: In V3.0 kann die Geräte-ID wahlweise auf zwei Arten ermittelt werden: • Durch Auswahl der folgenden Optionen in der "Netzsicht": – Gerät (graues Feld) – "Eigenschaften" des Geräts – "Hardwarekennung" **Hinweis**: Es wird jedoch nicht für alle Geräte die Hardwarekennung angezeigt. • Durch Auswahl der folgenden Optionen in der "Projektnavigation": – PLC-Variablen – Standardvariablentabelle – Register "Konstanten" des Systems Die Hardwarekennungen aller konfigurierten Geräte werden angezeigt. **Hinweis**: In V4.0 wird die Geräte-ID (Hardwarekennung) für das Schnittstellenmodul in der Variablentabelle über den Parameter "Gerätename [HEAD]" unter den Systemkonstanten ermittelt.
INDEX	IN	Byte, Word, USInt, UInt, SInt, Int, DInt	Datensatznummer
MLEN	IN	Byte, USInt, UInt	Maximale Länge in Bytes der abzurufenden Informationen aus dem Datensatz (RDREC)
VALID	OUT	Bool	Neuer Datensatz wurde empfangen und ist gültig (RDREC). Das Bit VALID ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
DONE	OUT	Bool	Datensatz wurde übertragen (WRREC). Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
BUSY	OUT	Bool	• BUSY = 1: Der Lesevorgang (RDREC) oder Schreibvorgang (WRREC) ist noch nicht beendet. • BUSY = 0: Die Übertragung des Datensatzes ist beendet.
ERROR	OUT	Bool	ERROR = 1: Ein Lesefehler (RDREC) bzw. Schreibfehler (WRREC) ist aufgetreten. Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	DWord	Bausteinstatus oder Fehlerinformationen (Seite 464)

Parameter und Datentyp		Datentyp	Beschreibung
LEN	OUT (RDREC) IN (WRREC)	UInt	• Länge der abgerufenen Informationen aus dem Datensatz (RDREC) • Maximale Byte-Länge des zu übertragenden Datensatzes (WRREC)
RECORD	IN_OUT	Variant	• Zielbereich für den abgerufenen Datensatz (RDREC) • Datensatz (WRREC)

Die Anweisungen RDREC und WRREC funktionieren asynchron, d.h. die Verarbeitung deckt mehrere Anweisungsaufrufe ab. Sie starten den Auftrag durch Aufruf von RDREC oder WRREC mit REQ = 1.

Der Auftragsstatus wird über den Ausgangsparameter BUSY und die beiden zentralen Bytes des Ausgangsparameters STATUS angezeigt. Die Übertragung des Datensatzes ist beendet, wenn der Ausgangsparameter BUSY auf den Wert FALSE gesetzt wurde.

Ist der Ausgangsparameter VALID (RDREC) oder DONE (WRREC) gleich TRUE (nur einen Zyklus lang), weist dies darauf hin, dass der Datensatz erfolgreich in den Zielbereich RECORD (RDREC) oder zum Zielgerät (WRREC) übertragen wurde. Bei RDREC enthält der Ausgangsparameter LEN die Länge der abgerufenen Daten in Bytes.

Der Ausgangsparameter ERROR (nur einen Zyklus lang, wenn ERROR = TRUE) zeigt an, dass beim Übertragen des Datensatzes ein Fehler aufgetreten ist. In diesem Fall enthält der Ausgangsparameter STATUS (nur während des einen Zyklus, in dem ERROR = TRUE) die Fehlerinformationen.

Datensätze werden vom Hersteller des Hardwaregeräts definiert. Ausführliche Informationen zu einem Datensatz finden Sie in der Gerätedokumentation des jeweiligen Hardwareherstellers.

Hinweis

Wenn ein DPV1-Slave über eine GSD-Datei (GSD ab Version 3) konfiguriert ist und die DP-Schnittstelle des DP-Masters auf "S7-kompatibel" gesetzt ist, dann können Sie im Anwenderprogramm mit "RDREC" keine Datensätze aus den E/A-Modulen lesen und mit "WRREC" keine Datensätze in die E/A-Module schreiben. In einem solchen Fall adressiert der DP-Master den falschen Steckplatz (konfigurierter Steckplatz + 3).

Abhilfe: Richten Sie als Schnittstelle des DP-Masters "DPV1" ein.

Hinweis

Die Schnittstellen der Anweisungen "RDREC" und "WRREC" sind mit den FBs "RDREC" und "WRREC" identisch, die in der "PROFIBUS-Richtlinie für die PROFIBUS-Kommunikation und Proxy-Funktionsbausteine nach IEC 61131-3" definiert sind.

Hinweis

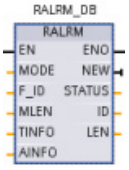
Wenn Sie "RDREC" oder "WRREC" verwenden, um einen Datensatz für PROFINET IO zu lesen oder zu schreiben, werden negative Werte in den Parametern INDEX, MLEN und LEN als vorzeichenlose 16-Bit-Ganzzahlen interpretiert.

9.3.3 Anweisung RALRM (Alarm empfangen)

Die Anweisung RALRM (Alarm lesen) können Sie für PROFINET und PROFIBUS einsetzen.

Tabelle 9- 72 Anweisung RALRM

KOP/FUP	SCL	Beschreibung

	`"RALRM_DB"(`
	`mode:=_int_in_,`
	`f_ID:=_word_in_,`
	`mlen:=_uint_in_,`
	`new=>_bool_out_,`
	`status=>_dword_out_,`
	`ID=>_word_out_,`
	`len=>_uint_out_,`
	`tinfo:=_variant_inout_,`
	`ainfo:=_variant_inout_);`
		Mit der Anweisung RALRM (Alarm lesen) lesen Sie Diagnosealarminformationen aus einem PROFIBUS- oder PROFINET I/O-Modul/Gerät.
		Die Informationen in den Ausgangsparametern enthalten die Startinformationen des aufgerufenen OBs sowie die Informationen der Alarmquelle.
		Rufen Sie RALRM in einem Alarm-OB auf, um zu den Ereignissen, die den Alarm verursacht haben, Informationen auszugeben. Bei der S7-1200 werden die folgenden Diagnosealarm-OBs unterstützt: Status, Aktualisierung, Profil, Diagnosefehleralarm, Ziehen oder Stecken von Modulen, Baugruppenträger- oder Stationsausfall

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

[2] Im SCL-Beispiel ist "RALRM_DB" der Name des Instanz-DBs.

Tabelle 9- 73 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
MODE	IN	Byte, USInt, SInt, Int	Betriebszustand
F_ID	IN	HW_IO (Word)	Logische Anfangsadresse der Komponente (des Moduls), von der Alarme empfangen werden sollen. **Hinweis**: Die Geräte-ID kann wahlweise auf zwei Arten ermittelt werden: • Durch Auswahl der folgenden Optionen in der "Netzsicht": – Gerät (graues Feld) – "Eigenschaften" des Geräts – "Hardwarekennung" **Hinweis**: Es wird nicht für alle Geräte die Hardwarekennung angezeigt. • Durch Auswahl der folgenden Optionen in der "Projektnavigation": – PLC-Variablen – Standardvariablentabelle – Register "Konstanten" des Systems – Die Hardwarekennungen aller konfigurierten Geräte werden angezeigt.
MLEN	IN	Byte, USInt, UInt	Maximale Länge in Byte der zu empfangenden Informationen aus dem Datensatz. Ist MLEN gleich 0, können so viele Alarminformationen empfangen werden, wie im Zielbereich AINFO zur Verfügung stehen.
NEW	OUT	Bool	Ein neuer Alarm wurde empfangen.
STATUS	OUT	DWord	Status der Anweisung RALRM. Weitere Informationen finden Sie unter "STATUS-Parameter für RDREC, WRREC und RALRM" (Seite 364).

Parameter und Datentyp		Datentyp	Beschreibung
ID	OUT	HW_IO (Word)	Hardwarekennung des E/A-Moduls, das den Diagnosealarm verursacht hat **Hinweis**: In der Beschreibung des Parameters F_ID finden Sie eine Erläuterung dazu, wie Sie die Geräte-ID ermitteln.
LEN	OUT	DWord, UInt, UDInt, DInt, Real, LReal	Länge der empfangenen Alarminformationen AINFO
TINFO	IN_OUT	Variant	Aufgabeninformationen: Zielbereich für Start- und Verwaltungsinformationen des OBs. Die Länge TINFO ist immer 32 Byte.
AINFO	IN_OUT	Variant	Alarminformationen: Zielbereich für Header-Informationen und zusätzliche Alarminformationen. Für AINFO müssen Sie eine Länge von mindestens der in MLEN angegebenen Bytezahl angeben, wenn MLEN größer als 0 ist. Die Länge von AINFO ist variabel.

Hinweis

Wenn Sie "RALRM" in einem OB aufrufen, dessen Startereignis kein E/A-Alarm ist, liefert die Anweisung entsprechend weniger Informationen in ihren Ausgängen.

Stellen Sie sicher, dass Sie unterschiedliche Instanz-DBs verwenden, wenn Sie "RALRM" in unterschiedlichen OBs aufrufen. Wenn Sie die von einem "RALRM"-Aufruf resultierenden Daten außerhalb des zugehörigen Alarm-OBs auswerten, müssen Sie für jedes OB-Startereignis einen eigenen Instanz-DB verwenden.

Hinweis

Die Schnittstelle der Anweisung "RALRM" ist mit dem FB "RALRM" identisch, der in der "PROFIBUS-Richtlinie für die PROFIBUS-Kommunikation und Proxy-Funktionsbausteine nach IEC 61131-3" definiert ist.

RALRM aufrufen

Die Anweisung RALRM können Sie in drei verschiedenen Betriebsarten (MODE) aufrufen.

Tabelle 9- 74 Betriebsarten der Anweisung RALRM

MODE	Beschreibung
0	• ID enthält die Hardwarekennung des E/A-Moduls, das den Alarm ausgelöst hat. • Der Ausgangsparameter NEW wird auf WAHR gesetzt. • LEN erzeugt einen Ausgang von 0. • AINFO und TINFO werden nicht mit Informationen aktualisiert.
1	• ID enthält die Hardwarekennung des E/A-Moduls, das den Alarm ausgelöst hat. • Der Ausgangsparameter NEW wird auf WAHR gesetzt. • LEN erzeugt einen Ausgang in der Größe der Byteanzahl der ausgegebenen AINFO-Daten. • AINFO und TINFO werden mit alarmbezogenen Informationen aktualisiert.
2	Wenn die Hardwarekennung des Eingangsparameters F_ID den Alarm ausgelöst hat, dann gilt Folgendes: • ID enthält die Hardwarekennung des E/A-Moduls, das den Alarm ausgelöst hat. Dieser Wert muss mit dem Wert von F_ID übereinstimmen. • Der Ausgangsparameter NEW wird auf WAHR gesetzt. • LEN erzeugt einen Ausgang in der Größe der byteanzahl der ausgegebenen AINFO-Daten. • AINFO und TINFO werden mit alarmbezogenen Informationen aktualisiert.

Hinweis

Wenn Sie einen zu kurzen Zielbereich für TINFO oder AINFO zuweisen, kann RALRM nicht die vollständigen Informationen zurückgeben.

MLEN kann die ausgegebene AINFO-Datenmenge begrenzen.

Informationen dazu, wie Sie die Daten von TINFO und AINFO interpretieren, finden Sie bei den Parametern von AINFO und TINFO im Online-Informationssystem von STEP 7.

Daten des Organisationsbausteins TInfo

Die folgende Tabelle zeigt, wie die TInfo-Daten für die Anweisung RALRM angeordnet sind:

Ebenso für OBs: Status, Aktualisierung, Profil, Diagnosefehleralarm, Ziehen oder Stecken von Modulen, Baugruppenträger- oder Stationsausfall	0	SI-Format	OB-Klasse	OB-Nr.	
	4	LADDR			
TI_Submodule - OBs: Status, Aktualisierung, Profil	4			Steckplatz	
	8	Spezifikator		0	
TI_DiagnosticInterrupt - OB: Diagnosefehleralarm	4			E/A-Zustand	
	8	Kanal		Multifehler	0
TI_PlugPullModule - OB: Ziehen oder Stecken von Modulen	4			Ereignisklasse	Fehler-ID
	8	0		0	
TI_StationFailure - OB: Baugruppenträger- oder Stationsausfall	4			Ereignisklasse	Fehler-ID
	8	0		0	
Ebenso für OBs: Status, Aktualisierung, Profil, Diagnosefehleralarm, Ziehen oder Stecken von Modulen, Baugruppenträger- oder Stationsausfall	12	0			
	16				
	20	Adresse		slv_prfl	intr_type
	24	Merker1	Merker2	ID	
	28[1]	Hersteller		Instanz	

[1] Bytes 28 - 31 (Hersteller und Instanz) werden bei PROFIBUS nicht verwendet.

Hinweis

Ausführlichere Informationen zu den TINFO-Daten finden Sie im Online-Informationssystem von STEP 7.

9.3.4 STATUS-Parameter für RDREC, WRREC und RALRM

Der Ausgangsparameter STATUS enthält Fehlerinformationen, die als ARRAY[1...4] OF BYTE mit der folgenden Struktur ausgewertet werden:

Tabelle 9- 75 Ausgangs-Array STATUS

Array-Element	Name	Beschreibung
STATUS[1]	Function_Num	• B#16#00, wenn kein Fehler vorliegt • Funktions-ID von DPV1-PDU: Tritt ein Fehler auf, wird , B#16#80 durch ODER verknüpft (beim Lesen des Datensatzes mit: B#16#DE; beim Schreiben des Datensatzes mit: B#16#DF). Wenn kein DPV1-Protokollelement verwendet wird, wird B#16#C0 ausgegeben.
STATUS[2]	Error_Decode	Adresse der Fehler-ID
STATUS[3]	Error_Code_1	Fehler-ID
STATUS[4]	Error_Code_2	Herstellerspezifische Erweiterung der Fehler-ID

Tabelle 9- 76 STATUS[2]-Werte

Error_decode (B#16#....)	Quelle	Beschreibung
00 bis 7F	CPU	Kein Fehler und keine Warnung
80	DPV1	Fehler gemäß IEC 61158-6
81 bis 8F	CPU	B#16#8x zeigt einen Fehler im "x-ten" Aufrufparameter der Anweisung.
FE, FF	DP-Profil	Profilspezifischer Fehler

Tabelle 9- 77 STATUS[3]-Werte

Error_decode (B#16#....)	Error_code_1 (B#16#....)	Erläuterung (DVP1)	Beschreibung
00	00		Kein Fehler, keine Warnung
70	00	Reserviert, ablehnen	Erster Aufruf; keine aktive Datensatzübertragung
	01	Reserviert, ablehnen	Erster Aufruf; Datensatzübertragung hat begonnen
	02	Reserviert, ablehnen	Zwischenzeitlicher Aufruf; Datensatzübertragung bereits aktiv
80	90	Reserviert, übergeben	Ungültige logische Anfangsadresse
	92	Reserviert, übergeben	Unzulässiger Typ für Pointer Variant
	93	Reserviert, übergeben	Die über ID oder F_ID adressierte DP-Komponente ist nicht konfiguriert.

Error_decode (B#16#....)	Error_code_1 (B#16#....)	Erläuterung (DVP1)	Beschreibung
	96		Die Anweisung RALRM (Seite 360) kann die Anlaufinformationen, Verwaltungsinformationen, Headerinformationen und weitere Alarminformationen des OBs nicht liefern. Bei den folgenden OBs können Sie die Anweisung DPNRM_DG (Seite 371) verwenden, um die aktuellen Diagnosemeldungen des betreffenden DP-Slaves asynchron zu lesen (Adressinformationen aus den Anlaufinformationen des OBs): • Prozessalarm (Seite 93) • Status (Seite 99), Aktualisierung (Seite 99) oder Profil (Seite 100) • Diagnosefehleralarm (Seite 95) • Ziehen oder Stecken von Modulen (Seite 97)
	A0	Lesefehler	Negative Quittierung beim Lesen aus Modul
	A1	Schreibfehler	Negative Quittierung beim Schreiben in Modul
	A2	Modulausfall	DP-Protokollfehler in Schicht 2 (Beispiel: Slave-Ausfall oder Busprobleme)
	A3	Reserviert, übergeben	• PROFIBUS DP: DP-Protokollfehler mit Direct-Data-Link-Mapper oder User-Interface/User • PROFINET IO: Allgemeiner CM-Fehler
	A4	Reserviert, übergeben	Kommunikation auf dem Kommunikationsbus unterbrochen
	A5	Reserviert, übergeben	-
	A7	Reserviert, übergeben	DP-Slave oder Module belegt (temporärer Fehler).
	A8	Versionskonflikt	DP-Slave oder Modul meldet nicht kompatible Versionen.
	A9	Funktion nicht unterstützt	Funktion wird von DP-Slave oder Modul nicht unterstützt
	AA bis AF	Benutzerspezifisch	DP-Slave oder Modul meldet herstellerspezifischen Fehler in der Anwendung. Prüfen Sie die Dokumentation des Herstellers des DP-Slaves oder des Moduls.
	B0	Ungültiger Index	Datensatz im Modul unbekannt; unzulässige Datensatznummer ≥ 256
	B1	Längenfehler beim Schreiben	Die Längeninformationen im Parameter RECORD sind falsch. • Bei RALRM: Längenfehler in AINFO **Hinweis**: Das Online-Informationssystem von STEP 7 bietet unmittelbaren Zugriff auf Informationen dazu, wie die von AINFO zurückgegebenen Puffer auszuwerten sind. • Bei RDREC (Seite 357) und WRREC (Seite 357): Längenfehler in MLEN
	B2	Ungültiger Steckplatz	Der konfigurierte Steckplatz ist nicht belegt.
	B3	Typkonflikt	Tatsächlicher Modultyp entspricht nicht dem angegebenen Modultyp.

Error_decode (B#16#....)	Error_code_1 (B#16#....)	Erläuterung (DVP1)	Beschreibung
	B4	Ungültiger Bereich	DP-Slave oder Modul meldet Zugriff auf ungültigen Datenbereich.
	B5	Statuskonflikt	DP-Slave oder Modul nicht betriebsbereit
	B6	Zugriff verweigert	DP-Slave oder Modul verweigert den Zugriff.
	B7	Ungültiger Bereich	DP-Slave oder Modul meldet einen ungültigen Bereich für einen Parameter oder Wert.
	B8	Ungültiger Parameter	DP-Slave oder Modul meldet einen ungültigen Parameter.
	B9	Ungültiger Typ	DP-Slave oder Modul meldet einen ungültigen Typ: • Bei RDREC (Seite 357): Puffer zu klein (Teilmengen können nicht gelesen werden) • Bei WRREC (Seite 357): Puffer zu klein (Teilmengen können nicht geschrieben werden)
	BA bis BF	Benutzerspezifisch	DP-Slave oder Modul meldet herstellerspezifischen Fehler beim Zugriff. Prüfen Sie die Dokumentation des Herstellers des DP-Slaves oder des Moduls.
	C0	Einschränkungskonflikt beim Lesen	• Bei WRREC (Seite 357): Die Daten können nur geschrieben werden, wenn sich die CPU in der Betriebsart STOP befindet. **Hinweis**: Das bedeutet, dass die Daten nicht vom Anwenderprogramm geschrieben werden können. Sie können die Daten nur online mit einem PG/PC schreiben. • Bei RDREC (Seite 357): Das Modul leitet den Datensatz, doch es liegen entweder keine Daten vor oder die Daten können nur gelesen werden, wenn sich die CPU in der Betriebsart STOP befindet. **Hinweis**: Wenn nur in der Betriebsart STOP Daten geschrieben werden können, ist keine Auswertung durch das Anwenderprogramm möglich. Sie können die Daten dann nur online mit einem PG/PC lesen.
	C1	Einschränkungskonflikt beim Schreiben	Die Daten der vorherigen Schreibanforderung an das Modul für denselben Datensatz wurden vom Modul noch nicht verarbeitet.
	C2	Ressource ausgelastet	Das Modul verarbeitet derzeit die maximal mögliche Anzahl von Aufträgen für eine CPU.
	C3	Ressource nicht verfügbar	Die erforderlichen Betriebsressourcen sind derzeit ausgelastet.
	C4		Interner temporärer Fehler. Der Auftrag konnte nicht durchgeführt werden. Wiederholen Sie den Auftrag. Tritt dieser Fehler häufig auf, prüfen Sie Ihre Installation auf Quellen elektrischer Störung.
	C5		DP-Slave oder Modul nicht verfügbar
	C6		Datensatzübertragung wurde wegen Prioritätsklassen-einordnung abgebrochen.

Error_decode (B#16#....)	Error_code_1 (B#16#....)	Erläuterung (DVP1)	Beschreibung
	C7		Der Auftrag wurde wegen Warmstarts oder Kaltstarts des DP-Masters abgebrochen.
	C8 bis CF		DP-Slave oder Modul meldet herstellerspezifischen Ressourcenfehler. Prüfen Sie die Dokumentation des Herstellers des DP-Slaves oder des Moduls.
	Dx	Benutzerspezifisch	DP-Slave-spezifisch. Siehe Beschreibung des DP-Slave.
81	00 bis FF		Fehler im ersten Aufrufparameter (bei RALRM (Seite 360): MODE)
	00		Unzulässige Betriebsart
82	00 bis FF		Fehler im zweiten Aufrufparameter
88	00 bis FF		Fehler im achten Aufrufparameter (bei RALRM (Seite 360): TINFO) **Hinweis**: Das Online-Informationssystem von STEP 7 bietet unmittelbaren Zugriff auf Informationen dazu, wie die von TINFO zurückgegebenen Puffer auszuwerten sind.
	01		Falsche Syntax-ID
	23		Quantitätsstruktur überschritten oder Zielbereich zu klein
	24		Falsche Bereichs-ID
	32		DB/DI-Nummer außerhalb des Anwenderbereichs
	3A		DB/DI-Nummer ist NULL für die Bereichs-ID DB/DI oder der angegebene DB/DI existiert nicht.
89	00 bis FF		Fehler im neunten Aufrufparameter (bei RALRM (Seite 360): AINFO) **Hinweis**: Das Online-Informationssystem von STEP 7 bietet unmittelbaren Zugriff auf Informationen dazu, wie die von AINFO zurückgegebenen Puffer auszuwerten sind.
	01		Falsche Syntax-ID
	23		Quantitätsstruktur überschritten oder Zielbereich zu klein
	24		Falsche Bereichs-ID
	32		DB/DI-Nummer außerhalb des Anwenderbereichs
	3A		DB/DI-Nummer ist NULL für die Bereichs-ID DB/DI oder der angegebene DB/DI existiert nicht.
8A	00 bis FF		Fehler im zehnten Aufrufparameter
8F	00 bis FF		Fehler im fünfzehnten Aufrufparameter
FE, FF	00 bis FF		Profilspezifischer Fehler

Array-Element STATUS[4]

Bei DPV1-Fehlern übergibt der DP-Master STATUS[4] an die CPU und die Anweisung. Ohne DPV1-Fehler ist dieser Wert auf 0 gesetzt, mit den folgenden Ausnahmen bei RDREC:

- STATUS[4] enthält die Zielbereichslänge von RECORD, wenn MLEN > die Zielbereichslänge aus RECORD.
- STATUS[4] = MLEN, wenn die tatsächliche Länge des Datensatzes < MLEN < die Länge des Zielbereichs aus RECORD.
- STATUS[4] = 0, wenn STATUS[4] > 255; müsste gesetzt werden

Bei PROFINET IO hat STATUS[4] den Wert 0.

9.3.5 Anweisungen DPRD_DAT und DPWR_DAT (Konsistente Daten für DP-Slaves lesen/schreiben)

Die Anweisungen DPRD_DAT (Konsistente Daten lesen) und DPWR_DAT (Konsistente Daten schreiben) können Sie für PROFINET und PROFIBUS einsetzen.

Tabelle 9- 78 Anweisungen DPRD_DAT und DPWR_DAT

KOP/FUP	SCL	Beschreibung
	`ret_val := DPRD_DAT( laddr:=_word_in_, rec-ord=>_variant_out_);`	Mit der Anweisung DPRD_DAT lesen Sie ein Byte oder mehrere Bytes an Daten aus einem der folgenden Speicherorte: • Modul in der lokalen Basis • DP-Standard-Slave • PROFINET IO-Gerät Die CPU überträgt die gelesenen Daten konsistent. Tritt während der Datenübertragung kein Fehler auf, werden die gelesenen Daten von der CPU in den vom Parameter RECORD angegebenen Zielbereich eingegeben. Der Zielbereich muss dieselbe Länge haben wie die, die Sie für das ausgewählte Modul in STEP 7 konfiguriert haben. Wenn Sie die Anweisung DPRD_DAT ausführen, können Sie auf die Daten lediglich eines Moduls zugreifen. Die Übertragung beginnt an der konfigurierten Anfangsadresse.
	`ret_val := DPWR_DAT( laddr:=_word_in_, rec-ord:=_variant_in_);`	Verwenden Sie die Anweisung DPWR_DAT, um die Daten in RECORD konsistent in die folgenden Speicherorte zu übertragen: • Adressiertes Modul in der lokalen Basis • DP-Standard-Slave • PROFINET IO-Gerät Der Quellbereich muss dieselbe Länge haben wie die, die Sie für das ausgewählte Modul in STEP 7 konfiguriert haben.

- Die S7-1200 CPU unterstützt bis zu 64 Bytes konsistenter Daten auf dem lokalen Bus. Verwenden Sie die Anweisungen DPRD_DAT und DPWR_DAT , um auf mehr als 64 Bytes an Daten konsistent zuzugreifen.
- PROFIBUS unterstützt bis zu 4 Bytes konsistenter Daten. Verwenden Sie die Anweisungen DPRD_DAT und DPWR_DAT, um auf mehr als 4 Bytes an Daten konsistent zuzugreifen.
- PROFINET unterstützt bis zu 1472 Bytes konsistenter Daten. Für konsistente Übertragungen zwischen der S7-1200 und PROFINET-Geräten brauchen Sie diese Anweisungen nicht zu verwenden.
- Diese Anweisungen können Sie für Datenbereiche von 1 Byte oder mehr verwenden. Bei Verweigerung des Zugriffs wird der Fehlercode W#16#8090 gemeldet.

Hinweis

Wenn Sie die Anweisungen DPRD_DAT und DPWR_DAT mit konsistenten Daten verwenden, müssen Sie diese konsistenten Daten aus der automatischen Aktualisierung des Prozessabbilds herausnehmen. Weitere Informationen hierzu finden Sie unter "PLC-Grundlagen: Ausführung des Anwenderprogramms" (Seite 81).

Tabelle 9- 79 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
LADDR	IN	HW_IO (Word)	• Konfigurierte Startadresse im Bereich "E" des Moduls, aus dem die Daten gelesen werden (DPRD_DAT) • Konfigurierte Startadresse aus dem Prozessabbild der Ausgänge des Moduls, in das die Daten geschrieben werden (DPWR_DAT) Die Adressen müssen im Hexadezimalformat eingegeben werden (Beispiel: eine Eingangs- oder Ausgangsadresse von 100 bedeutet: LADDR:=W#16#64).
RECORD	OUT	Variant	Zielbereich der gelesenen Anwenderdaten (DPRD_DAT) oder Quellbereich der zu schreibenden Anwenderdaten (DPWR_DAT). Dieser Bereich muss genauso groß sein, wie Sie im ausgewählten Modul mit STEP 7 konfiguriert haben.
RET_VAL	OUT	Int	Tritt ein Fehler auf, während die Funktion aktiv ist, enthält der Rückgabewert einen Fehlercode.

Funktionsweise von DPRD_DAT

Der Zielbereich muss dieselbe Länge haben wie die, die für das ausgewählte Modul in STEP 7 konfiguriert wurde. Tritt während der Datenübertragung kein Fehler auf, werden die ausgelesenen Daten in den mit RECORD angegebenen Zielbereich eingegeben.

Wenn Sie aus einem DP-Normslave mit modularem Aufbau oder mit mehreren DP-Kennungen lesen, können Sie pro Aufruf der Anweisung DPRD_DAT nur auf die Daten eines Moduls bzw. einer DP-Kennung zugreifen, indem Sie die konfigurierte Anfangsadresse angeben.

Funktionsweise von DPWR_DAT

Sie übertragen die Daten in RECORD konsistent in den adressierten DP-Normslave bzw. das PROFINET IO-Gerät. Die Daten werden synchron übertragen, d.h. der Schreibvorgang wird zusammen mit der Anweisung beendet.

Der Quellbereich muss dieselbe Länge haben wie die, die Sie für das ausgewählte Modul in STEP 7 konfiguriert haben.

Wenn der DP-Normslave modular aufgebaut ist, können Sie nur auf ein Modul des DP-Slaves zugreifen.

Tabelle 9- 80 Fehlercodes von DPRD_DAT und DPWR_DAT

Fehlercode	Beschreibung
0000	Kein Fehler
8090	Es gilt einer der folgenden Fälle: • Sie haben für die angegebene logische Basisadresse kein Modul konfiguriert. • Sie haben die Einschränkung bezüglich der Länge konsistenter Daten ignoriert. • Sie haben die Anfangsadresse im Parameter LADDR nicht im Hexadezimalformat eingegeben.
8092	Der Parameter RECORD unterstützt die folgenden Datentypen: Byte, Char, Word, DWord, Int, UInt, USInt, SInt, DInt, UDInt, and arrays of these types.
8093	An der in LADDR angegebenen logischen Adresse ist kein DP-Modul/PROFINET IO-Gerät vorhanden, aus dem konsistente Daten gelesen (DPRD_DAT) oder in das konsistente Daten geschrieben (DPWR_DAT) werden können.
80A0	Beim Zugriff auf die E/A-Geräte wurde ein Zugriffsfehler erkannt (DPRD_DAT).
80A1	Beim Zugriff auf die E/A-Geräte wurde ein Zugriffsfehler erkannt (DPWR_DAT).
80B0	Slave-Ausfall auf dem externen DP-Schnittstellenmodul (DPRD_DAT) und (DPWR_DAT)
80B1	Die Länge des angegebenen Zielbereichs (DPRD_DAT) oder Quellbereichs (DPWR_DAT) ist nicht mit der in STEP 7 konfigurierten Länge der Anwenderdaten identisch.
80B2	Systemfehler mit externem DP-Schnittstellenmodul (DPRD_DAT) und (DPWR_DAT)
80B3	Systemfehler mit externem DP-Schnittstellenmodul (DPRD_DAT) und (DPWR_DAT)
80C0	Die Daten wurden noch nicht vom Modul gelesen (DPRD_DAT).
80C1	Die Daten des vorherigen Schreibauftrags für das Modul wurden vom Modul noch nicht verarbeitet (DPWR_DAT).
80C2	Systemfehler mit externem DP-Schnittstellenmodul (DPRD_DAT) und (DPWR_DAT)
80Fx	Systemfehler mit externem DP-Schnittstellenmodul (DPRD_DAT) und (DPWR_DAT)
85xy	Systemfehler mit externem DP-Schnittstellenmodul (DPWR_DAT)
87xy	Systemfehler mit externem DP-Schnittstellenmodul (DPRD_DAT)
808x	Systemfehler mit externem DP-Schnittstellenmodul (DPRD_DAT)
8xyy	Allgemeine Fehlerinformation Weitere Informationen zu allgemeinen Fehlercodes finden Sie unter "Gemeinsame Fehlercodes für die erweiterten Anweisungen" (Seite 464).

x = Parameternummer

y = Ereignisnummer

Hinweis

Wenn Sie auf DPV1-Slaves zugreifen, können Fehlerinformationen von diesen Slaves vom DP-Master an die Anweisung weitergeleitet werden.

9.3.6 Anweisung DPNRM_DG (Diagnosedaten eines DP-Slaves lesen)

Die Anweisung DPNRM_DG (Diagnosedaten lesen) können Sie für PROFIBUS einsetzen.

Tabelle 9- 81 Anweisung DPNRM_DG

KOP/FUP	SCL	Beschreibung

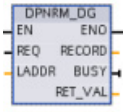
	`ret_val := DPNRM_DG(`
	`req:=_bool_in_,`
	`laddr:=_word_in_,`
	`record=>_variant_out_,`
	`busy=>_bool_out_);`
		Mit der Anweisung DPNRM_DG können Sie die aktuellen Diagnosedaten eines DP-Slaves in dem Format, das in EN 50 170 Band 2, PROFIBUS angegeben ist, lesen. Die ausgelesenen Daten werden nach einer fehlerfreien Datenübertragung in den von RECORD angegebenen Zielbereich eingegeben.

Tabelle 9- 82 Datentypen für die Parameter von DPNRM_DG

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	REQ=1: Leseanforderung
LADDR	IN	HW_DPSLAVE	Konfigurierte Diagnoseadresse des DP-Slaves: muss die Adresse der Station und nicht des E/A-Geräts sein. Um die Diagnoseadresse zu ermitteln, wählen Sie in der Netzsicht der "Gerätekonfiguration" die Station (und nicht das Abbild des Geräts). Geben Sie die Adressen im Hexadezimalformat ein. Beispiel: Diagnoseadresse 1022 bedeutet LADDR:=W#16#3FE.
RET_VAL	OUT	Int	Tritt ein Fehler auf, während die Funktion aktiv ist, enthält der Rückgabewert einen Fehlercode. Tritt kein Fehler auf, wird die Länge der tatsächlich übertragenen Daten in RET_VAL eingegeben.
RECORD	OUT	Variant	Zielbereich für die Diagnosedaten, die gelesen wurden. Die Mindestlänge des zu lesenden Datensatzes (bzw. der Zielbereich) beträgt 6 Byte. Die maximale Länge des zu sendenden Datensatzes beträgt 240 Byte. Normslaves können mehr als 240 Bytes an Diagnosedaten liefern, maximal 244 Bytes. Dann werden nur die ersten 240 Bytes in den Zielbereich übertragen, und das Überlaufbit wird in den Daten gesetzt.
BUSY	OUT	Bool	BUSY = 1: Der Leseauftrag ist noch nicht beendet

Sie starten den Leseauftrag, indem Sie dem Eingangsparameter REQ im Aufruf der Anweisung DPNRM_DG den Wert 1 zuweisen. Der Leseauftrag wird asynchron ausgeführt, in anderen Worten, es sind dafür mehrere Aufrufe der Anweisung DPNRM_DG erforderlich. Der Status des Auftrags wird von den Ausgangsparametern RET_VAL und BUSY angegeben.

Tabelle 9- 83 Struktur der Slave-Diagnosedaten

Byte	Beschreibung
0	Teilnehmerstatus 1
1	Teilnehmerstatus 2
2	Teilnehmerstatus 3
3	Master-Teilnehmernummer
4	Hersteller-ID (oberes Byte)
5	Hersteller-ID (unteres Byte)
6 ...	Weitere slave-spezifische Diagnoseinformationen

Tabelle 9- 84 Fehlercodes der Anweisung DPNRM_DG

Fehlercode	Beschreibung	Einschränkung
0000	Kein Fehler	-
7000	Erster Aufruf mit REQ = 0: Keine Datenübertragung aktiv; BUSY hat den Wert 0.	-
7001	Erster Aufruf mit REQ = 1: Keine Datenübertragung aktiv; BUSY hat den Wert 1.	Dezentrale E/A
7002	Zwischenzeitlicher Aufruf (REQ irrelevant): Datenübertragung ist bereits aktiv; BUSY hat den Wert 1.	Dezentrale E/A
8090	Angegebene logische Basisadresse ist ungültig: Keine Basisadresse vorhanden.	-
8092	Der Parameter RECORD unterstützt die folgenden Datentypen: Byte, Char, Word, DWord, Int, UInt, USInt, SInt, DInt, UDInt, and arrays of these types.	-
8093	• Diese Anweisung ist für das von LADDR angegebene Modul nicht zulässig (S7-DP-Module für die S7-1200 sind zulässig). • LADDR gibt statt der Station das E/A-Gerät an. Um die Diagnoseadresse für LADDR zu ermitteln, wählen Sie in der Netzsicht der "Gerätekonfiguration" die Station (und nicht das Abbild des Geräts).	-
80A2	• DP-Protokollfehler in Schicht 2 (Beispiel: Slave-Ausfall oder Busprobleme) • Bei ET200S kann der Datensatz nicht im DPV0-Modus gelesen werden.	Dezentrale E/A
80A3	DP-Protokollfehler mit User-Interface/User	Dezentrale E/A
80A4	Kommunikationsproblem auf dem Kommunikationsbus	Der Fehler tritt zwischen der CPU und dem externen DP-Schnittstellenmodul auf.

Fehlercode	Beschreibung	Einschränkung
80B0	• Die Anweisung ist für den Modultyp nicht zulässig. • Das Modul erkennt den Datensatz nicht. • Datensatznummer 241 ist nicht zulässig.	-
80B1	Die im Parameter RECORD angegebene Länge ist falsch.	Angegebene Länge > Datensatzlänge
80B2	Der konfigurierte Steckplatz ist nicht belegt.	-
80B3	Tatsächlicher Modultyp entspricht nicht dem erforderlichen Modultyp.	-
80C0	Keine Diagnoseinformationen vorhanden.	-
80C1	Die Daten des vorherigen Schreibauftrags für denselben Datensatz auf dem Modul wurden vom Modul noch nicht verarbeitet.	-
80C2	Das Modul verarbeitet derzeit die maximal mögliche Anzahl von Aufträgen für eine CPU.	-
80C3	Die erforderlichen Ressourcen (Speicher usw.) sind derzeit ausgelastet.	-
80C4	Interner temporärer Fehler. Der Auftrag konnte nicht verarbeitet werden. Wiederholen Sie den Auftrag. Tritt dieser Fehler häufig auf, prüfen Sie Ihr System auf Quellen elektrischer Störung.	-
80C5	Dezentrale E/A nicht verfügbar	Dezentrale E/A
80C6	Datensatzübertragung wurde wegen Prioritätsklasseneinordnung abgebrochen (Neustart oder Hintergrund)	Dezentrale E/A
8xyy[1]	Allgemeine Fehlercodes	

Weitere Informationen zu allgemeinen Fehlercodes finden Sie unter "Erweiterte Anweisungen, dezentrale E/A: Fehlerinformationen für RDREC, WRREC und RALRM" (Seite 364).

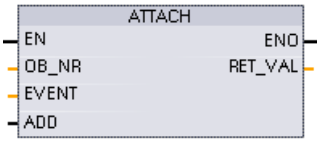
9.4 Alarme

9.4.1 Anweisungen ATTACH und DETACH (OB und Alarmereignis einander zuweisen/Zuweisung aufheben)

Mit den Anweisungen ATTACH und DETACH können Sie Unterprogramme, die durch Alarmereignisse angestoßen werden, aktivieren und deaktivieren.

Tabelle 9- 85 Anweisungen ATTACH und DETACH

KOP/FUP	SCL	Beschreibung

	`ret_val := ATTACH(`
	`ob_nr:=_int_in_,`
	
	`event:=_event_att_in_,`
	`add:=_bool_in_);`
		Die Anweisung ATTACH aktiviert die Ausführung des Unterprogramms eines Alarm-OBs bei einem Prozessereignis.

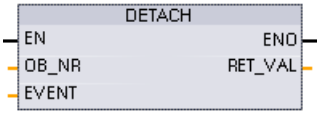
	`ret_val := DETACH(`
	`ob_nr:=_int_in_,`
	
	`event:=_event_att_`
	`in);`
		Die Anweisung DETACH deaktiviert die Ausführung des Unterprogramms eines Alarm-OBs bei einem Prozessereignis.

Tabelle 9- 86 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
OB_NR	IN	OB_ATT	Organisationsbausteinkennung: Wählen Sie einen der verfügbaren Prozessalarm-OBs aus, die über die Funktion "Neuen Baustein hinzufügen" angelegt wurden. Doppelklicken Sie auf das Parameterfeld und dann auf das Symbol, um die verfügbaren OBs anzuzeigen.
EVENT	IN	EVENT_ATT	Ereigniskennung: Wählen Sie eines der verfügbaren Prozessalarmereignisse aus, die in der PLC-Gerätekonfiguration für digitale Eingänge oder schnelle Zähler aktiviert wurden. Doppelklicken Sie auf das Parameterfeld und dann auf das Symbol, um die verfügbaren Ereignisse anzuzeigen.
ADD (nur ATTACH)	IN	Bool	• ADD = 0 (Standard): Dieses Ereignis ersetzt alle bisherigen Ereigniszuordnungen für diesen OB. • ADD = 1: Dieses Ereignis wird zu allen bisherigen Ereigniszuordnungen für diesen OB hinzugefügt.
RET_VAL	OUT	Int	Ausführungsbedingung

Prozessalarmereignisse

Die folgenden Prozessalarmereignisse werden von der CPU unterstützt:

- Ereignisse steigende Flanke: die ersten 12 integrierten Digitaleingänge der CPU (DEa.0 bis DEb.3) und alle SB-Digitaleingänge
  - Eine steigende Flanke tritt auf, wenn der digitale Eingang in Reaktion auf den Signalwechsel eines an den Eingang angeschlossenen Geräts von AUS nach EIN wechselt.
- Ereignisse fallende Flanke: die ersten 12 integrierten Digitaleingänge der CPU (DEa.0 bis DEb.3) und alle SB-Digitaleingänge
  - Eine fallende Flanke tritt auf, wenn der digitale Eingang von EIN nach AUS wechselt.
- Ereignisse Aktueller Wert des schnellen Zählers (HSC) = Referenzwert (CV = RV) (HSC 1 bis 6)
  - Ein Alarm CV = RV wird für einen HSC generiert, wenn der aktuelle Zählwert von einem benachbarten Wert zu dem Wert wechselt, der genau dem zuvor eingerichteten Referenzwert enspricht.
- Ereignisse HSC-Richtungswechsel (HSC 1 bis 6)
  - Ein Richtungswechselereignis tritt auf, wenn erkannt wird, dass der HSC von Erhöhen zu Verringern oder von Verringern zu Erhöhen wechselt.
- Ereignisse Externes Rücksetzen HSC (HSC 1 bis 6)
  - Bestimmte HSC-Betriebsarten gestatten die Zuweisung eines digitalen Eingangs für externes Rücksetzen, um den Zählwert des HSC auf Null zurückzusetzen. Ein Ereignis Externes Rücksetzen tritt für einen solchen HSC auf, wenn dieser Eingang von AUS nach EIN wechselt.

Aktivieren von Prozessalarmereignissen in der Gerätekonfiguration

Prozessalarme müssen während der Gerätekonfiguration aktiviert werden. Sie müssen das Kästchen zum Aktivieren der Ereignisse in der Gerätekonfiguration für einen digitalen Eingangskanal oder einen HSC selektieren, wenn Sie dieses Ereignis während der Konfiguration oder zur Laufzeit zuweisen möchten.

Mögliche Optionskästchen in der PLC-Gerätekonfiguration:

- Digitaleingang
  - Erkennung steigender Flanken aktivieren
  - Erkennung fallender Flanken aktivieren
- Schneller Zähler (HSC)
  - Diesen schnellen Zähler aktivieren
  - Alarm für Zählerwert gleich Referenzwert generieren
  - Alarm für Ereignis Externes Rücksetzen generieren
  - Alarm für Richtungswechselereignis generieren

Neue Prozessalarm-OBs in Ihr Programm einfügen

Standardmäßig wird bei der ersten Aktivierung des Ereignisses dem Ereignis kein OB zugeordnet. Dies wird durch die Kennung "<nicht verbunden>" in der Gerätekonfiguration "Prozessalarm:" gekennzeichnet. Nur Prozessalarm-OBs können einem Prozessalarmereignis zugeordnet werden. Alle vorhandenen Prozessalarm-OBs werden in der Klappliste "Prozessalarm:" aufgeführt. Wird kein OB aufgeführt, müssen Sie einen OB vom Typ "Prozessalarm" wie folgt anlegen. Gehen Sie in der Projektnavigation unter "Programmbausteine" folgendermaßen vor:

1. Doppelklicken Sie auf "Neuen Baustein hinzufügen", wählen Sie "Organisationsbaustein (OB)" und dann "Prozessalarm".
2. Optional können Sie den OB umbenennen, die Programmiersprache (KOP, FUP oder SCL) festlegen und die Bausteinnummer auswählen (schalten Sie auf manuelle Bearbeitung um und wählen Sie eine andere als die vorgeschlagene Bausteinnummer aus).
3. Bearbeiten Sie den OB und fügen Sie die Reaktion des Programms ein, die bei Auftreten des Ereignisses ausgeführt werden soll. Sie können bis zu einer Schachtelungstiefe von sechs Bausteinen FCs und FBs aus diesem OB aufrufen.

Parameter OB_NR

Alle vorhandenen Prozessalarm-OBs werden in der Klappliste "Prozessalarm:" in der Gerätekonfiguration und in der Klappliste OB_NR der Parameter ATTACH / DETACH aufgeführt.

Parameter EVENT

Wenn ein Prozessalarmereignis aktiviert wird, wird diesem bestimmten Ereignis ein eindeutiger Standardereignisname zugewiesen. Diesen Ereignisnamen können Sie durch Bearbeiten im Eingabefeld "Ereignisname:" ändern, er muss jedoch eindeutig sein. Die Ereignisnamen werden in Variablennamen in der Variablentabelle "Konstanten" umgewandelt und erscheinen in der Klappliste des Parameters "EVENT" in den Anweisungsfeldern ATTACH und DETACH. Der Wert dieser Variablen ist eine interne Nummer, die das Ereignis kennzeichnet.

Allgemeine Funktionsweise

Jeder Prozessalarm kann einem Prozessalarm-OB zugeordnet werden, der bei Auftreten des Prozessalarmereignisses in die Warteschlange für die Bearbeitung gestellt wird. Die Zuordnung von OB und Ereignis kann während der Konfiguration oder zur Laufzeit erfolgen.

Sie haben die Möglichkeit, zur Zeit der Konfiguration einem aktivierten Ereignis einen OB zuzuordnen bzw. die Zuordnung aufzuheben. Um während der Konfiguration einem Ereignis einen OB zuzuordnen, wählen Sie in der Klappliste der verfügbaren Prozessalarm-OBs unter "Prozessalarm:" einen OB aus (klicken Sie auf den Abwärtspfeil rechts an der Liste). Wählen Sie in dieser Liste den Namen des gewünschten OBs aus, oder wählen Sie "<nicht verbunden>", um die Zuordnung aufzuheben.

Sie können ein aktiviertes Prozessalarmereignis auch während der Laufzeit zuordnen bzw. die Zuordnung aufheben. Mit den Anweisungen ATTACH und DETACH können Sie während der Laufzeit (bei Bedarf mehrmals) ein aktiviertes Prozessalarmereignis dem entsprechenden OB zuordnen bzw. die Zuordnung aufheben. Wenn derzeit kein OB zugeordnet ist (entweder aufgrund der Auswahl "<nicht verbunden>" in der Gerätekonfiguration oder als Ergebnis der Ausführung von DETACH), wird der aktivierte Prozessalarm ignoriert.

Funktionsweise von DETACH

Mit der Anweisung DETACH heben Sie die Zuordnung eines bestimmten Ereignisses oder aller Ereignisse zu einem bestimmten OB auf. Ist ein EVENT angegeben, wird nur die Zuordnung dieses einen Ereignisses zum angegebenen Organisationsbaustein OB_NR aufgehoben; alle anderen Ereignisse, die diesem Organisationsbaustein OB_NR zugeordnet sind, bleiben davon unbeeinflusst. Ist kein EVENT angegeben, werden alle derzeit zu diesem Organisationsbaustein OB_NR zugeordneten Ereignisse getrennt.

Bedingungscodes

Tabelle 9- 87 Bedingungscodes

RET_VAL (W#16#....)	ENO	Beschreibung
0000	1	Kein Fehler
0001	1	Keine Zuordnung vorhanden (nur DETACH)
8090	0	OB ist nicht vorhanden
8091	0	OB ist vom falschen Typ
8093	0	Ereignis ist nicht vorhanden

9.4.2 Weckalarme

9.4.2.1 Anweisung SET_CINT (Weckalarm parametrieren)

Tabelle 9- 88 SET_CINT (Weckalarm parametrieren)

KOP/FUP	SCL	Beschreibung

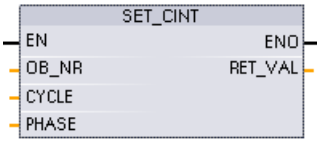
	`ret_val := SET_CINT(`
	`ob_nr:=_int_in_,`
	`cycle:=_udint_in_,`
	`phase:=_udint_in_);`
		Der angegebene Alarm-OB wird gesetzt, um die zyklische Ausführung zu starten, die den Programmzyklus unterbricht.

Tabelle 9- 89 Datentypen für die Parameter

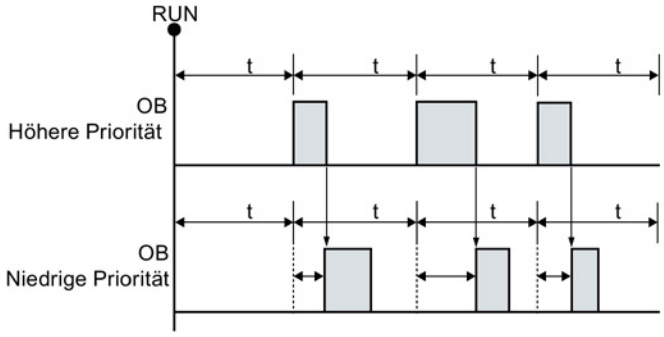
Parameter und Datentyp		Datentyp	Beschreibung
OB_NR	IN	OB_CYCLIC	OB-Nummer (symbolisch adressierbar)
CYCLE	IN	UDInt	Zeitintervall in Mikrosekunden
PHASE	IN	UDInt	Phasenverschiebung in Mikrosekunden
RET_VAL	OUT	Int	Ausführungsbedingung

Beispiele: Zeitparameter

- Ist die CYCLE-Zeit = 100 us, dann unterbricht der von OB_NR angegebene Alarm-OB den Programmzyklus alle 100 us. Der Alarm-OB wird ausgeführt und übergibt anschließend die Ausführungssteuerung wieder an den Programmzyklus an der Stelle, an der die Unterbrechung auftrat.
- Ist die CYCLE-Zeit = 0, dann wird das Alarmereignis deaktiviert und der Alarm-OB wird nicht ausgeführt.
- Die PHASE-Zeit (Phasenverschiebung) ist eine angegebene Verzögerungszeit, die auftritt, bevor das CYCLE-Zeitintervall beginnt. Mit der Phasenverschiebung können Sie die Ausführung von OBs mit niedrigerer Priorität zeitlich steuern.

Wenn OBs niedrigerer und höherer Priorität in demselben Zeitintervall aufgerufen werden, wird der OB niedrigerer Priorität erst aufgerufen, wenn die Bearbeitung des OBs höherer Priorität beendet ist. Der Ausführungsbeginn des OBs niedrigerer Priorität kann sich je nach Verarbeitungszeit des OBs höherer Priorität verschieben.

OB-Aufruf ohne Phasenverschiebung



Wenn Sie die Ausführung eines OBs niedrigerer Priorität mit fester Zykluszeit starten möchten, dann muss die Phasenverschiebung größer sein als die Verarbeitungszeit des OBs höherer Priorität.

OB-Aufruf mit Phasenverschiebung

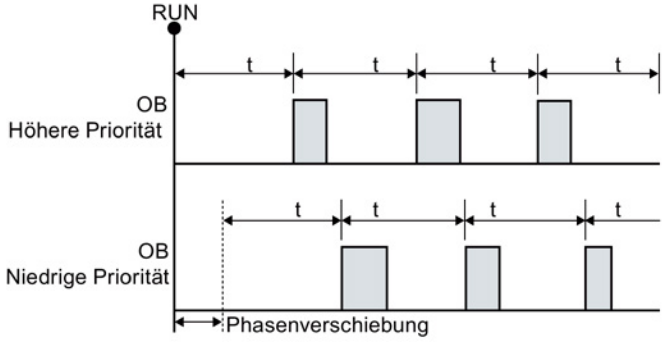


Tabelle 9- 90 Bedingungscodes

RET_VAL (W#16#....)	Beschreibung
0000	Kein Fehler
8090	OB ist nicht vorhanden oder hat den falschen Typ
8091	Ungültige Zykluszeit
8092	Ungültige Phasenverschiebungszeit
80B2	OB hat kein zugehöriges Ereignis

9.4.2.2 Anweisung QRY_CINT (Weckalarmparameter abfragen)

Tabelle 9- 91 **QRY_CINT (Weckalarm abfragen)**

KOP/FUP	SCL	Beschreibung

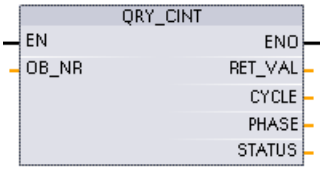
	`ret_val := QRY_CINT(`
	`ob_nr:=_int_in_,`
	`cycle=>_udint_out_,`
	`phase=>_udint_out__,`
	`status=>_word_out_);`
		Parameter und Ausführungszustand eines Weckalarm-OBs werden abgerufen. Die zurückgegebenen Werte existierten zur Zeit der Ausführung von QRY_CINT.

Tabelle 9- 92 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
OB_NR	IN	OB_CYCLIC	OB-Nummer (symbolisch adressierbar, z. B. OB_MyOBName)
RET_VAL	OUT	Int	Ausführungsbedingung
CYCLE	OUT	UDInt	Zeitintervall in Mikrosekunden
PHASE	OUT	UDInt	Phasenverschiebung in Mikrosekunden
STATUS	OUT	Word	Statuscode des Weckalarms: • Bits 0 bis 4, siehe STATUS-Tabelle unten • Sonstige Bits, immer 0

Tabelle 9- 93 Parameter STATUS

Bit	Wert	Beschreibung
0	0	Während CPU RUN
	1	Beim Anlauf
1	0	Der Alarm ist aktiviert.
	1	Der Alarm wird über die Anweisung DIS_IRT deaktiviert.
2	0	Der Alarm ist nicht aktiv oder abgelaufen.
	1	Der Alarm ist aktiv.
4	0	Der von OB_NR angegebene OB existiert nicht.
	1	Der von OB_NR angegebene OB existiert.
Sonstige Bits		Immer 0

Tritt ein Fehler auf, zeigt RET_VAL den entsprechenden Fehlercode und den Parameter STATUS = 0 an.

Tabelle 9- 94 Parameter RET_VAL

RET_VAL (W#16#....)	Beschreibung
0000	Kein Fehler
8090	OB ist nicht vorhanden oder hat den falschen Typ.
80B2	OB hat kein zugehöriges Ereignis.

9.4.3 Uhrzeitalarme

⚠ WARNUNG

Wenn ein Angreifer über die NTP-Synchronisierung (Network Time Protocol) auf Ihre Netzwerke zugreifen kann, kann er möglicherweise durch Veränderung der CPU-Systemzeit in gewissem Rahmen Ihren Prozess kontrollieren.

Die NTP-Client-Funktion der S7-1200 CPU ist standardmäßig deaktiviert und ermöglicht im aktivierten Zustand lediglich konfigurierten IP-Adressen, als NTP-Server zu fungieren. Die CPU deaktiviert diese Funktion standardmäßig, und Sie müssen die Funktion konfigurieren, um ferngesteuerte Korrekturen der CPU-Systemzeit zu gestatten.

Die S7-1200 CPU unterstützt Uhrzeitalarme und Uhrzeitoperationen, die von einer korrekten CPU-Systemzeit abhängig sind. Wenn Sie NTP konfigurieren und die Uhrzeitsynchronisierung von einem Server akzeptieren, müssen Sie sicherstellen, dass es sich bei dem Server um eine vertrauenswürdige Quelle handelt. Ist dies nicht der Fall, kann es zu einer Sicherheitsverletzung kommen, bei der ein unbekannter Benutzer die CPU-Systemzeit verändern und dadurch in gewissem Rahmen Ihren Prozess kontrollieren kann.

Informationen und Empfehlungen bezüglich der Sicherheit finden Sie in unseren Operational Guidelines für Industrial Security (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf) auf der Website "Service & Support" von Siemens.

9.4.3.1 SET_TINTL (Uhrzeitalarm festlegen)

Tabelle 9- 95 SET_TINTL (Datum und Uhrzeit für Uhrzeitalarm mit Datentyp DTL festlegen)

KOP/FUP	SCL	Beschreibung

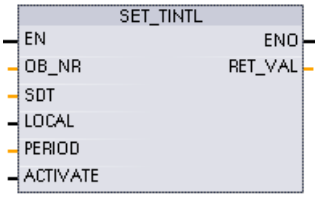
	`ret_val := SET_TINTL(`
	`OB_NR:=_int_in_,`
	`SDT:=_dtl_in_,`
	`LOCAL:=_bool_in_`
	`PERIOD:=_word_in_`
	`ACTIVATE:=_bool_in_);`
		Festlegen eines Uhrzeitalarms. Der OB kann für eine Ausführung oder für wiederholte Ausführungen in einem zugewiesenen Zeitraum festgelegt werden.

Tabelle 9- 96 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
OB_NR	IN	OB_TOD (INT)	OB-Nummer (symbolisch adressierbar)
SDT	IN	DTL	Anfangsdatum und -uhrzeit: Sekunden und Millisekunden werden ignoriert und können auf 0 gesetzt werden.
LOCAL	IN	Bool	0 = Systemzeit verwenden 1 = Ortszeit verwenden (sofern für die CPU die Ortszeit eingestellt ist, ansonsten verwenden Sie die Systemzeit)
PERIOD	IN	Word	Der Zeitraum ab Anfangsdatum und -uhrzeit für das wiederholt auftretende Alarmereignis. • W#16#0000 = einmalig • W#16#0201 = jede Minute • W#16#0401 = jede Stunde • W#16#1001 = täglich • W#16#1201 = wöchentlich • W#16#1401 = monatlich • W#16#1801 = jährlich • W#16#2001 = Monatsende
ACTIVATE	IN	Bool	0 = ACT_TINT muss ausgeführt werden, um das Alarmereignis zu aktivieren. 1 = Das Alarmereignis ist aktiviert.
RET_VAL	OUT	Int	Ausführungsbedingung

Ihr Programm kann SET_TINTL für die Einstellung eines Datums- und Uhrzeitalarmereignisses nutzen, das den zugewiesenen Alarm-OB ausführt. Mit Parameter SDT werden die Werte für Startdatum und -uhrzeit eingestellt, der Zeitraum für wiederkehrende Alarme (zum Beispiel täglich oder wöchentlich) wird mit Parameter PERIOD eingestellt. Wird für wiederkehrende Alarme "monatlich" eingestellt, muss als Startdatum ein Tag von 1 bis 28 eingegeben werden. Die Tage 29 bis 31 sind nicht zulässig, weil sie z. B. im Februar nicht vorkommen. Wenn Sie ein Alarmereignis am Monatsende programmieren möchten, dann verwenden Sie für den Parameter PERIOD die Monatsende-Einstellung.

Der Wochentagswert der DTL-Daten im Parameter SDT wird ignoriert. Das aktuelle Datum und die aktuelle Uhrzeit einer CPU stellen Sie über die Funktion "Uhrzeit stellen" in der Ansicht "Online & Diagnose" einer Online-CPU ein. Sie müssen Tag, Monat und Jahr einstellen. STEP 7 berechnet den Alarmzeitraum basierend auf Datum und Uhrzeit der CPU.

Hinweis

Bei der Umstellung von Sommer- nach Winterzeit ist die erste Stunde des Tages nicht vorhanden. Verwenden Sie deshalb eine Anfangszeit in der zweiten Stunde oder verwenden Sie einen zusätzlichen Verzögerungsalarm innerhalb der ersten Stunde.

Tabelle 9- 97 Bedingungscode

RET_VAL (W#16#....)	Beschreibung
0000	Kein Fehler
8090	Ungültiger Parameter OB_NR
8091	Ungültiger Parameter der SDT-Anfangszeit: (beispielsweise eine Anfangszeit in der übersprungenen Stunde zu Beginn der Sommerzeit)
8092	Ungültiger PERIOD-Parameter
80A1	Die Anfangszeit liegt in der Vergangenheit. (Dieser Fehlercode tritt nur bei PERIOD = W #16#0000 auf.)

9.4.3.2 CAN_TINT (Uhrzeitalarm löschen)

Tabelle 9- 98 CAN_TINT (Uhrzeitalarm löschen)

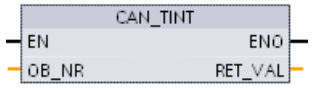
KOP/FUP	SCL	Beschreibung
	`ret_val:=CAN_TINT(_int_in);`	Löscht das Uhrzeitalarmereignis für den angegebenen Alarm-OB.

Tabelle 9- 99 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
OB_NR	IN	OB_TOD (INT)	OB-Nummer (symbolisch adressierbar)
RET_VAL	OUT	Int	Ausführungsbedingung

Tabelle 9- 100 Bedingungscodes

RET_VAL (W#16#....)	Beschreibung
0000	Kein Fehler
8090	Ungültiger Parameter OB_NR
80A0	Kein(e) Startdatum/-uhrzeit für diesen Alarm-OB eingestellt

9.4.3.3 ACT_TINT (Uhrzeitalarm aktivieren)

Tabelle 9- 101 ACT_TINT (Uhrzeitalarm aktivieren)

KOP/FUP	SCL	Beschreibung
	`ret_val:=ACT_TINT(_int_in_);`	Aktiviert das Uhrzeitalarmereignis für den angegebenen Alarm-OB.

Tabelle 9- 102 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
OB_NR	IN	OB_TOD (INT)	OB-Nummer (symbolisch adressierbar)
RET_VAL	OUT	Int	Ausführungsbedingung

Tabelle 9- 103 Bedingungscodes

RET_VAL (W#16#....)	Beschreibung
0000	Kein Fehler
8090	Ungültiger Parameter OB_NR
80A0	Startdatum und -uhrzeit für den Uhrzeitalarm-OB nicht eingestellt
80A1	Die aktivierte Zeit liegt in der Vergangenheit. Der Fehler tritt nur auf, wenn der Alarm-OB nur für die einmalige Ausführung eingestellt ist.

9.4.3.4 QRY_TINT (Status des Uhrzeitalarms abfragen)

Tabelle 9- 104 QRY_TINT (Uhrzeitalarm abfragen)

KOP/FUP	SCL	Beschreibung

	`ret_val:=QRY_TINT(`
	`  OB_NR:=_int_in_,`
	`  STATUS=>_word_out_);`
		Fragt den Uhrzeitalarmstatus für den angegebenen Alarm-OB ab.

Tabelle 9- 105 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
OB_NR	IN	OB_TOD (INT)	OB-Nummer (symbolisch adressierbar) des Alarm-OBs für die Abfrage
RET_VAL	OUT	Int	Ausführungsbedingung
STATUS	OUT	Word	Status des angegebenen Alarm-OBs

Tabelle 9- 106 Parameter STATUS

Bit	Wert	Beschreibung
0	0	In Run
	1	Beim Anlauf
1	0	Der Alarm ist aktiviert.
	1	Der Alarm ist deaktiviert.
2	0	Der Alarm ist nicht aktiv oder abgelaufen.
	1	Der Alarm ist aktiv.
4	0	Die zugewiesene OB_NR ist nicht vorhanden.
	1	Ein OB mit der zugewiesenen OB_NR ist vorhanden.
6	1	Uhrzeitalarm verwendet Ortszeit.
	0	Uhrzeitalarm verwendet Systemzeit.
Sonstiges		Immer 0

Tabelle 9- 107 Bedingungscode

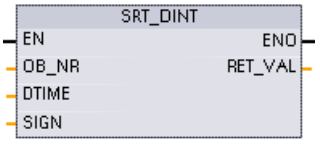
RET_VAL (W#16#....)	Beschreibung
0000	Kein Fehler
8090	Ungültiger Parameter OB_NR

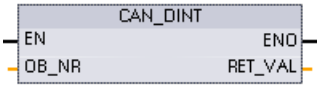
9.4.4 Verzögerungsalarme

Sie können die Bearbeitung des Verzögerungsalarms mit den Anweisungen SRT_DINT und CAN_DINT starten und abbrechen und den Alarmzustand mit der Anweisung QRY_DINT abfragen. Jeder Verzögerungsalarm ist ein einmaliges Ereignis, das nach einer vorgegebenen Verzögerungszeit auftritt. Wird das Zeitverzögerungsereignis gelöscht, bevor die Zeitverzögerung abgelaufen ist, tritt der Alarm nicht im Programm auf.

Tabelle 9- 108 Anweisungen SRT_DINT, CAN_DINT und QRY_DINT

KOP/FUP	SCL	Beschreibung

	`ret_val := SRT_DINT(`
	`ob_nr:=_int_in_,`
	`dtime:=_time_in_,`
	`sign:=_word_in_);`
		SRT_DINT startet einen Verzögerungsalarm, der einen OB ausführt, wenn die im Parameter DTIME festgelegte Verzögerungszeit abgelaufen ist.

	`ret_val := CAN_DINT(` `ob_nr:=_int_in_);`	CAN_DINT löscht einen Verzögerungsalarm, der bereits gestartet ist. Der Verzögerungsalarm-OB wird dann nicht ausgeführt.

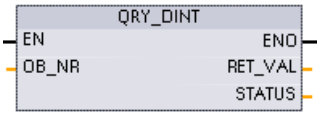
	`ret_val := QRY_DINT(`
	`ob_nr:=_int_in_,`
	`sta-`
	`tus=>_word_out_);`
		QRY_DINT fragt den Zustand des vom Parameter OB_NR angegebenen Verzögerungsalarms ab.

Tabelle 9- 109 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
OB_NR	IN	OB_DELAY	Organisationsbaustein (OB), der nach einer Zeitverzögerung gestartet werden soll: Wählen Sie einen der verfügbaren Verzögerungsalarm-OBs aus, die über die Funktion "Neuen Baustein hinzufügen" in der Projektnavigation angelegt wurden. Doppelklicken Sie auf das Parameterfeld und dann auf das Symbol, um die verfügbaren OBs anzuzeigen.
DTIME [1]	IN	Time	Verzögerungswert (1 bis 60000 ms)
SIGN [1]	IN	Word	Von der S7-1200 nicht verwendet: Jeder Wert wird angenommen. Um Fehler zu verhindern, muss ein Wert zugewiesen werden.
RET_VAL	OUT	Int	Ausführungsbedingung
STATUS	OUT	Word	Anweisung QRY_DINT: Zustand des angegebenen Verzögerungsalarm-OBs, siehe Tabelle unten

[1] Nur bei SRT_DINT

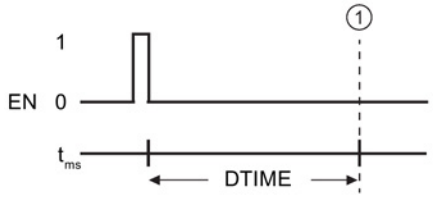
Bedienung

Ist EN=1, startet Anweisung SRT_DINT die interne Zeitverzögerung (DTIME). Wenn die angegebene Verzögerungszeit abgelaufen ist, generiert die CPU eine Programmunterbrechung, die die Ausführung des zugeordneten Verzögerungsalarm-OBs anstößt. Mit der Anweisung CAN_DINT können Sie einen gestarteten Verzögerungsalarm abbrechen, bevor die angegebene Verzögerungszeit erreicht ist. Ingesamt dürfen maximal vier Verzögerungsalarmereignisse aktiv sein.

Hinweis

SRT_DINT startet die Zeitverzögerung in jedem Zyklus, wenn EN=1 ist. Richten Sie EN=1 als Einmalimpuls ein, anstatt EN=1 für den Start der Zeitverzögerung einzustellen.

Zeitdiagramm für Anweisung SRT_DINT:



① Verzögerungsalarm wird ausgeführt

Verzögerungsalarm-OBs in Ihr Projekt einfügen

Sie können Verzögerungsalarm-OBs nur in den Anweisungen SRT_DINT und CAN_DINT hinzufügen. In einem neuen Projekt ist noch kein Verzögerungsalarm-OB vorhanden. Sie müssen Verzögerungsalarm-OBs in Ihr Projekt einfügen. Um einen Verzögerungsalarm-OB anzulegen, gehen Sie folgendermaßen vor:

1. Doppelklicken Sie in der Projektnavigation unter "Programmbausteine" auf "Neuen Baustein hinzufügen", wählen Sie "Organisationsbaustein (OB)" und dann "Verzögerungsalarm".
2. Sie können den OB umbenennen, die Programmiersprache festlegen und die Bausteinnummer auswählen. Schalten Sie auf manuelle Bearbeitung um, wenn Sie eine andere als die automatisch zugeordnete Bausteinnummer auswählen möchten.
3. Bearbeiten Sie das Unterprogramm des Verzögerungsalarm-OBs und fügen Sie die Reaktion des Programms ein, die bei Auftreten des Verzögerungsereignisses ausgeführt werden soll. Sie können bis zu einer Schachtelungstiefe von sechs Bausteinen andere FCs und FBs aus dem Verzögerungsalarm-OB aufrufen.
4. Die Namen der neu zugewiesenen Verzögerungsalarm-OBs stehen zur Verfügung, wenn Sie den Parameter OB_NR der Anweisungen SRT_DINT und CAN_DINT bearbeiten.

Parameter STATUS von QRY_DINT

Tabelle 9- 110 Liegt ein Fehler vor (REL_VAL <> 0), dann ist STATUS = 0.

Bit	Wert	Beschreibung
0	0	In RUN
	1	Beim Anlauf
1	0	Der Alarm ist aktiviert.
	1	Der Alarm ist deaktiviert.
2	0	Der Alarm ist nicht aktiv oder abgelaufen.
	1	Der Alarm ist aktiv.
4	0	Ein OB mit der in OB_NR angegebenen OB-Nummer existiert nicht.
	1	Ein OB mit der in OB_NR angegebenen OB-Nummer existiert.
Sonstige Bits		Immer 0

Bedingungscodes

Tabelle 9- 111 Bedingungscodes für SRT_DINT, CAN_DINT und QRY_DINT

RET_VAL (W#16#...)	Beschreibung
0000	Kein Fehler
8090	Falscher Parameter OB_NR
8091	Falscher Parameter DTIME
80A0	Verzögerungsalarm noch nicht gestartet.

9.4.5 Anweisungen DIS_AIRT und EN_AIRT (Ausführung von Alarmen höherer Priorität und asynchronen Fehlerereignissen verzögern/aktivieren)

Die Alarmbearbeitung kann mit den Anweisungen DIS_AIRT und EN_AIRT aktiviert und deaktiviert werden.

Tabelle 9- 112 Anweisungen DIS_AIRT und EN_AIRT

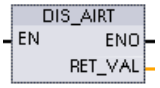
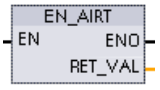
KOP/FUP	SCL	Beschreibung
	`DIS_AIRT();`	DIS_AIRT verzögert die Bearbeitung neuer Alarmereignisse. Sie können DIS_AIRT in einem OB mehrmals ausführen.
	`EN_AIRT();`	EN_AIRT aktiviert die Bearbeitung von Alarmereignissen, die zuvor mit der Anweisung DIS_AIRT deaktiviert wurden. Jede Ausführung von DIS_AIRT muss mit einer Ausführung von EN_AIRT gelöscht werden. Die Anweisungen EN_AIRT müssen in demselben OB oder in einer aus demselben OB aufgerufenen FC bzw. FB ausgeführt werden, bevor für diesen OB erneut Alarme aktiviert werden.

Tabelle 9- 113 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
RET_VAL	OUT	Int	Anzahl Verzögerungen = Anzahl ausgeführter DIS_AIRT in der Warteschlange.

Die Anzahl der Ausführungen von DIS_AIRT werden vom Betriebssystem gezählt. Jede dieser Anweisungen bleibt gültig, bis sie mit einer Anweisung EN_AIRT gelöscht wird oder bis der aktuelle OB vollständig abgearbeitet ist. Beispiel: Wenn Sie die Alarmbearbeitung fünfmal mit fünf Ausführungen von DIS_AIRT deaktiviert haben, müssen Sie diese mit fünf Ausführungen von EN_AIRT stornieren, damit die Alarmbearbeitung wieder aktiviert ist.

Nach der erneuten Aktivierung der Alarmereignisse werden die Alarme, die während der Ausführung von DIS_AIRT auftraten, bearbeitet bzw. die Alarme werden bearbeitet, sobald der aktuelle OB ausgeführt wurde.

Der Parameter RET_VAL zeigt an, wie häufig die Alarmverarbeitung deaktiviert wurde, also wie viele Ausführungen von DIS_AIRT in die Warteschlang gestellt wurden. Die Alarmverarbeitung wird erst wieder aktiviert, wenn der Parameter RET_VAL = 0 ist.

9.5 Diagnose (PROFINET oder PROFIBUS)

9.5.1 Diagnoseanweisungen

Die folgenden Diagnoseanweisungen können entweder mit PROFINET oder PROFIBUS verwendet werden:

- Anweisung LED (Seite 392): Sie können den Zustand der LEDS eines dezentralen Peripheriegeräts auslesen.
- Anweisung DeviceStates (Seite 393): Sie können die Betriebszustände dezentraler Peripheriegeräte in einem E/A-Untersystem abrufen.
- Anweisung ModuleStates (Seite 399): Sie können die Betriebszustände der Module in einem dezentralen Peripheriegerät abrufen.
- Anweisung GET_DIAG (Seite 405): Sie können die Diagnoseinformationen aus einem angegebenen Gerät auslesen.
- Anweisung Get_IM_Data (Seite 411): Sie können die Identifikations- und Wartungsdaten (I&M) für ein bestimmtes Modul oder Submodul prüfen.

9.5.2 Diagnoseereignisse für die dezentrale Peripherie

Hinweis

In einem PROFIBUS IO-System geht die CPU nach einem Ladevorgang oder dem Aus- und Wiedereinschalten in den Betriebszustand RUN, sofern die Hardwarekompatibilität nicht so eingestellt ist, dass akzeptable Ersatzmodule (Seite 168) zulässig sind und mindestens ein Modul fehlt oder es sich dabei nicht um ein akzeptables Ersatzmodul für das konfigurierte Modul handelt.

Wie Sie in der folgenden Tabelle sehen, unterstützt die CPU diejenige Diagnose, die für die Komponenten des dezentralen E/A-Systems konfiguriert werden kann. Jeder dieser Fehler erzeugt einen Protokolleintrag im Diagnosepuffer.

Tabelle 9- 114 Behandlung von Diagnoseereignissen bei PROFINET und PROFIBUS

Fehlertyp	Diagnoseinformation für die Station?	Eintrag im Diagnosepuffer?	CPU-Betriebszustand
Diagnosefehler	Ja	Ja	Bleibt im Betriebszustand RUN
Fehler bei Baugruppenträger oder Station	Ja	Ja	Bleibt im Betriebszustand RUN
Peripheriezugriffsfehler [1]	Nein	Ja	Bleibt im Betriebszustand RUN
Peripheriezugriffsfehler [2]	Nein	Ja	Bleibt im Betriebszustand RUN
Ziehen/Stecken-Ereignis	Ja	Ja	Bleibt im Betriebszustand RUN

[1] Beispielursache für einen Peripheriezugriffsfehler: Ein Modul wurde entfernt.

[2] Beispielursache für einen Peripheriezugriffsfehler: Azyklische Kommunikation mit einem Submodul, das nicht kommuniziert.

Über die Anweisung GET_DIAG (Seite 405) können Sie für jede Station die Diagnoseinformationen abrufen. Auf diese Weise können Sie die im Gerät aufgetretenen Fehler programmatisch behandeln und, sofern gewünscht, die CPU in den Betriebszustand STOP versetzen. Für dieses Verfahren müssen Sie das Hardwaregerät angeben, aus dem die Statusinformationen ausgelesen werden sollen.

Die Anweisung GET_DIAG verwendet die "L-Adresse" (LADDR) der Station, um den Zustand der gesamten Station abzurufen. Diese L-Adresse finden Sie in der Netzsicht der Gerätekonfiguration, indem Sie den gesamten Baugruppenträger der Station auswählen (den gesamten grauen Bereich). Die L-Adresse wird in der Registerkarte "Eigenschaften" der Station angezeigt. Sie finden den Parameter LADDR für jedes einzelne Modul entweder in den Eigenschaften des Moduls (in der Gerätekonfiguration) oder in der Standardvariablentabelle der CPU.

9.5.3 Anweisung LED (LED-Status lesen)

Tabelle 9- 115 Anweisung LED

KOP/FUP	SCL	Beschreibung

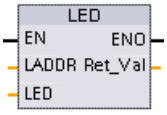
	`ret_val := LED(`
	`  laddr:=_word_in_,`
	`  LED:=_uint_in_);`
		Mit der Anweisung LED lesen Sie den Zustand der LEDs auf einer CPU oder Schnittstelle aus. Der angegebene LED-Zustand wird vom Ausgang RET_VAL zurückgegeben.

Tabelle 9- 116 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung		
LADDR	IN	HW_IO	Identifikationsnummer der CPU oder Schnittstelle[1]		
LED	IN	UInt	LED-Kennnummer		
			1	RUN/STOP	Farbe 1 = grün, Farbe 2 = gelb
			2	Fehler	Farbe 1 = rot
			3	Wartung	Farbe 1 = gelb
			4	Redundanz	-/-
			5	Verbindung	Farbe 1 = grün
			6	Tx/Rx	Farbe 1 = gelb
RET_VAL	OUT	Int	Zustand der LED		

[1] Sie können beispielsweise die CPU (wie "PLC_1") oder die PROFINET-Schnittstelle in der Klappliste des Parameters auswählen.

Tabelle 9- 117 Zustand von RET_VAL

RET_VAL (W#16#...)	Beschreibung	
LED-Zustand 0 bis 9	0	LED ist nicht vorhanden
	1	Aus
	2	Farbe 1 ein (leuchtet ständig)
	3	Farbe 2 ein (leuchtet ständig)
	4	Farbe 1 blinkt mit 2 Hz
	5	Farbe 2 blinkt mit 2 Hz
	6	Farben 1 & 2 blinken abwechselnd mit 2 Hz
	7	Farbe 1 ein (Tx/Rx)
	8	Farbe 2 ein (Tx/Rx)
	9	Zustand der LED ist nicht verfügbar
8091	Von LADDR angegebenes Gerät ist nicht vorhanden	
8092	Von LADDR angegebenes Gerät unterstützt keine LEDs	
8093	LED-Kennung nicht definiert	
80Bx	Von LADDR angegebene CPU unterstützt die Anweisung LED nicht	

9.5.4 Anweisung DeviceStates

Mit der Anweisung DeviceStates können Sie die Zustände aller dezentralen E/A-Slavegeräte, die an einen spezifischen dezentralen E/A-Master angeschlossen sind, ausgeben.

Tabelle 9- 118 Anweisung DeviceStates

KOP/FUP	SCL	Beschreibung

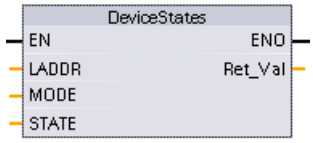
	`ret_val := DeviceStates(`
	`laddr:=_word_in_,`
	`mode:=_uint_in_,`
	`state:=_variant_inout_);`
		DeviceStates ruft die Betriebszustände von E/A-Geräten in einem E/A-Untersystem ab. Nach der Ausführung enthält der Parameter STATE den Fehlerzustand jedes einzelnen E/A-Geräts in einer Bitliste (für die zugewiesenen Parameter LADDR und MODE). Diese Informationen entsprechen dem Gerätestatus in der Diagnoseansicht von STEP 7.
		Der Eingang LADDR der Anweisung DeviceStates verwendet die Hardwarekennung einer dezentralen E/A-Schnittstelle. Im TIA Portal finden Sie die Hardwarekennungen für einen PLC über die Datentypen "Hw_IoSystem" im Register "Systemkonstanten" in der PLC-Variablentabelle.

Tabelle 9- 119 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
LADDR	IN	HW_IOSYSTEM	Logische Adresse: (Kennung des E/A-Systems)
MODE	IN	UInt	Unterstützt fünf Betriebsarten. Der Eingang MODE legt fest, welche Daten an der für STATE-Informationen angegebenen Stelle ausgegeben werden. Folgende Betriebsarten sind möglich: • 1: Gerätekonfiguration aktiv • 2: Gerät defekt • 3: Gerät deaktiviert • 4: Gerät vorhanden • 5: Problem im Gerät

Parameter und Datentyp		Datentyp	Beschreibung
RET_VAL	OUT	Int	Ausführungsbedingung
STATE[1]	InOut	Variant	Puffer, der den Fehlerzustand der einzelnen Geräte empfängt: Als Datentyp für den Parameter STATE können Sie einen beliebigen Bittyp (Bool, Byte, Word oder DWord) oder ein Array eines Bittyps auswählen. • Bit 0 des ersten Bytes der ausgegebenen STATE-Daten ist ein Zusammenfassungsbit. Ist es auf WAHR gesetzt, zeigt es an, dass andere Daten verfügbar sind. • Die vom Parameter STATE ausgegebenen Daten zeigen eine 1-zu-1-Übereinstimmung zwischen einer Bitposition und einer dezentralen E/A-Adresse. Diese Geräteadressierung ist WAHR für PROFIBUS und PROFINET. Beispiel: Bit 4 im ersten Byte stimmt mit PROFIBUS-Adresse 4 oder PROFINET-Gerätenummer 4 überein.

[1] Bei PROFIBUS DP beträgt die Länge der Zustandsinformationen 128 Bits. Bei PROFINET I/O beträgt die Länge 1024 Bit.

Nach der Ausführung enthält der Parameter STATE den Fehlerzustand jedes einzelnen E/A-Geräts als Bitliste (für die zugewiesenen LADDR und MODE).

Tabelle 9- 120 Bedingungscodes

RET_VAL (W#16#...)	Beschreibung
0	Kein Fehler
8091	LADDR ist nicht vorhanden.
8092	LADDR spricht kein E/A-System an.
8093	Ungültiger Datentyp für den Parameter STATE: Gültige Datentypen sind (Bool, Byte, Word oder Dword) oder ein Array aus (Bool, Byte, Word oder Dword)
80Bx	Die Anweisung DeviceStates wird von der CPU für diesen Parameter LADDR nicht unterstützt.
8452	Die vollständigen Zustandsdaten sind für den zugewiesenen Parameter STATE zu groß. Der Puffer STATE enthält ein Teilergebnis.

9.5.4.1 Beispiele für die Konfiguration von DeviceStates

Beispiel in PROFIBUS

Das Beispiel in PROFIBUS besteht aus den folgenden Komponenten:

- 16 PROFIBUS-Geräte mit den Namen "DPSlave_10" bis "DPSlave_25"
- Die 16 PROFIBUS-Geräte verwenden jeweils eine der PROFIBUS-Adressen 10 bis 25.
- Für jedes Slavegerät sind mehrere E/A-Module konfiguriert.
- Es werden die ersten vier Bytes der ausgegebenen STATE-Parameterinformationen angezeigt.

MODE	Beispiel 1: Normalbetrieb ohne Fehler	Beispiel 2: PROFIBUS Slavegerät DPSlave_12 mit einem gezogenen Modul	Beispiel 3: PROFIBUS Slavegerät DPSlave_12 getrennt
1: Gerätekonfiguration aktiv	0x01FC_FF03	0x01FC_FF03	0x01FC_FF03
2: Gerät defekt	0x0000_0000	0x0110_0000	0x0110_0000
3: Gerät deaktiviert	0x0000_0000	0x0000_0000	0x0000_0000
4: Gerät vorhanden	0x01FC_FF03	0x01FC_FF03	0x01EC_FF03
5: Problem im Gerät	0x0000_0000	0x0110_0000	0x0110_0000

Die folgenden vier Tabellen zeigen eine binäre Aufschlüsselung der vier analysierten Datenbytes:

Tabelle 9- 121 Beispiel 1: Keine Fehler: Für MODE 1 (Gerätekonfiguration aktiv) wird der Wert 0x01FC_FF03 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x01	Bit 7 0000-0001 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0xFC	Bit 15 1111-1100 Bit 8	
Byte 3 0xFF	Bit 23 1111-1111 Bit 16	
Byte 4 0x03	Bit 31 0000-0011 Bit 24	

Die Geräte sind an den Adressen 10 (Bit 10) bis 25 (Bit 25) konfiguriert.
An den Adressen 1 bis 9 sind keine Geräte konfiguriert.
MODE Daten 4 (Gerät vorhanden) stimmen mit MODE 1 (Gerätekonfiguration aktiv) überein, also entsprechen die konfigurierten Geräte den vorhandenen Geräten.

Tabelle 9- 122 Beispiel 2: Ein Modul wurde aus dem PROFIBUS-Slavegerät "DPSlave_12" gezogen. Für MODE 2 (Gerät defekt) wird der Wert 0x0110_0000 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x01	Bit 7 0000-0001 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0x10	Bit 15 000**1**-0000 Bit 8	
Byte 3 0x00	Bit 23 0000-0000 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Gerät 12 (Bit 12) ist als defekt gekennzeichnet.

MODE 5 (Problem im Gerät) gibt die gleichen Informationen wie MODE 2 (Gerät defekt) aus.

Tabelle 9- 123 Beispiel 2 (Fortsetzung): Ein Modul wurde aus dem PROFIBUS-Slavegerät "DPSlave_12" gezogen. Für MODE 4 (Gerät vorhanden) wird der Wert 0x01FC_FF03 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x01	Bit 7 0000-0001 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0xFC	Bit 15 111**1**-1100 Bit 8	
Byte 3 0xFF	Bit 23 1111-1111 Bit 16	
Byte 4 0x03	Bit 31 0000-0011 Bit 24	

Obwohl Gerät 12 (Bit 12) wie in MODE 2 (Gerät defekt) oben gezeigt einen Fehler aufweist, funktioniert das Gerät trotzdem noch im Netzwerk, weshalb MODE 4 (Gerät vorhanden) das Gerät als "vorhandenes Gerät" anzeigt.

Tabelle 9- 124 Beispiel 3: PROFIBUS-Slavegerät "DPSlave_12" ist vom PROFIBUS-Netzwerk getrennt (Kabel getrennt oder Spannungsverlust). "DPSlave_12" wird weiterhin als defektes Gerät sowie als Gerät mit Fehler erkannt. Der Unterschied ist, dass "DPSlave_12" nicht mehr als vorhandenes Gerät erkannt wird. Für MODE 4 (Gerät vorhanden) wird der Wert 0x01EC_FF03 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x01	Bit 7 0000-0001 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0xEC	Bit 15 111**0**-1100 Bit 8	
Byte 3 0xFF	Bit 23 1111-1111 Bit 16	
Byte 4 0x03	Bit 31 0000-0011 Bit 24	

Gerät 12 (Bit 12) ist als nicht vorhanden gekennzeichnet. Mit dieser Ausnahme werden die übrigen Geräte 10 bis 25 weiterhin als vorhanden gemeldet.

Beispiel in PROFINET

Das Beispiel in PROFINET besteht aus den folgenden Komponenten:

- 16 PROFINET-Slavegeräte mit den Namen "et200s_1" bis "et200s_16".
- Die 16 PROFINET-Geräte verwenden jeweils eine der PROFINET-Gerätenummern 1 bis 16.
- Für jedes Slavegerät sind mehrere E/A-Module konfiguriert.
- Es werden die ersten vier Bytes der ausgegebenen STATE-Parameterinformationen angezeigt.

MODE	Beispiel 1: Normalbetrieb ohne Fehler	Beispiel 2: PROFINET Gezogenes Modul an Slave et200s_1	Beispiel 3: PROFINET Slave et200s_1 getrennt
1: Gerätekonfiguration aktiv	0xFFFF_0100	0xFFFF_0100	0xFFFF_0100
2 - Gerät defekt	0x0000_0000	0x0300_0000	0x0300_0000
3 - Gerät deaktiviert	0x0000_0000	0x0000_0000	0x0000_0000
4 - Gerät vorhanden	0xFFFF_0100	0xFFFF_0100	0xFDFF_0100
5 - Problem im Gerät	0x0000_0000	0x0300_0000	0x0300_0000

Die folgenden vier Tabellen zeigen eine binäre Aufschlüsselung der vier analysierten Datenbytes:

Tabelle 9- 125 Beispiel 1: Keine Fehler: Für MODE 1 (Gerätekonfiguration aktiv) wird der Wert 0xFFFF_0100 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0xFF	Bit 7 1111-1111 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0xFF	Bit 15 1111-1111 Bit 8	
Byte 3 0x01	Bit 23 0000-0001 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Die Geräte sind an den Adressen 1 (Bit 1) bis 16 (Bit 16) konfiguriert.
An den Adressen 1 bis 9 sind keine Geräte konfiguriert.
MODE Daten 4 (Gerät vorhanden) stimmen mit MODE 1 (Gerätekonfiguration aktiv) überein, also entsprechen die konfigurierten Geräte den vorhandenen Geräten.

Tabelle 9- 126 Beispiel 2: Ein Modul wurde aus dem PROFINET-Slavegerät "et200s_1" gezogen. Für MODE 2 (Gerät defekt) wird der Wert 0x0300_0000 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x03	Bit 7 0000-0011 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0x00	Bit 15 0000-0000 Bit 8	
Byte 3 0x00	Bit 23 0000-0000 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Gerät 1 (Bit 1) ist als defekt gekennzeichnet. Da das Gerät weiterhin vorhanden ist, zeigt MODE 4 (Gerät vorhanden) die gleichen Daten wie im Normalbetrieb an.
MODE 5 (Problem im Gerät) gibt die gleichen Informationen wie MODE 2 (Gerät defekt) aus.

Tabelle 9- 127 Beispiel 2 (Fortsetzung): Ein Modul wurde aus dem PROFIBUS-Slavegerät "et200s_1" gezogen. Für MODE 4 (Gerät vorhanden) wird der Wert 0xFFFF_0100 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0xFF	Bit 7 1111-1111 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0xFF	Bit 15 1111-1111 Bit 8	
Byte 3 0x01	Bit 23 0000-0001 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Obwohl Gerät 1 (Bit 1) wie in MODE 2 (Gerät defekt) oben gezeigt einen Fehler aufweist, funktioniert das Gerät trotzdem noch im Netzwerk, weshalb MODE 4 (Gerät vorhanden) das Gerät als "vorhandenes Gerät" anzeigt.

Tabelle 9- 128 Beispiel 3: PROFINET-Slavegerät "et200s_1" ist vom PROFINET-Netzwerk getrennt (Kabel getrennt oder Spannungsverlust). Für MODE 4 (Gerät vorhanden) wird der Wert 0xFDFF_0100 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0xFD	Bit 7 1111-1101 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0xFF	Bit 15 1111-1111 Bit 8	
Byte 3 0x01	Bit 23 0000-0001 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Gerät 1 (Bit 1) ist nicht vorhanden. Geräte 2 (Bit 2) bis 16 (Bit 16) sind vorhanden.

9.5.5 Anweisung ModuleStates

Mit der Anweisung ModuleStates können Sie den Zustand aller Module in einer PROFIBUS- oder PROFINET-Station ausgeben.

Tabelle 9- 129 Anweisung ModuleStates

KOP/FUP	SCL	Beschreibung

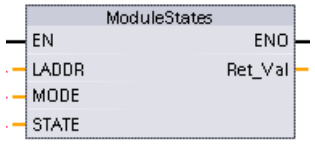
	`ret_val := ModuleStates(`
	`laddr:=_word_in_,`
	`mode:=_uint_in,`
	`state:=_variant_inout);`
		ModuleStates ruft die Betriebszustände von E/A-Modulen ab. Nach der Ausführung enthält der Parameter STATE den Fehlerzustand jedes einzelnen E/A-Moduls in einer Bitliste (für die zugewiesenen Parameter LADDR und MODE). Diese Informationen entsprechen dem Modulstatus in der Diagnoseansicht von STEP 7.
		Der Eingang LADDR der Anweisung ModuleStates verwendet die Hardwarekennung einer dezentralen E/A-Station und nicht des Kopfmoduls selbst. Die Hardwarekennung können Sie ermitteln, indem Sie die gesamte Station in der Netzsicht selektieren und dann in den Eigenschaften im Bereich "Hardwarekennung" nachsehen. Sie finden Sie außerdem über die Datentypen "Hw_Device" und "Hw_DpSlave" im Register "Systemkonstanten" in der PLC-Variablentabelle.

Tabelle 9- 130 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
LADDR	IN	HW_DEVICE	Logische Adresse (Kennung der E/A-Module)
MODE	IN	UInt	Unterstützt fünf Betriebsarten. Der Eingang MODE legt fest, welche Daten an der für STATE-Informationen angegebenen Stelle ausgegeben werden. Folgende Betriebsarten sind möglich: • 1: Modulkonfiguration aktiv • 2: Modul defekt • 3: Modul deaktiviert • 4: Modul vorhanden • 5: Problem im Modul

Parameter und Datentyp		Datentyp	Beschreibung
RET_VAL	OUT	Int	Status (Bedingungscode)
STATE[1]	InOut	Variant	Puffer, der den Fehlerzustand der einzelnen Module empfängt: Als Datentyp für den Parameter STATE können Sie einen beliebigen Bittyp (Bool, Byte, Word oder DWord) oder ein Array eines Bittyps auswählen. • Bit 0 des ersten Bytes der ausgegebenen STATE-Daten ist ein Zusammenfassungsbit. Ist es auf WAHR gesetzt, zeigt es an, dass andere Daten verfügbar sind. • Die vom Parameter STATE ausgegebenen Daten zeigen eine 1-zu-1-Übereinstimmung zwischen einer Bitposition und einer Modulposition. Diese Steckplatzadressierung ist WAHR für PROFIBUS und PROFINET. Beispiel: Bei einem ET 200S mit einem Kopfmodul, Powermodul und einem Paar E/A-Module stimmt Bit 1 im ersten Byte mit dem Kopfmodul überein, Bit 2 mit dem Powermodul und die Bits 3 und 4 mit den E/A-Modulen.

[1] Es können maximal 128 Bit zugewiesen werden. Die Anzahl der erforderlichen Bits ist von der Nutzung Ihres E/A-Moduls abhängig.

Tabelle 9- 131 Bedingungscodes

RET_VAL (W#16#...)	Beschreibung
0	Kein Fehler
8091	Von LADDR angegebenes Modul ist nicht vorhanden.
8092	Das von LADDR angegebene Modul spricht kein E/A-Gerät an.
8093	Ungültiger Datentyp für den Parameter STATE: Gültige Datentypen sind (Bool, Byte, Word oder Dword) oder ein Array aus (Bool, Byte, Word oder Dword).
80Bx	Die Anweisung ModuleStates wird von dieser CPU für diesen Parameter LADDR nicht unterstützt.
8452	Die vollständigen Zustandsdaten sind für den zugewiesenen Parameter STATE zu groß. Der Puffer STATE enthält ein Teilergebnis.

9.5.5.1 Beispiele für die Konfiguration von ModuleStates

Beispiel in PROFIBUS

Das Beispiel in PROFIBUS besteht aus den folgenden Komponenten:

- 16 PROFIBUS-Geräte mit den Namen "DPSlave_10" bis "DPSlave_25"
- Die 16 PROFIBUS-Geräte verwenden jeweils eine der PROFIBUS-Adressen 10 bis 25.
- Für jedes Slavegerät sind mehrere E/A-Module konfiguriert.
- Im Beispiel wird der Parameter LADDR des PROFIBUS-Slave "DPSlave_12" verwendet, der ein Kopfmodul, ein Powermodul und zwei E/A-Module enthält.
- Es werden die ersten vier Bytes der ausgegebenen STATE-Parameterinformationen angezeigt.

MODE	Beispiel 1: Normalbetrieb ohne Fehler	Beispiel 2: PROFIBUS Slavegerät DPSlave_12 mit gezogenem Modul	Beispiel 3: PROFIBUS Slavegerät DPSlave_12 getrennt
1: Modulkonfiguration aktiv	0x1F00_0000	0x1F00_0000	0x1F00_0000
2: Modul defekt	0x0000_0000	0x0900_0000	0x1F00_0000
3: Modul deaktiviert	0x0000_0000	0x0000_0000	0x0000_0000
4: Modul vorhanden	0x1F00_0000	0x1700_0000	0x0000_0000
5: Problem im Modul	0x0000_0000	0x0900_0000	0x1F00_0000

Die folgenden vier Tabellen zeigen eine binäre Aufschlüsselung der vier analysierten Datenbytes:

Tabelle 9- 132 Beispiel 1: Keine Fehler: Für MODE 1 (Modulkonfiguration aktiv) wird der Wert 0x1F00_0000 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x1F	Bit 7 0001-1111 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0x00	Bit 15 0000-0000 Bit 8	
Byte 3 0x00	Bit 23 0000-0000 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Die Steckplätze 1 (Bit 1) bis 4 (Bit 4) enthalten Module. Die Steckplätze 5 (Bit 5) und darüber hinaus enthalten keine Module. Die Daten von
MODE 4 (Modul vorhanden) stimmen mit MODE 1 (Modulkonfiguration aktiv) überein, also entsprechen die konfigurierten Module den vorhandenen Modulen.

Tabelle 9- 133 Beispiel 2: Ein Modul wurde aus dem PROFIBUS-Slavegerät "DPSlave_12" gezogen. Für MODE 2 (Modul defekt) wird der Wert 0x0900_0000 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x09	Bit 7 0000-**1**001 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0x00	Bit 15 0000-0000 Bit 8	
Byte 3 0x00	Bit 23 0000-0000 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Nur Modul 3 (Bit 3) ist als defekt gekennzeichnet. Alle anderen Module sind funktionsfähig.

Tabelle 9- 134 Beispiel 2 (Fortsetzung): Ein Modul wurde aus dem PROFIBUS-Slavegerät "DPSlave_12" gezogen. Für MODE 4 (Modul defekt) wird der Wert 0x1700_0000 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x17	Bit 7 0001-**0**111 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0x00	Bit 15 0000-0000 Bit 8	
Byte 3 0x00	Bit 23 0000-0000 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Modul 3 (Bit 3) wird als fehlend gezeigt. Die Module 1, 2 und 4 (Bits 1, 2 und 4) werden als vorhanden gezeigt.

Tabelle 9- 135 Beispiel 3: PROFIBUS-Slavegerät "DPSlave_12" ist vom PROFIBUS-Netzwerk getrennt (Kabel getrennt oder Spannungsverlust). Für MODE 2 (Modul defekt) wird der Wert 0x1F00_0000 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x1F	Bit 7 000**1**-1111 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0x00	Bit 15 0000-0000 Bit 8	
Byte 3 0x00	Bit 23 0000-0000 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Die Module in den Steckplätzen 1 bis 4 (Bits 1 bis 4) sind alle als defekt gekennzeichnet, weil das Gerät fehlt.

MODE 5 (Problem im Modul) zeigt die gleichen Informationen wie MODE 2 (Modul defekt).

Beispiel in PROFINET

Das Beispiel in PROFINET besteht aus den folgenden Komponenten:

- 16 PROFINET-Slavegeräte mit den Namen "et200s_1" bis "et200s_16".
- Die 16 PROFINET-Geräte verwenden jeweils eine der PROFINET-Gerätenummern 1 bis 16.
- Für jedes Slavegerät sind mehrere E/A-Module konfiguriert.
- Im Beispiel wird der PROFINET-Slave "et200s_1" verwendet, der ein Kopfmodul, ein Powermodul und 18 E/A-Module enthält.
- Es werden die ersten vier Bytes der ausgegebenen STATE-Parameterinformationen angezeigt.

MODE	Beispiel 1: Normalbetrieb ohne Fehler	Beispiel 2: PROFINET Gezogenes Modul an Slave et200s_1	Beispiel 3: PROFINET Slave et200s_1 getrennt
1: Modulkonfiguration aktiv	0xFFFF_1F00	0xFFFF_1F00	0xFFFF_1F00
2: Modul defekt	0x0000_0000	0x0180_0000	0xFFFF_1F00
3: Modul deaktiviert	0x0000_0000	0x0000_0000	0x0000_0000
4: Modul vorhanden	0xFFFF_1F00	0xFF7F_1F00	0x0000_0000
5: Problem im Modul	0x0000_0000	0x0180_0000	0xFFFF_1F00

Die folgenden vier Tabellen zeigen eine binäre Aufschlüsselung der vier analysierten Datenbytes:

Tabelle 9- 136 Beispiel 1: Keine Fehler: Für MODE 1 (Modulkonfiguration aktiv) wird der Wert 0xFFFF_1F00 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0xFF	Bit 7 1111-1111 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0xFF	Bit 15 1111-1111 Bit 8	
Byte 3 0x1F	Bit 23 0001-1111 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Die Steckplätze 1 (Bit 1) bis 20 (Bit 20) enthalten Module. Die Steckplätze 21 (Bit 21) und darüber hinaus enthalten keine Module. Die Daten von
MODE 4 (Modul vorhanden) stimmen mit MODE 1 (Modulkonfiguration aktiv) überein, also entsprechen die konfigurierten Module den vorhandenen Modulen.

Tabelle 9- 137 Beispiel 2: Ein Modul wurde aus dem PROFINET-Slavegerät "et200s_1" gezogen. Für MODE 2 (Modul defekt) wird der Wert 0x0180_0000 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0x01	Bit 7 0000-0001 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0x80	Bit 15 **1**000-0000 Bit 8	
Byte 3 0x00	Bit 23 0000-0000 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Nur Modul 15 (Bit 15) ist als defekt gekennzeichnet. Alle anderen Module sind funktionsfähig.

Tabelle 9- 138 Beispiel 2 (Fortsetzung): Ein Modul wurde aus dem PROFIBUS-Slavegerät "et200s_1" gezogen. Für MODE 4 (Modul defekt) wird der Wert 0xFF7F_1F00 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0xFF	Bit 7 1111-1111 Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0x7F	Bit 15 **0**111-1111 Bit 8	
Byte 3 0x1F	Bit 23 0001-1111 Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Modul 15 (Bit 15) wird als fehlend gezeigt. Die Module 1 bis 14 (Bits 1 bis 14) und 16 bis 20 (Bits 16 bis 20) werden als fehlend gezeigt.

Tabelle 9- 139 Beispiel 3: PROFINET-Slavegerät "et200s_1" ist vom PROFINET-Netzwerk getrennt (Kabel getrennt oder Spannungsverlust). Für MODE 2 (Modul defekt) wird der Wert 0xFFFF_1F00 ausgegeben.

Byte mit Wert	Bitmuster mit Wert	Hinweise
Byte 1 0xFF	Bit 7 **1111-1111** Bit 0	Bit 0 ist wahr; Daten sind verfügbar.
Byte 2 0xFF	Bit 15 **1111-1111** Bit 8	
Byte 3 0x1F	Bit 23 000**1-1111** Bit 16	
Byte 4 0x00	Bit 31 0000-0000 Bit 24	

Die Module in den Steckplätzen 1 bis 20 (Bits 1 bis 20) sind alle als defekt gekennzeichnet, weil das Gerät fehlt.

MODE 5 (Problem im Modul) zeigt die gleichen Informationen wie MODE 2 (Modul defekt).

9.5.6 Anweisung GET_DIAG (Diagnoseinformationen lesen)

Beschreibung

Mit der Anweisung GET_DIAG können Sie die Diagnoseinformationen eines Hardwaregeräts auslesen. Das Hardwaregerät wird über den Parameter LADDR ausgewählt. Mit dem Parameter MODE wählen Sie aus, welche Diagnoseinformationen ausgelesen werden sollen.

Tabelle 9- 140 Anweisung GET_DIAG

KOP/FUP	SCL	Beschreibung

	`ret_val := GET_DIAG(`
	`  mode:=_uint_in_,`
	`  laddr:=_word_in_,`
	`  cnt_diag=>_uint_out_,`
	`  diag:=_variant_inout_,`
	`  detail:=_variant_inout_);`
		Liest die Diagnoseinformationen aus einem zugewiesenen Hardwaregerät aus.

Parameter

Die folgende Tabelle zeigt die Parameter der Anweisung GET_DIAG:

Tabelle 9- 141 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
MODE	IN	UInt	Mit dem Parameter MODE geben Sie an, welche Diagnosedaten ausgegeben werden sollen.
LADDR	IN	HW_ANY (Word)	Hardwarekennung des Geräts
RET_VAL	OUT	Int	Zustand der Anweisung
CNT_DIAG	OUT	UInt	Anzahl der ausgegebenen Diagnosedetails
DIAG	InOut	Variant	Pointer auf den Datenbereich zum Speichern von Diagnoseinformationen des ausgewählten Modus
DETAILS	InOut	Variant	Pointer auf den Datenbereich zum Speichern von Diagnoseinformationen in Übereinstimmung mit dem ausgewählten Modus

Parameter MODE

Abhängig vom Wert des Parameters MODE werden unterschiedliche Diagnosedaten an den Ausgabeparametern DIAG, CNT_DIAG und DETAILS ausgegeben:

Tabelle 9- 142 Parameter MODE

MODE	Beschreibung	DIAG	CNT_DIAG	DETAILS
0	Ausgabe aller unterstützten Diagnoseinformationen für ein Modul als DWord, wobei Bit X=1 kenntlich macht, dass der Modus X unterstützt wird.	Bitzeichenfolge der unterstützten Modi als DWord, wobei Bit X=1 kenntlich macht, dass der Modus X unterstützt wird.	0	-
1	Ausgabe des zugehörigen Zustands des adressierten Hardwareobjekts.	Diagnosezustand: Ausgabe in Übereinstimmung mit der Struktur DIS. (Hinweis: Beachten Sie auch die Informationen zur "DIS-Struktur" unten und das Beispiel für die Anweisung GET_DIAG am Ende des Abschnitts.)	0	-
2	Ausgabe des Zustands aller untergeordneten Module des adressierten Hardwareobjekts.	Ausgabe der Diagnosedaten in Übereinstimmung mit der Struktur DNN. (Hinweis: Beachten Sie auch die Informationen zur "DNN-Struktur" unten und das Beispiel für die Anweisung GET_DIAG am Ende des Abschnitts.)	0	-

DIS-Struktur

Bei einem MODE-Parameter = 1 werden die Diagnoseinformationen in Übereinstimmung mit der Struktur DIS ausgegeben. Die folgende Tabelle zeigt die Bedeutung der einzelnen Parameterwerte:

Tabelle 9- 143 Struktur der Diagnoseinformationsquelle (DIS, Diagnostic Information Source)

Parameter	Datentyp	Wert	Beschreibung
MaintenanceState	DWord	Enum	
		0	Keine Wartung erforderlich
		1	Das Modul oder Gerät ist deaktiviert.
		2	-
		3	-
		4	-
		5	Wartung notwendig
		6	Wartung angefordert
		7	Fehler

Parameter	Datentyp	Wert	Beschreibung
		8	Zustand unbekannt/Fehler im untergeordneten Modul
		9	-
		10	Eingänge/Ausgänge sind nicht verfügbar.
Componentstate Detail	DWord	Bit-Array	Zustand der Submodule des Moduls: • Bit 0 bis 15: Statusmeldung des Moduls • Bit 16 bis 31: Statusmeldung der CPU
		0 bis 2 (Enum)	Weitere Informationen: • Bit 0: Keine weiteren Informationen • Bit 1: Übertragung nicht zulässig
		3	Bit 3 = 1: Mindestens ein Kanal unterstützt Kennzeichner für die Diagnose.
		4	Bit 4 = 1: Mindestens ein Kanal oder eine Komponente muss gewartet werden
		5	Bit 5 = 1: Für mindestens einen Kanal oder eine Komponente wurde die Wartung angefordert
		6	Bit 6 = 1: Fehler bei mindestens einem Kanal oder einer Komponente
		7 bis 10	Reserviert (immer = 0)
		11 bis 14	Bit 11 = 1: PNIO - Submodul korrekt Bit 12 = 1: PNIO - Ersatzmodul Bit 13 = 1: PNIO - falsches Modul Bit 14 = 1: PNIO - Modul getrennt
		15	Reserviert (immer = 0)
		16 bis 31	Von der CPU erzeugte Zustandsinformationen für Module: Bit 16 = 1: Modul deaktiviert Bit 17 = 1: CiR-Operation aktiv Bit 18 = 1: Eingang nicht verfügbar Bit 19 = 1: Ausgang nicht verfügbar Bit 20 = 1: Überlauf Diagnosepuffer Bit 21 = 1: Diagnose nicht verfügbar Bit 22 - 31: Reserviert (immer 0)
OwnState	Uint16	Enum	Der Wert des Parameters OwnState beschreibt den Wartungszustand des Moduls.
		0	Keine Störung
		1	Das Modul oder Gerät ist deaktiviert.
		2	Wartung notwendig
		3	Wartung angefordert
		4	Fehler
		5	Das Modul oder Gerät kann von der CPU nicht erreicht werden (gilt für Module und Geräte unter einer CPU).
		6	Eingänge/Ausgänge sind nicht verfügbar.
		7	-
IO State	Uint16	Bit-Array	E/A-Zustand des Moduls
		0	Bit 0 = 1: Keine Wartung erforderlich

Parameter	Datentyp	Wert	Beschreibung
		1	Bit 1 = 1: Das Modul oder Gerät ist deaktiviert.
		2	Bit 2 = 1: Wartung notwendig
		3	Bit 3 = 1: Wartung angefordert
		4	Bit 4 = 1: Fehler
		5	Bit 5 = 1: Das Modul oder Gerät kann von der CPU nicht erreicht werden (gilt für Module und Geräte unter einer CPU).
		6	Eingänge/Ausgänge sind nicht verfügbar.
		7	Kennzeichner; Bit 7 = 1, wenn Bit 0, 2 oder 3 gesetzt ist
		8 bis 15	Reserviert (immer = 0)
OperatingState	UInt16	Enum	
		0	-
		1	In STOP/Firmware-Update
		2	In STOP/Urlöschen des Speichers
		3	In STOP/Selbstanlauf
		4	In STOP
		5	Urlöschen des Speichers
		6	In START
		7	In RUN
		8	-
		9	In HOLD
		10	-
		11	-
		12	Modul defekt
		13	-
		14	Keine Spannung
		15	CiR
		16	In STOP/ohne DIS
		17	Eingang
		18	
		19	
		20	

DNN-Struktur

Bei einem MODE-Parameter = 2 werden die Diagnoseinformationsdetails in Übereinstimmung mit der Struktur DNN ausgegeben. Die folgende Tabelle zeigt die Bedeutung der einzelnen Parameterwerte:

Tabelle 9- 144 Struktur des Diagnosenavigationsknotens (DNN, Diagnostic Navigation Node)

Parameter	Datentyp	Wert	Beschreibung
SubordinateState	UINT	Enum	Zustand des untergeordneten Moduls (siehe Parameter OwnState der Struktur DIS)
SubordinateIOState	WORD	Bitarray	Zustand der Eingänge und Ausgänge des untergeordneten Moduls (siehe Parameter IO State der Struktur DIS)
DNNmode	WORD	Bitarray	• Bit 0 = 0: Diagnose aktiviert • Bit 0 = 1: Diagnose deaktiviert • Bit 1 bis 15: Reserviert

Parameter RET_VAL

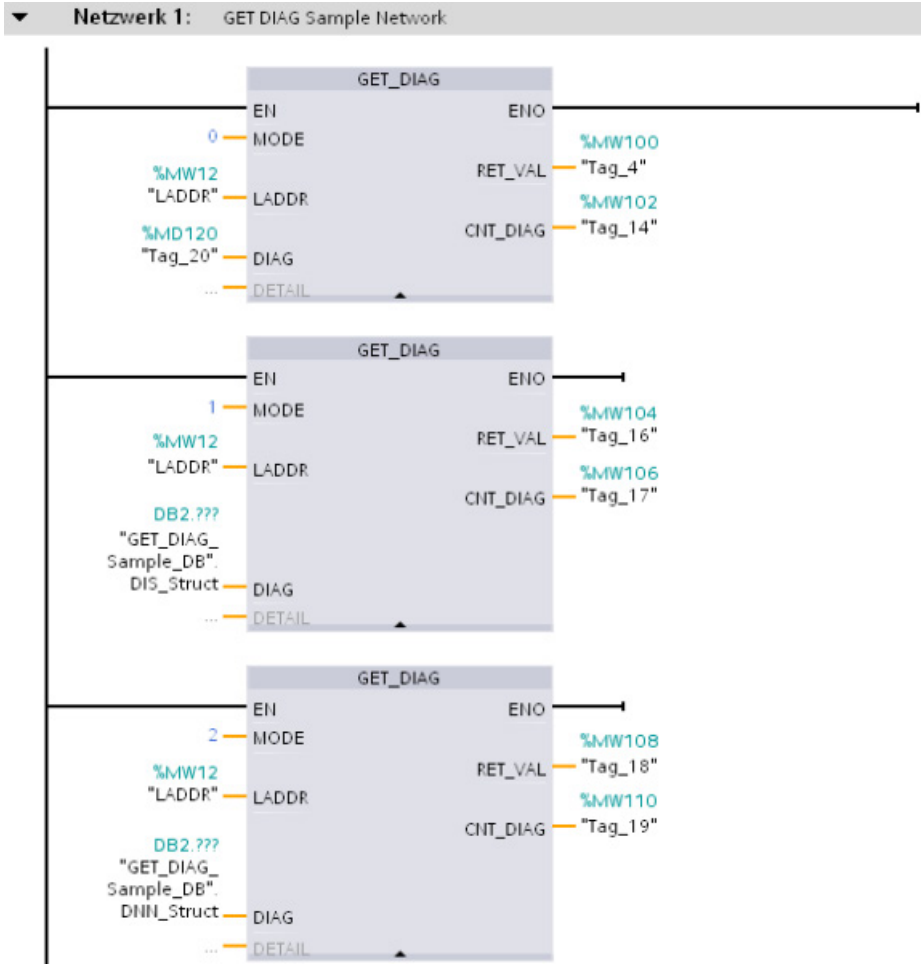
Tabelle 9- 145 Fehlercodes des Parameters RET_VAL

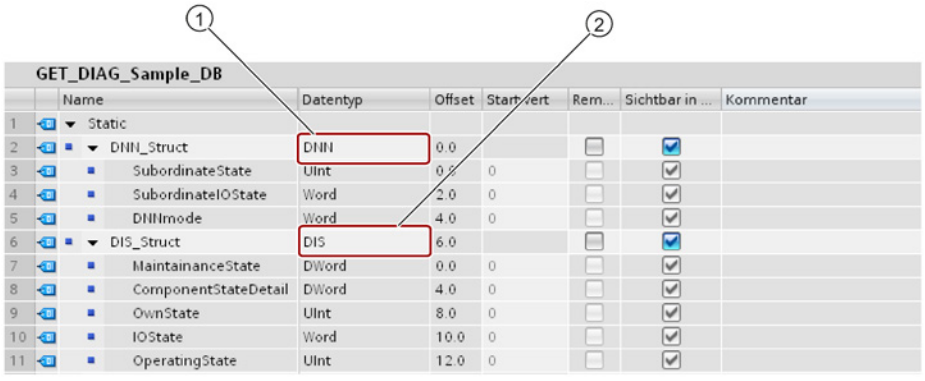
Fehlercode (W#16#...)	Beschreibung
0	Kein Fehler
8080	Wert im Parameter MODE wird nicht unterstützt.
8081	Typ im Parameter DIAG wird nicht mit dem ausgewählten Modus unterstützt (Parameter MODE).
8082	Typ im Parameter DETAILS wird nicht mit dem ausgewählten Modus unterstützt (Parameter MODE).
8090	LADDR ist nicht vorhanden.
8091	Der ausgewählte Kanal im Parameter CHANNEL ist nicht vorhanden.
80C1	Nicht genügend Ressourcen für die parallele Ausführung

Beispiel

Das folgende KOP-Netzwerk und der folgende DB zeigen, wie Sie die drei Betriebsarten mit den drei Strukturen verwenden:

- DIS
- DNN





① DNN
② DIS

Hinweis

Im DB müssen Sie den Datentyp manuell eingeben, um auf jede der drei Strukturen zuzugreifen. Eine Auswahl über eine Klappliste ist nicht möglich. Geben Sie die Datentypen genau wie im Folgenden gezeigt ein:

- DNN
- DIS

9.5.7 Anweisung Get_IM_Data (Identifikations- und Wartungsdaten lesen)

Mit der Anweisung Get_IM_Data prüfen Sie die Identifikations- und Wartungsdaten (I&M) für das angegebene Modul oder Submodul.

Tabelle 9- 146 Anweisung Get_IM_Data

KOP/FUP	SCL	Beschreibung

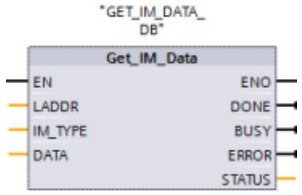
	`"GET_IM_DATA_DB"(LADDR:=16#0 ,`
	`IM_TYPE:=0,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	
	`DATA:=_variant_inout_);`
		Mit der Anweisung Get_IM_Data prüfen Sie die Identifikations- und Wartungsdaten (I&M) für das angegebene Modul oder Submodul.

Tabelle 9- 147 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
LADDR	Input	HW_EA	Kennung der Baugruppe
IM_TYPE	Input	UInt	Nummer der Identifikations- und Wartungdaten (I&M): • 0: I&M0 (MLFB, Seriennummer, Version und andere Informationen) • 1: I&M1 (Designatoren) • 2: I&M2 (Installationsdatum) • 3: I&M3 (Deskriptor) • 4: I&M4 (Signatur)
RET_VAL	Output	Int	Status (Bedingungscode)
DATA	InOut	Variante	I&M-Daten (STRING oder ein BYTE-Array)

Über die Identifikations- und Wartungsdaten (I&M) können Sie die Systemkonfiguration prüfen, Hardwareänderungen erkennen oder Wartungsdaten anzeigen. Die Modulidentifikationsdaten (I-Daten) sind schreibgeschützt. Die Modulwartungsdaten (M-Daten) sind von Systeminformationen wie z. B. dem Installationsdatum abhängig. M-Daten werden während der Wartungsplanung erstellt und in das Modul geschrieben:

- Wenn es sich bei dem Datentyp des Parameters DATA um eine Zeichenkette handelt, wird die aktuelle Länge der Zeichenkette entsprechend der Länge der I&M-Daten festgelegt.
- Wenn es sich bei dem Datentyp des Parameters DATA um ein Array aus Byte oder Char handelt, dann werden die I&M-Daten als Folge von Bytes kopiert.
- Wenn es sich bei dem Datentyp des Parameters DATA um eine Struktur handelt, dann werden die I&M-Daten als Folge von Bytes kopiert.
- Wenn das jeweilige Array aus Byte/Char in DATA länger ist als die angeforderten I&M-Daten, dann wird der Bytewert 16#00 angehängt.
- Andere Datentypen werden nicht unterstützt, es wird der Fehler 8093 zurückgegeben.

Tabelle 9- 148 Bedingungscodes

RET_VAL (W#16#...)	Beschreibung
0	Kein Fehler
8091	LADDR ist nicht vorhanden
8092	LADDR adressiert kein HW-Objekt, das I&M-Daten unterstützt
8093	Am Parameter DATA angegebener Datentyp wird nicht unterstützt
80B1	DATA Die Anweisung DATA wird von der CPU für diesen Parameter LADDR nicht unterstützt LADDR
80B2	IM_TYPE wird von der CPU nicht unterstützt
8452	Die vollständigen I&M-Informationen passen nicht in die am Parameter DATA angegebene Variable. Es wird ein Teilergebnis bis zur Bytelänge der Variable zurückgegeben.

9.6 Impuls

9.6.1 Anweisung CTRL_PWM (Impulsdauermodulation)

Tabelle 9- 149 Anweisung CTRL_PWM (Impulsdauermodulation)

KOP/FUP	SCL	Beschreibung

	`"CTRL_PWM_DB"(`
	`PWM:=W#16#0,`
	`enable:=FALSE,`
	`busy=>_bool_out_,`
	`status=>_word_out_);`
		Bietet eine feste Zykluszeit mit variabler relativer Einschaltdauer. Der PWM-Ausgang läuft nach dem Start kontinuierlich mit der angegebenen Frequenz (Zykluszeit). Die Impulsdauer wird nach Bedarf verändert, um die gewünschte Steuerung zu erzielen.

1 Wenn Sie die Anweisung einfügen, zeigt STEP 7 den Dialog "Aufrufoptionen" zum Erstellen des zugehörigen DB an.

2 Im SCL-Beispiel ist "CTRL_PWM_DB" der Name des Instanz-DBs.

Tabelle 9- 150 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
PWM	IN	HW_PWM (Word)	PWM-Kennung: Die Namen aktivierter Impulsgeneratoren werden zu Variablen in der Variablentabelle "Konstanten" und können als PWM-Parameter genutzt werden. (Standardwert: 0)
ENABLE	IN	Bool	1 = Impulsgenerator starten 0 = Impulsgenerator stoppen
BUSY	OUT	Bool	Funktion beschäftigt (Standardwert: 0)
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Die Anweisung CTRL_PWM speichert die Parameterinformationen im DB. Die Parameter des Datenbausteins werden nicht separat vom Anwender geändert, sondern von der Anweisung CTRL_PWM gesteuert.

Geben Sie den gewünschten Impulsgenerator über den Variablennamen für den Parameter PWM an.

Wenn der Eingang EN gleich WAHR ist, startet oder stoppt die Anweisung PWM_CTRL die angegebene PWM anhand des Werts am Eingang ENABLE. Die Impulsdauer wird von dem Wert der zugewiesenen Adresse des Ausgangsworts angegeben.

Weil die CPU die Anforderung verarbeitet, während die Anweisung CTRL_PWM ausgeführt wird, meldet der Parameter BUSY immer FALSCH. Wird ein Fehler erkannt, wird ENO auf FALSCH gesetzt und der Parameter STATUS enthält einen Fehlercode.

Die Impulsdauer wird beim ersten Wechsel der CPU in RUN auf den in der Gerätekonfiguration eingegebenen Anfangswert gesetzt. Um die Impulsdauer zu ändern, schreiben Sie die gewünschten Werte in die in der Gerätekonfiguration angegebene Ausgangswortadresse ("Ausgangsadressen"/"Anfangsadresse:"). Um die gewünschte Impulsdauer in das entsprechende Ausgangswort zu schreiben, nutzen Sie eine Anweisung wie Übertragen, Umwandeln, Arithmetik oder PID. Sie müssen den gültigen Bereich für den Wert des Ausgangsworts einhalten (Prozent, Tausendstel, Zehntausendstel oder S7-Analogformat).

Hinweis

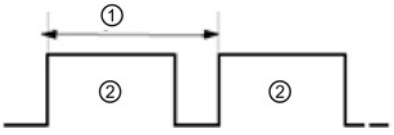
Zu PWM und PTO zugewiesene digitale E/A können nicht geforct werden.

Die von der Impulsdauermodulation (PWM) und der Impulsfolge (PTO) verwendeten digitalen E/A werden während der Gerätekonfiguration zugewiesen. Wenn diesen Funktionen digitale E/A zugewiesen werden, können die Werte der Adressen der zugewiesenen E/A nicht durch die Funktion zum Forcen in der Beobachtungstabelle geändert werden.

Tabelle 9- 151 Wert des Parameters STATUS

STATUS	Beschreibung
0	Kein Fehler
80A1	PWM-Kennung adressiert keine gültige PWM.

9.6.2 Funktionsweise der Impulsausgänge



① Zykluszeit

② Impulsdauer

Die Impulsdauer kann als Hundertstel der Zykluszeit (0 bis 100), als Tausendstel (0 bis 1000), als Zehntausendstel (0 bis 10000) oder als S7-Analogformat angegeben werden.

Die Impulsdauer kann zwischen 0 (kein Impuls, immer aus) und Vollausschlag (kein Impuls, immer ein) liegen.

Weil der PWM-Ausgang zwischen 0 und Vollausschlag liegen kann, bietet er einen digitalen Ausgang, der in vielerlei Hinsicht einem Analogausgang gleicht. Der PWM-Ausgang kann z. B. zur Steuerung der Drehzahl eines Motors vom Stillstand bis zur vollen Drehzahl dienen oder er kann dafür eingesetzt werden, die Position eines Ventils von geschlossen bis vollständig geöffnet zu steuern.

Zur Steuerung schneller Impulsausgänge stehen vier Impulsgeneratoren zur Verfügung: PWM und Impulsfolge (PTO). PTO wird von den Bewegungssteuerungsanweisungen genutzt. Sie können jeden Impulsgenerator entweder PWM oder PTO zuordnen, jedoch nicht beiden gleichzeitig.

Sie können integrierte CPU-Ausgänge oder die Ausgänge eines optionalen Signalboards nutzen. In der folgenden Tabelle werden die Adressen der Ausgänge aufgeführt (wobei die Standardkonfiguration der Ausgänge vorausgesetzt wird). Wenn Sie die Adressen der Ausgänge geändert haben, entsprechen die Adressen den von Ihnen zugewiesenen. Beachten Sie, dass PWM nur einen Ausgang benötigt, während PTO optional zwei Ausgänge je Kanal nutzen kann. Wenn ein Ausgang für eine Impulsfunktion nicht erforderlich ist, steht er zu anderen Zwecken zur Verfügung. Die E/A-Zuweisung entnehmen Sie der folgenden Tabelle.

Die nachstehende Tabelle zeigt die Standard-E/A-Zuweisungen. Die vier Impulsgeneratoren können jedoch für jeden integrierten oder digitalen SB-Ausgang einer CPU konfiguriert werden. Die verschiedenen Ausgänge unterstützen unterschiedliche Spannungen und Geschwindigkeiten. Berücksichtigen Sie dies bei der Zuweisung der PWM/PTO-Ausgänge.

Hinweis

Impulsfolgen können nicht von anderen Operationen im Anwenderprogramm verwendet werden.

Wenn Sie die Ausgänge der CPU oder des Signalboards als Impulsgeneratoren (für PWM- oder Bewegungssteuerungs-PTO-Anweisungen) konfigurieren, werden die entsprechenden Adressen der Ausgänge aus dem Speicher der Ausgänge entfernt und können nicht für andere Zwecke in Ihrem Anwenderprogramm verwendet werden. Wenn Ihr Anwenderprogramm einen Wert in einen Ausgang schreibt, der als Impulsgenerator genutzt wird, schreibt die CPU diesen Wert nicht in den physischen Ausgang.

Hinweis

PTO-Richtungsausgänge können für die Nutzung an anderer Stelle im Programm freigesetzt werden.

Für jeden PTO müssen zwei Ausgänge zugewiesen werden: ein Ausgang als Impulsausgang und ein Ausgang als Richtungsausgang. Sie können auch nur den Impulsausgang und nicht den Richtungsausgang verwenden. So können Sie den Richtungsausgang für andere Zwecke in Ihrem Anwenderprogramm freisetzen.

Tabelle 9- 152 Standard-Ausgangszuweisungen der Impulsgeneratoren[3]

Beschreibung	Impuls	Richtung
PTO1		
Integrierte E/A	A0.0	A0.1
SB-E/A	A4.0	A4.1
PWM1		
Integrierte Ausgänge	A0.0	-
SB-Ausgänge	A4.0	-
PTO2		
Integrierte E/A	A0.2	A0.3
SB-E/A	A4.2	A4.3
PWM2		
Integrierte Ausgänge	A0.2	-

Beschreibung	Impuls	Richtung
SB-Ausgänge	A4.2	-
PTO3		
Integrierte E/A	A0.4[1]	A0.5[1]
SB-E/A	A4.0	A4.1
PWM3		
Integrierte Ausgänge	A0.4[1]	-
SB-Ausgänge	A4.1	-
PTO4		
Integrierte E/A	A0.6[2]	A0.7[2]
SB-E/A	A4.2	A4.3
PWM4		
Integrierte Ausgänge	A0.6[2]	-
SB-Ausgänge	A4.3	-

[1] Die CPU 1211C hat keine Ausgänge A0.4, A0.5, A0.6 und A0.7. Deshalb können diese Ausgänge bei der CPU 1211C nicht verwendet werden.

[2] Die CPU 1212C hat keine Ausgänge A0.6 und A0.7. Deshalb können diese Ausgänge bei der CPU 1212C nicht verwendet werden.

[3] Diese Tabelle gilt für die PTO/PWM-Funktionen der CPU 1211C, CPU 1212C, CPU 1214C, CPU 1215C und CPU 1217C.

9.6.3 Konfigurieren eines Impulskanals für PWM

Um die Impulsdauermodulation vorzubereiten, konfigurieren Sie zunächst einen Impulskanal in der Gerätekonfiguration durch Auswahl der CPU, des Impulsgenerators (PTO/PWM) und PWM1 bis PWM4. Aktivieren Sie den Impulsgenerator (Optionskästchen). Wenn ein Impulsgenerator aktiviert wird, wird diesem bestimmten Impulsgenerator ein eindeutiger Standardname zugewiesen. Diesen Namen können Sie durch Bearbeiten im Eingabefeld "Name:" ändern, er muss jedoch eindeutig sein. Die Namen aktivierter Impulsgeneratoren werden zu Variablen in der Variablentabelle "Konstanten" und können als PWM-Parameter der Anweisung CTRL_PWM genutzt werden.

Tabelle 9- 153 CPU-Ausgang: Höchstfrequenz

CPU	CPU-Ausgangskanal:	Impuls- und Richtungsausgang	A/B, vorwärts/rückwärts und Impuls/Richtung
1211C	Aa.0 bis Aa.3	100 kHz	100 kHz
1212C	Aa.0 bis Aa.3	100 kHz	100 kHz
	Aa.4, Aa.5	20 kHz	20 kHz
1214C und 1215C	Aa.0 bis Aa.4	100 kHz	100 kHz
	Aa.5 bis Ab.1	20 kHz	20 kHz
1217C	DAa.0 bis DAa.3 (.0+, .0- bis .3+, .3-)	1 MHz	1 MHz
	DAa.4 bis DAb.1	100 kHz	100 kHz

Tabelle 9- 154 Signalboard-Ausgang: Höchstfrequenz (optionales Board)

Signalboard (SB)	SB-Ausgangskanal	Impuls- und Richtungsausgang	A/B, vorwärts/rückwärts und Impuls/Richtung
SB 1222, 200 kHz	DAe.0 bis DAe.3	200 kHz	200 kHz
SB 1223, 200 kHz	DAe.0, DAe.1	200 kHz	200 kHz
SB 1223	DAe.0, DAe.1	20 kHz	20 kHz

Hinweis

Die maximale Impulsfrequenz der Impulsgeneratoren beträgt 1 MHz bei der CPU 1217C und 100 kHz bei den CPUs 1211C, 1212C, 1214C und 1215C, 20 kHz (bei einem Standard-SB) bzw. 200 kHz (bei einem High-Speed-SB). Sie werden von STEP 7 jedoch nicht gewarnt, wenn Sie eine Achse mit einer maximalen Geschwindigkeit oder Frequenz, die diese Hardwareeinschränkung überschreitet, konfigurieren. Dies kann zu Problemen in Ihrer Anwendung führen. Stellen Sie also stets sicher, dass Sie die maximale Impulsfrequenz der Hardware nicht überschreiten.

Sie können den Impulsgenerator umbenennen, einen Kommentar einfügen und Parameter zuweisen. Dazu gehen Sie wie folgt vor:

- Der Impulsgenerator wird wie folgt genutzt: PWM oder PTO (wählen Sie PWM)
- Ausgangsquelle: integrierte CPU oder SB
- Zeitbasis: Millesekunden oder Mikrosekunden
- Impulsdauerformat:
  - Hundertstel (0 bis 100)
  - Tausendstel (0 bis 1000)
  - Zehntausendstel (0 bis 10000)
  - S7-Analogformat (0 bis 27648)
- Zykluszeit (Bereich ist 0 bis 16.777.215): Geben Sie Ihre Zykluszeit ein. Dieser Wert kann nur in der Gerätekonfiguration geändert werden.
- Anfangsimpulsdauer: Geben Sie die Anfangsimpulsdauer ein. Die Impulsdauer kann während der Laufzeit geändert werden.

Geben Sie die Anfangsadresse ein, um die Ausgangsadressen zu konfigurieren. Geben Sie die Adresse des Ausgansgworts ein, wo Sie den Wert für die Impulsdauer ablegen möchten.

Hinweis

Impulsfolgen können von anderen Anweisungen im Anwenderprogramm nicht verwendet werden

Wenn Sie die Ausgänge der CPU oder des Signalboards als Impulsgeneratoren (für PWM oder Bewegungssteuerungsanweisungen) konfigurieren, werden die entsprechenden Adressen der Ausgänge aus dem Speicher der Ausgänge entfernt und können nicht für andere Zwecke in Ihrem Anwenderprogramm verwendet werden. Wenn Ihr Anwenderprogramm einen Wert in einen Ausgang schreibt, der als Impulsgenerator genutzt wird, schreibt die CPU diesen Wert nicht in den physischen Ausgang.

Die Standardadressen für die Impulsdauerwerte sind:

- PWM1: AW1000
- PWM2: AW1002
- PWM3: AW1004
- PWM4: AW1006

Der Wert an dieser Adresse bestimmt die Dauer des Impulses und wird jedes Mal auf den oben angegebenen Wert für die "Anfangsimpulsdauer" initialisiert, wenn die CPU vom Betriebszustand STOP in den Betriebszustand RUN wechselt. Sie ändern diesen Ausgangswortwert während der Laufzeit, um die Impulsdauer zu verändern.

9.7 Rezepte und Datenprotokolle

9.7.1 Rezepte

9.7.1.1 Übersicht über Rezepte

Speicherung von Rezeptdaten

- Ein Rezeptdatenbaustein, den Sie in Ihrem Projekt erstellen, muss im **Lade**speicher der CPU abgelegt werden. Hierfür kann der interne CPU-Speicher oder eine externe, als Programmkarte verwendete Memory Card verwendet werden.
- Ein weiterer DB, den Sie erstellen müssen, ist der aktive Rezeptdatenbaustein. Dieser DB muss sich im **Arbeits**speicher befinden, in dem jeweils ein aktiver Rezeptdatensatz von Ihrer Programmlogik gelesen oder geschrieben wird.

Verwaltung von Rezeptdaten

Der Rezept-DB verwendet ein Array aus Produktrezeptdatensätzen. Jedes Element des Rezept-Arrays stellt eine unterschiedliche Rezeptvariante dar, die auf einem gemeinsamen Satz Komponenten basiert.

- Sie erstellen einen PLC-Datentyp oder einen Datentyp Struct, der alle Komponenten in einem Rezeptdatensatz definiert. Diese Datentypvorlage wird für alle Rezeptdatensätze wiederverwendet. Produktrezepte variieren je nach den Startwerten, die den Rezeptkomponenten zugewiesen sind.
- Eines der Rezepte kann jederzeit mit der Anweisung READ_DBL aus dem Rezept-DB (alle Rezepte im Ladespeicher) in den aktiven Rezept-DB (ein Rezept im Arbeitsspeicher) übertragen werden. Nachdem ein Rezeptdatensatz in den Arbeitsspeicher verschoben wurde, kann Ihre Programmlogik die Komponentenwerte lesen und mit dem Produktionslauf beginnen. Durch die Übertragung wird die Menge des für die Rezeptdaten erforderlichen CPU-Arbeitsspeichers minimiert.
- Werden die Komponentenwerte des aktiven Rezepts während eines Produktionslaufs von einem HMI-Gerät angepasst, können Sie die geänderten Werte mit der Anweisung WRIT_DBL in den Rezept-DB zurückschreiben.

Export von Rezepten (vom Rezept-DB in eine CSV-Datei)

Mit der Anweisung RecipeExport kann der vollständige Satz Rezeptdatensätze in eine CSV-Datei exportiert werden. Auch nicht verwendete Rezeptdatensätze werden dabei exportiert.

Import von Rezepten (aus einer CSV-Datei in einen Rezept-DB)

Nach dem Export der Rezepte können Sie die generierte CSV-Datei als Datenstrukturvorlage verwenden.

1. Eine vorhandene CSV-Rezeptdatei laden Sie über die Dateibrowser-Seite des CPU-Webservers von der CPU in einen PC.
2. Ändern Sie die CSV-Rezeptdatei mithilfe eines ASCII-Texteditors. Sie können die Startwerte der Komponenten ändern, jedoch nicht die Datentypen oder Datenstruktur.
3. Laden Sie die geänderte CSV-Datei vom PC wieder zurück in die CPU. Sie müssen dafür jedoch die alte CSV-Datei (mit dem gleichen Namen) im Ladespeicher der CPU löschen oder umbenennen, damit der CPU-Webserver den Ladevorgang durchführen kann.
4. Nachdem Sie die geänderte CSV-Datei in die CPU geladen haben, können Sie die neuen Startwerte mit der Anweisung RecipeImport aus der geänderten CSV-Datei (im CPU-Ladespeicher) in den Rezept-DB (im CPU-Ladespeicher) übertragen.

9.7.1.2 Beispiel für ein Rezept

Beispiel für Rezepte

In dieser Tabelle wird gezeigt, wie Sie Rezeptinformationen für die Verwendung in einem Rezept-DB vorbereiten. In diesem Beispiel für einen Rezept-DB sind fünf Datensätze gespeichert, von denen drei verwendet werden. Der vierte und fünfte Datensatz werden zur späteren Erweiterung freigehalten. Jede Zeile in der Tabelle stellt einen Datensatz mit Rezeptname, Komponenten-Datentypen und Komponentenwerten dar.

productname	water	barley	wheat	hops	yeast	waterTmp	mashTmp	mashTime	QTest
Pils	10	9	3	280	39	40	30	100	0
Lager	10	9	3	150	33	50	30	120	0
BlackBeer	10	9	3	410	47	60	30	90	1
Not_used	0	0	0	0	0	0	0	0	0
Not_used	0	0	0	0	0	0	0	0	0

Erstellen eines Rezeptdatenbausteins

Hinweis

Regeln für Rezeptdatenbausteine

- Der Rezept-DB muss ein eindimensionales Array vom PLC-Datentyp oder Datentyp Struct enthalten. Im Rezeptbeispiel wird gezeigt, wie Sie einen Rezept-DB mit PLC-Datentyp erstellen.
- Im Beispiel haben die Komponenteninhaltsstoffe alle den Datentyp UINT. Die Datentypen der Komponenten können auch eine Mischung verschiedener Datentypen außer Struct sein. In einem Array-Element im Rezept-DB ist ein Struct in einem PLC-Datentyp oder ein Struct im Datentyp Struct nicht zulässig.

Erstellen Sie zunächst einen neuen PLC-Datentyp

Fügen Sie einen neuen PLC-Datentyp hinzu, dessen Name den Rezepttyp angibt. In der folgenden Abbildung ist "Beer_Recipe" der neue komplexe PLC-Datentyp, der eine Folge einfacher Datentypen speichert. Der PLC-Datentyp "Beer_Recipe" ist eine Datenvorlage, die in jedem Datensatz des Rezept-DBs und auch im aktiven Rezept-DB wiederverwendet wird. Geben Sie die Komponentennamen und Datentypen ein, die in allen Beispielrezepten gemeinsam vorhanden sind. Die einzelnen Komponentenwerte werden später im Rezept-DB hinzugefügt.

Beer_Recipe

	Name	Data type	Default value
1	productname	String[20]	'Beer_Recipe'
2	water	UInt	0
3	barley	UInt	0
4	wheat	UInt	0
5	hops	UInt	0
6	yeast	UInt	0
7	waterTmp	UInt	0
8	mashTmp	UInt	0
9	mashTime	UInt	0
10	QTest	UInt	0

Erstellen Sie dann einen Rezeptdatenbaustein

- Erstellen Sie Ihren Rezept-DB als globalen Datenbaustein mit aktivierter DB-Eigenschaft "Nur im Ladespeicher ablegen".
- Der Name des Rezeptdatenbausteins wird als Dateiname der entsprechenden CSV-Datei verwendet. Die im DB-Namen enthaltenen Zeichen müssen sich an die Namenseinschränkungen des Windows-Dateisystems halten. Die Zeichen \ / : * ? " < > | und das Leerzeichen sind nicht zulässig.
- Die Zuweisung des Rezept-Arrays lautet "Products" als Array [1.. 5] of "Beer_Recipe". Die Array-Größe 5 ist die maximal mögliche Anzahl Rezeptvarianten.
- Die Werte für die Rezeptkomponenten werden als DB-Startwerte hinzugefügt.

In der folgenden Abbildung ist das Rezept "BlackBeer" erweitert, um alle Komponenten eines Rezeptdatensatzes zu zeigen.

Recipe_DB

	Name	Data type	Offset	Start value
1	Static			
2	Products	Array [1 .. 5] of "Beer_Recipe"	...	
3	Products[1]	"Beer_Recipe"	...	
4	Products[2]	"Beer_Recipe"	...	
5	Products[3]	"Beer_Recipe"	...	
6	productname	String[20]	...	'BlackBeer'
7	water	UInt	...	10
8	barley	UInt	...	9
9	wheat	UInt	...	3
10	hops	UInt	...	410
11	yeast	UInt	...	47
12	waterTmp	UInt	...	60
13	mashTmp	UInt	...	30
14	mashTime	UInt	...	90
15	QTest	UInt	...	1
16	Products[4]	"Beer_Recipe"	...	
17	Products[5]	"Beer_Recipe"	...	

Export von Rezepten (von Rezept-DB in CSV-Datei)

"Durch die Ausführung der Anweisung "RecipeExport (Seite 424)" werden die Daten aus dem Rezept-DB in eine CSV-Datei übertragen (siehe folgende Textdatei).

```
Recipe_DB.csv
index,productname,water,barley,wheat,hops,yeast,waterTmp,
mashTmp,mashTime,QTest
1,”Pils”,10,9,3,280,39,40,30,100,0
2,”Lager”,10,9,3,150,33,50,30,120,0
3,”BlackBeer”,10,9,3,410,47,60,30,90,1
4 ”Not_used”,0,0,0,0,0,0,0,0,0
5 ”Not_used”,0,0,0,0,0,0,0,0,0
```

Import von Rezepten (aus CSV-Datei in Rezept-DB)

1. Eine vorhandene CSV-Rezeptdatei laden Sie über die Dateibrowser-Seite des CPU-Webservers aus dem CPU-Ladespeicher in einen PC.
2. Ändern Sie die CSV-Rezeptdatei mithilfe eines ASCII-Texteditors. Sie können die Startwerte der Komponenten ändern, jedoch nicht die Datentypen oder Datenstruktur.
3. Laden Sie die geänderte CSV-Datei vom PC wieder zurück in die CPU. Sie müssen dafür jedoch die alte CSV-Datei (mit dem gleichen Namen) im Ladespeicher der CPU löschen oder umbenennen, damit der CPU-Webserver den Ladevorgang durchführen kann.
4. Nachdem Sie die geänderte CSV-Datei in die CPU geladen haben, können Sie die neuen Startwerte mit der Anweisung RecipeImport aus der geänderten CSV-Datei (im CPU-Ladespeicher) in den Rezept-DB (im CPU-Ladespeicher) übertragen.

CSV-Dateien müssen exakt mit der Struktur des entsprechenden Rezept-DBs übereinstimmen

- Die Werte in der CSV-Datei können geändert werden, doch die Struktur darf nicht geändert werden. Für die Anweisung RecipeImport ist es erforderlich, dass die Anzahl der Datensätze und Komponenten exakt der Struktur des Ziel-Rezept-DBs entspricht. Ansonsten schlägt die Ausführung der Anweisung RecipeImport fehl. Beispiel: Sind im Rezept-DB 10 Rezepte definiert, doch werden nur 6 davon verwendet, werden auch die Zeilen 7 bis 10 der CSV-Datei in den DB übertragen. Sie müssen regeln, ob diese Daten gültig sind oder nicht. So können Sie beispielsweise in nicht verwendeten Rezeptdatensätzen dem Produktnamen die Variable “Not_used” zuweisen.
- Wenn Sie der Textdatei Datensätze hinzufügen und die geänderte Datei importieren, müssen Sie darauf achten, dass der von Ihnen zugewiesene Array-Grenzwert des Rezept-DBs genügend Elemente für alle Rezeptdatensätze umfasst.
- Beim Export in die CSV-Datei wird automatisch eine Indexnummer generiert. Wenn Sie zusätzliche Datensätze erstellen, fügen Sie dementsprechend nachfolgende Indexnummern hinzu.
- Bei der Ausführung der Anweisung RecipeImport wird die CSV-Datei auf korrekte Struktur und daraufhin geprüft, ob die Werte in die im zugehörigen Rezept-DB zugewiesenen Datentypen passen. So kann beispielsweise der Datentyp Bool keinen ganzzahligen Wert speichern, und die Ausführung von RecipeImport schlägt in einem solchen Fall fehl.

Anzeige von CSV-Rezeptdaten in Excel

Die CSV-Datei kann zum einfacheren Lesen und Bearbeiten in Excel geöffnet werden. Werden die Kommas nicht als Dezimaltrennzeichen erkannt, verwenden Sie die Importfunktion von Excel, um die Daten strukturiert auszugeben.

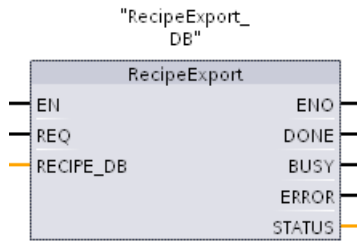
	A	B	C	D	E	F	G	H	I	J	K
1	index	product	water	barley	wheat	hops	yeast	waterTmp	mashTmp	mashTime	QTest
2	1	"Pils"	10	9	3	280	39	40	30	100	0
3	2	"Lager"	10	9	3	150	33	50	30	120	0
4	3	"BlackBeer"	10	9	3	410	47	60	30	90	1
5	4	"Not_used"	0	0	0	0	0	0	0	0	0
6	5	"Not_used"	0	0	0	0	0	0	0	0	0

9.7.1.3 Programmanweisungen zum Übertragen von Rezeptdaten

Anweisung RecipeExport (Rezeptexport)

Tabelle 9- 155 Anweisung RecipeExport

KOP/FUP	SCL	Beschreibung

	`"RecipeExport_DB"(`
	`req:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`Reci-`
	`pe_DB:=_variant_inout_);`
		Die Anweisung "RecipeExport" exportiert alle Rezeptdatensätze aus einem Rezeptdatenbaustein in das CSV-Dateiformat. Die CSV-Datei enthält Produktnamen, Komponentennamen und Startwerte. Die CSV-Datei wird im internen Ladespeicher abgelegt oder auch im externen Ladespeicher, sofern eine optionale als Programmkarte verwendete externe Memory Card installiert ist.
		Der Exportvorgang wird vom Parameter "REQ" angestoßen. Der Parameter BUSY wird während der Exportverarbeitung auf 1 gesetzt. Nach der Ausführung von Recipe-Export wird BUSY auf 0 zurückgesetzt und das Ende des Vorgangs wird am Parameter DONE mit 1 angezeigt. Tritt während der Ausführung ein Fehler auf, zeigen die Parameter ERROR und STATUS das Ergebnis an.

Damit der Export von Rezepten möglich ist, muss ein Rezept-DB erstellt werden. Der Name des Rezeptdatenbausteins wird als Dateiname der neuen CSV-Datei verwendet. Ist bereits eine CSV-Datei mit identischem Namen vorhanden, wird sie während des Exportvorgangs überschrieben.

Sie können über die Dateibrowser-Seite (Seite 844) des integrierten Webservers der CPU auf die CSV-Datei mit den Rezepten zugreifen. Die Datei wird im Rezeptordner im Stammverzeichnis des CPU-Ladespeichers abgelegt.

Tabelle 9- 156 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Steuerparameter REQUEST: Aktiviert den Export bei einer steigenden Flanke.
RECIPE_DB	Durchgang	Variant	Pointer auf den Rezeptdatenbaustein. Weitere Informationen finden Sie im Beispiel für einen Rezept-DB (Seite 420). Die im DB-Namen enthaltenen Zeichen müssen sich an die Namenseinschränkungen des Windows-Dateisystems halten. Die Zeichen \ / : * ? " < > | und das Leerzeichen sind nicht zulässig.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde. (Standardwert: Falsch)
BUSY	OUT	Bool	Ausführung von RecipeExport • 0: Keine Anweisung in Bearbeitung • 1: Anweisung in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist. • 0: Keine Warnung und kein Fehler • 1: Ein Fehler ist aufgetreten. Der Parameter STATUS enthält Informationen über die Fehlerart.
STATUS	OUT	Word	Ausführungsbedingung

Tabelle 9- 157 Werte von ERROR und STATUS

ERROR	STATUS (W#16#....)	Beschreibung
0	0000	Kein Fehler
0	7000	Aufruf ohne REQ-Flanke: BUSY = 0, DONE = 0
0	7001	Erster Aufruf mit REQ-Flanke (in Bearbeitung): BUSY = 1, DONE = 0
0	7002	N^{ter} Aufruf (in Bearbeitung): BUSY = 1, DONE = 0
1	8070	Gesamter Instanzspeicher ist belegt.
1	8090	Dateiname enthält ungültige Zeichen.
1	8091	Die in RECIPE_DB angegebene Datenstruktur kann nicht verarbeitet werden.
1	8092	Die in RECIPE_DB angegebene Datenstruktur überschreitet 5000 Byte.
1	80B3	Nicht genügend Platz auf MC oder im internen Ladespeicher.
1	80B4	MC ist schreibgeschützt.
1	80B6	Attribut "Nur im Ladespeicher ablegen" des Rezept-DBs ist nicht aktiviert.
1	80C0	CSV-Datei ist kurzzeitig gesperrt.
1	80C1	DB ist kurzzeitig gesperrt.

Anweisung RecipeImport (Rezeptimport)

Tabelle 9- 158 Anweisung RecipeImport

KOP/FUP	SCL	Beschreibung

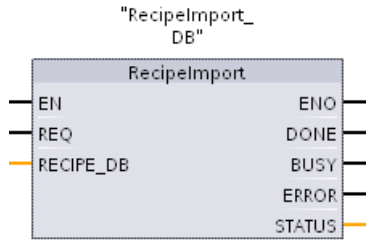
	`"RecipeImport_DB"(`
	`req:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`Reci-`
	`pe_DB:=_variant_inout_);`
		Die Anweisung "RecipeImport" importiert Rezeptdaten aus einer CSV-Datei im CPU-Ladespeicher in einen vom Parameter RECIPE_DB angegebenen Rezeptdatenbaustein. Während des Importvorgangs werden die Startwerte im Rezeptdatenbaustein überschrieben. Der Importvorgang wird vom Parameter "REQ" angestoßen. Der Parameter BUSY wird während der Importverarbeitung auf 1 gesetzt. Nach der Ausführung von RecipeImport wird BUSY auf 0 zurückgesetzt und das Ende des Vorgangs wird am Parameter DONE mit 1 angezeigt. Tritt während der Ausführung ein Fehler auf, zeigen die Parameter ERROR und STATUS das Ergebnis an.

Tabelle 9- 159 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Steuerparameter REQUEST: Aktiviert den Import bei einer steigenden Flanke.
RECIPE_DB	Durchgang	Variant	Pointer auf den Rezeptdatenbaustein. Weitere Informationen finden Sie im "Beispiel für einen Rezept-DB (Seite 420)". Die im DB-Namen enthaltenen Zeichen müssen sich an die Namenseinschränkungen des Windows-Dateisystems halten. Die Zeichen \ / : * ? " < > | und das Leerzeichen sind nicht zulässig.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde. (Standardwert: Falsch)
BUSY	OUT	Bool	• 0 - Keine Anweisung in Bearbeitung • 1 - Anweisung in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Damit ein Importvorgang durchgeführt werden kann, muss ein Rezept-DB mit einer Struktur, die der Datenstruktur der CSV-Datei entspricht, vorhanden sein.

Regeln für CSV-Dateien:

- Die CSV-Datei muss sich im Stammverzeichnis "Recipes" im internen Ladespeicher oder auch im externen Ladespeicher, sofern eine optionale als Programmkarte verwendete externe Memory Card installiert ist, befinden.
- Der Name der CSV-Datei muss dem Namen des Datenbausteins im Parameter RECIPE_DB entsprechen.
- Die erste Zeile (Überschrift) der CSV-Datei enthält den Namen der Rezeptkomponenten. Die erste Zeile wird beim Import ignoriert. Die Namen der Rezeptkomponenten in der CSV-Datei und im Datenbaustein werden während des Importvorgangs nicht aufeinander abgestimmt.
- In jedem Fall ist der erste Wert in jeder Zeile der CSV-Datei die Indexnummer des Rezepts. Die einzelnen Rezepte werden in der Reihenfolge des Index importiert. Hierfür muss sich der Index in der CSV-Datei in aufsteigender Reihenfolge befinden und darf keine Lücken enthalten (ist dies nicht der Fall, wird im Parameter STATUS die Fehlermeldung 80B0 ausgegeben).
- Die CSV-Datei darf nicht mehr Rezeptdatensätze enthalten als im Rezeptdatenbaustein vorgesehen. Die maximale Anzahl Datensätze wird von den Array-Grenzwerten im Datenbaustein angegeben.

Tabelle 9- 160 Werte von ERROR und STATUS

ERROR	STATUS (W#16#....)	Beschreibung
0	0000	Kein Fehler
0	7000	Aufruf ohne REQ-Flanke: BUSY = 0, DONE = 0
0	7001	Erster Aufruf mit REQ-Flanke (in Bearbeitung): BUSY = 1, DONE = 0
0	7002	N^{ter} Aufruf (in Bearbeitung): BUSY = 1, DONE = 0
1	8070	Gesamter Instanzspeicher ist belegt.
1	8090	Der Dateiname enthält ungültige Zeichen.
1	8092	Keine passende CSV-Datei zum Importieren gefunden. Mögliche Ursache: Der Name der CSV-Datei entspricht nicht dem Namen des Rezept-DBs.
1	80C0	CSV-Datei ist kurzzeitig gesperrt.
1	80C1	Datenbaustein ist kurzzeitig gesperrt.
1	80B0	Die Nummerierung im Index der CSV-Datei ist nicht kontinuierlich, nicht in aufsteigender Reihenfolge oder überschreitet die maximale Anzahl (Array-Grenzwert) im Datenbaustein.
1	80B1	Struktur des Rezeptdatenbausteins und der CSV-Datei stimmen nicht überein: Die CSV-Datei enthält zu viele Felder.
1	80B2	Struktur des Rezeptdatenbausteins und der CSV-Datei stimmen nicht überein: Die CSV-Datei enthält zu wenige Felder.
1	80B6	Attribut "Nur im Ladespeicher ablegen" des Rezept-DBs ist nicht aktiviert.
1	80D0 +n	Struktur des Rezeptdatenbausteins und der CSV-Datei stimmen nicht überein: Datentyp im Feld n stimmt nicht überein (n <= 46).
1	80FF	Struktur des Rezeptdatenbausteins und der CSV-Datei stimmen nicht überein: Datentyp im Feld n stimmt nicht überein (n > 46).

9.7.1.4 Beispielprogramm für Rezepte

Voraussetzungen für das Beispielprogramm für Rezepte

Für das Beispielprogramm für Rezepte müssen die folgenden Voraussetzungen erfüllt sein:

- Ein Rezept-DB mit allen Rezeptdatensätzen. Der Rezept-DB ist im Ladespeicher abgelegt.
- Ein aktiver Rezept-DB mit einer Kopie eines Rezepts im Arbeitsspeicher.

Weitere Informationen zum Rezept-DB und der entsprechenden CSV-Datei finden Sie unter "Beispiel für einen Rezept-DB (Seite 420)".

Erstellen Sie den aktiven Rezept-DB

Im Fenster "Neuen Baustein hinzufügen":

- Wählen Sie im Fenster "Neuen Baustein hinzufügen" die Schaltfläche "Datenbaustein".
- Wählen Sie in der Klappliste "Typ" den zuvor erstellten PLC-Datentyp "Beer_recipe" aus.

Startwerte sind nicht erforderlich. Wenn ein Rezept aus dem Rezept-DB in den aktiven Rezept-DB übertragen wird, werden die DB-Datenwerte festgelegt. In dem Beispiel ist der aktive Rezept-DB das Ziel für die Daten der Anweisung READ_DBL und bietet der Anweisung WRITE_DBL Quelldaten. Die folgende Abbildung zeigt den DB "Active_Recipe".

Active_Recipe

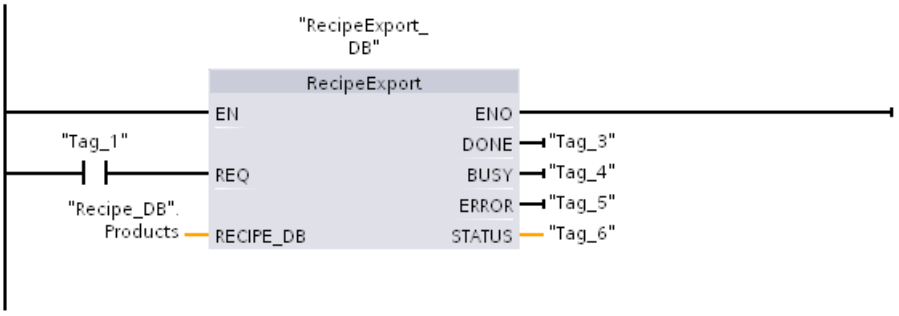
	Name	Data type	Start value
1	Static		
2	productname	String[20]	'Beer_Recipe'
3	water	UInt	0
4	barley	UInt	0
5	wheat	UInt	0
6	hops	UInt	0
7	yeast	UInt	0
8	waterTmp	UInt	0
9	mashTmp	UInt	0
10	mashTime	UInt	0
11	QTest	UInt	0

Instanz-DBs

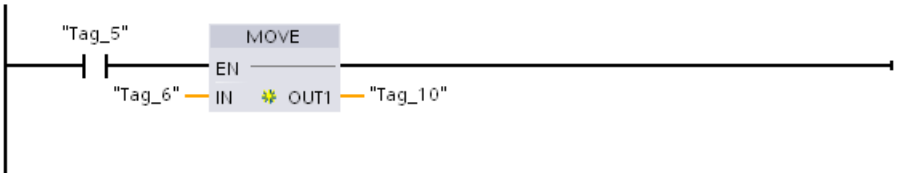
Die von den Anweisungen RecipeExport ("RecipeExport_DB") und RecipeImport ("RecipeImport_DB") verwendeten Instanz-DBs werden automatisch erstellt, wenn Sie die Anweisungen in Ihr Programm einfügen. Die Instanz-DBs dienen zum Steuern der Anweisungsausführung und werden in der Programmlogik nicht referenziert.

Beispielprogramm für Rezepte

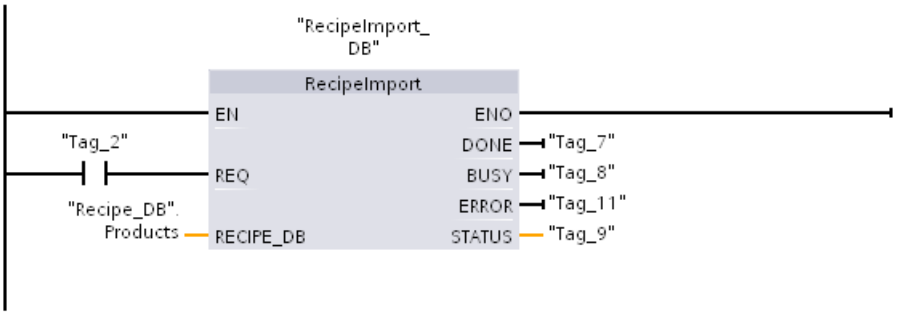
Netzwerk 1 Eine steigende Flanke an REQ startet den Exportvorgang. Aus den Daten des Rezept-DBs wird eine CSV-Datei generiert und im Rezeptordner im CPU-Speicher abgelegt.



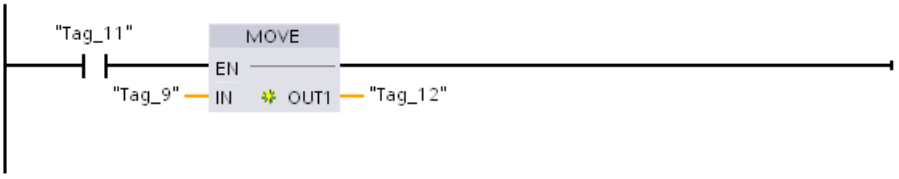
Netzwerk 2 Erfassen Sie den Ausgang STATUS der Ausführung von RecipeExport, weil er nur einen Zyklus lang gültig ist.



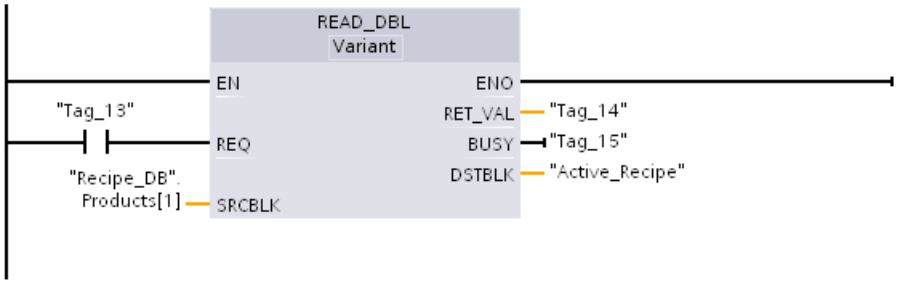
Netzwerk 3 Eine steigende Flanke an REQ startet den Importvorgang. In den vorhandenen Rezept-DB werden alle aus der entsprechenden CSV-Datei, die sich im Rezeptordner des CPU-Speichers befindet, gelesenen Rezeptdaten geladen.



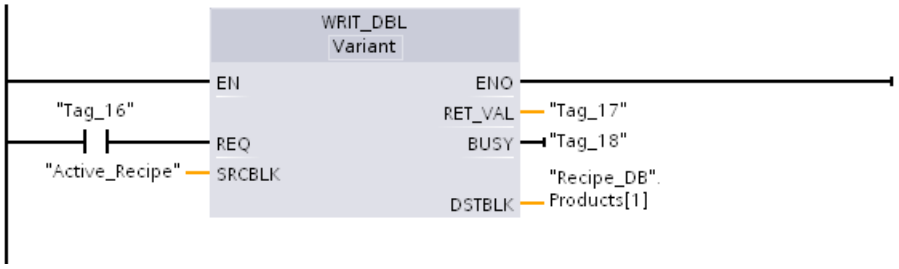
Netzwerk 4 Erfassen Sie den Ausgang STATUS der Ausführung von RecipeImport, weil er nur einen Zyklus lang gültig ist.



Netzwerk 5 READ_DBL kopiert die Startwerte von einem Rezept "Recipe_DB". Products[1] (im CPU-Ladespeicher) in die aktuellen Werte des DBs "Active_Recipe" (im CPU-Arbeitsspeicher). Nach der Ausführung von READ_DBL kann Ihre Programmlogik über die Adressen im Speicher des DBs "Active_Recipe" auf die Komponentenwerte des Rezepts zugreifen. So geben beispielsweise die symbolischen Adressen ("Active_Recipe".productname) und ("Active_Recipe".water) Ihrer Programmlogik den aktuellen Rezeptnamen und die Menge des Wassers an.



Netzwerk 6 Während der Laufzeit könnte ein HMI-Gerät einen im DB "Active_Recipe" gespeicherten Komponentenwert ändern. Verbesserte Rezeptdaten können durch Ausführung von WRIT_DBL gespeichert werden. In dem Beispiel werden alle Startwerte des Rezept-DBs für das einzelne Rezept "Recipe_DB". Products[1] durch die aktuellen Werte aus dem DB "Active_Recipe" überschrieben.



9.7.2 Datenprotokolle

Ihr Steuerungsprogramm kann mit den Anweisungen Data log Laufzeitdatenwerte in beständigen Protokolldateien speichern. Die Datenprotokolldateien werden im Flash-Speicher (CPU oder Memory Card) gespeichert. Die Daten der Protokolldateien werden im herkömmlichen CSV-Format (durch Komma getrennte Werte) gespeichert. Die Datensätze sind in einer zirkulären Protokolldatei vordefinierter Größe organisiert.

Die Anweisungen Data log dienen in Ihrem Programm dazu, einen Datensatz anzulegen, zu öffnen, zu schreiben und die Protokolldateien zu schließen. Sie entscheiden, welche Programmwerte protokolliert werden, indem Sie einen Datenpuffer anlegen, der einen einzigen Protokolldatensatz definiert. Ihr Datenpuffer wird als temporärer Speicher für einen neuen Protokolldatensatz verwendet. Neue aktuelle Werte müssen während der Laufzeit programmatisch in den Puffer übertragen werden. Wenn alle aktuellen Datenwerte aktualisiert sind, können Sie die Anweisung DataLogWrite ausführen, um Daten aus dem Puffer in einen Protokolldatensatz zu übertragen.

Datenprotokolldateien können Sie auf der Dateibrowser-Seite des Webservers öffnen, bearbeiten, speichern, umbenennen und löschen. Um den Dateibrowser anzuzeigen, benötigen Sie Leserechte, und um Datenprotokolldateien zu bearbeiten, zu löschen oder umzubenennen, benötigen Sie Änderungsrechte.

9.7.2.1 Datensatzstruktur der Datenprotokolle

Die Parameter DATA und HEADER der Anweisung DataLogCreate weisen den Datentyp und die Spaltenkopfbeschreibung aller Datenelemente in einem Protokolldatensatz zu.

Parameter DATA für die Anweisung DataLogCreate

Der Parameter DATA verweist auf Speicher, der als temporärer Puffer für einen neuen Protokolldatensatz verwendet wird. Ihm muss eine M- oder DB-Adresse zugewiesen sein.

Zuweisen können Sie einen kompletten DB (abgeleitet von einem PLC-Datentyp, den Sie bei der Erstellung des DBs angeben) oder einen Teil eines DBs (bei dem angegebenen DB-Element kann es sich um einen beliebigen Datentyp, eine Datentypstruktur, einen PLC-Datentyp oder ein Daten-Array handeln).

Die Strukturdatentypen sind auf eine einzige Schachtelungsebene begrenzt. Die Gesamtzahl der deklarierten Datenelemente muss der Anzahl der im Parameter HEADER angegebenen Spalten entsprechen. Sie können maximal 253 Datenelemente (mit Zeitstempel) bzw. 255 Datenelemente (ohne Zeitstempel) zuweisen. Durch diese Einschränkung bleibt Ihr Datensatz innerhalb des Grenzwerts von maximal 256 Spalten eines Excel-Arbeitsblatts.

Der Parameter DATA kann entweder remanente oder nicht remanente Datenelemente in einem DB vom Typ "Standard" (kompatibel mit S7-300/400) oder "Optimiert" zuweisen.

Um einen Protokolldatensatz zu schreiben, müssen Sie zunächst neue Prozesswerte in den temporären Datensatz DATA laden und dann die Anweisung DataLogWrite ausführen, wodurch neue Datensatzwerte in der Datenprotokolldatei gespeichert werden.

Parameter HEADER für die Anweisung DataLogCreate

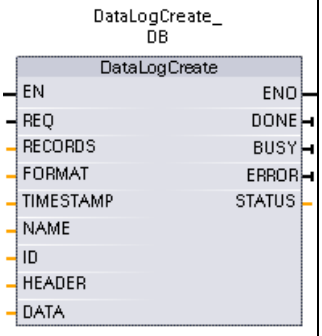
Der Parameter HEADER zeigt auf die Spaltenköpfe in der obersten Zeile der Datenmatrix in der CSV-Datei. HEADER-Daten müssen sich im DB- oder M-Speicher befinden und die Zeichen müssen den üblichen Formatregeln für CSV-Dateien entsprechen, wobei die einzelnen Spaltennamen durch Komma zu trennen sind. Bei dem Datentyp kann es sich um Strings, Byte-Arrays oder Zeichen-Arrays handeln. Zeichen- bzw. Byte-Arrays ermöglichen eine größere Größe, während der Datentyp String auf maximal 255 Bytes begrenzt ist. Der Parameter HEADER ist optional. Wenn der HEADER nicht zugewiesen ist, wird keine Kopfzeile in der Datenprotokolldatei angelegt.

9.7.2.2 Programmanweisungen zum Steuern von Datenprotokollen

Anweisung DataLogCreate (Datenprotokoll erstellen)

Tabelle 9- 161 Anweisung DataLogCreate

KOP/FUP	SCL	Beschreibung

	`"DataLogCreate_DB"(`
	`req:=_bool_in_,`
	`records:=_udint_in_,`
	`format:=_uint_in_,`
	`timestamp:=_uint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`name:=_string_inout_,`
	`ID:=_dword_inout_,`
	`header:=_variant_inout_,`
	`data:=_variant_inout_);`
		Erstellt und initialisiert eine Datenprotokolldatei. Die Datei wird im PLC-Verzeichnis \Datenprotokolle angelegt, trägt den im Parameter NAME angegebenen Namen und wird implizit für Schreibanweisungen geöffnet. Mit den Anweisungen Data log können Sie Laufzeit-Prozessdaten programmatisch im Flash-Speicher der CPU speichern.
		STEP 7 erstellt automatisch den zugehörigen Instanz-DB, wenn Sie die Anweisung einfügen.

[1] Im SCL-Beispiel ist "DataLogCreate_DB" der Name des Instanz-DBs.

Tabelle 9- 162 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet. (Standardwert: Falsch)
RECORDS	IN	UDint	Die maximale Anzahl Datensätze, die das kreisförmige Datenprotokoll enthalten kann, bevor der älteste Eintrag überschrieben wird: Der Datensatz mit der Kopfzeile ist in dieser Anzahl nicht enthalten. Es muss ausreichend PLC-Ladespeicher verfügbar sein, damit das Datenprotokoll erfolgreich angelegt wird. (Voreinstellung - 1)

Parameter und Datentyp		Datentyp	Beschreibung
FORMAT	IN	UInt	Format des Datenprotokolls: • 0 - Internes Format (nicht unterstützt) • 1 - Durch Komma getrennte Werte, "csv-eng" (Standardwert)
TIMESTAMP	IN	UInt	Format des Zeitstempels der Daten: Spaltenköpfe für Datum- und Uhrzeitfelder sind nicht erforderlich. Der Zeitstempel zeigt die Systemzeit (Coordinated Universal Time - UTC) und nicht die Lokalzeit. • 0 - Kein Zeitstempel • 1 - Datum und Zeitstempel (Standardwert)
NAME	IN	Variant	Name des Datenprotokolls: Sie geben einen Namen ein. Diese Variante unterstützt nur den Datentyp String und kann nur im lokalen Speicher, im DB- oder M-Speicher abgelegt werden. (Standardwert: ' ') Diese Zeichenkette wird auch für den Namen der Datenprotokolldatei verwendet. Die im Namen enthaltenen Zeichen müssen sich an die Namenseinschränkungen des Windows-Dateisystems halten. Die Zeichen \ / : * ? " < > | und das Leerzeichen sind nicht zulässig.
ID	Durchgang	DWord	Numerische Kennung des Datenprotokolls: Sie speichern diesen generierten Wert zur Verwendung mit anderen Datenprotokollanweisungen. Der Parameter ID wird bei der Anweisung DataLogCreate nur als Ausgang verwendet. (Standardwert: 0) Der Zugriff auf diesen Parameter über einen symbolischen Namen ist nicht zulässig.
HEADER	Durchgang	Variant	Pointer auf die Spaltenköpfe des Datenprotokolls in der obersten Zeile der Datenmatrix in der CSV-Datei. (Standardwert: null). HEADER-Daten müssen sich im DB- oder M-Speicher befinden. Die Zeichen müssen den üblichen Formatregeln für CSV-Dateien entsprechen, und die einzelnen Spaltennamen sind durch Komma zu trennen. Bei dem Datentyp kann es sich um Strings, Byte-Arrays oder Zeichen-Arrays handeln. Zeichen- bzw. Byte-Arrays ermöglichen eine größere Größe, während der Datentyp String auf maximal 255 Bytes begrenzt ist. Der Parameter HEADER ist optional. Wenn der HEADER nicht parametriert ist, wird keine Kopfzeile in der Datenprotokolldatei angelegt.

Parameter und Datentyp		Datentyp	Beschreibung
DATA	Durchgang	Variant	Pointer auf die Datenstruktur des Datensatzes, anwenderdefinierter Datentyp (UDT) oder Array. Datensatzdaten müssen sich im DB- oder M-Speicher befinden. Der Parameter DATA gibt die einzelnen Datenelemente (Spalten) eines Protokolldatensatzes und deren Datentyp an. Die Strukturdatentypen sind auf eine einzige Schachtelungsebene begrenzt. Die Anzahl der deklarierten Datenelemente muss der Anzahl der im Parameter HEADER angegebenen Spalten entsprechen. Sie können maximal 253 Datenelemente (mit Zeitstempel) bzw. 255 Datenelemente (ohne Zeitstempel) zuweisen. Durch diese Einschränkung bleibt Ihr Datensatz innerhalb des Grenzwerts von maximal 256 Spalten eines Excel-Arbeitsblatts.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde. (Standardwert: Falsch)
BUSY	OUT	Bool	• 0 - Keine Anweisung in Bearbeitung • 1 - Anweisung in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Es wird eine Datenprotokolldatei mit einer anhand der Parameter RECORDS und DATA fest vorgegebenen Größe angelegt. Die Datensätze sind als kreisförmige Protokolldatei aufgebaut. Neue Datensätze werden in die Datenprotokolldatei eingefügt, bis die maximale Anzahl Datensätze, die im Parameter RECORDS vorgegeben ist, gespeichert ist. Danach überschreibt der nächste Datensatz, der geschrieben wird, den ältesten Datensatz. Eine weitere Schreibanweisung überschreibt den zweitältesten Datensatz usw.

Speicherauslastung:

- Die Datenprotokolle belegen nur Ladespeicher.
- Es gibt keinen festen Grenzwert für die Gesamtzahl der Datenprotokolle. Die Größe aller Datenprotokolle insgesamt ist durch die verfügbare Kapazität des Ladespeichers begrenzt. Maximal acht Protokolle dürfen gleichzeitig geöffnet sein.
- Die maximal mögliche Anzahl für den Parameter RECORDS ist der Grenzwert einer UDint-Zahl (4.294.967.295). Der tatsächliche Grenzwert für den Parameter RECORD richtet sich nach der Größe eines einzelnen Datensatzes, der Größe anderer Datenprotokolle und der verfügbaren Kapazität des Ladespeichers. Zudem ist in Excel die Anzahl der Zeilen in einem Excel-Arbeitsblatt begrenzt.

Hinweis

Die Datenprotokollerstellung muss abgeschlossen sein, bevor eine Anweisung zum Schreiben des Datenprotokolls angestoßen werden kann

- Die Operationen DataLogCreate und DataLogNewFile zum Erstellen von Datenprotokollen erstrecken sich über viele Programmzyklen. Die tatsächliche für die Erstellung der Protokolldatei benötigte Zeit hängt von der Datensatzstruktur und der Anzahl der Datensätze ab. Ihre Programmlogik muss den Wechsel des DONE-Bits nach WAHR, der die abgeschlossene Erstellung einer Protokolldatei meldet, überwachen und erfassen. Wird eine Anweisung DataLogWrite vor Abschluss der Datenprotokollerstellung ausgeführt, kann die Schreiboperation den neuen Protokolldatensatz nicht wie erwartet schreiben.
- In bestimmten Situationen, wenn ein sehr schneller Zyklus abläuft, kann die Erstellung des Datenprotokolls länger dauern. Wenn die langsame Erstellung zu lange dauert, müssen Sie sicherstellen, dass das Kontrollkästchen zum Aktivieren der Mindestzykluszeit für zyklische OBs aktiviert und die Mindestzykluszeit auf 1 ms oder mehr gesetzt ist. Weitere Informationen finden Sie unter Konfigurieren von Zykluszeit und Kommunikationslast (Seite 104).

Hinweis

Die Anweisung DataLogNewFile kopiert die Datensatzstruktur eines vorhandenen Datenprotokolls

Wenn Sie das Überschreiben von Datensätzen verhindern möchten, können Sie mit der Anweisung DataLogNewFile ein neues Datenprotokoll basierend auf dem aktuellen Datenprotokoll anlegen, nachdem das aktuelle Datenprotokoll die maximale Anzahl von Datensätzen gespeichert hat. Neue Datensätze werden dann in der neuen Datenprotokolldatei gespeichert. Die alte Datenprotokolldatei und die Datensatzdaten bleiben im Flash-Speicher erhalten.

Tabelle 9- 163 Werte von ERROR und STATUS

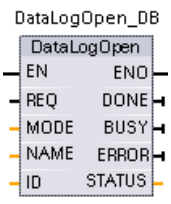
ERROR	STATUS (W#16#....)	Beschreibung
0	0000	Kein Fehler
0	7000	Aufruf ohne REQ-Flanke: BUSY = 0, DONE = 0
0	7001	Erster Aufruf mit REQ-Flanke (in Bearbeitung): BUSY = 1, DONE = 0
0	7002	N^{ter} Aufruf (in Bearbeitung): BUSY = 1, DONE = 0
1	8070	Gesamter interner Instanzspeicher ist belegt.
1	807F	Interner Fehler
1	8090	Ungültiger Dateiname
1	8091	Der Namensparameter ist keine Zeichenkette.
1	8093	Es ist bereits ein Datenprotokoll mit dem Namen vorhanden. Verwenden Sie einen anderen Namen, vergewissern Sie sich, dass die CSV-Datei des Datenprotokolls nicht geöffnet ist, und löschen Sie dann das vorhandene Datenprotokoll über die Dateibrowser-Seite (Seite 844) des Webservers.

ERROR	STATUS (W#16#....)	Beschreibung
1	8097	Gewünschte Dateilänge überschreitet maximale Dateigröße für das Dateisystem.
1	80B3	Nicht genügend Ladespeicher.
1	80B4	MC (Speichermodul) ist schreibgeschützt.
1	80C1	Zu viele geöffnete Dateien: Maximal acht Datenprotokolldateien dürfen gleichzeitig geöffnet sein.
1	8253	Ungültige Anzahl Datensätze
1	8353	Ungültige Formatauswahl
1	8453	Ungültige Zeitstempelauswahl
1	8B24	Ungültige HEADER-Bereichszuweisung: Beispiel: Verweist auf lokalen Speicher
1	8B51	Ungültiger Datentyp für Parameter HEADER
1	8B52	Zu viele Datenelemente im Parameter HEADER
1	8C24	Ungültige DATA-Bereichszuweisung: Beispiel: Zeigt auf lokalen Speicher
1	8C51	Ungültiger Datentyp für Parameter DATA
1	8C52	Zu viele Datenelemente im Parameter DATA

Anweisung DataLogOpen (Datenprotokoll öffnen)

Tabelle 9- 164 Anweisung DataLogOpen

KOP/FUP	SCL	Beschreibung

	`"DataLogOpen_DB"(`
	`req:=_bool_in_,`
	`mode:=_uint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`name:=_string_inout_,`
	`ID:=_dword_inout_);`
		Öffnet eine bereits vorhandene Datenprotokolldatei. Ein Datenprotokoll muss geöffnet sein, damit Sie neue Datensätze in das Protokoll schreiben können. Datenprotokolle lassen sich einzeln öffnen und schließen. Maximal acht Datenprotokolle können gleichzeitig geöffnet sein.
		Wenn Sie die Anweisung einfügen, erstellt STEP 7 automatisch den zugehörigen Instanz-DB.

[2] Im SCL-Beispiel ist "DataLogOpen_DB" der Name des Instanz-DBs.

Tabelle 9- 165 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet. (Standardwert: Falsch)
MODE	IN	UInt	Betriebsart: • 0 - An vorhandene Daten anhängen (Standardwert) • 1 - Alle vorhandenen Datensätze löschen
NAME	IN	Variant	Name eines vorhandenen Datenprotokolls: Diese Variante unterstützt nur den Datentyp String und kann nur im lokalen Speicher, im DB- oder M-Speicher abgelegt werden. (Standardwert: ' ')

Parameter und Datentyp		Datentyp	Beschreibung
ID	Durchgang	DWord	Numerische Kennung eines Datenprotokolls. (Standardwert: 0) **Hinweis:** Der Zugriff auf diesen Parameter über einen symbolischen Namen ist nicht zulässig.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde. (Standardwert: Falsch)
BUSY	OUT	Bool	• 0 - Keine Anweisung in Bearbeitung • 1 - Anweisung in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Sie können entweder den NAMEN oder eine ID (Parameter ID als Eingang) eines bereits vorhandenen Datenprotokolls eingeben. Wenn Sie beide Parameter angeben, und eine gültige ID entspricht dem Datenprotokoll NAME, dann wird die ID verwendet und der NAME wird ignoriert.

Der NAME muss dem Namen eines mit der Anweisung DataLogCreate angelegten Datenprotokolls entsprechen. Wird nur der NAME angegeben, und der NAME verweist auf ein gültiges Datenprotokoll, dann wird die entsprechende ID ausgegeben (Parameter ID als Ausgang).

Hinweis

Allgemeine Verwendung von Datenprotokolldateien

- Nach Ausführung der Anweisungen DataLogCreate und DataLogNewFile werden automatisch Datenprotokolldateien geöffnet.
- Nach einem Wechsel des PLC-Geräts von RUN in STOP oder nach einem Neustart des PLC-Geräts werden Datenprotokolldateien automatisch geschlossen.
- Damit eine neue Anweisung DataLogWrite durchgeführt werden kann, muss eine Datenprotokolldatei geöffnet sein.
- Maximal acht Datenprotokolldateien dürfen gleichzeitig geöffnet sein. Mehr als acht Datenprotokolldateien dürfen vorhanden sein, von diesen müssen jedoch einige geschlossen werden, so dass maximal acht gleichzeitig geöffnet sind.

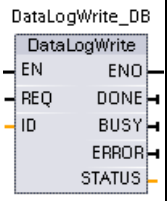
Tabelle 9- 166 Werte von ERROR und STATUS

ERROR	STATUS (W#16#)	Beschreibung
0	0000	Kein Fehler
0	0002	Warnung: Datenprotokolldatei wurde von diesem Anwendungsprogramm bereits geöffnet
0	7000	Aufruf ohne REQ-Flanke: BUSY = 0, DONE = 0
0	7001	Erster Aufruf mit REQ-Flanke (in Bearbeitung): BUSY = 1, DONE = 0
0	7002	N^{ter} Aufruf (in Bearbeitung): BUSY = 1, DONE = 0
1	8070	Gesamter interner Instanzspeicher ist belegt.
1	8090	Datenprotokolldefinition stimmt nicht mit der vorhandenen Datenprotokolldatei überein.
1	8091	Der Namensparameter ist keine Zeichenkette.
1	8092	Datenprotokoll ist nicht vorhanden.
1	80C0	Datenprotokolldatei ist gesperrt.
1	80C1	Zu viele geöffnete Dateien: Maximal acht Datenprotokolldateien dürfen gleichzeitig geöffnet sein.

Anweisung DataLogWrite (Datenprotokoll schreiben)

Tabelle 9- 167 Anweisung DataLogWrite

KOP/FUP	SCL	Beschreibung

	`"DataLogWrite_DB"(`
	`req:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`ID:=_dword_inout_);`
		Schreibt einen Datensatz in das angegebene Datenprotokoll. Das bereits vorhandene Zieldatenprotokoll muss geöffnet sein, damit eine Anweisung DataLogWrite durchgeführt werden kann.
		Wenn Sie die Anweisung einfügen, erstellt STEP 7 automatisch den zugehörigen Instanz-DB.

[2] Im SCL-Beispiel ist "DataLogWrite_DB" der Name des Instanz-DBs.

Tabelle 9- 168 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet. (Standardwert: Falsch)
ID	Durchgang	DWord	Numerische Kennung des Datenprotokolls: Wird bei der Anweisung DataLogWrite nur als Eingang verwendet. (Standardwert: 0) **Hinweis:** Der Zugriff auf diesen Parameter über einen symbolischen Namen ist nicht zulässig.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
BUSY	OUT	Bool	• 0 - Keine Anweisung in Bearbeitung • 1 - Anweisung in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Die Speicheradresse und die Datenstruktur des Datensatzpuffers werden vom Parameter DATA der Anweisung DataLogCreate konfiguriert. Um neue Datensatzdaten aus dem Puffer in ein Datenprotokoll zu übertragen, müssen Sie den Datensatzpuffer programmatisch mit aktuellen Laufzeitprozesswerten laden und dann die Anweisung DataLogWrite ausführen.

Der Parameter ID gibt eine Datenprotokoll- und Datensatzkonfiguration an. Die ID-Nummer wird beim Anlegen eines Datenprotokolls generiert.

Enthält die kreisförmige Datenprotokolldatei leere Datensätze, wird in den nächsten freien Datensatz geschrieben. Sind alle Datensätze belegt, wird der älteste Datensatz überschrieben.

ACHTUNG
Die Datenprotokollerstellung muss abgeschlossen sein, bevor eine Anweisung zum Schreiben des Datenprotokolls angestoßen werden kann Die Operationen DataLogCreate und DataLogNewFile zum Erstellen von Datenprotokollen erstrecken sich über viele Programmzyklen. Die tatsächliche für die Erstellung der Protokolldatei benötigte Zeit hängt von der Datensatzstruktur und der Anzahl der Datensätze ab. Ihre Programmlogik muss den Wechsel des DONE-Bits nach WAHR, der die abgeschlossene Erstellung einer Protokolldatei meldet, überwachen und erfassen. Wird eine Anweisung DataLogWrite vor Abschluss der Datenprotokollerstellung ausgeführt, kann die Schreiboperation den neuen Protokolldatensatz nicht wie erwartet schreiben.

Hinweis

Auswirkung von Datenprotokollen auf den internen CPU-Speicher

Jeder Schreibvorgang eines Datenprotokolls verbraucht mindestens 2 KB Speicher. Wenn Ihr Programm häufig kleinere Mengen von Daten schreibt, werden bei jedem Schreibvorgang mindestens 2 KB Speicher verbraucht. Eine bessere Umsetzung wäre die Ansammlung kleiner Datenelemente in einem Datenbaustein (DB), der dann weniger häufig ins Datenprotokoll geschrieben würde.

Wenn Ihr Programm sehr häufig viele Datenprotokolleinträge schreibt, können Sie auch eine austauschbare SD Memory Card verwenden.

ACHTUNG
Möglicher Protokolldatenverlust während eines CPU-Spannungsausfalls Tritt während einer noch nicht beendeten Anweisung DataLogWrite ein Spannungsausfall auf, geht der ins Datenprotokoll übertragene Datensatz möglicherweise verloren.

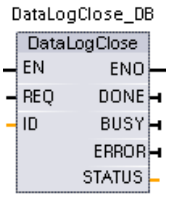
Tabelle 9- 169 Werte von ERROR und STATUS

ERROR	STATUS (W#16#)	Beschreibung
0	0000	Kein Fehler
0	0001	Zeigt an, dass das Datenprotokoll voll ist: Jedes Datenprotokoll wird mit einer vorgegebenen Maximalanzahl von Datensätzen angelegt. Der letzte Datensatz dieser Maximalanzahl wurde geschrieben. Die nächste Schreibanweisung überschreibt den ältesten Datensatz.
0	7000	Aufruf ohne REQ-Flanke: BUSY = 0, DONE = 0
0	7001	Erster Aufruf mit REQ-Flanke (in Bearbeitung): BUSY = 1, DONE = 0
0	7002	N^{ter} Aufruf (in Bearbeitung): BUSY = 1, DONE = 0
1	8070	Gesamter interner Instanzspeicher ist belegt.
1	8092	Datenprotokoll ist nicht vorhanden.
1	80B0	Datenprotokolldatei ist nicht geöffnet (nur beim Modus mit explizitem Öffnen).

Anweisung DataLogClose (Datenprotokoll schließen)

Tabelle 9- 170 Anweisung DataLogClose

KOP/FUP	SCL	Beschreibung

	`"DataLogClose_DB"(`
	`req:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`ID:=_dword_inout_);`
		Schließt eine geöffnete Datenprotokolldatei. Eine für ein geschlossenes Datenprotokoll ausgeführte Anweisung DataLogWrite führt zu einem Fehler. Schreibanweisungen für dieses Datenprotokoll sind erst zulässig, nachdem eine Anweisung DataLogOpen ausgeführt wurde.
		Beim Wechsel in den Betriebszustand STOP werden alle geöffneten Datenprotokolldateien geschlossen.
		Wenn Sie die Anweisung einfügen, erstellt STEP 7 automatisch den zugehörigen Instanz-DB.

[2] Im SCL-Beispiel ist "DataLogClose_DB" der Name des Instanz-DBs.

Tabelle 9- 171 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet. (Standardwert: Falsch)
ID	Durchgang	DWord	Numerische Kennung eines Datenprotokolls. Wird bei der Anweisung DataLogClose nur als Eingang verwendet. (Standardwert: 0) **Hinweis:** Der Zugriff auf diesen Parameter über einen symbolischen Namen ist nicht zulässig.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
BUSY	OUT	Bool	• 0 - Keine Anweisung in Bearbeitung • 1- Anweisung in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

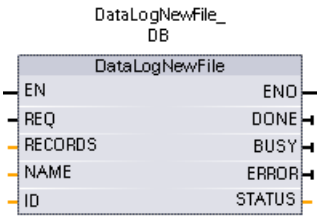
Tabelle 9- 172 Werte von ERROR und STATUS

ERROR	STATUS (W#16#)	Beschreibung
0	0000	Kein Fehler
0	0001	Datenprotokoll nicht geöffnet
0	7000	Aufruf ohne REQ-Flanke: BUSY = 0, DONE = 0
0	7001	Erster Aufruf mit REQ-Flanke (in Bearbeitung): BUSY = 1, DONE = 0
0	7002	N^{ter} Aufruf (in Bearbeitung): BUSY = 1, DONE = 0
1	8092	Datenprotokoll ist nicht vorhanden.

Anweisung DataLogNewFile (Datenprotokoll in neuer Datei)

Tabelle 9- 173 Anweisung DataLogNewFile

KOP/FUP	SCL	Beschreibung

	`"DataLogNewFile_DB"(`
	`req:=_bool_in_,`
	`records:=_udint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`name:=_DataLog_out_,`
	`ID:=_dword_inout_);`
		Ermöglicht Ihrem Programm, eine neue Datenprotokolldatei basierend auf einer bestehenden Datenprotokolldatei anzulegen.
		Wenn Sie die Anweisung einfügen, erstellt STEP 7 automatisch den zugehörigen Instanz-DB.

[2] Im SCL-Beispiel ist "DataLogNewFile_DB" der Name des Instanz-DBs.

Tabelle 9- 174 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet. (Standardwert: Falsch)
RECORDS	IN	UDInt	Die maximale Anzahl Datensätze, die das kreisförmige Datenprotokoll enthalten kann, bevor der älteste Eintrag überschrieben wird. (Standardwert: 1) Der Datensatz mit der Kopfzeile ist in dieser Anzahl nicht enthalten. Es muss ausreichend CPU-Ladespeicher verfügbar sein, damit das Datenprotokoll erfolgreich angelegt wird.
NAME	IN	Variant	Name des Datenprotokolls: Sie geben einen Namen ein. Diese Variante unterstützt nur den Datentyp String und kann nur im lokalen Speicher, im DB- oder M-Speicher abgelegt werden. (Standardwert: ' ') Diese Zeichenkette wird auch für den Namen der Datenprotokolldatei verwendet. Die im Namen enthaltenen Zeichen müssen sich an die Namenseinschränkungen des Windows-Dateisystems halten. Die Zeichen \ / : * ? " < > | und das Leerzeichen sind nicht zulässig.)
ID	Durchgang	DWord	Numerische Kennung des Datenprotokolls (Standardwert: 0): • Während der Ausführung gibt der Eingang ID ein gültiges Datenprotokoll an. Die neue Datenprotokollkonfiguration wird aus diesem Datenprotokoll kopiert. • Nach der Ausführung wird der Parameter ID zu einem Ausgang, der die ID der neu angelegten Datenprotokolldatei ausgibt. **Hinweis:** Der Zugriff auf diesen Parameter über einen symbolischen Namen ist nicht zulässig.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
BUSY	OUT	Bool	• 0 - Keine Anweisung in Bearbeitung • 1 - Anweisung in Bearbeitung

Parameter und Datentyp		Datentyp	Beschreibung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Sie können die Anweisung DataLogNewFile ausführen, wenn ein Datenprotokoll voll ist oder nahezu voll ist und Sie keine Daten verlieren möchten, die im Datenprotokoll gespeichert sind. Basierend auf der Struktur der vollen Datenprotokolldatei kann eine neue, leere Datenprotokolldatei angelegt werden. Der Kopfdatensatz wird mit den Eigenschaften des ursprünglichen Datenprotokolls (DATA-Einstellungen für Datensatzpuffer, Datenformat und Zeitstempel) aus dem ursprünglichen Datenprotokoll dupliziert. Die ursprüngliche Datenprotokolldatei wird implizit geschlossen und die neue Datenprotokolldatei implizit geöffnet.

Auslösung von DataLogWrite durch Parameter: Ihr Programm muss die Parameter ERROR und STATUS jeder Anweisung DataLogWrite überwachen. Ist der letzte Datensatz geschrieben und das Datenprotokoll voll, werden das DataLogWrite ERROR-Bit und das DataLogWrite STATUS-Wort auf 1 gesetzt. Diese ERROR- und STATUS-Werte sind nur einen Zyklus lang gültig. Deshalb muss Ihre Überwachungslogik ERROR = 1 als Zeitfenster nutzen, um den STATUS-Wert zu erfassen und dann auf STATUS = 1 (Datenprotokoll ist voll) zu prüfen.

Funktionsweise von DataLogNewFile: Wenn Ihre Programmlogik das Signal "Datenprotokoll voll" erhält, wird mit diesem Zustand eine Anweisung DataLogNewFile aktiviert. Sie müssen Anweisung DataLogNewFile mit der ID eines vorhandenen (üblicherweise vollen) und geöffneten Datenprotokolls ausführen, der Parameter NAME muss jedoch neu und eindeutig sein. Nach Ausführung der Anweisung DataLogNewFile wird ein neuer ID-Wert für das Datenprotokoll ausgegeben (als Ausgangsparameter), der dem Namen des neuen Datenprotokolls entspricht. Die neue Datenprotokolldatei wird implizit geöffnet und kann neue Datensätze speichern. Neue Anweisungen DataLogWrite, die sich an die neue Datenprotokolldatei richten, müssen den von der Anweisung DataLogNewFile ausgegebenen ID-Wert nutzen.

ACHTUNG

Die Datenprotokollerstellung muss abgeschlossen sein, bevor eine Anweisung zum Schreiben des Datenprotokolls angestoßen werden kann

Die Operationen DataLogCreate und DataLogNewFile zum Erstellen von Datenprotokollen erstrecken sich über viele Programmzyklen. Die tatsächliche für die Erstellung der Protokolldatei benötigte Zeit hängt von der Datensatzstruktur und der Anzahl der Datensätze ab. Ihre Programmlogik muss den Wechsel des DONE-Bits nach WAHR, der die abgeschlossene Erstellung einer Protokolldatei meldet, überwachen und erfassen. Wird eine Anweisung DataLogWrite vor Abschluss der Datenprotokollerstellung ausgeführt, kann die Schreiboperation den neuen Protokolldatensatz nicht wie erwartet schreiben.

Tabelle 9- 175 Werte von ERROR und STATUS

ERROR	STATUS (W#16#)	Beschreibung
0	0000	Kein Fehler
0	7000	Aufruf ohne REQ-Flanke: BUSY = 0, DONE = 0
0	7001	Erster Aufruf mit REQ-Flanke (in Bearbeitung): BUSY = 1, DONE = 0
0	7002	N^{ter} Aufruf (in Bearbeitung): BUSY = 1, DONE = 0
1	8070	Gesamter interner Instanzspeicher ist belegt.
1	8090	Ungültiger Dateiname
1	8091	Der Namensparameter ist keine Zeichenkette.
1	8092	Datenprotokoll ist nicht vorhanden.
1	8093	Das Datenprotokoll ist bereits vorhanden.
1	8097	Gewünschte Dateilänge überschreitet maximale Dateigröße für das Dateisystem.
1	80B3	Nicht genügend Ladespeicher.
1	80B4	MC ist schreibgeschützt.
1	80C1	Zu viele geöffnete Dateien.

9.7.2.3 Arbeiten mit Datenprotokollen

Die Datenprotokolldateien werden im beständigen Flash-Speicher im CSV-Format (durch Komma getrennte Werte) gespeichert. Sie können die Datenprotokolle über den PLC-Webserver anzeigen oder indem Sie die Memory Card des PLC-Geräts entnehmen und in einen herkömmlichen PC-Kartenleser einlegen.

Datenprotokolle über den PLC-Webserver anzeigen

Wenn der PROFINET-Anschluss des PLC-Geräts und ein PC an ein Netzwerk angeschlossen sind, können Sie einen PC-Webbrowser wie Microsoft Internet Explorer oder Mozilla Firefox nutzen, um auf den integrierten PLC-Webserver zuzugreifen. Das PLC-Gerät kann sich, während Sie den PLC-Webserver bedienen, in der Betriebsart RUN oder STOP befinden. Befindet sich das PLC-Gerät im Betriebszustand RUN, wird Ihr Steuerungsprogramm weiterhin ausgeführt, während der PLC-Webserver über das Netzwerk Protokolldaten überträgt.

Zugriff auf den Webserver:

1. Aktivieren Sie den Webserver in der Gerätekonfiguration der Ziel-CPU (Seite 819).
2. Verbinden Sie Ihren PC über das PROFINET-Netzwerk mit dem Zielsystem (Seite 823).
3. Greifen Sie über den integrierten Webserver auf die CPU zu (Seite 827).
4. Laden, bearbeiten und löschen Sie Datenprotokolldateien mit der Standard-Webseite "Dateibrowser" (Seite 844).
5. Öffnen Sie die .csv-Datei mit einem Tabellenkalkulationsprogramm wie Excel.

Hinweis

Verwaltung von Datenprotokollen

Speichern Sie maximal 1000 Datenprotokolle in einem Dateisystem. Bei Überschreiten dieser Anzahl kann es passieren, dass der Webserver nicht mehr genügend CPU-Ressourcen zur Verfügung hat, um die Datenprotokolle anzuzeigen.

Wenn Sie bemerken, dass die Webseite des Dateibrowsers die Datenprotokolle nicht anzeigen kann, müssen Sie die CPU in den Betriebszustand STOP versetzen, um Datenprotokolle anzeigen und löschen zu können.

Stellen Sie über die Verwaltung Ihrer Datenprotolle sicher, dass Sie lediglich die erforderliche Anzahl Datenprotokolle speichern und die maximale Anzahl von 1000 Datenprotokollen nicht überschreiten.

Datenprotokolle auf einer Memory Card des PLC-Geräts anzeigen

Ist in der S7-1200 CPU eine Memory Card vom Typ "Programmkarte" gesteckt, können Sie die Memory Card entnehmen und in einen Standardkartensteckplatz wie SD (Secure Digital) oder MMC (MultiMediaCard) an einem PC oder PG einfügen. Das PLC-Gerät ist beim Entnehmen der Memory Card im Betriebszustand STOP und Ihr Steuerungsprogramm wird nicht ausgeführt.

Öffnen Sie dann den Windows Explorer für Dateien und navigieren Sie zum Verzeichnis \Datenprotokoll auf der Memory Card. In diesem Verzeichnis befinden sich alle Ihre CSV-Datenprotokolldateien.

Kopieren Sie die Datenprotokolldateien in ein lokales Laufwerk auf Ihrem PC. Dann können Sie eine lokale Kopie einer CSV-Datei mit Excel öffnen; dies ist nicht die ursprüngliche Datei, die auf der Memory Card gespeichert ist.

ACHTUNG
Sie können Datenprotokolldateien über einen PC-Kartenleser von einer S7-1200 Memory Card kopieren, jedoch noch ändern oder löschen. Es ist empfehlenswert, den Standard-Dateibrowser des Webservers zu nutzen, um Datenprotokolldateien anzuzeigen, herunterzuladen (zu kopieren) und zu löschen. Das direkte Aufrufen des Dateisystems der Memory Card im Windows Explorer birgt das Risiko, dass Sie versehentlich Datenprotokoll- oder andere Systemdateien löschen oder ändern können, so dass möglicherweise eine Datei beschädigt oder die Memory Card unbrauchbar wird.

ACHTUNG
Auswirkung von Datenprotokollen auf Memory Cards Begrenzen Sie die Datenprotokollrate auf einen Wert von maximal alle 200 ms, um das Gesamtverhalten und die Robustheit Ihres Systems zu gewährleisten.

9.7.2.4 Grenzwert für die Größe von Datenprotokolldateien

Datenprotokolldateien nutzen den Ladespeicher des PLC-Geräts gemeinsam mit dem Programm, mit Programmdaten, Konfigurationsdaten, benutzerdefinierten Webseiten und PLC-Systemdaten. Ein großes Programm, das den internen Ladespeicher nutzt, belegt viel Platz im Ladespeicher. Möglicherweise steht dadurch nicht genügend freier Speicherplatz für Datenprotokolldateien zur Verfügung. In diesem Fall können Sie eine Memory Card als "Programmkarte" (Seite 142) nutzen, um die Kapazität des Ladespeichers zu erhöhen. S7-1200 CPUs können entweder internen oder externen Ladespeicher nutzen, jedoch nicht beide gleichzeitig.

Regel für die maximale Größe von Datenprotokolldateien

Die maximale Größe einer Datenprotokolldatei darf die Größe des freien Ladespeichers bzw. 500 Megabyte (je nachdem, was kleiner ist) nicht überschreiten. Die Größe von 500 Megabyte bezieht sich in diesem Fall auf die Dezimaldefinition von Megabyte, sodass die maximale Größe von Datenprotokolldateien 500.000.000 Byte bzw. 500 x 1000^2 Byte beträgt.

Tabelle 9- 176 Größe des Ladespeichers

Datenbereich	CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C, CPU 1217C	Datenspeicherung
Interner Ladespeicher Flash-Speicher	1 MB	1 MB	4 MB	4 MB	Anwenderprogramm und Programmdaten, Konfigurationsdaten, Datenprotokolle, benutzerdefinierte Webseiten und PLC-Systemdaten
Externer Ladespeicher Optionale Memory Card als "Programmkarte"	4 MB, 12 MB, 24 MB, 256 MB, 2 GB oder 32 GB, je nach Größe der SD-Karte				

Ermitteln des freien Ladespeichers

Die Kapazität des freien Ladespeichers variiert im Normalbetrieb, weil das Betriebssystem ständig Speicher benötigt und wieder freigibt. Um die Größe des Ladespeichers anzuzeigen, gehen Sie wie folgt vor.

1. Stellen Sie eine Online-Verbindung zwischen STEP 7 und dem S7-1200 Zielsystem her.
2. Laden Sie das Programm, das Ihre Datenprotokollanweisungen steuert.
3. Erstellen Sie optional nach Bedarf anwenderdefinierte Webseiten. Die Standard-Webseiten für den Zugriff auf Datenprotokolle sind in der Firmware des PLC-Geräts gespeichert und belegen keinen Platz im Ladespeicher.
4. Fragen Sie mit den Online- und Diagnosefunktionen die Gesamtgröße des Ladespeichers und den freien Speicherplatz ab (Seite 1115).

Berechnen der Größe einer Datenprotokolldatei (alle Datensätze)

Beim Anlegen der Datenprotokolldatei wird die maximale Speichergröße zugeordnet. Neben der erforderlichen Größe für alle Datensätze müssen Sie den Speicherplatz für einen Datenprotokoll-Header (sofern verwendet), Zeitstempel-Header (sofern verwendet), Datensatzindex-Header und die Mindestbausteingröße für die Speicherzuordnung berücksichtigen.

Ermitteln Sie die Größe Ihrer Datenprotokolldateien mithilfe der folgenden Formel und achten Sie darauf, dass Sie die Regel für die maximale Größe einhalten.

Datenbytes des Datenprotokolls = ((Datenbytes in einem Datensatz + Zeitstempel-Bytes + 12 Bytes) * Anzahl der Datensätze)

Header

Headerbytes des Datenprotokolls = Header-Zeichenbytes + 2 Byte

Header-Zeichenbytes

- Kein Daten-Header und keine Zeitstempel = 7 Byte
- Kein Daten-Header und Zeitstempel (mit Zeitstempel-Header) = 21 Byte
- Daten-Header und keine Zeitstempel = Anzahl der Zeichenbytes in allen Spaltenüberschriften einschließlich Trennzeichen-Kommas
- Daten-Header und Zeitstempel (mit Zeitstempel-Header) = Anzahl der Zeichenbytes in allen Spaltenüberschriften einschließlich Trennzeichen-Kommas + 21 Byte

Daten

Datenbytes des Datenprotokolls = ((Datenbytes in einem Datensatz + Zeitstempel-Bytes + 12 Bytes) * Anzahl der Datensätze)

Datenbytes in einem Datensatz

Der Parameter DATA der Anweisung DataLogCreate zeigt auf eine Struktur, die die Anzahl der Datenfelder und den Datentyp jedes Datenfelds für einen Protokolldatensatz zuweist.

Multiplizieren Sie die Anzahl des jeweiligen Datentyps mit der erforderlichen Anzahl Bytes für diesen Datentyp. Wiederholen Sie den Vorgang für jeden Datentyp in einem Datensatz und addieren Sie alle Datenbytes, um die Summe aller Datenelemente in einem Datensatz zu erhalten.

Größe der einzelnen Datenelemente

Die Daten in Datenprotokolldateien werden als Zeichenbytes im CSV-Format (durch Komma getrennte Werte) gespeichert. Die folgende Tabelle zeigt die Anzahl Bytes, die erforderlich sind, um jedes Datenelement zu speichern.

Datentyp	Anzahl der Bytes (einschließlich Daten plus ein Kommabyte)
Bool	2
Byte	5
Wort	7
DWord	12
Char	4
String	**Beispiel 1**: MyString[10] Die maximale Zeichenkettengröße wird mit 10 Zeichen angegeben. Textzeichen + automatische Auffüllung mit Leerzeichen = 10 Byte Anführungszeichen am Anfang und am Ende + Kommazeichen = 3 Byte 10 + 3 = 13 Byte insgesamt **Beispiel 2**: Mystring2 Wird keine Größe in eckigen Klammern angegeben, dann werden standardmäßig 254 Byte zugewiesen. Textzeichen + automatische Auffüllung mit Leerzeichen = 254 Byte Anführungszeichen am Anfang und am Ende + Kommazeichen = 3 Byte 254 + 3 = 257 Byte insgesamt
USInt	5
UInt	7
UDInt	12
SInt	5
Int	7
DInt	12
Real	16
LReal	25
Time	15
DTL	24

Anzahl der Datensätze in einer Datenprotokolldatei

Der Parameter RECORDS der Anweisung DataLogCreate legt die maximale Anzahl Datensätze in einer Datenprotokolldatei fest.

Zeitstempelbytes in einem Datensatz

- Kein Zeitstempel = 0 Byte
- Zeitstempel = 20 Byte

9.7.2.5 Beispielprogramm für Datenprotokolle

Dieses Beispielprogramm für Datenprotokolle zeigt nicht die gesamte Programmlogik, die erforderlich ist, um Abtastwerte eines dynamischen Prozesses abzurufen, es zeigt jedoch die wesentliche Funktionsweise der Anweisungen für Datenprotokolle. Die Struktur und Anzahl der verwendeten Protokolldateien richtet sich nach Ihren Prozesssteuerungsanforderungen.

Hinweis

Allgemeine Verwendung von Datenprotokolldateien

- Nach Ausführung der Anweisungen DataLogCreate und DataLogNew werden automatisch Datenprotokolldateien geöffnet.
- Nach einem Wechsel des PLC-Geräts von RUN in STOP oder nach einem Neustart des PLC-Geräts werden Datenprotokolldateien automatisch geschlossen.
- Damit eine Anweisung DataLogWrite durchgeführt werden kann, muss eine Datenprotokolldatei geöffnet sein.
- Maximal acht Datenprotokolldateien dürfen gleichzeitig geöffnet sein. Mehr als acht Datenprotokolldateien dürfen vorhanden sein, von diesen müssen jedoch einige geschlossen werden, so dass maximal acht gleichzeitig geöffnet sind.

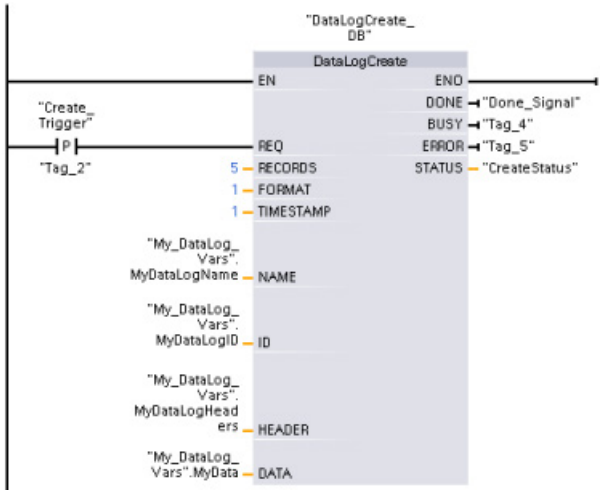
Beispielprogramm für Datenprotokolle

Beispielhafte Datenprotokollnamen, Kopfzeilentext und die Struktur MyData werden in einem Datenbaustein erstellt. Die drei MyData-Variablen speichern neue Abtastwerte temporär. Die Prozessabtastwerte an diesen DB-Adressen werden durch Ausführung der Anweisung DataLogWrite in eine Datenprotokolldatei übertragen.

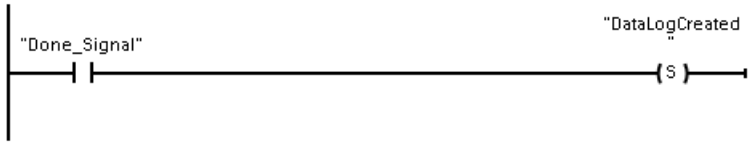
My_Datalog_Vars

	Name	Datentyp	Startwert
1	Static		
2	MyNewDataLogName	String	'MyNEWDataLog'
3	MyDataLogName	String	'MyDataLog'
4	MyDataLogID	DWord	0
5	MyDataLogHeaders	String	'Count,Temperature,Pressure'
6	MyData	Struct	
7	MyCount	Int	0
8	MyTemperature	Real	0.0
9	MyPressure	Real	0.0

Netzwerk 1 Eine steigende Flanke an REQ startet die Datenprotokollerstellung.

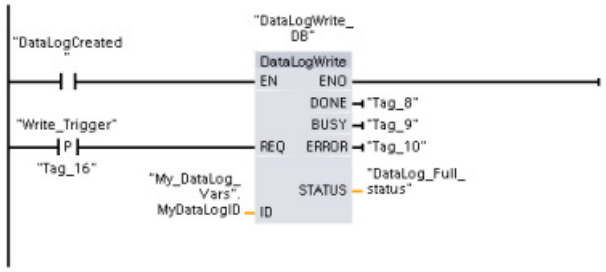


Netzwerk 2 Erfassen Sie den Ausgang DONE von DataLogCreate, weil er nur einen Zyklus lang gültig ist.

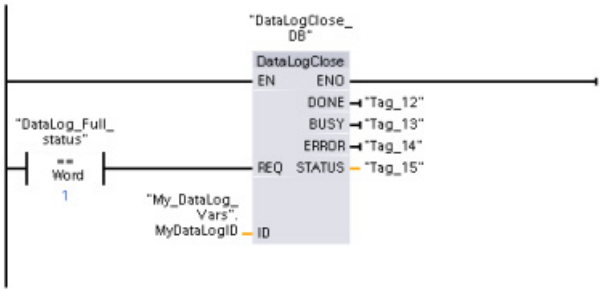


Netzwerk 3 Eine positive Flanke löst den Zeitpunkt aus, an dem neue Prozesswerte in der Struktur MyData gespeichert werden.

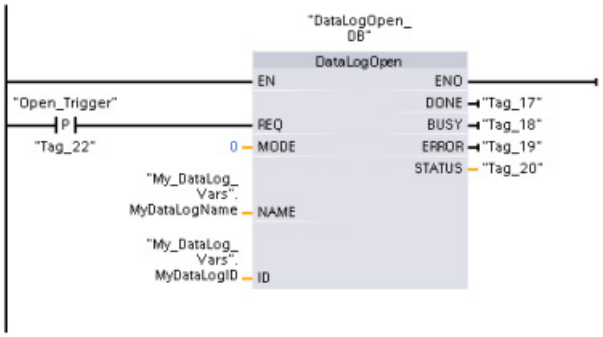
Netzwerk 4 Der Zustand des Eingangs EN basiert auf dem Zeitpunkt, zu dem die Ausführung von DataLogCreate beendet ist. Ein Erstellungsvorgang erstreckt sich über viele Zyklen und muss beendet sein, damit ein Schreibvorgang durchgeführt werden kann. Die positive Flanke am Eingang REQ ist das Ereignis, das eine aktivierte Schreiboperation auslöst.



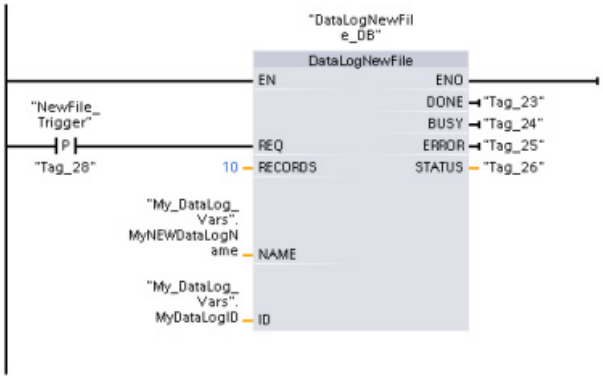
Netzwerk 5 Schließen Sie das Datenprotokoll, nachdem der letzte Datensatz geschrieben wurde. Nach Ausführung der Anweisung DataLogWrite, die den letzten Datensatz schreibt, wird der Zustand "Protokolldatei voll" gemeldet, wenn der Ausgang STATUS von DataLogWrite = 1 ist.



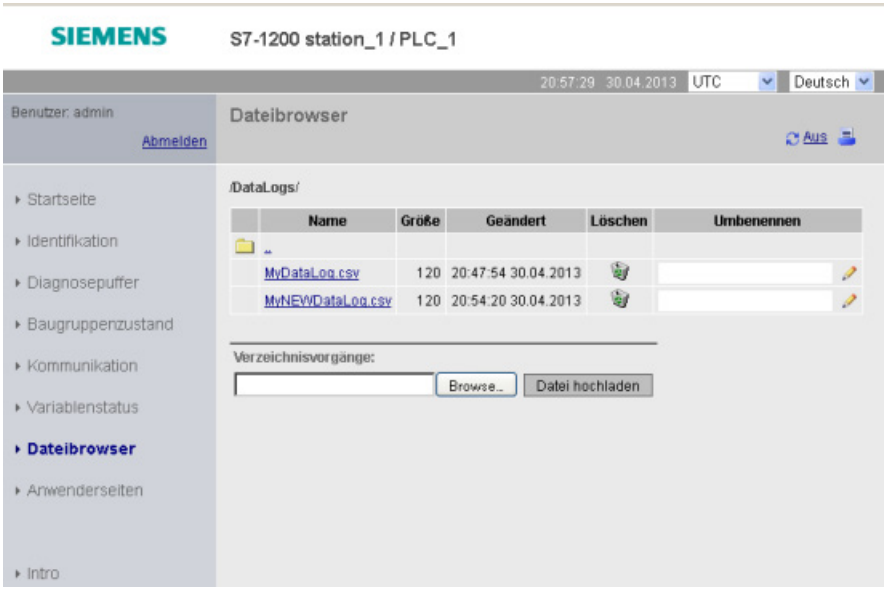
Netzwerk 6 Eine positive Flanke am Eingang REQ der Anweisung DataLogOpen simuliert, dass der Anwender an einem HMI-Gerät eine Taste drückt, die eine Datenprotokolldatei öffnet. Wenn Sie eine Datenprotokolldatei öffnen, in der alle Datensätze mit Prozessdaten belegt sind, dann überschreibt die nächste Ausführung der Anweisung DataLogWrite den ältesten Datensatz. Sie können jedoch auch das alte Datenprotokoll aufbewahren und stattdessen ein neues Datenprotokoll anlegen. Dies wird in Netzwerk 7 gezeigt.



Netzwerk 7 Der ID-Parameter ist ein IN/OUT-Typ. Zunächst geben Sie den ID-Wert des vorhandenen Datenprotokolls an, dessen Struktur Sie kopieren möchten. Nachdem die Anweisung DataLogNewFile durchgeführt wurde, wird ein neuer und eindeutiger ID-Wert für das neue Datenprotokoll in die Adresse des ID-Verweises zurückgeschrieben. Die erforderliche Erfassung ONE-Bit = WAHR wird nicht gezeigt. Ein Beispiel für die Logik des DONE-Bits finden Sie in den Netzwerken 1, 2 und 4.



Vom Beispielprogramm angelegte und mit dem Webserver der S7-1200 CPU angezeigte Datenprotokolldateien



① Die Option "Delete" ist nur verfügbar, wenn Sie mit Änderungsrechten angemeldet sind.

② Die Option "Rename" ist nur verfügbar, wenn Sie mit Änderungsrechten angemeldet sind.

Tabelle 9- 177 In Excel angezeigte, heruntergeladene Beispiele für CSV-Dateien

Zwei geschriebene Datensätze in einer Datei, die maximal fünf Datensätze enthalten kann	
Fünf Datensätze in einer Datenprotokolldatei, die maximal fünf Datensätze enthalten kann	
Nachdem ein weiterer Datensatz in die oben abgebildete Datei, die bereits voll ist, geschrieben wurde, überschreibt der sechste Schreibvorgang den ältesten Datensatz 1 mit Datensatz 6. Ein weiterer Schreibvorgang überschreibt Datensatz 2 mit Datensatz 7 usw.	

	A	B	C	D	E	F
1	Record	Date	UTC Time	Count	Temperature	Pressure
2	1	9/29/2010	21:01:46	5	5.00E+00	5.00E+00
3	2	9/29/2010	21:01:47	5	5.00E+00	5.00E+00
4						
5						

	A	B	C	D	E	F
1	Record	Date	UTC Time	Count	Temperature	Pressure
2	1	9/30/2010	20:26:56	1	9.86E+01	3.52E+01
3	2	9/30/2010	20:28:43	2	1.00E+02	3.73E+01
4	3	9/30/2010	20:29:03	3	9.99E+01	3.68E+01
5	4	9/30/2010	20:29:21	4	9.95E+01	3.64E+01
6	5	9/30/2010	20:30:19	5	9.92E+01	3.74E+01
7						

	A	B	C	D	E	F
1	Record	Date	UTC Time	Count	Temperature	Pressure
2	6	9/30/2010	20:32:03	6	9.86E+01	3.58E+01
3	2	9/30/2010	20:28:43	2	1.00E+02	3.73E+01
4	3	9/30/2010	20:29:03	3	9.99E+01	3.68E+01
5	4	9/30/2010	20:29:21	4	9.95E+01	3.64E+01
6	5	9/30/2010	20:30:19	5	9.92E+01	3.74E+01
7						

Hinweis

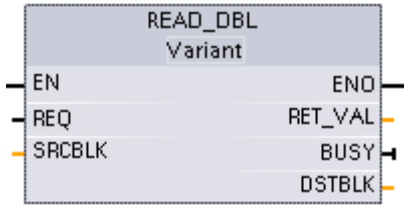
Es werden keine //END-Merker mehr am Ende eines Datenprotokolls gesetzt, das nicht voll ist. Vor der S7-1200 CPU V4.1 enthielten Datenprotokolle, die nicht voll waren, einen //END-Merker.

9.8 Datenbausteinsteuerung

9.8.1 Anweisungen READ_DBL und WRIT_DBL (Datenbaustein im Ladespeicher lesen/schreiben)

Tabelle 9- 178 Anweisungen READ_DBL und WRIT_DBL

KOP/FUP	SCL	Beschreibung

	`READ_DBL(`
	`req:=_bool_in_,`
	`srcblk:=_variant_in_,`
	`busy=>_bool_out_,`
	`dstblk=>_variant_out_);`
		Kopiert DB-Startwerte oder einen Teil der Werte aus dem Ladespeicher in einen Ziel-DB im Arbeitsspeicher.
		Der Inhalt des Ladespeichers wird während des Kopiervorgangs nicht verändert.

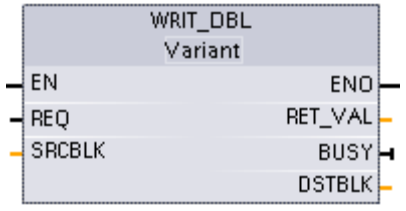
	`WRIT_DBL(`
	`req:=_bool_in_,`
	`srcblk:=_variant_in_,`
	`busy=>_bool_out_,`
	`dstblk=>_variant_out_);`
		Kopiert aktuelle DB-Werte oder einen Teil der Werte aus dem Arbeitsspeicher in einen Ziel-DB im Ladespeicher.
		Der Inhalt des Arbeitsspeichers wird während des Kopiervorgangs nicht verändert.

Tabelle 9- 179 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	BOOL	Ein Signal mit hohem Pegel startet die Operation, wenn BUSY = 0.
SRCBLK	IN	VARIANT	READ_DBL: Pointer auf den Quelldatenbaustein im Ladespeicher WRIT_DBL: Pointer auf den Quelldatenbaustein im Arbeitsspeicher
RET_VAL	OUT	INT	Ausführungsbedingung
BUSY	OUT	BOOL	BUSY = 1 meldet, dass der Lese-/Schreibvorgang unvollständig ist.
DSTBLK	OUT	VARIANT	READ_DBL: Pointer auf den Zieldatenbaustein im Arbeitsspeicher WRIT_DBL: Pointer auf den Zieldatenbaustein im Ladespeicher

Typischerweise wird ein DB sowohl im Ladespeicher (Flash) als auch im Arbeitsspeicher (RAM) gespeichert. Die Startwerte (Anfangswerte) werden immer im Ladespeicher gespeichert und die aktuellen Werte werden immer im Arbeitsspeicher gespeichert. Mit READ_DBL können Sie einen Satz Startwerte aus dem Ladespeicher in die aktuellen Werte eines DBs im Arbeitsspeicher kopieren, der von Ihrem Programm referenziert wird. Mit WRIT_DBL können Sie die im internen Ladespeicher oder auf einer Memory Card gespeicherten aktuellen Werte im Arbeitsspeicher aktualisieren.

Hinweis

Auswirkung der Anweisungen WRIT_DBL und READ_DBL auf den Flash-Speicher

Die Anweisung WRIT_DBL führt Schreibvorgänge im Flash-Speicher durch (im internen Ladespeicher oder auf einer Memory Card). Um eine Reduzierung der Lebensdauer des Flash-Speichers zu verhindern, verwenden Sie die Anweisung WRIT_DBL für weniger häufige Aktualisierungen wie das Aufzeichnen von Änderungen an einem Produktionsprozess. Vermeiden Sie aus ähnlichen Gründen häufige Aufrufe von READ_DBL für Lesevorgänge.

Sie müssen die Datenbausteine für READ_DBL und WRIT_DBL vor dem Aufrufen dieser Anweisungen im STEP 7-Programm erstellen. Wenn Sie den Quell-DB als Baustein vom Typ "Standard" erstellt haben, muss der Ziel-DB ebenfalls vom Typ "Standard" sein. Wenn der Quelldatenbaustein als Baustein vom Typ "Optimiert" erstellt wurde, muss der Zieldatenbaustein ebenfalls vom Typ "Optimiert" sein.

Wenn es sich um Standard-DBs handelt, können Sie entweder einen Variablennamen oder einen P#-Wert angeben. Mit dem P#-Wert können Sie eine beliebige Anzahl von Elementen der angegebenen Größe (Byte, Word oder DWord) angeben und kopieren. So können Sie einen Teil eines DBs oder den gesamten DB kopieren. Handelt es sich um optimierte DBs, können Sie lediglich einen Variablennamen angeben. Den Operator P# können Sie nicht verwenden. Wenn Sie für einen Standard-DB oder einen optimierten DB (oder für andere Arten von Arbeitsspeichern) einen Variablennamen angeben, kopiert die Anweisung die Daten, auf die dieser Variablenname verweist. Hierbei kann es sich um einen benutzerdefinierten Typ, ein Array oder ein grundlegendes Element handeln. Wenn es sich um einen Standard-DB und nicht um einen optimierten DB handelt, können Sie bei diesen Anweisungen nur den Datentyp Struct verwenden. Sie müssen einen benutzerdefinierten Datentyp (UDT) verwenden, wenn es sich um eine Struktur im optimierten Speicher handelt. Nur mit einem benutzerdefinierten Typ wird sichergestellt, dass die "Datentypen" sowohl für die Quellstruktur als auch für die Zielstruktur identisch sind.

Hinweis

Struktur (Datentyp Struct) in einem "optimierten" DB verwenden

Wenn Sie einen Datentyp Struct mit "optimierten" DBs verwenden, müssen Sie zunächst einen benutzerdefinierten Datentyp (UDT) für den Datentyp Struct erstellen. Dann konfigurieren Sie den Quell- und Ziel-DB mit dem Datentyp UDT. Der Datentyp UDT gewährleistet, dass die Datentypen im Datentyp Struct für beide DBs konsistent bleiben.

Bei "Standard"-DBs verwenden Sie den Datentyp Struct, ohne einen Datentyp UDT zu erstellen.

READ_DBL und WRIT_DBL werden asynchron zum Programmzyklus ausgeführt. Die Verarbeitung erstreckt sich über mehrere Aufrufe von READ_DBL und WRIT_DBL. Sie starten den DB-Übertragungsauftrag durch Aufruf von REQ = 1 und überwachen anschließend die Ausgänge BUSY und RET_VAL, um zu ermitteln, wann die Datenübertragung beendet und korrekt ist.

Hinweis

Auswirkung der Anweisungen WRIT_DBL und READ_DBL auf die Kommunikationslast

Wenn die Anweisung WRIT_DBL oder READ_DBL kontinuierlich aktiv ist, kann sie dabei so viele Kommunikationsressourcen verbrauchen, dass STEP 7 die Kommunikation mit der CPU verliert. Verwenden Sie aus diesem Grund für den REQ-Parameter eine positive Flanke am Eingang (Seite 223) und keinen Schließer- oder Öffnereingang (Seite 217), der mehrere Zyklen lang eingeschaltet bleiben würde (Signalpegel High).

Um Datenkonsistenz sicherzustellen, ändern Sie den Zielbereich während der Bearbeitung von READ_DBL bzw. den Quellbereich während der Bearbeitung von WRIT_DBL nicht (d.h. solange der Parameter BUSY WAHR ist).

Einschränkungen bei den Parametern SRCBLK und DSTBLK:

- Ein Datenbaustein muss, damit er referenziert werden kann, zuvor erstellt worden sein.
- Die Länge eines VARIANT-Pointers vom Typ BOOL muss durch 8 teilbar sein.
- Die Länge eines VARIANT-Pointers vom Typ STRING muss in den Quell- und Ziel-Pointern identisch sein.

Informationen zu Rezepten und zur Maschineneinrichtung

Mit den Anweisungen READ_DBL und WRIT_DBL können Sie Rezepte oder Informationen für die Maschineneinrichtung verwalten. Dies ist prinzipiell eine weitere Vorgehensweise, um für Werte, die sich nicht sehr häufig ändern, remanente Daten zu speichern. Sie sollten jedoch die Anzahl der Schreibvorgänge begrenzen, um den Flash-Speicher nicht übermäßig zu beanspruchen. Sie können auf diese Weise den Umfang des remanenten Speichers über den normalen Umfang der remanenten Daten hinaus erweitern, zumindest für Werte, die sich nicht häufig ändern. Sie können mit der Anweisung WRIT_DBL Rezeptinformationen oder Informationen für die Maschineneinrichtung aus dem Arbeitsspeicher im Ladespeicher speichern, und Sie können diese Informationen mit der Anweisung READ_DBL wieder aus dem Ladespeicher in den Arbeitsspeicher abrufen.

Tabelle 9- 180 Bedingungscodes

RET_VAL (W#16#...)	Beschreibung
0000	Kein Fehler
0081	Warnung: Der Quellbereich ist kleiner als der Zielbereich. Die Quelldaten werden vollständig kopiert, wobei die zusätzlichen Bytes im Zielbereich nicht verändert werden.
7000	Aufruf mit REQ = 0: BUSY = 0
7001	Erster Aufruf mit REQ = 1 (in Bearbeitung): BUSY = 1
7002	N^{ter} Aufruf (in Bearbeitung): BUSY = 1

RET_VAL (W#16#...)	Beschreibung
8051	Typfehler Datenbaustein
8081	Der Quellbereich ist größer als der Zielbereich. Der Zielbereich wird vollständig belegt und die verbleibenden Bytes der Quelle werden ignoriert.
8251	Typfehler Quelldatenbaustein
82B1	Fehlender Quelldatenbaustein
82C0	Der Quell-DB wird von einer anderen Anweisung oder Kommunikationsfunktion bearbeitet.
8551	Typfehler Zieldatenbaustein
85B1	Fehlender Zieldatenbaustein
85C0	Der Ziel-DB wird von einer anderen Anweisung oder Kommunikationsfunktion bearbeitet.
80C3	Mehr als 50 READ_DBL- oder 50 WRIT_DBL-Anweisungen befinden sich derzeit in der Warteschlange für die Ausführung.

Siehe auch Rezepte (Seite 419)

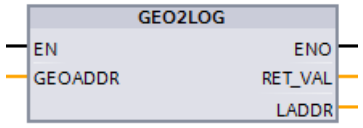
9.9 Adressverarbeitung

9.9.1 Anweisung GEO2LOG (Hardwarekennung über Steckplatz ermitteln)

Mit Anweisung GEO2LOG wird die Hardwarekennung anhand der Steckplatzinformationen ermittelt.

Tabelle 9- 181 Anweisung GEO2LOG

KOP/FUP	SCL	Beschreibung

	`ret_val := GEO2LOG(`
	`  GEOADDR:=_variant_in_out_,`
	`laddr:=_word_out_);`
		Mit Anweisung GEO2LOG wird die Hardwarekennung anhand der Steckplatzinformationen ermittelt.

Mit Anweisung GEO2LOG wird die Hardwarekennung anhand der Steckplatzinformationen ermittelt, die mit dem Systemdatentyp GEOADDR festgelegt werden:

Je nach dem Hardwaretyp, der in Parameter HWTYPE festgelegt ist, werden die folgenden Informationen der anderen GEOADDR-Parameter ausgewertet:

- Bei HWTYPE = 1 (PROFINET IO-System):
  - Nur IOSYSTEM wird ausgewertet. Die anderen Parameter von GEOADDR werden nicht berücksichtigt.
  - Die Hardwarekennung des PROFINET IO-Systems wird ausgegeben.
- Bei HWTYPE = 2 (PROFINET IO-Gerät):
  - IOSYSTEM und STATION werden ausgewertet. Die anderen Parameter von GEOADDR werden nicht berücksichtigt.
  - Die Hardwarekennung des PROFINET IO-Geräts wird ausgegeben.
- Bei HWTYPE = 3 (Baugruppenträger):
  - Nur IOSYSTEM und STATION werden ausgewertet. Die anderen Parameter von GEOADDR werden nicht berücksichtigt.
  - Die Hardwarekennung des Baugruppenträgers wird ausgegeben.
- Bei HWTYPE = 4 (Modul):
  - IOSYSTEM, STATION, und SLOT werden ausgewertet. Parameter SUBSLOT von GEOADDR wird nicht berücksichtigt.
  - Die Hardwarekennung des Moduls wird ausgegeben.
- Bei HWTYPE = 5 (Submodul):
  - Alle Parameter von GEOADDR werden ausgewertet.
  - Die Hardwarekennung des Submoduls wird ausgegeben.

Parameter AREA des Systemdatentyps GEOADDR wird nicht ausgewertet.

Tabelle 9- 182 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
GEOADDR	IN/OUT or IN ?	Variant	Pointer auf die Struktur des Systemdatentyps GEOADDR. Der Systemdatentyp GEOADDR enthält die Steckplatzinformationen, aus denen die Hardware-ID ermittelt wird. Weitere Informationen finden Sie unter "Systemdatentyp GEOADDR (Seite 463)".
RET_VAL	OUT or RETURN ?	Int	Ausgabe der Fehlerinformationen.
LADDR	OUT	HW_ANY	Hardwarekennung des Bauteils oder des Moduls. Die Nummer wird automatisch zugewiesen und ist in den Eigenschaften der Hardwarekonfiguration gespeichert.

Weitere Informationen über gültige Datentypen finden Sie unter "Übersicht der gültigen Datentypen" in der STEP 7 Online-Hilfe.

Tabelle 9- 183 Bedingungscodes

RET_VAL* (W#16#...)	Erklärung
0	Kein Fehler.
8091	Ungültiger Wert in GEOADDR für HWTYPE.
8094	Ungültiger Wert in GEOADDR für IOSYSTEM.
8095	Ungültiger Wert in GEOADDR für STATION.
8096	Ungültiger Wert in GEOADDR für SLOT.
8097	Ungültiger Wert in GEOADDR für SUBSLOT.
* Die Fehlercodes können als ganzzahlige oder als Hexadezimalwerte im Programmeditor angezeigt werden.	

9.9.2 Anweisung LOG2GEO (Steckplatz über Hardwarekennung ermitteln)

Mit der Anweisung LOG2GEO ermitteln Sie die geografische Adresse (Modulsteckplatz) aus der logischen Adresse einer Hardwarekennung.

Tabelle 9- 184 Anweisung LOG2GEO

KOP/FUP	SCL	Beschreibung

	`ret_val := LOG2GEO(`
	`  laddr:=_word_in_,`
	
	`GEOADDR:=_variant_in_out_);`
		Mit der Anweisung LOG2GEO ermitteln Sie den zu einer Hardwarekennung gehörigen Modulsteckplatz.

Die Anweisung LOG2GEO bestimmt die geografische Adresse einer logischen Adresse anhand der Hardwarekennung:

- Mit dem Parameter LADDR wählen Sie die logische Adresse anhand der Hardwarekennung aus.
- GEOADDR enthält die geografische Adresse der am Eingang LADDR angegebenen logischen Adresse.

Hinweis

Wenn der Typ HW eine Komponente nicht unterstützt, wird eine Untersteckplatznummer für ein Modul 0 ausgegeben.

Ein Fehler wird gemeldet, wenn der Eingang LADDR kein HW-Objekt adressiert.

Tabelle 9- 185 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
LADDR	IN	HW_ANY	Hardwarekennung des IO-Systems oder des Moduls. Die Nummer wird automatisch zugewiesen und in den Eigenschaften der CPU oder der Schnittstelle der Hardwarekonfiguration gespeichert.
RET_VAL	OUT	Int	Fehlercode der Anweisung
GEOADDR	IN_OUT	Variant	Pointer auf den Systemdatentyp GEOADDR. Der Systemdatentyp GEOADDR enthält die Steckplatzinformationen. Weitere Informationen finden Sie unter "Systemdatentyp GEOADDR (Seite 463)".

Für weitere Informationen über gültige Datentypen, siehe "Übersicht der gültigen Datentypen" in der STEP 7 Online-Hilfe.

Tabelle 9- 186 Bedingungscodes

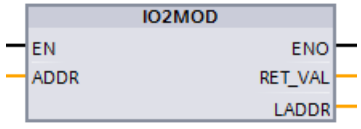
RET_VAL (W#16#...)	Beschreibung
0000	Kein Fehler
8090	Die im Parameter LADDR angegebene Adresse ist ungültig.
* Die Fehlercodes können als ganzzahlige oder als Hexadezimalwerte im Programmeditor angezeigt werden.	

9.9.3 Anweisung IO2MOD (Hardwarekennung über E/A-Adresse ermitteln)

Mit Anweisung IO2MOD wird die Hardwarekennung eines Moduls anhand der E/A-Adresse eines Submoduls ermittelt.

Tabelle 9- 187 Anweisung IO2MOD

KOP/FUP	SCL	Beschreibung

	`ret_val := IO2MOD(`
	`  ADDR:=_word_in_,`
	`  LADDR:=_word_out_);`
		Mit der Anweisung IO2MOD ermitteln Sie den zu einer Hardwarekennung gehörigen Modulsteckplatz.

Mit IO2MOD wird die Hardwarekennung des Moduls anhand einer E/A-Adresse (E, A, PE, PA) eines Submoduls ermittelt.

Geben Sie die E/A-Adresse in Parameter ADDR ein. Werden für diesen Parameter mehrere E/A-Adressen verwendet, so wird nur die erste Adresse für die Ermittlung der Hardwarekennung ausgewertet. Wird die erste Adresse richtig angegeben, so ist die Länge für die Adressangabe in ADDR nicht von Belang. Wird ein Adressbereich angegeben, der mehrere Module oder unbenutzte Adressen umfasst, kann auch die Hardwarekennung des ersten Moduls ermittelt werden.

Wird in Parameter ADDR keine E/A-Adresse eines (Sub-)Moduls angegeben, wird am Parameter RET_VAL der Fehlercode "8090" ausgegeben.

Hinweis

Eingabe der E/A-Adresse in SCL

Mit der E/A-Zugriffs-ID "%QWx:P" ist keine Programmierung in SCL möglich. In diesem Fall ist der symbolische Variablenname oder die absolute Adresse im Prozessbild zu verwenden.

Tabelle 9- 188 Datentypen für die Parameter

Parameter	Deklaration	Datentyp	Speicherbereich	Beschreibung
ADDR	IN or IN/OUT ?	Variant	E, A, M, D, L	E/A-Adresse (E, A, PE, PA) in einem Submodul. Achten Sie darauf, dass für Parameter ADDR kein Scheibenzugriff verwendet wird. In diesem Fall würden an Parameter LADDR fehlerhafte Werte ausgegeben.
RET_VAL	OUT or RETURN ?	Int	E, A, M, D, L	Fehlercode der Anweisung.
LADDR	OUT	HW_IO	E, A, M, D, L	Ermittelte Hardwarekennung (logische Adresse) des E/A-Submoduls.

Weitere Informationen über gültige Datentypen finden Sie unter "Übersicht der gültigen Datentypen" in der STEP 7 Online-Hilfe.

Tabelle 9- 189 Bedingungscodes

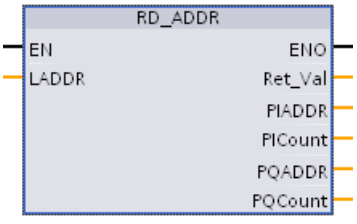
RET_VAL* (W#16#...)	Erklärung
0	Kein Fehler.
8090	Die in Parameter ADDR angegebene E/A-Adresse wird von keiner Hardwarekomponente benutzt.
* Die Fehlercodes können als ganzzahlige oder als Hexadezimalwerte im Programmeditor angezeigt werden.	

9.9.4 Anweisung RD_ADDR (E/A-Adressen über Hardwarekennung ermitteln)

Mit der Anweisung RD_ADDR rufen Sie die E/A-Adressen eines Submoduls ab.

Tabelle 9- 190 Anweisung RD_ADDR

KOP/FUP	SCL	Beschreibung

	`ret_val := RD_ADDR(`
	`  laddr:=_word_in_,`
	`  PIADDR=>_udint_out_,`
	`  PICount=>_uint_out_,`
	`  PQADDR=>_udint_out_,`
	`  PQCount=>_uint_out_,);`
		Mit der Anweisung RD_ADDR rufen Sie die E/A-Adressen eines Submoduls ab.

Die Anweisung RD_ADDR ermittelt die Länge und die Anfangsadresse der Eingänge oder Ausgänge anhand der Hardwarekennung eines Submoduls:

- Mit dem Parameter LADDR wählen Sie das Eingangs- oder Ausgangsmodul anhand der Hardwarekennung aus.
- Abhängig davon, ob es sich um ein Eingangs- oder ein Ausgangsmodul handelt, werden die folgenden Ausgangsparameter verwendet:
  - Bei einem Eingangsmodul werden die ermittelten Werte an den Parametern PIADDR und PICOUNT ausgegeben.
  - Bei einem Ausgangsmodul werden die ermittelten Werte an den Parametern PQADDR und PQCOUNT ausgegeben.
- Die Parameter PIADDR und PQADDR enthalten jeweils die Anfangsadresse der E/A-Adressen des Moduls.
- Die Parameter PICOUNT und PQCOUNT enthalten jeweils die Anzahl der Bytes der Eingänge oder Ausgänge (1 Byte für 8 Eingänge/Ausgänge, 2 Bytes für 16 Eingänge/Ausgänge).

Tabelle 9- 191 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
LADDR	IN	HW_IO	Hardwarekennung des (Sub-)Moduls
RET_VAL	OUT	Int	Fehlercode der Anweisung
PIADDR	OUT	UDInt	Anfangsadresse des Eingangsmoduls
PICOUNT	OUT	UInt	Anzahl der Bytes der Eingänge
PQADDR	OUT	UDInt	Anfangsadresse des Ausgangsmoduls
PQCOUNT	OUT	UInt	Anzahl der Bytes der Ausgänge

Weitere Informationen über gültige Datentypen finden Sie unter "Übersicht der gültigen Datentypen" in der STEP 7 Online-Hilfe.

Tabelle 9- 192 Bedingungscodes

RET_VAL (W#16#...)	Beschreibung
0000	Kein Fehler
8090	Die Hardwarekennung des Moduls am Parameter LADDR ist ungültig.
* Die Fehlercodes können als ganzzahlige oder als Hexadezimalwerte im Programmeditor angezeigt werden.	

9.9.5 Systemdatentyp GEOADDR

Geografische Adresse

Der Systemdatentyp GEOADDR enthält die geografische Adresse eines Moduls (oder die Steckplatzinformationen).

- Geografische Adresse für PROFINET IO:

  Für PROFINOT IO besteht die geografische Adresse aus der ID des PROFINET IO-Systems, der Gerätenummer, der Steckplatznummer und dem Submodul (wenn ein Submodul verwendet wird).

- Geografische Adresse für PROFIBUS DP:

  Für PROFIBUS DP besteht die geografische Adresse aus der ID des DP-Mastersystems, der Stationsnummer und der Steckplatznummer.

Die Steckplatzinformation der Module ist in der Hardwarekonfiguration jedes Moduls zu finden.

Struktur des Systemdatentyps GEOADDR

Die Struktur GEOADDR wird automatisch erzeugt, wenn als Datentyp "GEOADDR" im Datenbaustein eingegeben wird.

Parametername	Datentyp	Beschreibung
GEOADDR	STRUCT	
HWTYPE	UINT	Hardwaretyp: • 1: IO-System (PROFINET/PROFIBUS) • 2: IO-Gerät/DP-Slave • 3: Baugruppenträger • 4: Modul • 5: Submodul Wird ein Hardwaretyp von der Anweisung nicht unterstützt, wird der HWTYPE "0" ausgegeben.
AREA	UINT	Bereichskennung: • 0 = CPU • 1 = PROFINET IO • 2 = PROFIBUS DP • 3 = AS-i
IOSYSTEM	UINT	PROFINET IO-System (0 = zentrale Einheit im Baugruppenträger)
STATION	UINT	• Nummer des Baugruppenträgers, wenn die Bereichskennung AREA = 0 (Zentralmodul). • Stationsnummer, wenn die Bereichskennung AREA > 0.
SLOT	UINT	Steckplatznummer
SUBSLOT	UINT	Nummer des Submoduls. Dieser Parameter hat den Wert "0", wenn kein Submodul verfügbar oder steckbar ist.

9.10 Gemeinsame Fehlercodes für die erweiterten Anweisungen

Tabelle 9- 193 Gemeinsame Bedingungscodes für die erweiterten Anweisungen

Bedingungscode (W#16#....)[1]	Beschreibung
8x22[2]	Bereich für Eingang zu klein
8x23	Bereich für Ausgang zu klein
8x24	Unzulässiger Eingangsbereich
8x25	Unzulässiger Ausgangsbereich
8x28	Unzulässige Eingangsbitzuweisung
8x29	Unzulässige Ausgangsbitzuweisung
8x30	Ausgangsbereich ist ein schreibgeschützter DB.
8x3A	DB ist nicht vorhanden.

[1] Tritt einer dieser Fehler während der Ausführung eines Codebausteins auf, bleibt die CPU im Betriebszustand RUN (Voreinstellung) oder geht in STOP, wenn dies entsprechend konfiguriert wurde. Optional kann mit den Anweisungen GetError oder GetErorID in diesem Codebaustein die lokale Fehlerbearbeitung (CPU bleibt in RUN) veranlasst und eine programmierte Reaktion auf den Fehler erzeugt werden.

[2] Das "x" steht für die Nummer des Parameters mit dem Fehler. Parameternummern beginnen mit 1.

Technologieanweisungen

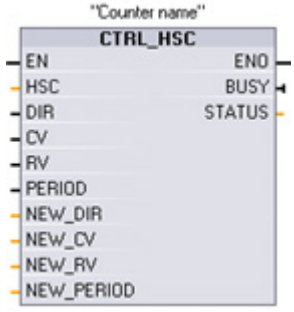
10

10.1 Schneller Zähler

10.1.1 Anweisung CTRL_HSC (Schnellen Zähler steuern)

Tabelle 10- 1 Anweisung CTRL_HSC (für allgemeine Zählaufgaben)

KOP/FUP	SCL	Beschreibung

	`"CTRL_HSC_1_DB" (`
	`hsc:=W#16#0,`
	`dir:=False,`
	`cv:=False,`
	`rv:=False,`
	`period:=False,`
	`new_dir:=0,`
	`new_cv:=L#0,`
	`new_rv:=L#0,`
	`new_period:=0,`
	`busy=>_bool_out_,`
	`status=>_word_out_);`
		Jede Anweisung CTRL_HSC (Schnellen Zähler steuern) nutzt eine in einem DB abgelegte Struktur, um Zählerdaten zu speichern. Sie weisen den DB zu, wenn Sie die Anweisung CTRL_HSC im Editor einfügen.

1 Wenn Sie die Anweisung einfügen, zeigt STEP 7 den Dialog "Aufrufoptionen" zum Erstellen des zugehörigen DB an.

2 Im SCL-Beispiel ist "CTRL_HSC_1_DB" der Name des Instanz-DBs.

Tabelle 10- 2 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
HSC	IN	HW_HSC	HSC-Kennung
DIR[1, 2]	IN	Bool	1 = Neue Richtung anfordern
CV[1]	IN	Bool	1 = Anforderung zum Setzen eines neuen Zählwerts
RV[1]	IN	Bool	1 = Anforderung zum Setzen eines neuen Referenzwerts
PERIOD[1]	IN	Bool	1 = Anforderung zum Setzen eines neuen Zeitintervalls (nur bei Frequenzmessung)
NEW_DIR	IN	Int	Neue Richtung: 1= vorwärts, -1= rückwärts
NEW_CV	IN	DInt	Neuer Zählwert
NEW_RV	IN	DInt	Neuer Referenzwert
NEW_PERIOD	IN	Int	Neuer Zeitintervallwert in Sekunden (nur bei Frequenzmessung): 1 = 1 s 2 = 0,1 s 3 = 0,1 s

Parameter und Datentyp		Datentyp	Beschreibung
BUSY[3]	OUT	Bool	Funktion besetzt
STATUS	OUT	Word	Ausführungsbedingung

[1] Wird keine Aktualisierung eines Parameterwerts angefordert, so werden die entsprechenden Eingangswerte ignoriert.

[2] Der Parameter DIR ist nur gültig, wenn für die konfigurierte Zählrichtung die Programmsteuerung (interne Richtungssteuerung) vorgegeben ist. Sie legen fest, wie dieser Parameter in der HSC-Gerätekonfiguration genutzt wird.

[3] Bei einem HSC in der CPU oder im SB hat der Parameter BUSY immer den Wert 0.

Sie konfigurieren die Parameter für jeden HSC in der Gerätekonfiguration der CPU: Zähl-/Frequenzfunktion, Rücksetzoptionen, Alarmereigniskonfiguration, Hardware-E/A und Adresse des Zählwerts.

Einige der Parameter für den HSC können von Ihrem Anwenderprogramm geändert werden, um die Programmsteuerung für den Zählvorgang vorzugeben:

- Setzen der Zählrichtung auf einen Wert NEW_DIR
- Setzen das aktuellen Zählwerts auf einen neuen Wert NEW_CV
- Setzen des Referenzwerts auf einen neuen Wert NEW_RV
- Setzen des Zeitintervallwerts (nur bei Frequenzmessung) auf einen neuen Wert NEW_PERIOD

Sind die folgenden Booleschen Merker auf 1 gesetzt, wenn die Anweisung CTRL_HSC ausgeführt wird, so wird der entsprechende Wert NEW_xxx in den Zähler geladen. Mehrere Anforderungen (mehrere Merker sind gleichzeitig gesetzt) werden in einer Ausführung der Anweisung CTRL_HSC verarbeitet.

- DIR = 1 ist eine Anforderung zum Laden eines Werts NEW_DIR, 0 = keine Änderung
- CV = 1 ist eine Anforderung zum Laden eines Werts NEW_CV, 0 = keine Änderung
- RV = 1 ist eine Anforderung zum Laden eines Werts NEW_RV, 0 = keine Änderung
- PERIOD = 1 ist eine Anforderung zum Laden eines Werts NEW_PERIOD, 0 = keine Änderung

Die Anweisung CTRL_HSC wird typischerweise in einem Prozessalarm-OB platziert, der ausgeführt wird, wenn das Prozessalarmereignis des Zählers ausgelöst wird. Wird zum Beispiel der Zählerinterrupt durch ein Ereignis CV=RV ausgelöst, so führt ein Prozessalarm-OB die Anweisung CTRL_HSC aus und kann den Referenzwert durch Laden eines Werts NEW_RV ändern.

Der aktuelle Zählwert ist in den Parametern für CTRL_HSC nicht vorgesehen. Die Adresse des Prozessabbilds, in der der aktuelle Zählwert gespeichert wird, wird bei der Hardwarekonfiguration dem schnellen Zähler zugewiesen. Sie können den Zählwert über die Programmlogik direkt auslesen. Der an Ihr Programm ausgegebene Wert ist ein korrekter Zählwert für den Moment, in dem der Zähler gelesen wurde. Der Zähler setzt die Zählung schneller Ereignisse fort. Der tatsächliche Zählwert kann sich deshalb ändern, bevor Ihr Programm einen Prozess mit einem alten Zählwert beendet.

Aktueller Zählwert des HSC: Programmzugriff, Wertebereich und Rollover-Verhalten

Die CPU speichert den aktuellen Wert jedes HSC in der Adresse eines Eingangs (E). Die folgende Tabelle zeigt die Standardadressen für den aktuellen Wert jedes HSC. Sie können die E-Adresse für den aktuellen Wert ändern, indem Sie die Eigenschaften der CPU in der Gerätekonfiguration ändern.

Schnelle Zähler speichern den aktuellen Zählwert in einem DInt-Wert. Ein DInt-Wert hat einen Bereich von -2147483648 bis +2147483647. Der Zähler springt beim Hochzählen vom maximalen positiven Wert zum maximalen negativen Wert und beim Herunterzählen vom maximalen negativen Wert zum maximalen positiven Wert.

HSC	Datentyp des aktuellen Werts	Standardadresse des aktuellen Werts
HSC1	DInt	ED1000
HSC2	DInt	ED1004
HSC3	DInt	ED1008
HSC4	DInt	ED1012
HSC5	DInt	ED1016
HSC6	DInt	ED1020

Bei einem Fehler wird ENO auf 0 gesetzt und der Ausgang STATUS gibt einen Bedingungscode an.

Tabelle 10- 3 Execution condition codes

STATUS (W#16#)	Beschreibung
0	Kein Fehler
80A1	HSC-Kennung adressiert keinen HSC
80B1	Ungültiger Wert in NEW_DIR
80B2	Ungültiger Wert in NEW_CV
80B3	Ungültiger Wert in NEW_RV
80B4	Ungültiger Wert in NEW_PERIOD
80C0	Mehrmaliger Zugriff auf den schnellen Zähler
80D0	Schneller Zähler (HSC) nicht in der CPU-Hardwarekonfiguration aktiviert

10.1.2 Anweisung CTRL_HSC_EXT (Schnellen Zähler steuern (erweitert))

Die Anweisung CTRL_HSC_EXT bietet Programmzugriff auf die Anzahl von Eingangsimpulsen für einen bestimmten HSC über einen angegebenen Zeitraum. Diese Anweisung ermöglicht es dem Programm, den Zeitraum zwischen den Eingangsimpulsen mit einer Auflösung im Nanosekundenbereich zu ermitteln.

Zur Verwendung der Anweisung CTRL_HSC_EXT gehen Sie wie folgt vor:

1. Konfigurieren Sie für den zugehörigen HSC den Periodenmodus. Wählen Sie die gewünschte Betriebsphase. Wenn Sie die interne Richtungssteuerung auswählen, geben Sie den Richtungseingang für andere Zwecke frei.
2. Fügen Sie die Anweisung CTRL_HSC_EXT in das KOP-Netzwrk ein, wodurch ebenfalls ein Instanz-Datenbaustein CTRL_HSC_EXT_DB erstellt wird.
3. Erstellen Sie einen benutzerdefinierten Global_DB = z.B. "MEINDB", bei dem es sich um einen Eingangsparameter für CTRL_HSC_EXT handelt. Dieser DB enthält die vom SFB benötigten Informationen.
4. Suchen Sie in MEINDB eine leere Zeile und fügen Sie den Variablennamen = z.B. "MeinZeitraum" ein.
5. Fügen Sie den Datentyp hinzu, indem Sie "HSC_Period" <Enter> eingeben (das Dropdown-Menü enthält diese Option derzeit noch nicht). Sie müssen diesen Namen genau wie gezeigt eingeben.
6. Prüfen Sie, ob es sich bei der Variablen "MeinZeitraum" jetzt um eine erweiterbare Kommunikationsdatenstruktur handelt.
7. Fügen Sie am Anschluss "CTRL" der KOP-Anweisung CTRL_HSC_EXT die DB-Variable "MEINDB".MeinZeitraum ein.

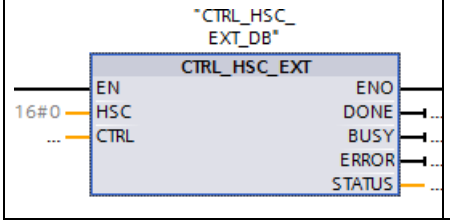
Hinweis

Einstellung der Zeiten für den Digitaleingangsfilter

Bei digitalen HSC-Eingängen verwenden Sie die kleinste erwartete Impulsbreite für die zugehörige Einstellung des Digitaleingangsfilters.

Tabelle 10- 4 Anweisung CTRL_HSC_EXT

KOP/FUP	SCL	Beschreibung

	`"CTRL_HSC_1_DB" (`
	`hsc:=_hw_hsc_in_,`
	`done:=_done_out_,`
	`busy:=_busy_out_,`
	`error:=_error_out_,`
	`status:=_status_out_,`
	`ctrl:=MYDB.MyPeriod);`
		Jede Anweisung CTRL_HSC_EXT (Schnellen Zähler steuern) nutzt eine in einem benutzerdefinierten globalen DB abgelegte systemdefinierte Datenstruktur, um Zählerdaten zu speichern. Der Datentyp HSC_Period wird als Eingangsparameter für CTRL_HSC_EXT zugewiesen.

[1] STEP 7 erstellt den DB automatisch, wenn Sie die Anweisung einfügen.

[2] Im SCL-Beispiel ist "CTRL_HSC_1_DB" der Name des Instanz-DBs.

Tabelle 10- 5 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
HSC	IN	HW_HSC	HSC-Kennung
CTRL	IN	HSC_Period	SFB-Eingang und Rückgabedaten (siehe Tabelle "Struktur von HSC_Period")
DONE	OUT	Bool	1 = Zeigt an, dass der SFB beendet ist. Immer 1, weil der SFB synchron ist.
BUSY	OUT	Bool	Immer 0, die Funktion ist niemals besetzt.
ERROR	OUT	Bool	1 = Weist auf einen Fehler hin.
STATUS	OUT	Word	Ausführungsbedingung (siehe Tabelle "Ausführungsbedingungen")

Tabelle 10- 6 Struktur von HSC_Period

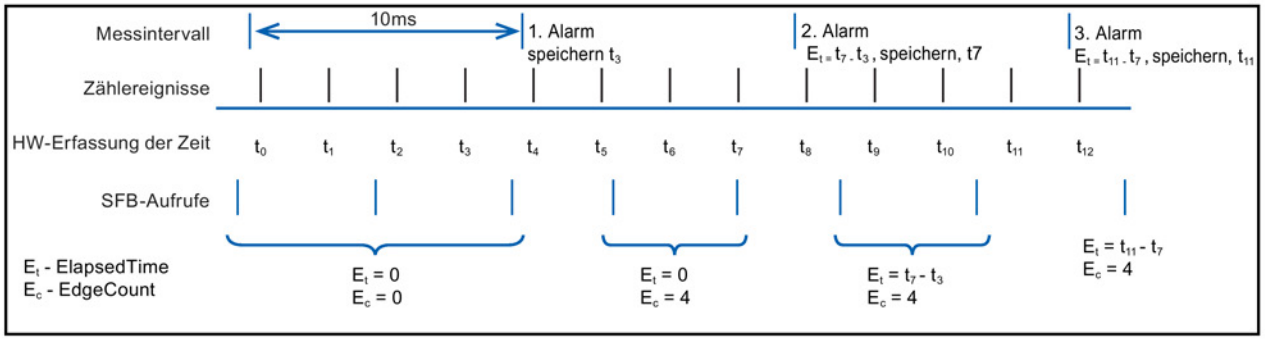
Strukturelement		Datentyp	Beschreibung
ElapsedTime	OUT	UDINT	Abgelaufene Zeit zwischen den letzten Eingangsimpulsen von aufeinander folgenden Intervallen.
EdgeCount	OUT	UDINT	Gibt die Anzahl von Eingangsimpulsen der letzten beendeten Perioden zurück.
EnHSC	IN	Bool	Aktiviert HSC: 1 = Aktiviert SFB, 0 = Deaktiviert SFB
EnPeriod	IN	Bool	Ermöglicht die Aktualisierung der Periode: 1 = Ändert SFB-Zeitraum in NewPeriod
NewPeriod	IN	INT	NewPeriod gibt das Intervall für die Periodendauermessung an (benötigter Zeitraum für die Durchführung einer Periodendauermessung). Zulässig sind einzig die Werte 10, 100 oder 1000 Millisekunden.

HSC_Period Definitionen der Strukturelemente

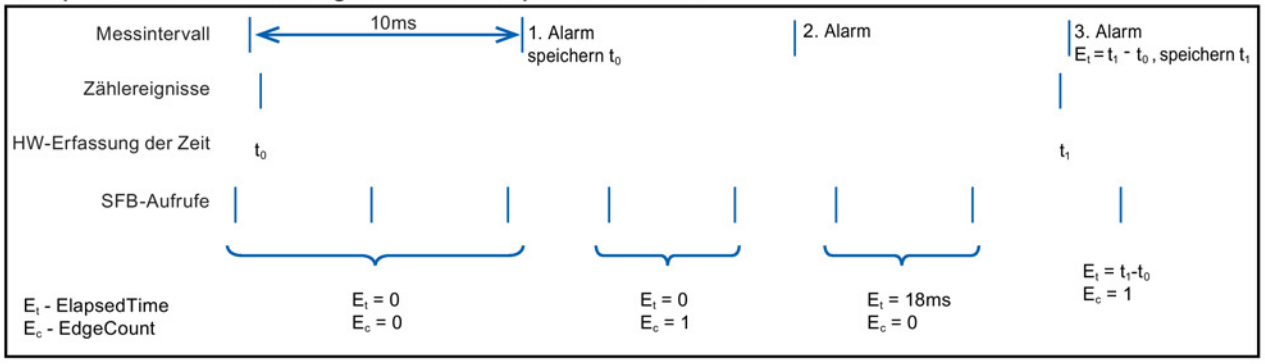
- ElapsedTime gibt die abgelaufene Zeit in Nanosekunden zwischen dem letzten Impuls in der aktuellen Periodendauermessung und dem letzten Impuls in der vorherigen Periodendauermessung zurück.
  Wenn EdgeCount = 0 ist, dann ist ElapsedTime die kumulierte Zeit seit dem letzten Impuls. ElapsedTime hat einen Bereich von 0 bis 4.294.967.280 Nanosekunden (0x0000 0000 bis 0xFFFF FFF0). Ein Periodenüberlauf wird durch den Rückgabewert 4.294.96.295 (0xFFFF FFFF) angezeigt. Die Werte von 0xFFFF FFF1 bis 0xFFFF FFFE sind reserviert.
  Wenn ElapsedTime entweder 0 (keine Eingangsimpulse empfangen) oder 0xFFFF FFFF (Periodenüberlauf) ist, dann ist EdgeCount nicht gültig.
- EdgeCount gibt die Anzahl von Eingangsimpulsen des letzten Messintervalls zurück. EdgeCount muss "1" oder mehr sein, damit die Periode zwischen den Impulsen berechnet werden kann. Die Periode wird anhand der folgenden Formel berechnet: Periode = ElapsedTime/EdgeCount.

Die folgenden Beispiele zeigen, wie die Anweisung Periodendauermessungen durchführt.

Beispiel 1: Mehrere Zählereignisse in einem Messintervall



Beispiel 2: Null und ein Zählereignis in einem multiplen Messintervall



Regeln:
1. Wenn $E1_t = 0$, ist der Zeitraum ungültig
2. Andernfalls, Zeitraum = E_t / E_c

Die CPU speichert den aktuellen Wert jedes HSC in der Adresse eines Eingangs (E). Die folgende Tabelle zeigt die Standardadressen für den aktuellen Wert jedes HSC. Sie können die Adresse des Eingangs (E) für den aktuellen Wert ändern, indem Sie die Eigenschaften der CPU in der Gerätekonfiguration ändern.

Schnelle Zähler speichern den aktuellen Zählwert in einem DInt-Wert. Ein DInt-Wert hat einen Bereich von -2147483648 bis +2147483647. Der Zähler springt beim Hochzählen vom maximalen positiven Wert zum maximalen negativen Wert und beim Herunterzählen vom maximalen negativen Wert zum maximalen positiven Wert.

Tabelle 10- 7 Standardadressen des aktuellen Werts jedes HSC

HSC	Datentyp des aktuellen Werts	Standardadresse des aktuellen Werts
HSC1	DInt	ED1000
HSC2	DInt	ED1004
HSC3	DInt	ED1008
HSC4	DInt	ED1012
HSC5	DInt	ED1016
HSC6	DInt	ED1020

Bei einem Fehler wird ENO auf 0 gesetzt und der Ausgang STATUS gibt den Bedingungscode an.

Tabelle 10- 8 Ausführungsbedingungen

STATUS (W#16#)	Beschreibung
0	Kein Fehler
80A1	HSC-Kennung adressiert keinen HSC
80D0	SBF 124 nicht verfügbar
80B5	Ungültiger Wert für NewPeriod

10.1.3 Funktionsweise von schnellen Zählern

Schnelle Zähler (HSC) zählen Ereignisse, die schneller als die Ausführungsrate des Zyklus-OBs auftreten. Wenn die zu zählenden Ereignisse langsamer als die Ausführungsrate des OBs auftreten, können Sie die Standard-Zähleranweisungen CTU, CTD oder CTUD verwenden. Wenn die Ereignisse schneller als die Ausführungsrate des OBs auftreten, verwenden Sie den schnelleren HSC. Über die Anweisung CTRL_HSC kann Ihr Programm programmatisch einige der HSC-Parameter ändern.

Beispiel: Sie können den HSC als Eingang für einen Winkelschrittgeber nutzen. Der Winkelschrittgeber sorgt für eine bestimmte Anzahl von Zählwerten pro Umdrehung sowie für einen Rücksetzimpuls einmal pro Umdrehung. Der bzw. die Taktgeber und der Rücksetzimpuls des Winkelschrittgebers liefern die Eingänge für den schnellen Zähler.

Der erste von mehreren voreingestellten Werten wird in den schnellen Zähler geladen. Die Ausgänge werden für die Zeitspanne aktiviert, während der der aktuelle Wert des Zählers kleiner als der voreingestellte Wert ist. Der HSC löst einen Alarm aus, wenn der aktuelle Wert des Zählers gleich dem voreingestellten Wert ist, wenn der Zähler zurückgesetzt wird oder wenn ein Richtungswechsel auftritt.

Wenn der aktuelle Wert gleich dem voreingestellten Wert ist und es zu einem Alarmereignis kommt, dann wird ein neuer voreingestellter Wert geladen und der nächste Signalzustand für die Ausgänge gesetzt. Tritt ein Alarmereignis auf, weil der Zähler zurückgesetzt wird, dann werden der erste voreingestellte Wert und die ersten Signalzustände der Ausgänge gesetzt und der Zyklus wiederholt.

Da die Alarme in einer sehr viel geringeren Geschwindigkeit auftreten als der schnelle Zähler zählt, kann eine präzise Steuerung der schnellen Anweisungen mit relativ geringem Einfluss auf den Zyklus der CPU implementiert werden. Da Sie Alarme bestimmten Interruptprogrammen zuordnen können, kann jede neue Voreinstellung in einem getrennten Interruptprogramm geladen werden, damit so der Zustand einfach gesteuert werden kann. Sie können alternativ auch alle Alarmereignisse in einem einzigen Interruptprogramm bearbeiten.

Auswahl des HSC-Eingangskanals

Nutzen Sie bei der Auswahl die folgende Tabelle und achten Sie darauf, dass die verwendeten Eingangskanäle von CPU und SB die maximalen Impulsraten in Ihren Prozesssignalen unterstützen.

Hinweis

Eingangskanäle von CPU und SB (ab Firmware V4) haben konfigurierbare Eingangsfilterzeiten

Bei früheren Firmware-Versionen waren die HSC-Eingangskanäle und auch die Filterzeiten festgelegt und konnten nicht geändert werden.

Ab Version V4 können Sie die Eingangskanäle und Filterzeiten festlegen. Die Standardeinstellung des Eingangsfilters von 6,4 ms ist für Ihre Prozesssignale möglicherweise zu langsam. Sie müssen die Zeiten der digitalen Eingangsfilter (Seite 170) für die HSC-Eingänge für Ihre HSC-Anwendung optimieren.

Tabelle 10- 9 CPU-Eingang: maximale Frequenz

CPU	CPU-Eingangskanal	1- oder 2-Phasenbetrieb	A/B-Zähler
1211C	Ea.0 bis Ea.5	100 kHz	80 kHz
1212C	Ea.0 bis Ea.5	100 kHz	80 kHz
	Ea.6, Ea.7	30 kHz	20 kHz
1214C und 1215C	Ea.0 bis Ea.5	100 kHz	80 kHz
	Ea.6 bis Eb.5	30 kHz	20 kHz
1217C	Ea.0 bis Ea.5	100 kHz	80 kHz
	Ea.6 bis Eb.1	30 kHz	20 kHz
	Eb.2 bis Eb.5 (0,2+, 0,2- bis 0,5+, 0,5-)	1 MHz	1 MHz

Tabelle 10- 10 SB-Eingang: maximale Frequenz (optionales Board)

Signalboard (SB)	SB-Eingangskanal	1- oder 2-Phasenbetrieb	A/B-Zähler
SB 1221, 200 kHz	Ee.0 bis Ee.3	200 kHz	160 kHz
SB 1223, 200 kHz	Ee.0, Ee.1	200 kHz	160 kHz
SB 1223	Ee.0, Ee.1	30 kHz	20 kHz

Funktionalität für den HSC auswählen

Alle HSCs arbeiten in der gleichen Zählerart auf die gleiche Weise. Zählerart, Richtungssteuerung und anfängliche Richtung für die HSC-Funktionseigenschaften werden in der CPU-Gerätekonfiguration zugewiesen.

Es gibt vier grundlegende Arten von schnellen Zählern:

- Einphasenzähler mit interner Richtungssteuerung
- Einphasenzähler mit externer Richtungssteuerung
- Zweiphasenzähler mit 2 Takteingängen
- A/B-Zähler

Sie können jeden HSC-Typ mit oder ohne Rücksetzeingang verwenden. Wenn Sie den Rücksetzeingang aktivieren (mit einigen Einschränkungen, siehe folgende Tabelle), wird der aktuelle Wert zurückgesetzt. Er bleibt so lange zurückgesetzt, bis Sie den Rücksetzeingang deaktivieren.

- Frequenzfunktion: Der HSC kann in bestimmten Betriebsarten so konfiguriert werden (Zählart), dass er eine Frequenz statt der aktuellen Impulszahl ausgibt. Es gibt drei verschiedene Frequenzmesszeiten: 0,01, 0,1 oder 1,0 Sekunden.

  Die Frequenzmesszeit legt fest, wie oft der HSC einen neuen Frequenzwert berechnet und ausgibt. Die ausgegebene Frequenz ist ein Mittelwert, der anhand der Gesamtzahl der Zählwerte im letzten Messzeitraum berechnet wird. Ändert sich die Frquenz schnell, ist der ausgegebene Wert ein Zwischenergebnis zwischen der höchsten und der niedrigsten während des Messzeitraums aufgetretenen Frequenz. Die Frequenz wird unabhängig von der eingestellten Frequenzmesszeit immer in Hertz (Impulse pro Sekunde) angegeben.

- Zählerarten und Zählereingänge: Die folgende Tabelle zeigt die Eingänge, die für Funktionen wie Taktgeber, Richtungssteuerung und Rücksetzen des HSC verwendet werden.

- Periodendauermessung: Die Periodendauermessung erfolgt über das konfigurierte Messintervall (10 ms, 100 ms oder 1000 ms). Der SDT HSC_Period gibt Periodendauermessungen zurück und liefert zwei Werte als Messergebnis: ElapsedTime und EdgeCount. Die HSC-Eingänge ID1000 bis ID1020 sind von Periodendauermessungen nicht betroffen:

  – ElapsedTime ist ein vorzeichenloser doppelter Ganzzahlenwert in Nanosekunden, der die Zeit vom ersten Zählereignis bis zum letzten Zählereignis im Messintervall darstellt. Wenn EdgeCount = 0 ist, dann ist ElapsedTime die Zeit seit dem letzten Zählereignis in einem vorherigen Intervall. ElapsedTime hat einen Bereich von 0 bis 4.294.967.280 ns (0x0000 0000 bis 0xFFFF FFF0). Überlauf wird durch den Wert 4.294.967.295 (0xFFFF FFFF) angegeben. Die Werte von 0xFFFF FFF1 bis 0xFFFF FFFE sind reserviert.

  – EdgeCount ist ein vorzeichenloser doppelter Ganzzahlenwert, der die Anzahl der Zählereignisse im Messintervall darstellt.

Ein Eingang kann nicht für zwei verschiedene Funktionen verwendet werden. Wird ein Eingang jedoch nicht von der aktuellen Zählerart des definierten schnellen Zählers benötigt, kann er für andere Zwecke genutzt werden. Befindet sich beispielsweise HSC1 in einer Betriebsart, in der die zwei integrierten Eingänge, aber nicht der dritte externe Rücksetzeingang (Standardzuweisung an E0.3) verwendet wird, so kann E0.3 für Flankenalarme oder für HSC2 belegt werden.

Tabelle 10- 11 Zählarten für den HSC

Typ	Eingang 1	Eingang 2	Eingang 3	Funktion
Einphasenzähler mit interner Richtungssteuerung	Takt	-	-	Zählwert oder Frequenz
			Rücksetzen	Zählwert
Einphasenzähler mit externer Richtungssteuerung	Takt	Richtung	-	Zählwert oder Frequenz
			Rücksetzen	Zählwert
Zweiphasenzähler mit 2 Takteingängen	Takt vorwärts	Takt rückwärts	-	Zählwert oder Frequenz
			Rücksetzen	Zählwert
A/B-Zähler	Phase A	Phase B	-	Zählwert oder Frequenz
			Zurücksetzen[1]	Zählwert

[1] Bei einem Winkelschrittgeber: Phase Z, Referenzpunkt

Eingangsadressen für den HSC

Beim Konfigurieren der CPU können Sie die "Hardware-Eingänge" für jeden HSC aktivieren und konfigurieren.

Alle HSC-Eingänge müssen an Klemmen an der CPU oder am optionalen Signalboard (das in die Front der CPU gesteckt wird) angeschlossen sein.

Hinweis

Wie Sie in den folgenden Tabellen sehen, überschneiden sich die Standardzuweisungen der optionalen Signale der verschiedenen HSCs. Zum Beispiel wird für das optionale externe Rücksetzen von HSC 1 der gleiche Eingang verwendet wie für einen der Eingänge von HSC 2.

Bei CPUs ab V4 können Sie die HSC-Eingänge während der CPU-Konfiguration neu zuweisen. Sie brauchen nicht die Standard-Eingangszuweisungen zu verwenden.

Stellen Sie stets sicher, dass Sie Ihre HSCs so konfigurieren, dass **kein** Eingang von zwei HSCs verwendet wird.

Die folgenden Tabellen zeigen die standardmäßen HSC-Eingangszuweisungen für die integrierten E/A der CPUs und eines optionalen SBs. (Wenn die ausgewählte SB-Ausführung nur zwei Eingänge hat, sind nur die Eingänge 4.0 und 4.1 verfügbar.)

Definitionen für die Tabelle der HSC-Eingänge

- **Einphasenzähler: C ist der Zähl**eingang, **[d] ist der Richtungs**eingang (optional) und **[R]** ist der Eingang für **externes Rücksetzen** (optional)
  (Rücksetzen ist nur im Zählmodus verfügbar).
- **Zweiphasenzähler: CU ist der Vorwärtszähl**eingang, **CD ist der Rückwärtszähl**eingang und **[R]** ist der Eingang für **externes Rücksetzen** (optional)
  (Rücksetzen ist nur im Zählmodus verfügbar).
- **A/B-Zähler: A ist der Zähleingang A**, **B ist der Zähleingang B** und **[R]** ist der Eingang für **externes Rücksetzen** (optional)**.** (Rücksetzen ist nur im Zählmodus verfügbar.)

Tabelle 10- 12 CPU 1211C: Zuweisungen der HSC-Standardadressen

HSC-Zählermodus		Integrierter CPU-Eingang (Standard 0.x)						Eingang eines optionalen SBs (Standard 4.x) [1]			
		0	1	2	3	4	5	0	1	2	3
HSC 1	Einphasig	C	[d]		[R]			C	[d]		[R]
	Zweiphasig	CU	CD		[R]			CU	CD		[R]
	A/B-Zähler	A	B		[R]			A	B		[R]
HSC 2	Einphasig		[R]	C	[d]				[R]	C	[d]
	Zweiphasig		[R]	CU	CD				[R]	CU	CD
	A/B-Zähler		[R]	A	B				[R]	A	B
HSC 3	Einphasig					C	[d]	C	[d]		[R]
	Zweiphasig										
	A/B-Zähler										
HSC4	Einphasig					C	[d]	C	[d]		[R]
	Zweiphasig					CU	CD				
	A/B-Zähler					A	B				
HSC 5	Einphasig							C	[d]		[R]
	Zweiphasig							CU	CD		[R]
	A/B-Zähler							A	B		[R]
HSC 6	Einphasig								[R]	C	[d]
	Zweiphasig								[R]	CU	CD
	A/B-Zähler								[R]	A	B

[1] Ein SB mit nur zwei Digitaleingängen bietet nur die Eingänge 4.0 und 4.1.

Tabelle 10- 13 CPU 1212C: Zuweisungen der HSC-Standardadressen

HSC-Zählermodus		Integrierter CPU-Eingang (Standard 0.x)								Eingang eines optionalen SBs (Standard 4.x) [1]			
		0	1	2	3	4	5	6	7	0	1	2	3
HSC 1	Einphasig	C	[d]		[R]					C	[d]		[R]
	Zweiphasig	CU	CD		[R]					CU	CD		[R]
	A/B-Zähler	A	B		[R]					A	B		[R]
HSC 2	Einphasig		[R]	C	[d]						[R]	C	[d]
	Zweiphasig		[R]	CU	CD						[R]	CU	CD
	A/B-Zähler		[R]	A	B						[R]	A	B
HSC 3	Einphasig					C	[d]		[R]	C	[d]		[R]
	Zweiphasig					CU	CD		[R]				
	A/B-Zähler					A	B		[R]				
HSC 4	Einphasig						[R]	C	[d]	C	[d]		[R]
	Zweiphasig						[R]	CU	CD				
	A/B-Zähler						[R]	A	B				
HSC 5	Einphasig									C	[d]		[R]
	Zweiphasig									CU	CD		[R]
	A/B-Zähler									A	B		[R]
HSC 6	Einphasig										[R]	C	[d]
	Zweiphasig										[R]	CU	CD
	A/B-Zähler										[R]	A	B

[1] Ein SB mit nur zwei Digitaleingängen bietet nur die Eingänge 4.0 und 4.1.

Tabelle 10- 14 CPU 1214C, CPU 1215C und CPU1217C:
Zuweisungen der HSC-Standardadressen
(nur integrierte Eingänge, Adressen des optionalen SBs siehe nächste Tabelle)

HSC-Zählermodus		Digitaleingangsbyte 0 (Standard: 0.x)								Digitaleingangsbyte 1 (Standard: 1.x)					
		0	1	2	3	4	5	6	7	0	1	2	3	4	5
HSC 1	Einphasig	C	[d]		[R]										
	Zweiphasig	CU	CD		[R]										
	A/B-Zähler	A	B		[R]										
HSC 2	Einphasig		[R]	C	[d]										
	Zweiphasig		[R]	CU	CD										
	A/B-Zähler		[R]	A	B										
HSC 3	Einphasig					C	[d]		[R]						
	Zweiphasig					CU	CD		[R]						
	A/B-Zähler					A	B		[R]						
HSC 4	Einphasig						[R]	C	[d]						
	Zweiphasig						[R]	CU	CD						
	A/B-Zähler						[R]	A	B						
HSC 5	Einphasig									C	[d]	[R]			
	Zweiphasig									CU	CD	[R]			
	A/B-Zähler									A	B	[R]			
HSC 6	Einphasig												C	[d]	[R]
	Zweiphasig												CU	CD	[R]
	A/B-Zähler												A	B	[R]

Tabelle 10- 15 Optionales SB in CPUs der obigen Tabelle: Zuweisungen der HSC-Standardadressen

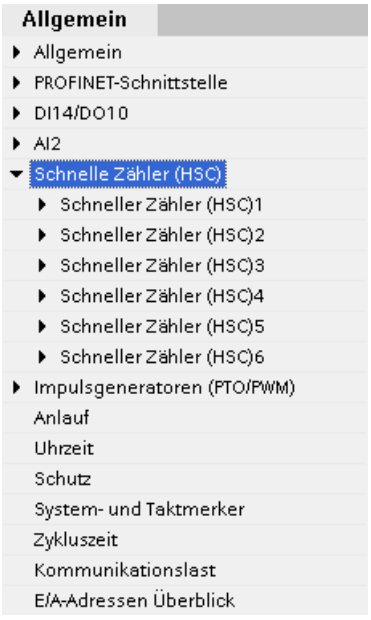
HSC		Eingänge des optionalen SBs (Standard: 4.x) [1]			
		0	1	2	3
HSC 1	Einphasig	C	[d]		[R]
	Zweiphasig	CU	CD		[R]
	A/B-Zähler	A	B		[R]
HSC 2	Einphasig		[R]	C	[d]
	Zweiphasig		[R]	CU	CD
	A/B-Zähler		[R]	A	B
HSC 5	Einphasig	C	[d]		[R]
	Zweiphasig	CU	CD		[R]
	A/B-Zähler	A	B		[R]
HSC 6	Einphasig		[R]	C	[d]
	Zweiphasig		[R]	CU	CD
	A/B-Zähler		[R]	A	B

[1] Ein SB mit nur zwei Digitaleingängen bietet nur die Eingänge 4.0 und 4.1.

Hinweis

Die von schnellen Zählern verwendeten digitalen E/A werden während der CPU-Gerätekonfiguration zugewiesen. Wenn HSC-Geräten Adressen digitaler E/A zugewiesen werden, können die Adresswerte der zugewiesenen E/A nicht durch die Forcefunktion einer Beobachtungstabelle geändert werden.

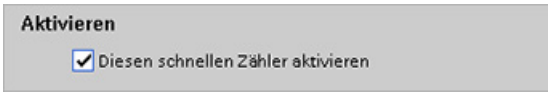
10.1.4 Konfiguration eines schnellen Zählers



Es können bis zu 6 schnelle Zähler konfiguriert werden. Öffnen Sie die CPU-Gerätekonfiguration und weisen Sie die Eigenschaften der einzelnen HSC zu.

Aktivieren Sie einen HSC durch Auswahl der Option "Aktivieren" für den HSC.

Mit der Anweisung CTRL_HSC und/oder CTRL_HSC_EXT in Ihrem Anwenderprogramm können Sie die Funktionsweise des HSC steuern.



⚠ WARNUNG

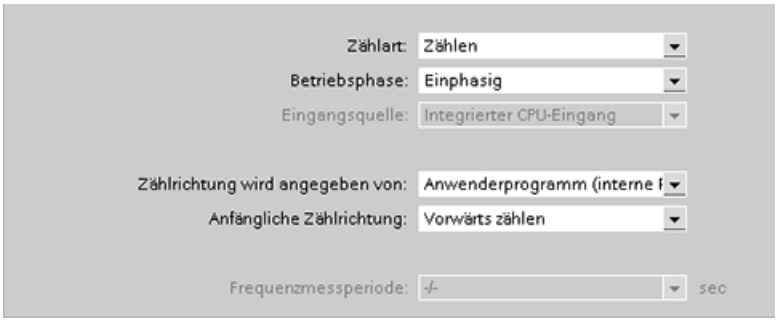
Gefahren beim Ändern der Filterzeiteinstellung für digitale Eingangskanäle

Wenn die bisherige Einstellung der Filterzeit eines Digitaleingangskanals verändert wird, muss möglicherweise bis zu 20,0 ms lang ein neuer Eingangswert mit Pegel "0" vorhanden sein, damit der Filter vollständig auf neue Eingänge reagiert. Während dieses Zeitraums werden kurze Impulse mit Pegel "0", die kürzer als 20,0 ms sind, möglicherweise nicht erkannt oder gezählt.

Das Verändern der Filterzeiten kann ein unerwartetes Verhalten der Maschine oder des Prozesses verursachen, was zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen kann.

Damit die neue Filterzeit sofort wirksam wird, ist die CPU aus- und wieder einzuschalten.

Nach der Aktivierung des HSC konfigurieren Sie die anderen Parameter wie Zählerfunktion, Anfangswerte, Rücksetzoptionen und Interruptereignisse.



Weitere Informationen zum Konfigurieren des HSC finden Sie im Abschnitt zum Konfigurieren der CPU (Seite 168).

10.2 PID-Regelung

STEP 7 bietet die folgenden PID-Anweisungen für die S7-1200 CPU:

- Die Anweisung PID_Compact dient zum Regeln technischer Prozesse mit kontinuierlichen Eingangs- und Ausgangsvariablen.
- Die Anweisung PID_3Step dient zum Regeln von motorbetätigten Geräten wie Ventilen, die digitale Signale zum Öffnen und Schließen benötigen.
- Die Anweisung PID_Temp bietet einen universellen PID-Regler für die speziellen Anforderungen der Temperaturregelung.

Hinweis

Änderungen, die Sie an der PID-Konfiguration vornehmen und im Betriebszustand RUN ins Zielsystem laden, werden erst dann wirksam, wenn die CPU von STOP in RUN geht. Im Dialog "PID-Parameter" über "Startwertsteuerung" vorgenommene Änderungen werden sofort wirksam.

Alle drei PID-Anweisungen (PID_Compact, PID_3Step und PID_Temp) können den P-, I- und D-Anteil während des Anlaufs berechnen (sofern die "Erstoptimierung" konfiguriert ist). Sie können die Anweisung auch für die "Feineinstellung" konfigurieren, um die Parameter zu optimieren. Sie brauchen die Parameter nicht manuell festzulegen.

Hinweis

Führen Sie die PID-Anweisung in einem wiederholten konstanten Zeitintervall der Abtastzeit aus (nach Möglichkeit in einem Zyklus-OB).

Weil der PID-Regler eine bestimmte Zeit benötigt, um auf Änderungen des Stellwerts zu reagieren, berechnen Sie den Ausgangswert nicht in jedem Zyklus. Führen Sie die PID-Anweisung nicht im Zyklus-OB des Hauptprogramms aus (z. B. OB 1).

Die Abtastzeit des PID-Algorithmus stellt die Zeit zwischen zwei Berechnungen des Ausgangswerts (Stellwerts) dar. Der Ausgangswert wird während der Selbsteinstellung berechnet und auf ein Vielfaches der Zykluszeit gerundet. Alle anderen Funktionen der PID-Anweisung werden bei jedem Aufruf ausgeführt.

PID-Algorithmus

Der PID-Regler (Proportional/Integral/Differential) misst das Zeitintervall zwischen zwei Aufrufen und wertet dann die Ergebnisse aus, um die Abtastzeit zu überwachen. Bei jedem Wechsel des Betriebszustands sowie beim ersten Anlauf wird ein Mittelwert der Abtastzeit errechnet. Dieser Wert dient als Referenzwert für die Überwachungsfunktion und zur Berechnung. Bei der Überwachung wird die aktuelle Messzeit zwischen zwei Aufrufen und der Mittelwert der für den Regler festgelegten Abtastzeit überwacht.

Der Ausgangswert des PID-Reglers besteht aus drei Anteilen:

- P (Proportional): Bei Berechnung mit P-Anteil ist der Ausgangswert proportional zur Differenz zwischen dem Sollwert und dem Prozesswert (Eingangswert).
- I (Integral): Bei Berechnung mit I-Anteil steigt der Ausgangswert proportional zur Dauer der Differenz zwischen dem Sollwert und dem Prozesswert (Eingangswert), um letztlich die Differenz zu korrigieren.
- D (Differential): Bei Berechnung mit D-Anteil steigt der Ausgangswert als Funktion der zunehmenden Änderungsgeschwindigkeit der Differenz zwischen dem Sollwert und dem Prozesswert (Eingangswert). Der Ausgangswert wird so schnell wie möglich an den Sollwert angeglichen.

Der PID-Regler berechnet den Ausgangswert für die Anweisung PID_Compact anhand folgender Formel.

$$y = K_p \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

y	Ausgangswert	x	Prozesswert
w	Sollwert	s	Laplace-Operator
K_p	Proportionale Verstärkung (P-Anteil)	a	Koeffizient für den Differenzierverzug (D-Anteil)
T_1	Integralzeit (I-Anteil)	b	Gewichtung des Proportionalanteils (P-Anteil)
T_D	Differentialzeit (D-Anteil)	c	Gewichtung des Differentialanteils (D-Anteil)

Der PID-Regler berechnet den Ausgangswert für die Anweisung PID_3Step anhand folgender Formel.

$$\Delta y = K_p \cdot s \cdot \left[(b \cdot w - x) + \frac{1}{T_I \cdot s} (w - x) + \frac{T_D \cdot s}{a \cdot T_D \cdot s + 1} (c \cdot w - x) \right]$$

y	Ausgangswert	x	Prozesswert
w	Sollwert	s	Laplace-Operator
K_p	Proportionale Verstärkung (P-Anteil)	a	Koeffizient für den Differenzierverzug (D-Anteil)
T_1	Integralzeit (I-Anteil)	b	Gewichtung des Proportionalanteils (P-Anteil)
T_D	Differentialzeit (D-Anteil)	c	Gewichtung des Differentialanteils (D-Anteil)

10.2.1 Anweisung PID und Technologieobjekt einfügen

STEP 7 bietet zwei Anweisungen für den PID-Regler:

- Die Anweisung PID_Compact und das zugehörige Technologieobjekt bieten einen universellen PID-Regler mit Einstellung. Das Technologieobjekt enthält alle Einstellungen für den Regelkreis.
- Die Anweisung PID_3Step und das zugehörige Technologieobjekt bieten einen PID-Regler mit bestimmten Einstellungen für motorbetätigte Ventile. Das Technologieobjekt enthält alle Einstellungen für den Regelkreis. Der Regler PID_3Step bietet zwei zusätzliche Boolesche Ausgänge.

Nach dem Anlegen des Technologieobjekts müssen Sie die Parameter konfigurieren (Seite 516). Sie passen auch die Parameter für die Selbsteinstellung an ("Erstoptimierung" während des Anlaufs oder manuelle "Feineinstellung"), um den PID-Regler in Betrieb zu nehmen (Seite 535).

Tabelle 10- 16 Anweisung PID und Technologieobjekt einfügen

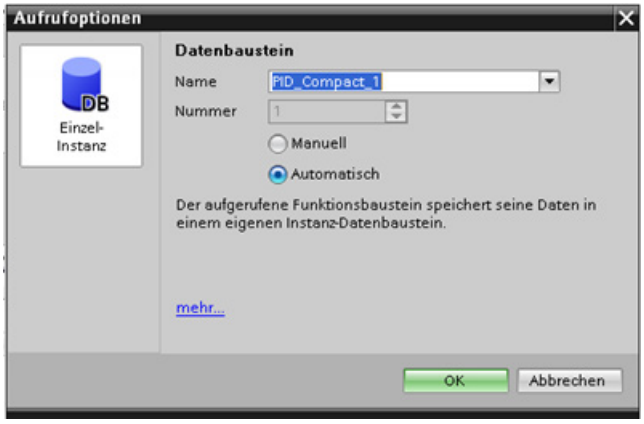
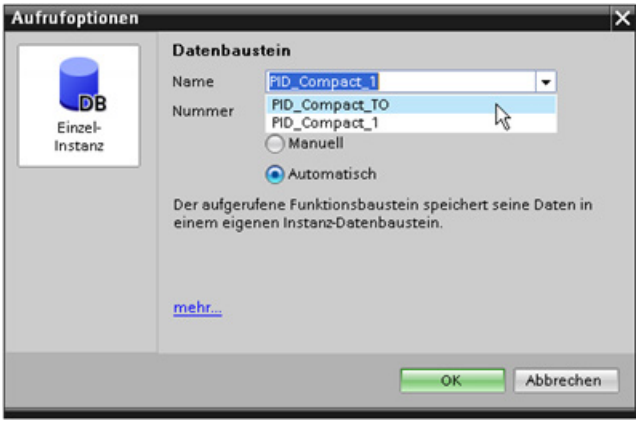
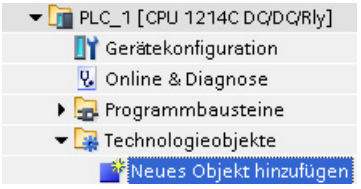
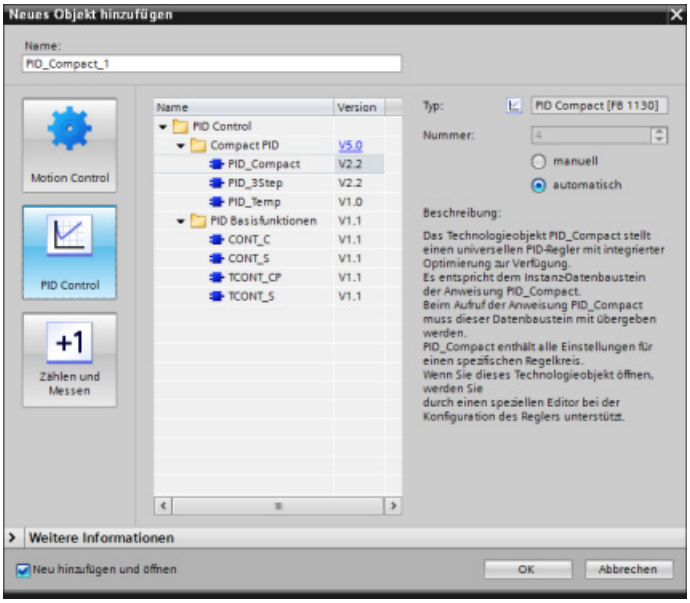
Wenn Sie eine PID-Anweisung in Ihr Anwenderprogramm einfügen, legt STEP 7 automatisch ein Technologieobjekt und einen Instanz-DB für die Anweisung an. Der Instanz-DB enthält alle Parameter für die PID-Anweisung. Jede PID-Anweisung benötigt einen eigenen, eindeutigen Instanz-DB, um ordnungsgemäß zu funktionieren. Nach dem Einfügen der PID-Anweisung und dem Erstellen des Technologieobjekts sowie des Instanz-DBs konfigurieren Sie die Parameter für das Technologieobjekt (Seite 516).	

Tabelle 10- 17 (Optional) Technologieobjekt in der Projektnavigation anlegen

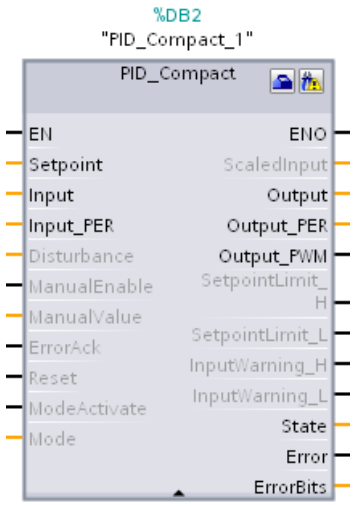
Sie können Technologieobjekte für Ihr Projekt auch **vor** dem Einfügen der PID-Anweisung anlegen. Wenn Sie das Technologieobjekt vor dem Einfügen der PID-Anweisung in Ihr Anwenderprogramm anlegen, können Sie das Technologieobjekt später beim Einfügen der PID-Anweisung auswählen.	
Um ein Technologieobjekt auszuwählen, doppelklicken Sie in der Projektnavigation auf das Symbol "Neues Objekt hinzufügen".	
Klicken Sie auf das Symbol "Regeln" und wählen Sie das Technologieobjekt für die Art des PID-Reglers aus (PID_Compact oder PID_3Step). Sie können wahlweise auch einen Namen für das Technologieobjekt eingeben. Klicken Sie auf "OK", um das Technologieobjekt anzulegen.	

10.2.2 Operation PID_Compact

Die Anweisung PID_Compact bietet einen universellen PID-Regler mit integrierter Selbsteinstellung für den Automatik- und Handbetrieb.

Tabelle 10- 18 Anweisung PID_Compact

KOP/FUP	SCL	Beschreibung

	`"PID_Compact_1"(`
	`  Setpoint:=_real_in_,`
	`  Input:=_real_in_,`
	`  Input_PER:=_word_in_,`
	`  Disturbance:=_real_in_,`
	`  ManualEnable:=_bool_in_,`
	`  ManualValue:=_real_in_,`
	`  ErrorAck:=_bool_in_,`
	`  Reset:=_bool_in_,`
	`  ModeActivate:=_bool_in_,`
	`  Mode:=_int_in_,`
	`  ScaledInput=>_real_out_,`
	`  Output=>_real_out_,`
	`  Output_PER=>_word_out_,`
	`  Output_PWM=>_bool_out_,`
	`  SetpointLimit_H=>_bool_out_,`
	`  SetpointLimit_L=>_bool_out_,`
	`  InputWarn-`
	`ing_H=>_bool_out_,`
	`  InputWarn-`
	`ing_L=>_bool_out_,`
	`  State=>_int_out_,`
	`  Error=>_bool_out_,`
	`  ErrorBits=>_dword_out_);`
		PID_Compact stellt einen PID-Regler mit Selbsteinstellung für den Automatik- und Handbetrieb bereit. PID_Compact ist ein PIDT1-Regler mit Anti-Windup und Gewichtung des P- und D-Anteils.

1 STEP 7 erstellt das Technologieobjekt und den Instanz-DB automatisch, wenn Sie die Anweisung einfügen. Der Instanz-DB enthält die Parameter des Technologieobjekts.

2 Im SCL-Beispiel ist "PID_Compact_1" der Name des Instanz-DBs.

Tabelle 10- 19 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
Setpoint	IN	Real	Sollwert des PID-Reglers im Automatikbetrieb. (Standardwert: 0.0)
Input	IN	Real	Eine Variable des Anwenderprogramms wird als Quelle des Prozesswerts verwendet. (Standardwert: 0.0) Wenn Sie den Parameter Input verwenden, müssen Sie Config.InputPerOn = FALSE einstellen.
Input_PER	IN	Word	Ein Analogeingang wird als Quelle des Prozesswerts verwendet. (Standardwert: W#16#0) Wenn Sie den Parameter Input_PER verwenden, müssen Sie Config.InputPerOn = TRUE einstellen.
Disturbance	IN	Real	Störgröße oder Vorsteuerungswert
ManualEnable	IN	Bool	Aktiviert oder deaktiviert den Handbetrieb. (Standardwert: FALSE): • Eine Flanke von FALSE nach TRUE aktiviert den "Handbetrieb", während sich bei Zustand = 4 die Betriebsart nicht verändert. Solange ManualEnable = TRUE ist, können Sie die Betriebsart nicht mit einer steigenden Flanke an ModeActivate wechseln und auch nicht das Dialogfeld für die Inbetriebnahme verwenden. • Eine Flanke von TRUE nach FALSE aktiviert die von Mode zugewiesene Betriebsart. Hinweis: Es wird empfohlen, die Betriebsart lediglich über ModeActivate zu ändern.
ManualValue	IN	Real	Ausgangswert für Handbetrieb. (Standardwert: 0.0) Verwendbar sind Werte von Config.OutputLowerLimit bis Config.OutputUpperLimit.
ErrorAck	IN	Bool	Setzt die ErrorBits und Warnausgänge zurück. Flanke von FALSE nach TRUE
Reset	IN	Bool	Startet den Regler neu. (Standardwert: FALSE): • Flanke von FALSE nach TRUE: – Wechselt in den "inaktiven" Modus – Setzt die ErrorBits und Warnausgänge zurück – Löscht den Integralanteil – Speichert die PID-Parameter • Solange Reset = TRUE ist, bleibt PID_Compact im "inaktiven" Modus (Zustand = 0). • Flanke von TRUE nach FALSE: – PID_Compact wechselt in die Betriebsart, die im Parameter Mode gespeichert ist.
ModeActivate	IN	Bool	PID_Compact schaltet in den Betriebszustand, der im Parameter Mode gespeichert ist. Flanke von FALSE nach TRUE:
Mode	IN	Int	Die gewünschte PID-Betriebsart; wird bei einer steigenden Flanke am Eingang Mode Activate aktiviert.
ScaledInput	OUT	Real	Skalierter Prozesswert. (Standardwert: 0.0)
Output[1]	OUT	Real	Ausgangswert im Format REAL. (Standardwert: 0.0)
Output_PER[1]	OUT	Word	Analoger Ausgangswert. (Standardwert: W#16#0)

Parameter und Datentyp		Datentyp	Beschreibung
Output_PWM[1]	OUT	Bool	Ausgangswert für die Impulsdauermodulation. (Standardwert: FALSE) Der Ausgangswert wird aus Ein- und Aus-Zeiten gebildet.
SetpointLimit_H	OUT	Bool	Sollwert oberer Grenzwert. (Standardwert: FALSE) Wenn SetpointLimit_H = TRUE ist, dann ist der absolute obere Grenzwert des Sollwerts erreicht (Setpoint ≥ Config.SetpointUpperLimit). Der Sollwert ist auf Config.SetpointUpperLimit begrenzt.
SetpointLimit_L	OUT	Bool	Sollwert unterer Grenzwert. (Standardwert: FALSE) Wenn SetpointLimit_L = TRUE ist, dann ist der absolute untere Grenzwert des Sollwerts erreicht (Setpoint ≥ Config.SetpointLowerLimit). Der Sollwert ist auf Config.SetpointLowerLimit begrenzt.
InputWarning_H	OUT	Bool	Wenn InputWarning_H = TRUE, hat der Prozesswert die obere Warngrenze erreicht oder überschritten. (Standardwert: FALSE)
InputWarning_L	OUT	Bool	Wenn InputWarning_L = TRUE, hat der Prozesswert die untere Warngrenze erreicht oder unterschritten. (Standardwert: FALSE)
State	OUT	Int	Aktuelle Betriebsart des PID-Reglers. (Standardwert: 0) Die Betriebsart können Sie mit dem Eingangsparameter Mode und einer steigenden Flanke an ModeActivate ändern: • State = 0: Inaktiv • State = 1: Erstoptimierung • State = 2: Manuelle Feineinstellung • State = 3: Automatikbetrieb • State = 4: Handbetrieb • State = 5: Ersatzausgangswert mit Fehlerüberwachung
Error	OUT	Bool	Wenn Error = TRUE, steht mindestens eine Fehlermeldung in diesem Zyklus an. (Standardwert: FALSE) Hinweis: Der Parameter Error in V1.x PID war das Feld der ErrorBits mit den Fehlercodes. Jetzt zeigt ein Boolescher Merker an, dass ein Fehler aufgetreten ist.
ErrorBits	OUT	DWord	In der Parametertabelle (Seite 489) der ErrorBits für die Anweisung PID_Compact werden die anstehenden Fehlermeldungen beschrieben. (Standardwert: DW#16#0000 (kein Fehler)). ErrorBits ist remanent und wird bei einer steigenden Flanke an Reset oder ErrorAck zurückgesetzt. Hinweis: In V1.x war der Parameter ErrorBits als Parameter Error definiert und nicht vorhanden.

[1] Sie können die Ausgänge der Parameter Output, Output_PER und Output_PWM parallel verwenden.

Funktionsweise des Reglers PID_Compact

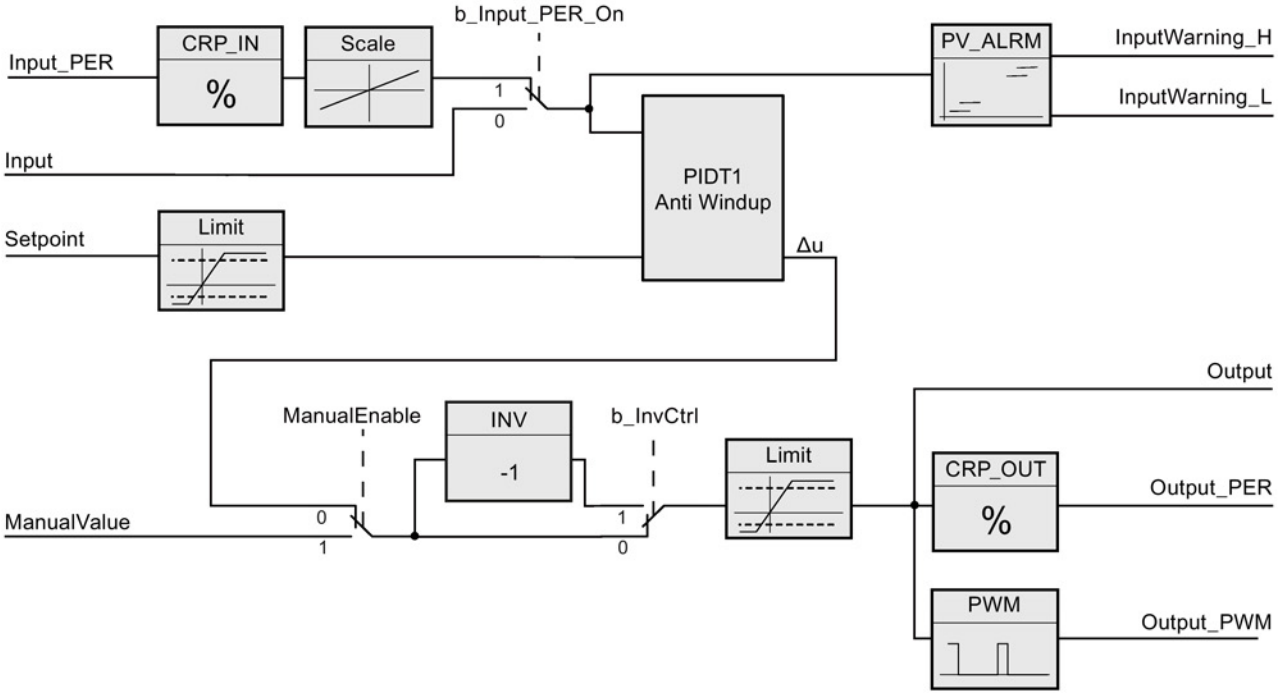


Bild 10-1 Funktionsweise des Reglers PID_Compact

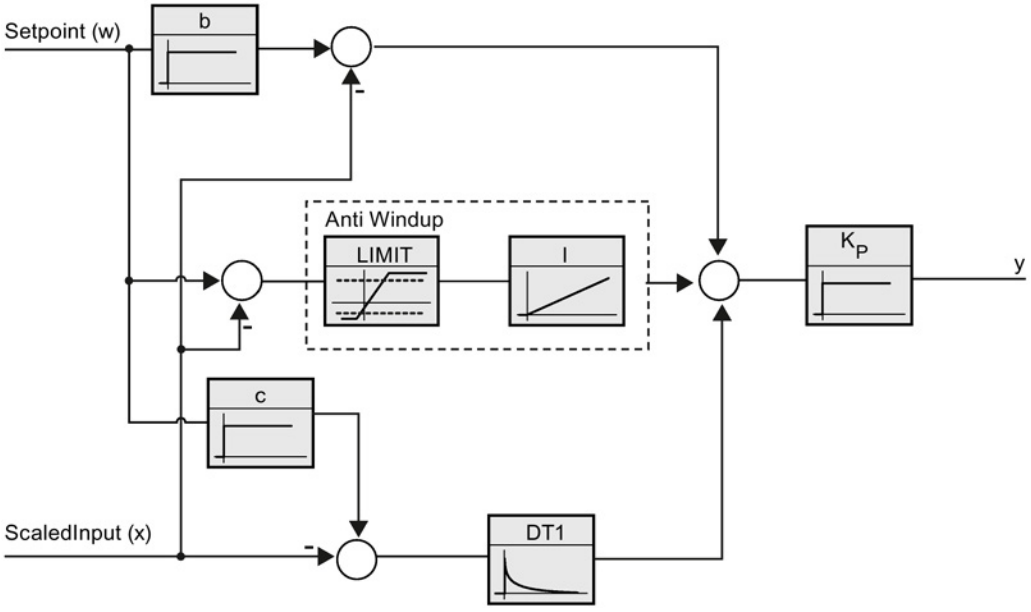


Bild 10-2 Funktionsweise des Reglers PID_Compact als PIDT1-Regler mit Anti-Windup

10.2.3 ErrorBit-Parameter der Anweisung PID_Compact

Stehen mehrere Fehler an, werden die Werte der Fehlercodes mittels binärer Addition angezeigt. Die Anzeige von Fehlercode 0003 beispielsweise weist darauf hin, dass auch die Fehler 0001 und 0002 anstehen.

Tabelle 10- 20 ErrorBit-Parameter der Anweisung PID_Compact

ErrorBit (DW#16#...)	Beschreibung
0000	Kein Fehler
0001 [1, 2]	Der Parameter Input liegt außerhalb der Prozesswertgrenzen. Input > Config.InputUpperLimit Input < Config.InputLowerLimit
0002 [2, 3]	Ungültiger Wert am Parameter Input_PER. Prüfen Sie, ob am Analogeingang ein Fehler anliegt.
0004 [4]	Fehler während der Feineinstellung. Schwingung des Prozesswerts konnte nicht erhalten werden.
0008 [4]	Fehler beim Start der Erstoptimierung. Der Prozesswert ist zu nah am Sollwert. Starten Sie die Feineinstellung.
0010 [4]	Der Sollwert wurde während der Einstellung geändert. Hinweis: Sie können die zulässige Schwankung am Sollwert in der Variablen CancelTuningLevel festlegen.
0020	Während der Feineinstellung ist keine Erstoptimierung zulässig. Hinweis: Wenn vor dem Auftreten des Fehlers ActivateRecoverMode = TRUE war, bleibt PID_Compact im Feineinstellungsmodus.
0080 [4]	Fehler während der Erstoptimierung. Inkorrekte Konfiguration der Grenzen des Ausgangswerts. Prüfen Sie, ob die Grenzen des Ausgangswerts richtig konfiguriert sind, und passen Sie die Steuerungslogik an.
0100 [4]	Fehler während der Feineinstellung führte zu ungültigen Parametern.
0200 [2, 3]	Ungültiger Wert am Parameter Input: Wert hat ungültiges Zahlenformat.
0400 [2, 3]	Berechnung des Ausgangswerts fehlgeschlagen. Prüfen Sie die PID-Parameter.
0800 [1, 2]	Abtastzeitfehler: PID_Compact wird nicht während der Abtastzeit des Weckalarm-OBs aufgerufen.
1000 [2, 3]	Ungültiger Wert am Parameter Setpoint: Wert hat ungültiges Zahlenformat.
10000	Ungültiger Wert am Parameter ManualValue: Wert hat ungültiges Zahlenformat. Hinweis: Wenn vor dem Auftreten des Fehlers ActivateRecoverMode = TRUE war, verwendet PID_Compact SubstituteOutput als Ausgangswert. Sobald Sie im Parameter ManualValue einen gültigen Wert zuweisen, verwendet PID_Compact diesen als Ausgangswert.

ErrorBit (DW#16#...)	Beschreibung
20000	Ungültiger Wert der Variablen SubstituteValue: Wert hat ungültiges Zahlenformat. PID_Compact verwendet die untere Grenze des Ausgangswerts als Ausgangswert. Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv war, ActivateRecoverMode = TRUE ist und der Fehler nicht mehr ansteht, gehtPID_Compact wieder in den Automatikbetrieb.
40000	Ungültiger Wert am Parameter Disturbance: Wert hat ungültiges Zahlenformat. Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv und ActivateRecoverMode = FALSE war, wird Disturbance auf null gesetzt. PID_Compact bleibt im Automatikbetrieb. Hinweis: Wenn vor dem Auftreten des Fehlers die Erstoptimierung oder die Feineinstellung aktiv und ActivateRecoverMode = TRUE war, wechselt PID_Compact in die im Parameter Mode gespeicherte Betriebsart. Wenn Störungen in der aktuellen Phase keine Auswirkungen auf den Ausgangswert haben, wird die Einstellung nicht abgebrochen.

[1] Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv und ActivateRecoverMode = TRUE war, bleibt PID_Compact im Automatikbetrieb.

[2] Hinweis: Wenn vor dem Auftreten des Fehlers die Erstoptimierung oder die Feineinstellung aktiv war und ActivateRecoverMode = TRUE ist, wechselt PID_Compact in die im Parameter Mode gespeicherte Betriebsart.

[3] Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv war und ActivateRecoverMode = TRUE ist, gibt PID_Compact den konfigurierten Ersatzausgangswert aus. Sobald der Fehler nicht mehr ansteht, wechselt PID_Compact wieder in den Automatikbetrieb.

[4] Hinweis: Wenn vor dem Auftreten des Fehlers ActivateRecoverMode = TRUE war, bricht PID_Compact die Einstellung ab und wechselt in die im Parameter Mode gespeicherte Betriebsart.

10.2.4 Warnparameter der Anweisung PID_Compact

Liegen am PID-Regler mehrere Warnungen an, werden die Werte der Fehlercodes mittels binärer Addition angezeigt. Die Anzeige von Fehlercode 0003 beispielsweise weist darauf hin, dass die Fehler 0001 und 0002 anstehen.

Tabelle 10- 21 Parameter Warning der Anweisung PID_Compact

Warning (DW#16#...)	Beschreibung
0000	Keine Warnung vorhanden.
0001 [1]	Der Wendepunkt wurde in der Erstoptimierung nicht gefunden.
0002	Bei der Einstellung im Betrieb wurde die Schwingung erzwungen. (Der Parameter "Warning" unterdrückt diese Warnung, sie ist nur im Parameter “WarningInternal” zu Diagnosezwecken sichtbar.)
0004 [1]	Der Sollwert war auf die konfigurierten Grenzwerte begrenzt.
0008 [1]	Für die ausgewählte Berechnungsmethode waren nicht alle erforderlichen Reglereigenschaften definiert. Stattdessen wurden die PID-Parameter mit der Methode TIR.TuneRuleHeat / TIR.TuneRuleCool = 3 berechnet.
0010	Der Betriebszustand konnte nicht gewechselt werden, weil Reset = TRUE oder ManualEnable = TRUE.
0020	Die Zykluszeit des aufrufenden OB begrenzt die Abtastzeit des PID-Algorithmus. Durch kürzere OB-Zykluszeiten können Sie die Ergebnisse verbessern.
0040 [1]	Der Prozesswert hat einen seiner Warngrenzwerte überschritten.
0080	Ungültiger Wert an "Mode". Der Betriebszustand wird nicht gewechselt.
0100 [1]	Der manuelle Wert war auf die Grenzwerte des Reglerausgangs begrenzt.
0200	Die angegebene Regel für die Einstellung wird nicht unterstützt. Es werden keine PID-Parameter berechnet.
1000	Der Ersatzausgangswert kann nicht erreicht werden, weil er außerhalb der Grenzwerte für den Ausgangswert liegt.

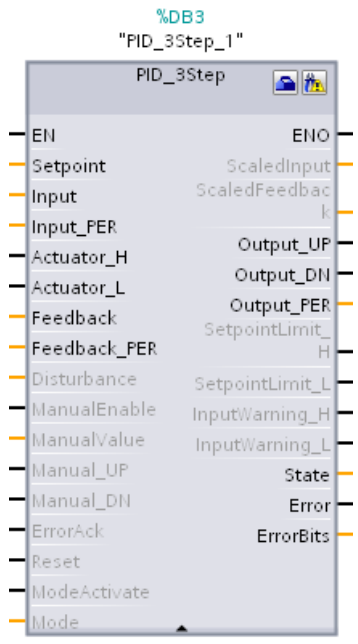
[1] Hinweis: Der PID-Regler hat, sobald die Ursache behoben war oder die Anwenderaktion mit gültigen Parametern wiederholt wurde, die folgenden Warnungen automatisch gelöscht: 0001, 0004, 0008, 0040 und 0100.

10.2.5 Anweisung PID_3Step

Die Anweisung PID_3Step konfiguriert einen PID-Regler mit Selbsteinstellungsfunktionen, der für motorbetätigte Ventile und Stellglieder optimiert wurde.

Tabelle 10- 22 Anweisung PID_3Step

KOP/FUP	SCL	Beschreibung

	`"PID_3Step_1"(`
	`  SetpoInt:=_real_in_,`
	`  Input:=_real_in_,`
	`  ManualValue:=_real_in_,`
	`  Feedback:=_real_in_,`
	`  InputPer:=_word_in_,`
	`  FeedbackPer:=_word_in_,`
	`Disturbance:=_real_in_,`
	`ManualEnable:=_bool_in_,`
	`  ManualUP:=_bool_in_,`
	`  ManualDN:=_bool_in_,`
	`  ActuatorH:=_bool_in_,`
	`  ActuatorL:=_bool_in_,`
	`  ErrorAck:=_bool_in_,`
	`Reset:=_bool_in_,`
	`  ModeActivate:=_bool_in_,`
	`Mode:=_int_in_,`
	`ScaledInput=>_real_out_,`
	`  ScaledFeedback=>_real_out_,`
	`  ErrorBits=>_dword_out_,`
	`  OutputPer=>_word_out_,`
	`  State=>_int_out_,`
	`  OutputUP=>_bool_out_,`
	`  OutputDN=>_bool_out_,`
	`  SetpoIntLimitH=>_bool_out_,`
	`  SetpoIntLimitL=>_bool_out_,`
	`  InputWarningH=>_bool_out_,`
	`  InputWarningL=>_bool_out_,`
	`  Error=>_bool_out_,`
	`ErrorBits=>_dword_out_);`
		PID_3Step konfiguriert einen PID-Regler mit Selbsteinstellungsfunktionen, der für motorbetätigte Ventile und Stellglieder optimiert wurde. Er bietet zwei Boolesche Ausgänge.
		PID_3Step ist ein PID T1-Regler mit Anti-Windup und Gewichtung des P- und D-Anteils.

[1] STEP 7 erstellt das Technologieobjekt und den Instanz-DB automatisch, wenn Sie die Anweisung einfügen. Der Instanz-DB enthält die Parameter des Technologieobjekts.

[2] Im SCL-Beispiel ist "PID_3Step_1" der Name des Instanz-DBs.

Tabelle 10- 23 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
Setpoint	IN	Real	Sollwert des PID-Reglers im Automatikbetrieb. (Standardwert: 0.0)
Input	IN	Real	Eine Variable des Anwenderprogramms wird als Quelle des Prozesswerts verwendet. (Standardwert: 0.0) Wenn Sie den Parameter Input verwenden, müssen Sie Config.InputPerOn = FALSE einstellen.
Input_PER	IN	Word	Ein Analogeingang wird als Quelle des Prozesswerts verwendet. (Standardwert: W#16#0) Wenn Sie den Parameter Input_PER verwenden, müssen Sie Config.InputPerOn = TRUE einstellen.
Actuator_H	IN	Bool	Digitale Positionsrückmeldung des Ventils für den oberen Endpunkt Wenn Actuator_H = TRUE ist, befindet sich das Ventil am oberen Endpunkt und wird nicht weiter in diese Richtung bewegt. (Standardwert: FALSE)
Actuator_L	IN	Bool	Digitale Positionsrückmeldung des Ventils für den unteren Endpunkt Wenn Actuator_L = TRUE ist, befindet sich das Ventil am unteren Endpunkt und wird nicht weiter in diese Richtung bewegt. (Standardwert: FALSE)
Feedback	IN	Real	Zurückgemeldete Position des Ventils. (Standardwert: 0.0) Wenn Sie den Parameter Feedback verwenden, müssen Sie Config.FeedbackPerOn = FALSE einstellen.
Feedback_PER	IN	Int	Zurückgemeldeter Analogwert der Ventilposition. (Standardwert: W#16#0) Wenn Sie den Parameter Feedback_PER verwenden, müssen Sie Config.FeedbackPerOn = TRUE. einstellen. Feedback_PER wird basierend auf den folgenden Variablen skaliert: • Config.FeedbackScaling.LowerPointIn • Config.FeedbackScaling.UpperPointIn • Config.FeedbackScaling.LowerPointOut • Config.FeedbackScaling.UpperPointOut
Disturbance	IN	Real	Störgröße oder Vorsteuerungswert
ManualEnable	IN	Bool	Aktiviert oder deaktiviert den Handbetrieb. (Standardwert: FALSE): • Eine Flanke von FALSE nach TRUE aktiviert den "Handbetrieb", während sich bei Zustand = 4 die Betriebsart nicht verändert. Solange ManualEnable = TRUE ist, können Sie die Betriebsart nicht mit einer steigenden Flanke an ModeActivate wechseln und auch nicht das Dialogfeld für die Inbetriebnahme verwenden. • Eine Flanke von TRUE nach FALSE aktiviert die von Mode zugewiesene Betriebsart. Hinweis: Es wird empfohlen, die Betriebsart lediglich über ModeActivate zu ändern.
ManualValue	IN	Real	Prozesswert für Handbetrieb. (Standardwert: 0.0) Im Handbetrieb geben Sie die absolute Position des Ventils an. ManualValue wird nur ausgewertet, wenn Sie OutputPer verwenden **oder** wenn eine zurückgemeldete Position verfügbar ist.

Parameter und Datentyp		Datentyp	Beschreibung
ManualUP	IN	Bool	• Manual_UP = TRUE: – Das Ventil wird geöffnet, auch wenn Sie Output_PER oder eine Positionsrückmeldung verwenden. Das Ventil wird nicht weiter bewegt, wenn der obere Endpunkt erreicht ist. – Siehe auch Config.VirtualActuatorLimit • Manual_UP = FALSE: – Wenn Sie Output_PER oder eine Positionsrückmeldung verwenden, wird das Ventil auf ManualValue bewegt. Andernfalls wird das Ventil nicht weiter bewegt. Hinweis: Wenn Manual_UP und Manual_DN gleichzeitig auf TRUE gesetzt werden, wird das Ventil nicht bewegt.
ManualDN	IN	Bool	• Manual_DN = TRUE: – Das Ventil wird geöffnet, auch wenn Sie Output_PER oder eine Positionsrückmeldung verwenden. Das Ventil wird nicht weiter bewegt, wenn der obere Endpunkt erreicht ist. – Siehe auch Config.VirtualActuatorLimit • Manual_DN = FALSE: – Wenn Sie Output_PER oder eine Positionsrückmeldung verwenden, wird das Ventil auf ManualValue bewegt. Andernfalls wird das Ventil nicht weiter bewegt.
ErrorAck	IN	Bool	Setzt die ErrorBits und Warnausgänge zurück. Flanke von FALSE nach TRUE
Reset	IN	Bool	Startet den Regler neu. (Standardwert: FALSE): • Flanke von FALSE nach TRUE: – Wechselt in den "inaktiven" Modus – Setzt die ErrorBits und Warnausgänge zurück – Löscht den Integralanteil – Speichert die PID-Parameter • Solange Reset = TRUE ist, bleibt PID_3Step im "inaktiven" Modus (Zustand = 0). • Flanke von TRUE nach FALSE: – PID_3Step wechselt in die Betriebsart, die im Parameter Mode gespeichert ist.
ModeActivate	IN	Bool	PID_3Step schaltet in den Betriebszustand, der im Parameter Mode gespeichert ist. Flanke von FALSE nach TRUE:
Mode	IN	Int	Die gewünschte PID-Betriebsart; wird bei einer steigenden Flanke am Eingang Mode Activate aktiviert.
ScaledInput	OUT	Real	Skalierter Prozesswert
ScaledFeedback	OUT	Real	Skalierte Positionsrückmeldung des Ventils Hinweis: Für einen Aktor ohne Positionsrückmeldung ist die mit ScaledFeedback angegebene Stellung des Aktors sehr ungenau. ScaledFeedback ist in diesem Fall nur für eine grobe Schätzung der aktuellen Position verwendbar.

Parameter und Datentyp		Datentyp	Beschreibung
Output_UP	OUT	Bool	Digitaler Ausgangswert zum Öffnen des Ventils. (Standardwert: FALSE) Wenn Config.OutputPerOn = FALSE, dann wird der Parameter Output_UP verwendet.
Output_DN	OUT	Bool	Digitaler Ausgangswert zum Schließen des Ventils. (Standardwert: FALSE) Wenn Config.OutputPerOn = FALSE, dann wird der Parameter Output_DN verwendet.
Output_PER	OUT	Word	Analoger Ausgangswert. Wenn Config.OutputPerOn = TRUE, dann wird der Parameter Output_PER verwendet.
SetpointLimitH	OUT	Bool	Sollwert oberer Grenzwert. (Standardwert: FALSE) Wenn SetpointLimitH = TRUE, dann ist der absolute obere Grenzwert des Sollwerts erreicht (Setpoint ≥ Config.SetpointUpperLimit). Hinweis: Der Sollwert ist auf (Setpoint ≥ Config.SetpointUpperLimit) begrenzt.
SetpointLimitL	OUT	Bool	Sollwert unterer Grenzwert. (Standardwert: FALSE) Wenn SetpointLimitL = TRUE, dann ist der absolute untere Grenzwert des Sollwerts erreicht (Setpoint ≥ Config.SetpointLowerLimit). Hinweis: Der Sollwert ist auf (Setpoint ≥ Config.SetpointLowerLimit) begrenzt.
InputWarningH	OUT	Bool	Wenn InputWarningH = TRUE, hat der Eingangswert die obere Warngrenze erreicht oder überschritten. (Standardwert: FALSE)
InputWarningL	OUT	Bool	Wenn InputWarningL = TRUE, hat der Eingangswert die untere Warngrenze erreicht oder überschritten. (Standardwert: FALSE)
State	OUT	Int	Aktuelle Betriebsart des PID-Reglers. (Standardwert: 0) Die Betriebsart können Sie mit dem Eingangsparameter Mode und einer steigenden Flanke an ModeActivate: ändern: • State = 0: Inaktiv • State = 1: Erstoptimierung • State = 2: Manuelle Feineinstellung • State = 3: Automatikbetrieb • State = 4: Handbetrieb • State = 5: Ersatzausgangswert • State = 6: Übergangszeitmessung • State = 7: Fehlerüberwachung • State = 8: Ersatzausgangswert mit Fehlerüberwachung\ • State = 10: Handbetrieb ohne Endpunktsignale

Parameter und Datentyp		Datentyp	Beschreibung
Error	OUT	Bool	Wenn Error = TRUE, steht mindestens eine Fehlermeldung an. (Standardwert: FALSE) Hinweis: Der Parameter Error in V1.x PID war das Feld der ErrorBits mit den Fehlercodes. Jetzt zeigt ein Boolescher Merker an, dass ein Fehler aufgetreten ist.
ErrorBits	OUT	DWord	In der Parametertabelle (Seite 499) der ErrorBits für die Anweisung PID_3Step werden die anstehenden Fehlermeldungen beschrieben. (Standardwert: DW#16#0000 (kein Fehler)). ErrorBits ist remanent und wird bei einer steigenden Flanke an Reset oder ErrorAck zurückgesetzt. Hinweis: In V1.x war der Parameter ErrorBits als Parameter Error definiert und nicht vorhanden.

Funktionsweise des PID_3Step-Reglers

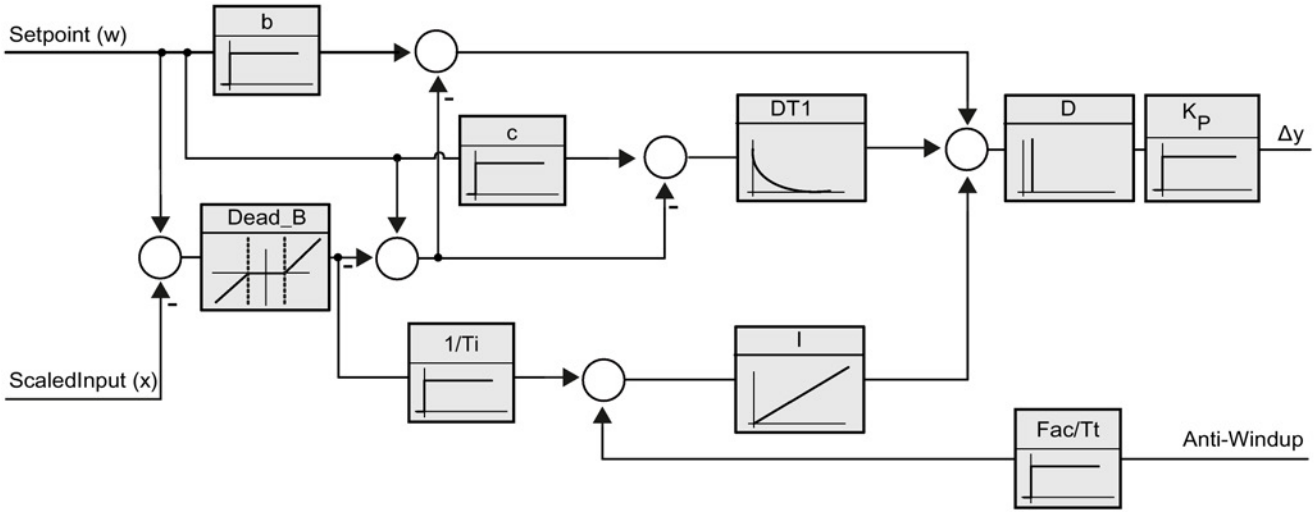


Bild 10-3 Funktionsweise des Reglers PID_3Step als PID T1-Regler mit Anti-Windup

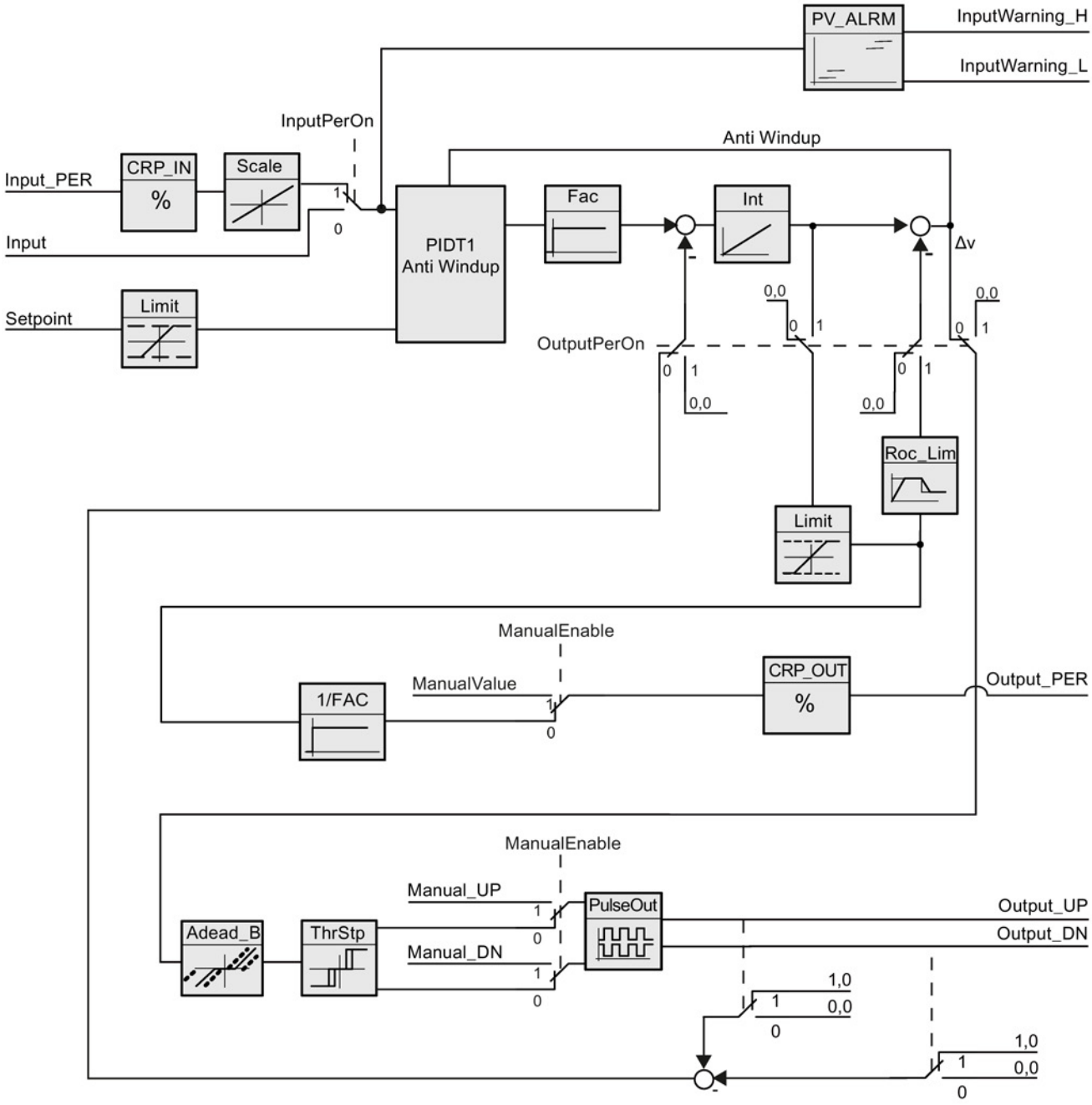


Bild 10-4 Funktionsweise des Reglers PID_3Step ohne Positionsrückmeldung

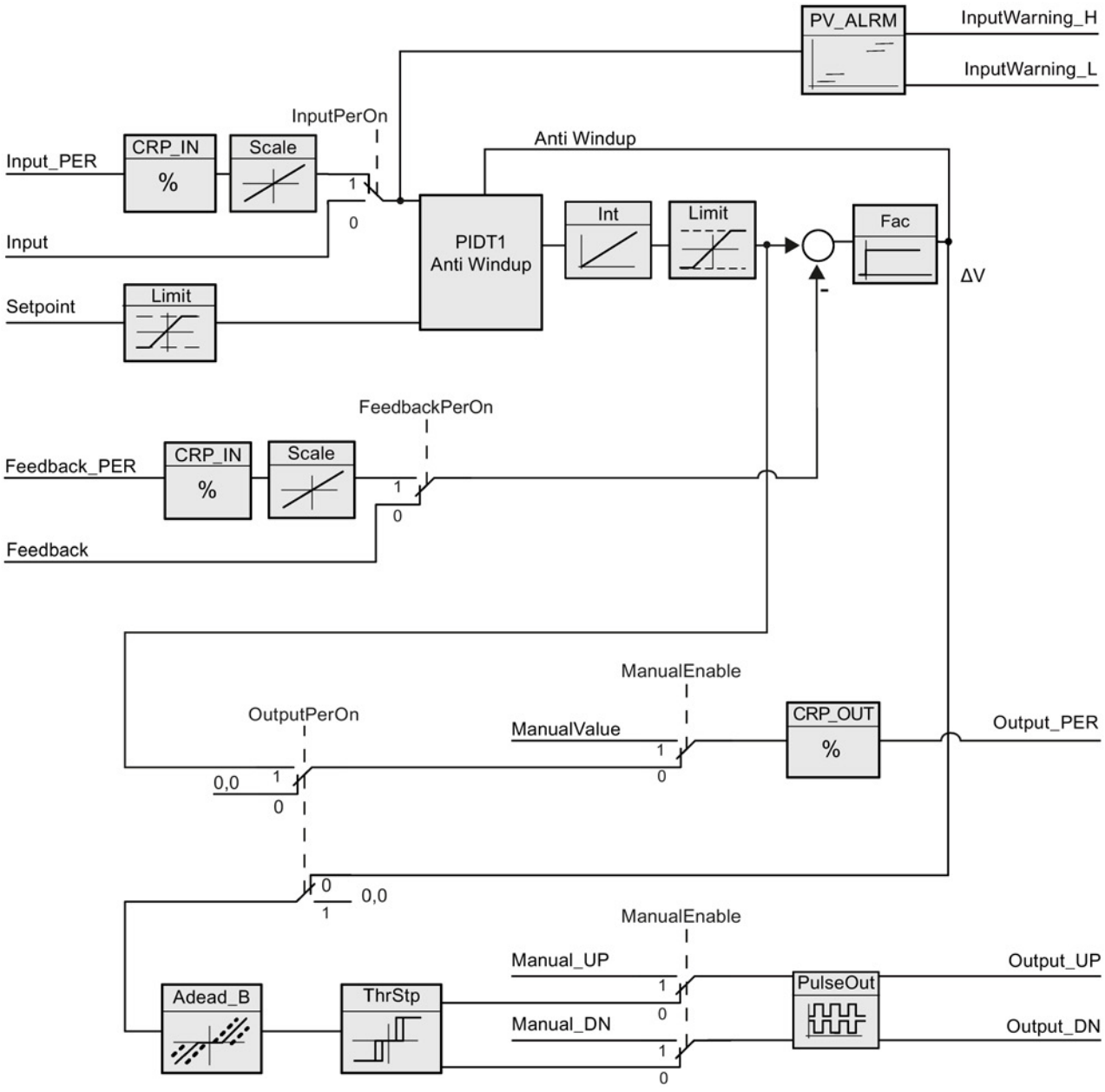


Bild 10-5 Funktionsweise des Reglers PID_3Step mit aktivierter Positionsrückmeldung

10.2.6 ErrorBit-Parameter der Anweisung PID_3Step

Stehen mehrere Fehler an, werden die Werte der Fehlercodes mittels binärer Addition angezeigt. Die Anzeige von Fehlercode 0003 beispielsweise weist darauf hin, dass auch die Fehler 0001 und 0002 anstehen.

Tabelle 10- 24 ErrorBit-Parameter der Anweisung PID_3STEP

ErrorBit (DW#16#...)	Beschreibung
0000	Kein Fehler
0001 [1, 2]	Der Parameter Input liegt außerhalb der Prozesswertgrenzen. Input > Config.InputUpperLimit Input < Config.InputLowerLimit
0002 [2, 3]	Ungültiger Wert am Parameter Input_PER. Prüfen Sie, ob am Analogeingang ein Fehler anliegt.
0004 [4]	Fehler während der Feineinstellung. Schwingung des Prozesswerts konnte nicht erhalten werden.
0010 [4]	Der Sollwert wurde während der Einstellung geändert. Hinweis: Sie können die zulässige Schwankung am Sollwert in der Variablen CancelTuningLevel festlegen.
0020	Während der Feineinstellung ist keine Erstoptimierung zulässig. Hinweis: Wenn vor dem Auftreten des Fehlers ActivateRecoverMode = TRUE war, bleibt PID_3Step im Feineinstellungsmodus.
0080 [4]	Fehler während der Erstoptimierung. Inkorrekte Konfiguration der Grenzen des Ausgangswerts. Prüfen Sie, ob die Grenzen des Ausgangswerts richtig konfiguriert sind, und passen Sie die Steuerungslogik an.
0100 [4]	Fehler während der Feineinstellung führte zu ungültigen Parametern.
0200 [2, 3]	Ungültiger Wert am Parameter Input: Wert hat ungültiges Zahlenformat.
0400 [2, 3]	Berechnung des Ausgangswerts schlug fehl. Prüfen Sie die PID-Parameter.
0800 [1, 2]	Abtastzeitfehler: PID_3Step wird nicht während der Abtastzeit des Weckalarm-OBs aufgerufen.
1000 [2, 3]	Ungültiger Wert am Parameter Setpoint: Wert hat ungültiges Zahlenformat.
2000 [1, 2, 5]	Ungültiger Wert am Parameter Feedback_PER. Prüfen Sie, ob am Analogeingang ein Fehler ansteht.
4000 [1, 2, 5]	Ungültiger Wert am Parameter Feedback: Wert hat ungültiges Zahlenformat.
8000 [1, 2]	Fehler bei der digitalen Positionsrückmeldung. Actuator_H = TRUE und Actuator_L = TRUE. Das Stellglied kann nicht auf den Ersatzausgangswert wechseln und bleibt in seiner aktuellen Position. In diesem Zustand ist kein Handbetrieb möglich. Um das Stellglied aus diesem Zustand zu bewegen, müssen Sie den "Endpunkt des Stellglieds" deaktivieren (Config.ActuatorEndStopOn = FALSE) oder in den Handbetrieb ohne Endpunktsignale wechseln (Mode = 10).

ErrorBit (DW#16#...)	Beschreibung
10000	Ungültiger Wert am Parameter ManualValue: Wert hat ungültiges Zahlenformat. Das Stellglied kann nicht auf den manuellen Wert wechseln und bleibt in seiner aktuellen Position. Weisen Sie in ManualValue einen gültigen Wert zu oder bewegen Sie das Stellglied im Handbetrieb mit Manual_UP und Manual_DN.
20000	Ungültiger Wert der Variablen SavePosition: Wert hat ungültiges Zahlenformat. Das Stellglied kann nicht auf den Ersatzausgangswert wechseln und bleibt in seiner aktuellen Position.
40000	Ungültiger Wert am Parameter Disturbance: Wert hat ungültiges Zahlenformat. Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv und ActivateRecoverMode = FALSE war, wird Disturbance auf null gesetzt. PID_3Step bleibt im Automatikbetrieb. Hinweis: Wenn vor dem Auftreten des Fehlers die Erstoptimierung oder die Feineinstellung aktiv und ActivateRecoverMode = TRUE war, wechselt PID_3Step in die im Parameter Mode gespeicherte Betriebsart. Wenn Störungen in der aktuellen Phase keine Auswirkungen auf den Ausgangswert haben, wird die Einstellung nicht abgebrochen. Der Fehler wirkt sich nicht während der Übergangszeitmessung aus.

[1] Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv und ActivateRecoverMode = TRUE war, bleibt PID_3Step im Automatikbetrieb.

[2] Hinweis: Wenn vor dem Auftreten des Fehlers die Erstoptimierung, die Feineinstellung oder die Übergangszeitmessung aktiv und ActivateRecoverMode = TRUE war, wechselt PID_3Step in die im Parameter Mode gespeicherte Betriebsart.

[3] Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv und ActivateRecoverMode = TRUE war, wechselt PID_3Step in die Betriebsart "Ersatzausgangswert mit Fehlerüberwachung" oder "Fehlerüberwachung". Sobald der Fehler nicht mehr ansteht, wechselt PID_3Step wieder in den Automatikbetrieb.

[4] Hinweis: Wenn vor dem Auftreten des Fehlers ActivateRecoverMode = TRUE war, bricht PID_3Step die Einstellung ab und wechselt in die im Parameter Mode gespeicherte Betriebsart.

[5] Das Stellglied kann nicht auf den Ersatzausgangswert wechseln und bleibt in seiner aktuellen Position. Im Handbetrieb können Sie die Position des Stellglieds nur mit Manual_UP und Manual_DN und nicht mit ManualValue ändern.

10.2.7 Warnparameter der Anweisung PID_3Step

Liegen am PID-Regler mehrere Warnungen an, werden die Werte der Fehlercodes mittels binärer Addition angezeigt. Die Anzeige von Fehlercode 0003 beispielsweise weist darauf hin, dass die Fehler 0001 und 0002 anstehen.

Tabelle 10- 25 Parameter Warning der Anweisung PID_Compact

Warning (DW#16#...)	Beschreibung
0000	Keine Warnung vorhanden.
0001 [1]	Der Wendepunkt wurde in der Erstoptimierung nicht gefunden.
0002	Bei der Einstellung im Betrieb wurde die Schwingung erzwungen. (Der Parameter "Warning" unterdrückt diese Warnung, sie ist nur im Parameter “WarningInternal” zu Diagnosezwecken sichtbar.)
0004 [1]	Der Sollwert war auf die konfigurierten Grenzwerte begrenzt.
0008 [1]	Für die ausgewählte Berechnungsmethode waren nicht alle erforderlichen Reglereigenschaften definiert. Stattdessen wurden die PID-Parameter mit der Methode TIR.TuneRuleHeat / TIR.TuneRuleCool = 3 berechnet.
0010	Der Betriebszustand konnte nicht gewechselt werden, weil Reset = TRUE oder ManualEnable = TRUE.
0020	Die Zykluszeit des aufrufenden OB begrenzt die Abtastzeit des PID-Algorithmus. Durch kürzere OB-Zykluszeiten können Sie die Ergebnisse verbessern.
0040 [1]	Der Prozesswert hat einen seiner Warngrenzwerte überschritten.
0080	Ungültiger Wert an "Mode". Der Betriebszustand wird nicht gewechselt.
0100 [1]	Der manuelle Wert war auf die Grenzwerte des Reglerausgangs begrenzt.
0200	Die angegebene Regel für die Einstellung wird nicht unterstützt. Es werden keine PID-Parameter berechnet.
1000	Der Ersatzausgangswert kann nicht erreicht werden, weil er außerhalb der Grenzwerte für den Ausgangswert liegt.

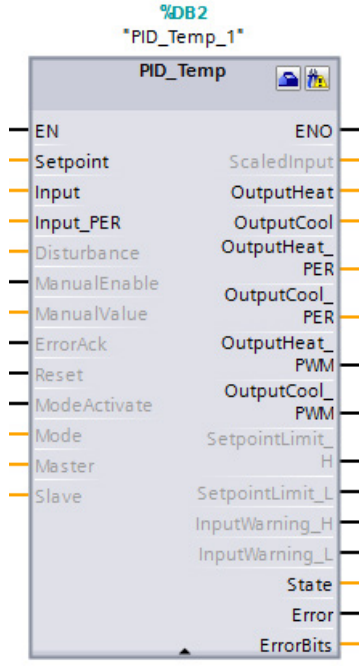
[1] Hinweis: Der PID-Regler hat, sobald die Ursache behoben war oder die Anwenderaktion mit gültigen Parametern wiederholt wurde, die folgenden Warnungen automatisch gelöscht: 0001, 0004, 0008, 0040 und 0100.

10.2.8 Anweisung PID_Temp

Die Anweisung PID_Temp bietet einen universellen PID-Regler für die speziellen Anforderungen der Temperaturregelung.

Tabelle 10- 26 Anweisung PID_Temp

KOP/FUP	SCL	Beschreibung

	`"PID_Temp_1"(`
	`  Setpoint:=_real_in_,`
	`  Input:=_real_in_,`
	`  Input_PER:=_int_in_,`
	`  Disturbance:=_real_in_,`
	`ManualEnable:=_bool_in_,`
	`  ManualValue:=_real_in_,`
	`  ErrorAck:=_bool_in_,`
	`Reset:=_bool_in_,`
	`  ModeActivate:=_bool_in_,`
	`Mode:=_int_in_,`
	`Master:=_dword_in`
	`Save:=_dword_in`
	`ScaledInput=>_real_out_,`
	`  OutputHeat=>_real_out_,`
	`  OutputCool=>_real_out_,`
	`  OutputHeat_PER=>_int_out_,`
	`  OutputCool_PER=>_int_out_,`
	`  Out-`
	`putHeat_PWM=>_bool_out_,`
	`  Out-`
	`putCool_PWM=>_bool_out_,`
	`SetpointLimit_H=>_bool_out_,`
	`SetpointLimit_L=>_bool_out_,`
	`  InputWarn-`
	`ing_H=>_bool_out_,`
	`  InputWarn-`
	`ing_L=>_bool_out_,`
	`  State=>_int_out_,`
	`  Error=>_bool_out_,`
	`ErrorBits=>_dword_out_);`
		PID_Temp bietet die folgenden Möglichkeiten:
		• Heizen und Kühlen im Prozess mit verschiedenen Aktoren
		• Integrierte Selbsteinstellung für die Temperaturregelung
		• Kaskadenschaltung von mehreren Temperaturen im Prozess, die vom gleichen Aktor geregelt werden

[1] STEP 7 erstellt das Technologieobjekt und den Instanz-DB automatisch, wenn Sie die Anweisung einfügen. Der Instanz-DB enthält die Parameter des Technologieobjekts.

[2] Im SCL-Beispiel ist "PID_Temp_1" der Name des Instanz-DBs.

Tabelle 10- 27 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
Setpoint	IN	Real	Sollwert des PID-Reglers im Automatikbetrieb. (Standardwert: 0.0)
Input	IN	Real	Eine Variable des Anwenderprogramms wird als Quelle des Prozesswerts verwendet. (Standardwert: 0.0) Wenn Sie den Parameter Input verwenden, müssen Sie Config.InputPerOn = FALSE einstellen.
Input_PER	IN	Int	Ein Analogeingang wird als Quelle des Prozesswerts verwendet. (Standardwert: 0) Wenn Sie den Parameter Input_PER verwenden, müssen Sie Config.InputPerOn = TRUE einstellen.
Disturbance	IN	Real	Störgröße oder Vorsteuerungswert
ManualEnable	IN	Bool	Aktiviert oder deaktiviert den Handbetrieb. (Standardwert: FALSE): • Eine Flanke von FALSE nach TRUE aktiviert den "Handbetrieb", während sich bei Zustand = 4 die Betriebsart nicht verändert. Solange ManualEnable = TRUE ist, können Sie die Betriebsart nicht mit einer steigenden Flanke an ModeActivate wechseln und auch nicht das Dialogfeld für die Inbetriebnahme verwenden. • Eine Flanke von TRUE nach FALSE aktiviert die von Mode zugewiesene Betriebsart. Hinweis: Es wird empfohlen, die Betriebsart lediglich über ModeActivate zu ändern.
ManualValue	IN	Real	Ausgangswert für Handbetrieb. (Standardwert: 0.0) Verwendbar sind Werte von Config.OutputLowerLimit bis Config.OutputUpperLimit.
ErrorAck	IN	Bool	Setzt die ErrorBits und Warnausgänge bei einer Flanke von FALSE nach TRUE zurück. (Standardwert: FALSE)
Reset	IN	Bool	Startet den Regler neu. (Standardwert: FALSE): • Flanke von FALSE nach TRUE: – Wechselt in den "inaktiven" Modus – Setzt die ErrorBits und Warnausgänge zurück – Löscht den Integralanteil – Speichert die PID-Parameter • Solange Reset = TRUE ist, bleibt PID_Temp im "inaktiven" Modus (Zustand = 0). • Flanke von TRUE nach FALSE: – PID_Temp wechselt in die Betriebsart, die im Parameter Mode gespeichert ist.
ModeActivate	IN	Bool	PID_Temp schaltet bei einer Flanke von FALSE nach TRUE in den Betriebszustand, der im Parameter Mode gespeichert ist. (Standardwert: FALSE)

Parameter und Datentyp		Datentyp	Beschreibung
Mode	IN/OUT	Int	Wird bei einer steigenden Flanke an Eingang Mode Activate aktiviert. Auswahl der Betriebsart (Standardwert: 0.0): • Mode = 0: Inaktiv • Mode = 1: Erstoptimierung • Mode = 2: Feineinstellung • Mode = 3: Automatikbetrieb • Mode = 4: Handbetrieb "Ersatzausgangswert mit Fehlerüberwachung" (State = 5). Kann nicht vom Benutzer aktiviert werden; dies ist eine automatische Fehlerreaktion.
Master	IN/OUT	DWord	Kaskadenaufschaltung auf Master (Anti-Windup- und Einstellbedingungen). (Standardwert: DW#16#0000) • Bits 0 - 15: In Anweisung PID_Temp nicht benutzt • Bits 16 - 23: Grenzwertzähler: Ein Slave erhöht diesen Wert, wenn der Grenzwert erreicht ist. Die Anzahl der Slaves mit Begrenzung wird für die Anti-Windup-Funktion verarbeitet (siehe Parameter Config.Cascade.AntiWindUpMode). • Bit 24: IsAutomatic: Dieses Bit wird auf "1" gesetzt, wenn alle Slaves dieses Reglers im Automatikbetrieb sind und für die Prüfung der Einstellbedingungen in einer Kaskade verarbeitet werden. Dieses Bit ist identisch mit Parameter AllSlaveAutomaticState. • Bit 25: "IsReplacement-Setpoint": Dieses Bit wird auf "1" gesetzt, wenn für einen Slave dieses Reglers der "Ersatzsollwert" aktiviert ist und er für die Prüfung der Einstellbedingungen in einer Kaskade verarbeitet wird. Der invertierte Wert wird in Parameter NoSlaveReplacementSetpoint gespeichert.
Slave	IN/OUT	DWord	
ScaledInput	OUT	Real	Skalierter Prozesswert. (Standardwert: 0.0)
OutputHeat[1]	OUT	Real	Ausgangswert für Heizen im Format REAL. (Standardwert: 0.0) Dieser Ausgangswert wird unabhängig von der Wahl des Ausgangs mit Parameter Config.Output.Heat.Select berechnet.
OutputCool[1]	OUT	Real	Ausgangswert für Kühlen im Format REAL. (Standardwert: 0.0) Dieser Ausgangswert wird unabhängig von der Wahl des Ausgangs mit Parameter Config.Output.Cool.Select berechnet.
OutputHeat_PER[1]	OUT	Int	Ausgangswert für Heizen im Peripherieformat (Standardwert: 0) Dieser Ausgangswert wird nur berechnet, wenn er mit Parameter Config.Output.Heat.Select = 2 ausgewählt wurde. Ist er nicht ausgewählt, ist der Ausgang immer "0".
OutputCool_PER[1]	OUT	Int	Ausgangswert für Kühlen im Peripherieformat (Standardwert: 0) Dieser Ausgangswert wird nur berechnet, wenn er mit Parameter Config.Output.Cool.Select = 2 ausgewählt wurde. Ist er nicht ausgewählt, ist der Ausgang immer "0".
OutputHeat_PWM[1]	OUT	Bool	Pulsbreitenmodulierter Ausgangswert für Heizen. (Standardwert: FALSE) Dieser Ausgangswert wird nur berechnet, wenn er mit Parameter Config.Output.Heat.Select = 1 (Standardwert) ausgewählt wurde. Ist er nicht ausgewählt, ist der Ausgang immer FALSE.

Parameter und Datentyp		Datentyp	Beschreibung
OutputCool_PWM[1]	OUT	Bool	Pulsbreitenmodulierter Ausgangswert für Kühlen. (Standardwert: FALSE) Dieser Ausgangswert wird nur berechnet, wenn er mit Parameter Config.Output.Cool.Select = 1 (Standardwert) ausgewählt wurde. Ist er nicht ausgewählt, ist der Ausgang immer FALSE.
SetpointLimit_H	OUT	Bool	Sollwert oberer Grenzwert. (Standardwert: FALSE) Wenn SetpointLimit_H = TRUE ist, dann ist der absolute obere Grenzwert des Sollwerts erreicht (Setpoint ≥ Config.SetpointUpperLimit). Der Sollwert ist auf Config.SetpointUpperLimit begrenzt.
SetpointLimit_L	OUT	Bool	Sollwert unterer Grenzwert. (Standardwert: FALSE) Wenn SetpointLimit_L = TRUE ist, dann ist der absolute untere Grenzwert des Sollwerts erreicht (Setpoint ≥ Config.SetpointLowerLimit). Der Sollwert ist auf Config.SetpointLowerLimit begrenzt.
InputWarning_H	OUT	Bool	Wenn InputWarning_H = TRUE, hat der Prozesswert die obere Warngrenze erreicht oder überschritten. (Standardwert: FALSE)
InputWarning_L	OUT	Bool	Wenn InputWarning_L = TRUE, hat der Prozesswert die untere Warngrenze erreicht oder unterschritten. (Standardwert: FALSE)
State	OUT	Int	Aktuelle Betriebsart des PID-Reglers. (Standardwert: 0) Die Betriebsart können Sie mit dem Eingangsparameter Mode und einer steigenden Flanke an ModeActivate ändern: • State = 0: Inaktiv • State = 1: Erstoptimierung • State = 2: Feineinstellung • State = 3: Automatikbetrieb • State = 4: Handbetrieb • State = 5: Ersatzausgangswert mit Fehlerüberwachung
Error	OUT	Bool	Wenn Error = TRUE, steht mindestens eine Fehlermeldung in diesem Zyklus an. (Standardwert: FALSE) Hinweis: Der Parameter Error in V1.x PID war das Feld der ErrorBits mit den Fehlercodes. Jetzt zeigt ein Boolescher Merker an, dass ein Fehler aufgetreten ist.
ErrorBits	OUT	DWord	In der Parametertabelle ErrorBits (Seite 513) für die Anweisung PID_Temp werden die anstehenden Fehlermeldungen beschrieben. (Standardwert: DW#16#0000 (kein Fehler)). ErrorBits ist remanent und wird bei einer steigenden Flanke an Reset oder ErrorAck zurückgesetzt. Hinweis: In V1.x war der Parameter ErrorBits als Parameter Error definiert und nicht vorhanden.
Warning	OUT	DWord	In der Parametertabelle Warning (Seite 515) für die Anweisung PID_Temp werden die anstehenden anwenderrelevanten Warnmeldungen beschrieben. (Standardwert: DW#16#0000 (keine Warnung)).
WarningInternal	OUT	DWord	In der Parametertabelle WarningInternal für die Anweisung PID_Temp werden die anstehenden internen Warnmeldungen beschrieben (umfasst alle Warnungen). (Standardwert: DW#16#0000 (keine interne Warnung)).

[1] Sie können die Ausgänge der Parameter Output, Output_PER und Output_PWM parallel verwenden.

10.2.8.1 Funktionsweise des PID_Temp-Reglers

Regelung von Heizung und/oder Kühlung auswählen

Zunächst müssen Sie auswählen, ob Sie zusätzlich zum Heizausgang am Parameter "ActivateCooling" ein Kühlgerät benötigen. Anschließend müssen Sie festlegen, ob Sie zwei PID-Parametersätze (erweiterter Modus) oder nur einen PID-Parametersatz mit einem zusätzlichen Heiz-/Kühlfaktor am Parameter "AdvancedCooling" verwenden möchten.

CoolFactor verwenden

Falls Sie einen Heiz-/Kühlfaktor anwenden möchten, müssen Sie den Wert manuell definieren. Sie ermitteln den Wert anhand der technischen Daten Ihrer Anwendung (Verhältnis der proportionalen Verstärkung der Stellglieder (z. B. das Verhältnis der maximalen Heiz- und Kühlleistung der Stellglieder) und weisen ihn dem Parameter "CoolFactor" zu. Ein Heiz-/Kühlfaktor von 2.0 bedeutet, dass das Heizgerät zweimal so effektiv ist wie das Kühlgerät. Wenn Sie den Kühlfaktor verwenden, berechnet PID_Temp das Ausgangssignal und multipliziert, abhängig vom Vorzeichen, das Ausgangssignal mit dem Heiz-/Kühlfaktor (bei negativem Vorzeichen) oder nicht (bei positivem Vorzeichen).

Zwei PID-Parametersätze verwenden

Unterschiedliche PID-Parametersätze zum Heizen und Kühlen können während der Inbetriebnahme automatisch erkannt werden. Im Vergleich zum Heiz-/Kühlfaktor können Sie eine bessere Regelungsleistung erwarten, weil Sie mit zwei Parametersätzen neben unterschiedlichen Proportionalverstärkungen unterschiedliche Verzögerungszeiten berücksichtigen können. Nachteil ist jedoch, dass der Einstellungsprozess länger dauern kann. Wenn die PID-Parameterumschaltung aktiviert ist (Config.AdvancedCooling = WAHR), erkennt der PID_Temp-Regler im "Automatikbetrieb" (Regelung ist aktiv), ob derzeit Heizen oder Kühlen erforderlich ist, und verwendet die PID-Parametersätze für die Regelung.

ControlZone

Beim PID_Temp-Regler können Sie für jeden Parametersatz am Parameter "ControlZone" eine Regelzone festlegen. Wenn sich die Regelabweichung (Sollwert - Eingang) innerhalb der Regelzone befindet, berechnet PID_Temp die Ausgangssignale mithilfe des PID-Algorithmus. Wenn die Regelabweichung jedoch den festgelegten Bereich verlässt, wird der Ausgang auf den maximalen Heiz- oder den maximalen Kühlausgangswert gesetzt (Kühlausgang aktiviert) / minimaler Heizausgangswert (Kühlausgang deaktiviert). Mithilfe dieser Funktionalität können Sie den gewünschten Sollwert schneller erreichen, insbesondere zum ersten Aufheizen bei langsamen Temperaturprozessen.

DeadZone

Mit dem Parameter "DeadZone" können Sie eine Breite der Regelabweichung beim Heizen und Kühlen festlegen, die vom PID-Algorithmus vernachlässigt wird. Das bedeutet, dass eine Regelabweichung innerhalb dieses Bereichs unterdrückt wird, und der PID_Temp-Regler verhält sich so, als ob der Sollwert und Prozesswert identisch sind. So können Sie unnötige Eingriffe durch den Regler um den Sollwert reduzieren und das Stellglied beibehalten. Falls Sie eine DeadZone verwenden möchten, müssen Sie den Wert manuell definieren. Bei der Selbsteinstellung wird der DeadZone-Wert nicht automatisch festgelegt. DeadZone ist bei Heizreglern ohne Kühlung oder Heiz-/Kühlreglern mit CoolFactor symmetrisch (zwischen -Retain.CtrlParams.Heat.DeadZone und +Retain.CtrlParams.Heat.DeadZone). DeadZone kann bei Heiz-/Kühlreglern mit zwei PID-Parametersätzen asymmetrisch sein (zwischen -Retain.CtrlParams.Cool.DeadZone und +Retain.CtrlParams.Heat.DeadZone).

Funktionsweise von PID_Temp-Reglern

Die folgenden Blockschaltbilder veranschaulichen den Standard- und Kaskadenbetrieb der Anweisung PID_Temp:

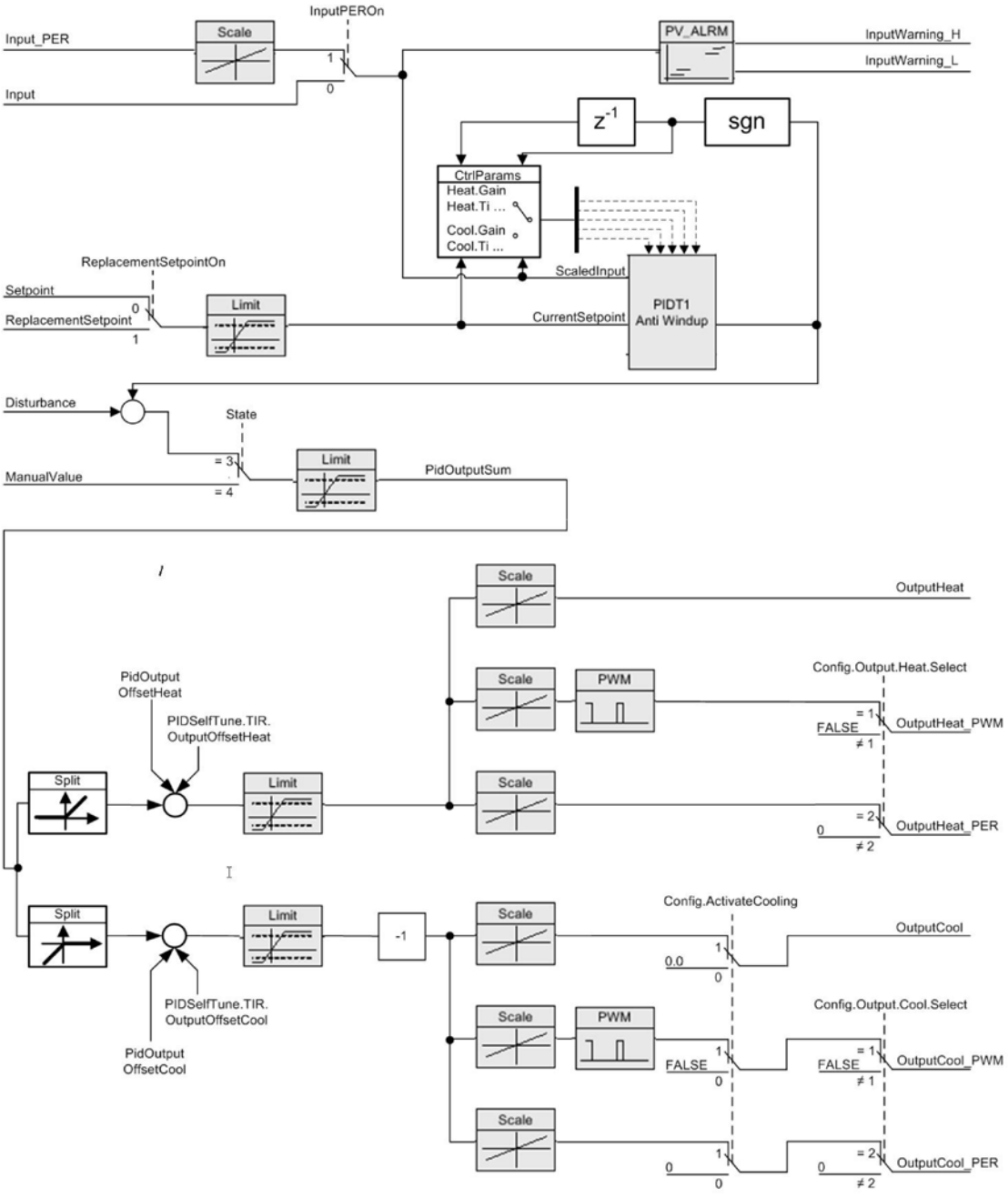


Bild 10-6 PID_Temp_Operation_Block_Diagram

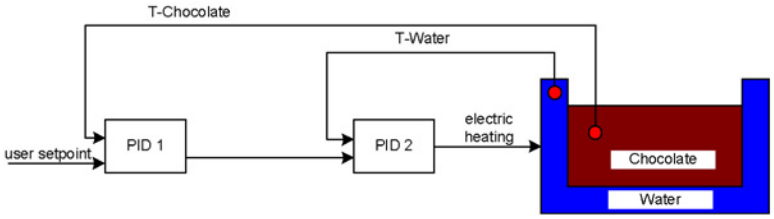


Bild 10-7 PID_Temp_Cascade_Operation_Block_Diagram

10.2.8.2 Kaskadieren von Reglern

Sie können Temperatur-PID-Regler kaskadieren, um mehrere Temperaturen, die vom gleichen Stellglied geregelt werden, zu verarbeiten.

Aufrufreihenfolge

Sie müssen kaskadierende PID-Regler in einem OB-Zyklus aufrufen. Zunächst müssen Sie den Master aufrufen, dann die nächsten Slaves im Regelsignalfluss und schließlich den letzten Slave in der Kaskade. Die Anweisung PID_Temp führt keine automatische Prüfung der Aufrufreihenfolge durch.

Kommunikationsverbindungen

Beim Kaskadieren von Reglern müssen Sie den Master und Slave so verbinden, dass sie miteinander Informationen austauschen können. Sie müssen den Master-IN/OUT-Parameter des Slaves mit dem Slave-IN/OUT-Parameter des Masters in der Richtung des Signalflusses verbinden.

Im Folgenden sehen Sie eine Verbindung von PID_Temp-Reglern in einer Kaskade mit zwei Unterkaskaden: "PID_Temp1" liefert den Sollwert. In der Konfiguration sind die Ausgänge von "PID_Temp2", "PID_Temp3", "PID_Temp5", "PID_Temp6" und "PID_Temp8" mit dem Prozess verbunden:

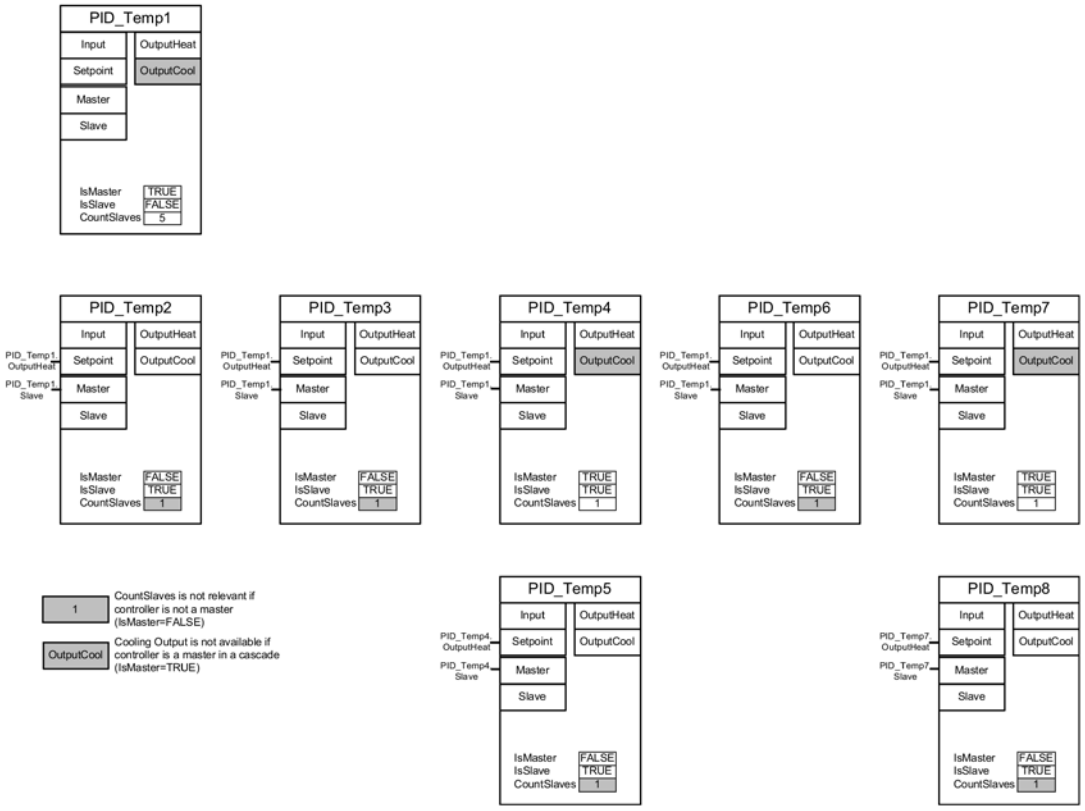


Bild 10-8 PID_Temp_Cascading_communication_connection

Ersatzsollwert

Die Anweisung PID_Temp liefert einen zweiten Sollwerteingang am Parameter “ReplacementSetpoint”, den Sie aktivieren können, indem Sie den Parameter “ReplacementSetpointOn” = WAHR einstellen. Sie können “ReplacementSetpoint” während der Inbetriebnahme oder Einstellung eines Slavereglers als Ihren Sollwerteingang verwenden, ohne die Verbindung von Ausgang und Sollwert zwischen Master und Slave trennen zu müssen. Diese Verbindung ist für den Normalbetrieb der Kaskade erforderlich.

Auf diese Weise brauchen Sie Ihr Programm nicht zu ändern und zu laden, wenn Sie kurzzeitig einen Slave von seinem Master trennen möchten. Sie brauchen nur den "ReplacementSetpoint" zu aktivieren und abschließend wieder zu deaktivieren. Wenn Sie den Sollwert am Parameter “CurrentSetpoint” sehen können, wirkt sich der Wert auf den PID-Algorithmus aus.

Selbsteinstellung

Eine Selbsteinstellung für einen kaskadierenden Masterregler muss diese Anforderungen erfüllen:

- Inbetriebnahme vom internen Slave zum ersten Master.
- Alle Slaves des Masters müssen im Automatikbetrieb sein.
- Der Ausgang des Masters muss der Sollwert für die Slaves sein.

Die Anweisung PID_Temp liefert die folgende Unterstützung für die Selbsteinstellung in der Kaskade:

- Wenn Sie die Selbsteinstellung für einen Masterregler starten, prüft der Master, ob alle Slaves im Automatikbetrieb sind und ob die Ersatzsollwertfunktion für alle Slaves deaktiviert ist (“ReplacementSetpointOn” = FALSCH). Werden diese Bedingungen nicht erfüllt, können Sie die Selbsteinstellung des Masters nicht durchführen. Der Master bricht die Einstellung ab, geht in den inaktiven Modus (wenn “ActivateRecoverMode” = FALSCH) oder zurück in die im Parameter "Modus" gespeicherte Betriebsart (wenn “ActivateRecoverMode” = WAHR). Der Master zeigt die Fehlermeldung 200000hex an ("Fehler beim Master in der Kaskade. Slaves sind nicht im Automatikbetrieb oder haben einen Ersatzsollwert aktiviert und verhindern die Einstellung des Masters.").
- Wenn sich alle Slaves im Automatikbetrieb befinden, setzt das System den Parameter “AllSlaveAutomaticState” = WAHR. Sie können diesen Parameter in Ihren Programmen verwenden oder die Fehlerursache 200000hex ermitteln.
- Wenn der "ReplacementSetpoint" bei allen Slaves deaktiviert ist, setzt das System den Parameter “NoSlaveReplacementSetpoint” = WAHR. Sie können diesen Parameter in Ihren Programmen verwenden oder die Fehlerursache 200000hex ermitteln.

Wenn Sie den Inbetriebnahmedialog der Anweisung PID_Temp verwenden, finden Sie dort weitere Unterstützung für die Einstellung von Kaskaden (Seite 537).

Betriebsarten und Fehlerbehandlung

Der PID_Temp-Regler gestattet das Wechseln der Betriebsart durch seinen Master oder die Slaves nicht. Das bedeutet, dass ein Master in der Kaskade in der aktuellen Betriebsart bleibt, wenn ein Slave einen Fehler meldet. Dies ist ein Vorteil, wenn zwei oder mehr parallele Slaves mit diesem Masterregler arbeiten: Ein Fehler in einer Kette führt nicht zum Abschalten der parallelen Kette.

Ebenso bleibt ein Slave innerhalb der Kaskade in seiner aktuellen Betriebsart, wenn sein Master einen Fehler aufweist. Der weitere Betrieb des Slaves richtet sich dann jedoch nach der Konfiguration des Masters, weil der Sollwert des Slaves der Ausgang des Masters ist. Das bedeutet, dass, wenn bei Konfiguration des Masters mit “ActivateRecoverMode” = WAHR ein Fehler auftritt, der Master den letzten gültigen Ausgangswert oder einen Ersatzausgangswert als Sollwert für den Slave ausgibt. Wenn Sie für den Master “ActivateRecoverMode” = FALSCH konfigurieren, wechselt der Master in den inaktiven Modus und legt für alle Ausgänge "0.0" fest, so dass der Slave "0.0" als Sollwert verwendet.

Weil nur die Slaveregler direkten Zugriff auf die Stellglieder haben und diese bei einem Fehler des Masters in ihrer Betriebsart bleiben, können Sie Schäden am Prozess vermeiden. So ist es beispielsweise bei Kunststoff verarbeitenden Geräten kritisch, wenn die Slaves den Betrieb einstellen, die Stellglieder herunterfahren und dadurch der Kunststoff im Gerät aushärtet, nur weil im Masterregler ein Fehler aufgetreten ist.

Anti-Windup

Ein Slave in einer Kaskade erhält seinen Sollwert vom Ausgang seines Masters. Wenn der Slave seine eigenen Ausgangsgrenzwerte erreicht, während der Master immer noch eine Regelabweichung (Sollwert - Eingang) erkennt, wird der Master eingefroren oder er reduziert seinen Integrationsbeitrag, um einen sogenannten "Windup" zu verhindern. Bei einem "Windup" erhöht der Master seinen Integrationsbeitrag auf einen sehr großen Wert und muss diesen zunächst reduzieren, damit der Regler wieder eine normale Reaktion haben kann. Ein solcher "Windup" wirkt sich negativ auf die Dynamik der Regelung aus. Der PID_Temp-Regler bietet Wege, um diese Auswirkung in einer Kaskade zu verhindern. Hierfür wird der Parameter “Config.Cascade.AntiWindUpMode” des Masterreglers konfiguriert:

Wert	Beschreibung
0	Deaktivierung der Anti-Windup-Funktion.
1	Reduzierung des Integrationsbeitrags des Masterreglers im Verhältnis “begrenzte Slaves” zu “vorhandene Slaves” (Parameter “CountSlaves”).
2	Einfrieren des Integrationsbeitrags des Masters, sobald ein Slave seinen Grenzwert erreicht. Nur relevant, wenn "Config.Cascade.IsMaster" = WAHR.

10.2.9 Anweisung PID_Temp ErrorBit-Parameter

Liegen am PID-Regler mehrere Warnungen an, werden die Werte der Fehlercodes mittels binärer Addition angezeigt. Die Anzeige von Fehlercode 0003 beispielsweise weist darauf hin, dass die Fehler 0001 und 0002 anstehen.

Tabelle 10- 28 ErrorBit-Parameter der Anweisung PID_Temp

ErrorBit (DW#16#...)	Beschreibung
0000	Kein Fehler
0001 [1, 2]	Der Parameter Input liegt außerhalb der Prozesswertgrenzen. Input > Config.InputUpperLimit Input < Config.InputLowerLimit
0002 [2, 3]	Ungültiger Wert am Parameter Input_PER. Prüfen Sie, ob am Analogeingang ein Fehler anliegt.
0004 [4]	Fehler während der Feineinstellung. Schwingung des Prozesswerts konnte nicht erhalten werden.
0008 [4]	Fehler beim Start der Erstoptimierung. Der Prozesswert ist zu nah am Sollwert. Starten Sie die Feineinstellung.
0010 [4]	Der Sollwert wurde während der Einstellung geändert. Hinweis: Sie können die zulässige Schwankung am Sollwert in der Variablen CancelTuningLevel festlegen.
0020	Während der Feineinstellung ist keine Erstoptimierung zulässig. Hinweis: Wenn vor dem Auftreten des Fehlers ActivateRecoverMode = TRUE war, bleibt PID_Temp im Feineinstellungsmodus.
0040 [4]	Fehler während der Erstoptimierung. Die Kühlung konnte den Prozesswert nicht verringern.
0080 [4]	Fehler während der Erstoptimierung. Inkorrekte Konfiguration der Grenzen des Ausgangswerts. Prüfen Sie, ob die Grenzen des Ausgangswerts richtig konfiguriert sind, und passen Sie die Steuerungslogik an.
0100 [4]	Fehler während der Feineinstellung führte zu ungültigen Parametern.
0200 [2, 3]	Ungültiger Wert am Parameter Input: Wert hat ungültiges Zahlenformat.
0400 [2, 3]	Berechnung des Ausgangswerts fehlgeschlagen. Prüfen Sie die PID-Parameter.
0800 [1, 2]	Abtastzeitfehler: PID_Temp wird nicht während der Abtastzeit des Weckalarm-OBs aufgerufen.
1000 [2, 3]	Ungültiger Wert am Parameter Setpoint: Wert hat ungültiges Zahlenformat.
10000	Ungültiger Wert am Parameter ManualValue: Wert hat ungültiges Zahlenformat. Hinweis: Wenn vor dem Auftreten des Fehlers ActivateRecoverMode = TRUE war, verwendet PID_Temp SubstituteOutput als Ausgangswert. Sobald Sie in Parameter ManualValue einen gültigen Wert zuweisen, verwendet PID_Temp diesen als Ausgangswert.

ErrorBit (DW#16#...)	Beschreibung
20000	Ungültiger Wert der Variablen SubstituteValue: Wert hat ungültiges Zahlenformat. PID_Temp verwendet die untere Grenze des Ausgangswerts als Ausgangswert. Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv war, ActivateRecoverMode = TRUE ist und der Fehler nicht mehr ansteht, gehtPID_Temp wieder in den Automatikbetrieb.
40000	Ungültiger Wert am Parameter Disturbance: Wert hat ungültiges Zahlenformat. Hinweis: War vor dem Auftreten des Fehlers der Automatikbetrieb aktiv und ActivateRecoverMode = FALSE, wird Disturbance auf Null gesetzt. PID_Temp bleibt im Automatikbetrieb. Hinweis: Wenn vor dem Auftreten des Fehlers die Erstoptimierung oder die Feineinstellung aktiv und ActivateRecoverMode = TRUE war, wechselt PID_Temp in die im Parameter Mode gespeicherte Betriebsart. Wenn Störungen in der aktuellen Phase keine Auswirkungen auf den Ausgangswert haben, wird die Einstellung nicht abgebrochen.
200000	Fehler beim Master in der Kaskade. Slaves sind nicht im Automatikbetrieb oder haben einen Ersatzsollwert aktiviert und verhindern die Einstellung des Masters.
400000	Der PID-Regler gestattet die Erstoptimierung zum Heizen nicht, solange Kühlung aktiv ist.
800000	Der Prozesswert muss nah am Sollwert liegen, um die Erstoptimierung für die Kühlung zu starten.
1000000	Fehler beim Starten der Einstellung. “Heat.EnableTuning” und “Cool.EnableTuning” sind nicht festgelegt oder entsprechen nicht der Konfiguration.
2000000	Für die Erstoptimierung zum Kühlen ist eine erfolgreiche Erstoptimierung zum Heizen erforderlich.
4000000	Fehler beim Starten der Feineinstellung. “Heat.EnableTuning” und “Cool.EnableTuning” können nicht gleichzeitig gesetzt sein.
8000000	Fehler während der PID-Parameterberechnung führte zu ungültigen Parametern (z. B. negative Verstärkung; die aktuellen PID-Parameter bleiben unverändert und die Einstellung wirkt sich nicht aus).

[1] Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv und ActivateRecoverMode = TRUE war, bleibt PID_Temp im Automatikbetrieb.

[2] Hinweis: Wenn vor dem Auftreten des Fehlers die Erstoptimierung oder die Feineinstellung aktiv war und ActivateRecoverMode = TRUE ist, wechselt PID_Temp in die im Parameter Mode gespeicherte Betriebsart.

[3] Hinweis: Wenn vor dem Auftreten des Fehlers der Automatikbetrieb aktiv war und ActivateRecoverMode = TRUE ist, gibt PID_Compact den konfigurierten Ersatzausgangswert aus. Sobald der Fehler nicht mehr ansteht, wechselt PID_Temp wieder in den Automatikbetrieb.

[4] Hinweis: Wenn vor dem Auftreten des Fehlers ActivateRecoverMode = TRUE war, bricht PID_Temp die Einstellung ab und wechselt in die im Parameter Mode gespeicherte Betriebsart.

10.2.10 Warnparameter der Anweisung PID_Temp

Liegen am PID-Regler mehrere Warnungen an, werden die Werte der Fehlercodes mittels binärer Addition angezeigt. Die Anzeige von Fehlercode 0003 beispielsweise weist darauf hin, dass die Fehler 0001 und 0002 anstehen.

Tabelle 10- 29 Parameter Warning der Anweisung PID_Temp

Warning (DW#16#...)	Beschreibung
0000	Keine Warnung vorhanden.
0001 [1]	Der Wendepunkt wurde in der Erstoptimierung nicht gefunden.
0002	Bei der Einstellung im Betrieb wurde die Schwingung erzwungen. (Der Parameter "Warning" unterdrückt diese Warnung, sie ist nur im Parameter “WarningInternal” zu Diagnosezwecken sichtbar.)
0004 [1]	Der Sollwert war auf die konfigurierten Grenzwerte begrenzt.
0008 [1]	Für die ausgewählte Berechnungsmethode waren nicht alle erforderlichen Reglereigenschaften definiert. Stattdessen wurden die PID-Parameter mit der Methode TIR.TuneRuleHeat / TIR.TuneRuleCool = 3 berechnet.
0010	Der Betriebszustand konnte nicht gewechselt werden, weil Reset = TRUE oder ManualEnable = TRUE.
0020	Die Zykluszeit des aufrufenden OB begrenzt die Abtastzeit des PID-Algorithmus. Durch kürzere OB-Zykluszeiten können Sie die Ergebnisse verbessern.
0040 [1]	Der Prozesswert hat einen seiner Warngrenzwerte überschritten.
0080	Ungültiger Wert an "Mode". Der Betriebszustand wird nicht gewechselt.
0100 [1]	Der manuelle Wert war auf die Grenzwerte des Reglerausgangs begrenzt.
0200	Die angegebene Regel für die Einstellung wird nicht unterstützt. Es werden keine PID-Parameter berechnet.
1000	Der Ersatzausgangswert kann nicht erreicht werden, weil er außerhalb der Grenzwerte für den Ausgangswert liegt.
4000	Die angegebene Ausgangsauswahl für Heizen und/oder Kühlen wird nicht unterstützt. Nur OutputHeat und OutputCool sind aktiv.
8000	Der angegebene Wert für den Parameter PIDSelfTune.SUT.AdaptDelayTime wird nicht unterstützt, deshalb wird der Standardwert "0" verwendet.
10000	Der angegebene Wert für den Parameter PIDSelfTune.SUT.CoolingMode wird nicht unterstützt, deshalb wird der Standardwert "0" verwendet.

[1] Hinweis: Der PID-Regler hat, sobald die Ursache behoben war oder die Anwenderaktion mit gültigen Parametern wiederholt wurde, die folgenden Warnungen automatisch gelöscht: 0001, 0004, 0008, 0040 und 0100.

10.2.11 PID_Compact- und PID_3Step-Regler konfigurieren

Die Parameter des Technologieobjekts legen die Funktionsweise des PID-Reglers fest. Öffnen Sie den Konfigurationseditor über das Symbol.

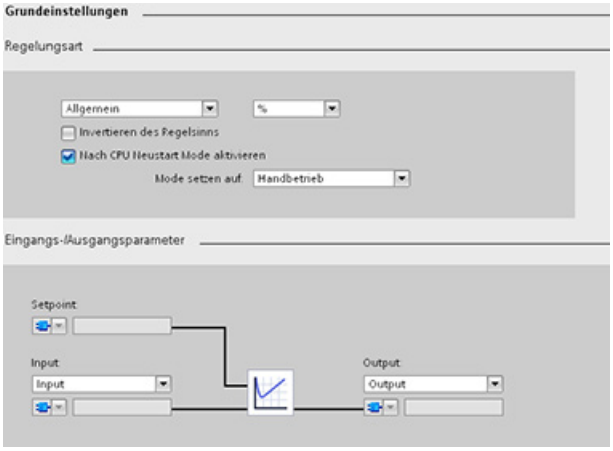


Tabelle 10- 30 Beispielhafte Konfigurationseinstellungen für die Anweisung PID_Compact

Einstellungen		Beschreibung
Grundlagen	Reglertyp	Wählt die physikalischen Einheiten aus.
	Reglerlogik invertieren	Ermöglicht die Auswahl eines invers funktionierenden PID-Reglers. • Ist diese Option nicht ausgewählt, verhält sich der PID-Regler entsprechend der direkten Funktionsweise und der Ausgang des PID-Regelkreises steigt, wenn der Eingangswert < Sollwert ist. • Ist diese Option ausgewählt, steigt der Ausgang des PID-Regelkreises, wenn der Eingangswert > Sollwert ist.
	Nach CPU-Neustart letzte Betriebsart aktivieren	Startet den PID-Regler neu, nachdem er zurückgesetzt wurde oder wenn eine Eingangsgrenze überschritten und in den gültigen Bereich zurückgeführt wurde.
	Eingang	Wählt entweder den Parameter Input oder den Parameter Input_PER (bei analog) für den Prozesswert aus. Input_PER kann direkt von einem analogen Eingangsmodul kommen.
	Ausgang	Wählt entweder den Parameter Output oder den Parameter Output_PER (bei analog) für den Ausgangswert aus. Output_PER kann direkt an ein analoges Ausgangsmodul gehen.
Prozesswert	Skaliert sowohl den Bereich als auch die Grenzen für den Prozesswert. Wenn der Prozesswert unter die untere Grenze fällt oder über die obere Grenze steigt, wechselt der PID-Regelkreis in die inaktive Betriebsart und setzt den Ausgangswert auf 0. Um Input_PER zu verwenden, **müssen** Sie den analogen Prozesswert (Eingangswert) skalieren.	

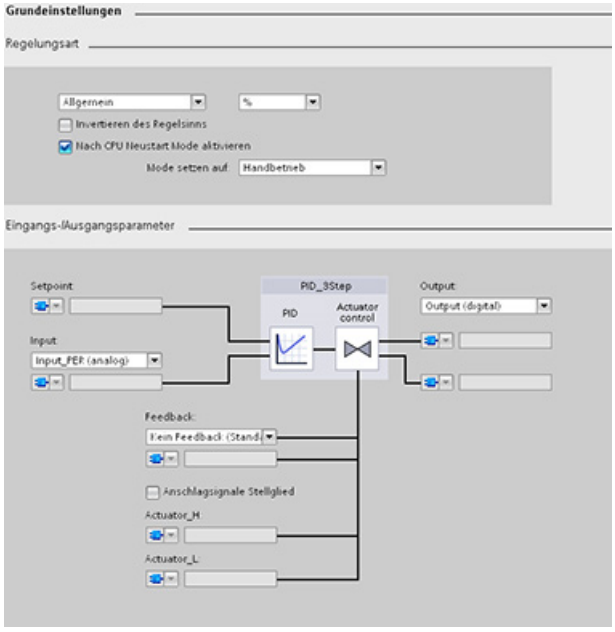


Tabelle 10- 31 Beispielhafte Konfigurationseinstellungen für die Anweisung PID_3Step

Einstellungen		Beschreibung
Grundlagen	Reglertyp	Wählt die physikalischen Einheiten aus.
	Reglerlogik invertieren	Ermöglicht die Auswahl eines invers funktionierenden PID-Reglers. • Ist diese Option nicht ausgewählt, verhält sich der PID-Regler entsprechend der direkten Funktionsweise und der Ausgang des PID-Regelkreises steigt, wenn der Eingangswert < Sollwert ist. • Ist diese Option ausgewählt, steigt der Ausgang des PID-Regelkreises, wenn der Eingangswert > Sollwert ist.
	Betriebsart nach CPU-Neustart aktivieren	Startet den PID-Regler neu, nachdem er zurückgesetzt wurde oder wenn eine Eingangsgrenze überschritten und in den gültigen Bereich zurückgeführt wurde. Betriebsart einstellen: Definiert die Betriebsart, die der PID-Regler nach dem Neustart einnehmen soll.
	Eingang	Wählt entweder den Parameter Input oder den Parameter Input_PER (bei analog) für den Prozesswert aus. Input_PER kann direkt von einem analogen Eingangsmodul kommen.
	Ausgang	Gibt an, ob die digitalen Ausgänge (Output_UP und Output_DN) oder der analoge Ausgang (Output_PER) für den Ausgangswert verwendet werden soll.
	Rückmeldung	Gibt die Art des an den PID-Regler zurückgemeldeten Gerätezustands an: • Keine Rückmeldung (Standard) • Rückmeldung • Feedback_PER
Prozesswert	Skaliert sowohl den Bereich als auch die Grenzen für den Prozesswert. Wenn der Prozesswert unter die untere Grenze fällt oder über die obere Grenze steigt, wechselt der PID-Regelkreis in die inaktive Betriebsart und setzt den Ausgangswert auf 0. Um Input_PER zu verwenden, **müssen** Sie den analogen Prozesswert (Eingangswert) skalieren.	

Einstellungen		Beschreibung
Stellglied	Motorübergangs-zeit	Legt die Zeit vom offenen zum geschlossenen Ventil fest. (Ermitteln Sie diesen Wert auf dem Datenblatt oder anhand des Ventil-Typenschilds.)
	Kleinste EIN-Zeit	Legt die minimale Betätigungszeit für das Ventil fest. (Ermitteln Sie diesen Wert auf dem Datenblatt oder anhand des Ventil-Typenschilds.)
	Kleinste AUS-Zeit	Legt die minimale Pausenzeit für das Ventil fest. (Ermitteln Sie diesen Wert auf dem Datenblatt oder anhand des Ventil-Typenschilds.)
	Reaktion auf den Fehler	Definiert das Verhalten des Ventils, wenn ein Fehler erkannt oder der PID-Regler zurückgesetzt wird. Wenn Sie festlegen, dass eine Ersatzposition verwendet werden soll, geben Sie die "Sicherheitsposition" ein. Bei analoger Rückmeldung oder analogem Ausgang wählen Sie einen Wert zwischen der unteren und oberen Grenze für den Ausgang. Bei Digitalausgängen können Sie nur 0 % (aus) oder 100 % (ein) wählen.
	Positionsrückmeldung skalieren[1]	• "Oberer Endpunkt" und "Unterer Endpunkt" definieren die maximale positive Position (vollständig geöffnet) und die maximale negative Position (vollständig geschlossen). "Oberer Endpunkt" muss größer als "Unterer Endpunkt" sein. • "Prozesswert obere Grenze" und "Prozesswert untere Grenze" definieren die untere und obere Position des Ventils während der Einstellung und im Automatikbetrieb. • "FeedbackPER" ("Low" und "High") definieren die analoge Rückmeldung der Ventilposition. "FeedbackPER High" muss höher als "FeedbackPER Low" sein.
Erweitert	Prozesswertüberwachung	Legt den oberen und unteren Grenzwert für den Prozesswert fest.
	PID-Parameter	In diesem Fenster kann der Anwender bei Bedarf seine eigenen PID-Einstellungsparameter eingeben. Hierfür muss das Kontrollkästchen "Manuelle Eingabe aktivieren" aktiviert sein.

[1] "Positionsrückmeldung skalieren" kann nur geändert werden, wenn Sie in den grundlegenden Einstellungen "Rückmeldung" aktiviert haben.

10.2.12 PID_Temp-Regler konfigurieren

Die Parameter des Technologieobjekts legen die Funktionsweise des PID-Reglers fest. Öffnen Sie den Konfigurationseditor über das Symbol.

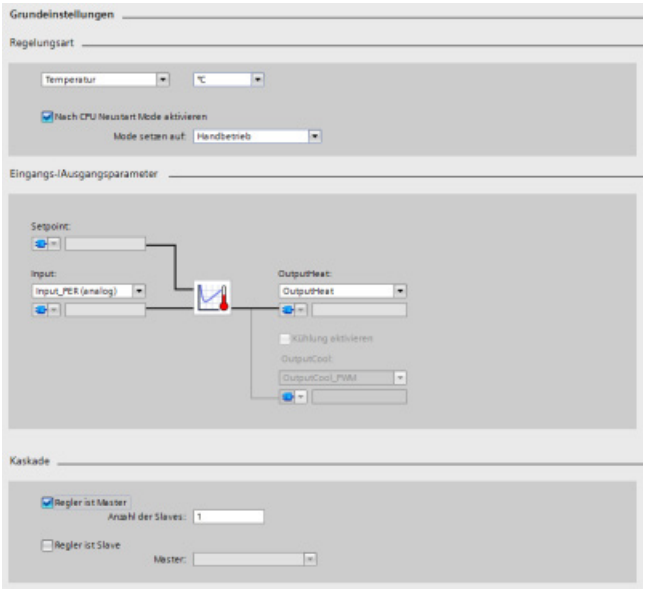


Tabelle 10- 32 Beispielhafte Konfigurationseinstellungen für die Anweisung PID_Temp

Einstellungen		Beschreibung
Grundlagen	Reglertyp	Wählt die physikalischen Einheiten aus.
	Betriebsart nach CPU-Neustart aktivieren	Startet den PID-Regler neu, nachdem er zurückgesetzt wurde oder wenn eine Eingangsgrenze überschritten und in den gültigen Bereich zurückgeführt wurde. Betriebsart einstellen: Definiert die Betriebsart, die der PID-Regler nach dem Neustart einnehmen soll.
	Eingang	Wählt entweder den Parameter Input oder den Parameter Input_PER (bei analog) für den Prozesswert aus. Input_PER kann direkt von einem analogen Eingangsmodul kommen.
	Ausgang Heizen	Gibt an, ob die digitalen Ausgänge (OutputHeat und OutputHeat_PWM) oder der analoge Ausgang (OutputHeat_PER (analog)) für den Ausgangswert verwendet werden soll.
	Ausgang Kühlen	Gibt an, ob die digitalen Ausgänge (OutputCool und OutputCool_PWM) oder der analoge Ausgang (OutputCool_PER (analog)) für den Ausgangswert verwendet werden soll.

Einstellungen		Beschreibung
Prozesswert	Skaliert sowohl den Bereich als auch die Grenzen für den Prozesswert. Wenn der Prozesswert unter die untere Grenze fällt oder über die obere Grenze steigt, wechselt der PID-Regelkreis in die inaktive Betriebsart und setzt den Ausgangswert auf 0. Um Input_PER zu verwenden, **müssen** Sie den analogen Prozesswert (Eingangswert) skalieren.	
Kaskade	Regler ist Master	Festlegung des Reglers als Master und Auswahl der Anzahl der Slaves.
	Regler ist Slave	Festlegung des Reglers als Slave und Auswahl der Anzahl der Master.

Reglertyp

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Physikalische Größe	"PhysicalQuantity"	Int (Enum)	• Allgemein • Temperatur (= Standard)	Vorauswahl für Wert der physikalischen Einheit Keine Regelung mit mehreren Werten und nicht bearbeitbar im Onlinemodus der funktionalen Ansicht.
Maßeinheit	"PhysicalUnit"	Int (Enum)	• Allgemein: Einheiten = % • Temperatur: Einheiten (mögliche Auswahl) = – °C (= Standard) – °F – K	Einheitenauswahl des Anwenders wird wieder auf “0” gesetzt, wenn Sie die physikalische Größe ändern.

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Betriebsart nach CPU-Neustart aktivieren	"RunModeByStartup"	Bool	Kontrollkästchen	Wenn auf WAHR gesetzt (= Standard), wechselt der Regler nach dem Aus- und Wiedereinschalten (Spannung ein - aus - ein) oder nach einem Wechsel des PLCs von STOP in RUN in den Zustand, der in der Variable "Mode" gespeichert ist. Andernfalls bleibt PID_Temp im Modus "inaktiv".
Betriebsart einstellen	"Mode"	Int (Enum)	Betriebsarten (mögliche Auswahl): • 0: Inaktiv • 1: Erstoptimierung • 2: Feineinstellung • 3: Automatikbetrieb • 4: Handbetrieb (= Standard)	Die Engineering-Station (ES) legt den Startwert der Variable “Mode” entsprechend der Benutzerauswahl fest. Der Standardwert von Mode (im TO-DB gespeichert) ist der Handbetrieb.

Eingangs-/Ausgangsparameter

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Sollwert	Sollwert	Real)	Real	Nur auf der Eigenschaftsseite zugänglich. Keine Regelung mit mehreren Werten im Onlinemodus der funktionalen Ansicht.
Auswahl Eingang	"Config.InputPerOn"	Bool (Enum)	Bool	Auswahl des zu verwendenden Eingangs. Mögliche Auswahl: • FALSCH: “Input” (Real) • WAHR: “Input_PER (analog)”
Eingang	Eingang oder Input_PER	Real oder Int	Real oder Int	Nur auf der Eigenschaftsseite zugänglich. Keine Regelung mit mehreren Werten im Onlinemodus der funktionalen Ansicht.
Auswahl Ausgang (Heizung)	"Config.Output.Heat.Select"	Int (Enum)	2 >= Config.Output.Heat.Select >= 0	Auswahl des zum Heizen zu verwendenden Ausgangs. Mögliche Auswahl: • “OutputHeat” (Real) • “OutputHeat_PWM” (Bool) (= Standard) • “OutputHeat_PER (analog)” (Wort) Wird einmal auf “OutputHeat” gesetzt, wenn das Kontrollkästchen "Dieser Regler ist ein Master" vom Anwender im Abschnitt "Kaskade" aktiviert wurde.
Ausgang (Heizung)	OutputHeat, OutputHeat_PER oder OutputHeat_PWM	Real oder Int oder Bool	Real, Int oder Bool	Nur auf der Eigenschaftsseite zugänglich. Keine Regelung mit mehreren Werten im Onlinemodus der funktionalen Ansicht.

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Ausgang aktivieren (Kühlung)	"Config.ActivateCooling"	Bool	Bool	Wenn dieses Kontrollkästchen aktiviert wird: • Wird der "Config.Output. Heat.PidLowerLimit = 0.0 einmal festgelegt. • Wird der Parameter "Config.ActivateCooling" auf WAHR gesetzt, statt auf FALSCH, wenn nicht aktiviert (= Standard). • Werden alle anderen Regler “Ausgang (Kühlung)” aktiviert (in den “Grundeinstellungen” und anderen Ansichten). • Wird die Linie vom PID-Symbol zu den Reglern nicht mehr grau, sondern schwarz angezeigt. • Wird das Kontrollkästchen “Dieser Regler ist ein Master” im Abschnitt "Kaskade" deaktiviert. Hinweis: Nur verfügbar, wenn Sie den Regler nicht als Master für eine Kaskade konfigurieren (das Kontrollkästchen “Dieser Regler ist ein Master” im Abschnitt “Kaskade” ist deaktiviert; "Config.Cascade.IsMaster" = FALSCH).

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Auswahl Ausgang (Kühlung)	"Config.Output.Cool.Select"	Int (Enum)	2 >= Config.Output.Heat.Select >= 0	Auswahl des zum Kühlen zu verwendenden Ausgangs. Mögliche Auswahl: • “OutputCool” (Real) • “OutputCool_PWM” (Bool) (= Standard) • “OutputCool_PER (analog)” (Wort) Nur verfügbar, wenn Sie "Ausgang aktivieren (Kühlung)" aktivieren; (Config.ActivateCooling = WAHR).
Ausgang (Kühlung)	OutputCool, OutputCool_PER oder OutputCool_PWM	Real oder Int oder Bool	Real, Int oder Bool	Nur auf der Eigenschaftsseite zugänglich. Keine Regelung mit mehreren Werten im Onlinemodus der funktionalen Ansicht.

Kaskadenparameter

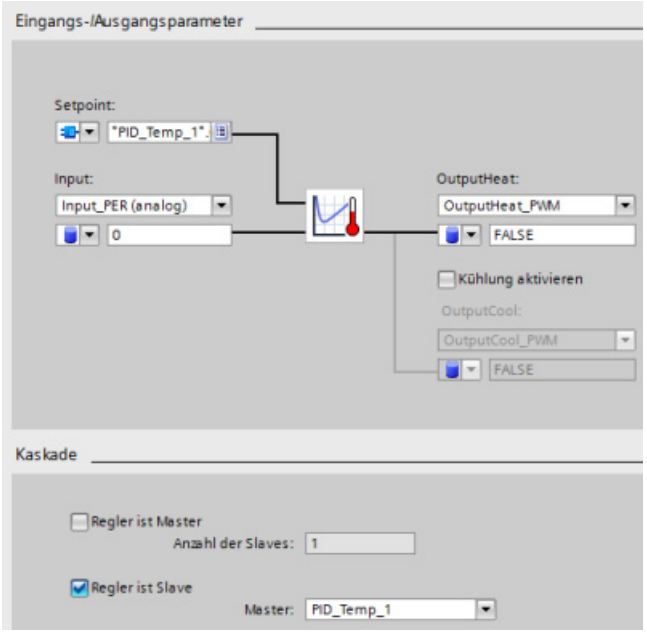
Mit den folgenden Parametern können Sie Regler als Master oder Slaves auswählen und die Anzahl der Slaveregler festlegen, die ihren Sollwert direkt vom Masterregler erhalten:

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Dieser Regler ist ein Master	"Config.Cascade.IsMaster"	Bool	Bool	Zeigt, ob dieser Regler ein Master in einer Kaskade ist. Wenn Sie dieses Kontrollkästchen aktivieren, führen Sie Folgendes durch: • Der Parameter "Config.Cascade.IsMaster " wird auf WAHR gesetzt, statt auf FALSCH, wenn nicht aktiviert (= Standard). • Setzt “Auswahl Ausgang (Heizung)” im Abschnitt “Eingangs-/Ausgangsparameter ” einmal auf “OutputHeat” (Config.Output.Heat.Select = 0). • Das Eingangsfeld "Anzahl der Slaves" wird aktiviert. • Das Kontrollkästchen “Ausgang aktivieren (Kühlung)” im Abschnitt “Eingangs-/Ausgangsparameter " wird deaktiviert. Hinweis: Nur verfügbar, wenn der Kühlausgang dieses Reglers deaktiviert ist (Kontrollkästchen “Ausgang aktivieren (Kühlung)” im Abschnitt “Eingangs-/Ausgangsparameter” ist deaktiviert (Config.ActivateCooling = FALSCH).

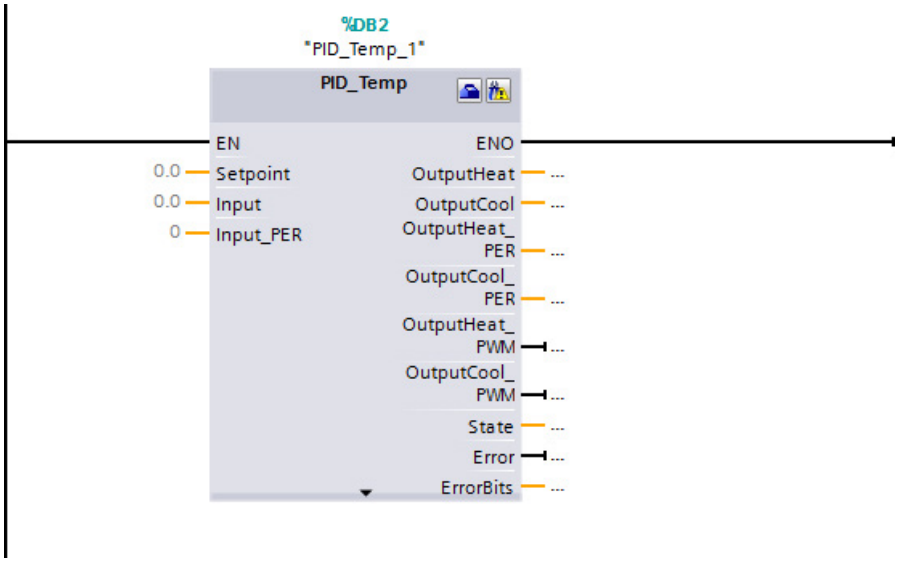
Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Anzahl der Slaves	"Config.Cascade.CountSlaves"	Int	255 >= Config.Cascade.CountSlaves >= 1	Anzahl der Slaveregler, die ihren Sollwert direkt von diesem Masterregler erhalten. Die Anweisung PID_Temp verarbeitet diesen Wert, zusammen mit anderen, für die Anti-Windup-Behandlung. "Anzahl der Slaves" ist nur verfügbar, wenn das Kontrollkästchen “Dieser Regler ist ein Master” aktiviert ist (Config.Cascade.IsMaster = WAHR).
Dieser Regler ist ein Slave	"Config.Cascade.IsSlave"	Bool	Bool	Zeigt, ob dieser Regler ein Slave in einer Kaskade ist. Wenn Sie dieses Kontrollkästchen aktivieren, wird der Parameter "Config.Cascade.IsSlave" auf WAHR gesetzt, statt auf FALSCH, wenn nicht aktiviert (= Standard). Sie müssen dieses Kontrollkästchen auf der Eigenschaftsseite aktivieren, um die Klappliste “SelectionMaster” freizugeben.

Beispiel: Kaskadieren von Reglern

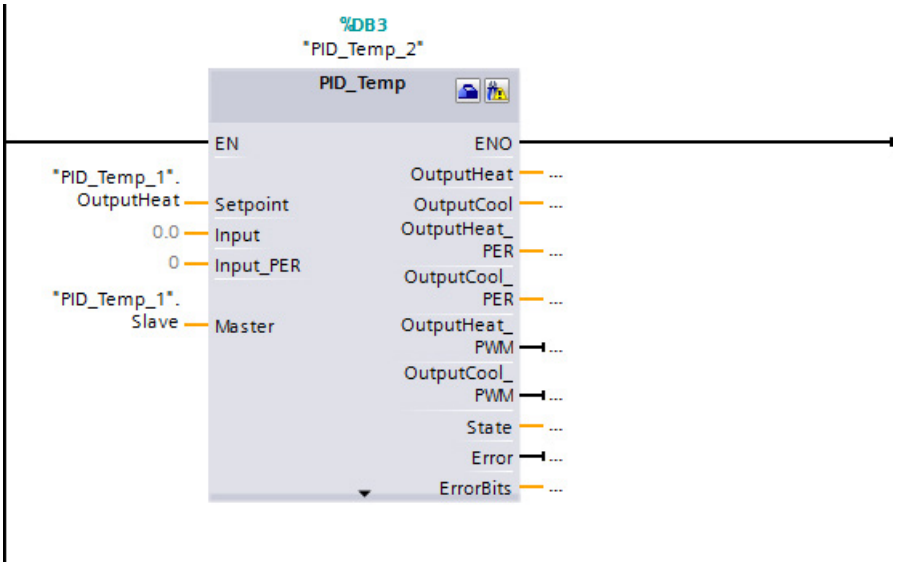
Im nachstehenden Dialog “Grundeinstellungen” sehen Sie den Abschnitt "Eingangs-/Ausgangsparameter" und den Abschnitt "Kaskade" für den Slaveregler "PID_Temp_2" nach Auswahl von "PID_Temp_1" als Master. Sie stellen die Verbindungen zwischen Master- und Slavereglern her:



Netzwerk 1: In diesen Netzwerken stellen Sie die Verbindung zwischen dem Master "PID_Temp_1" und dem Slave "PID_Temp_2" im Programmiereditor her:



Netzwerk 2: Sie stellen die Verbindung zwischen den Parametern “OutputHeat” und "Slave" des Masters "PID_Temp_1" mit den Parametern “Setpoint” und "Master" des Slaves "PID_Temp_2" her:



Selbsteinstellung des Temperaturprozesses

Die Anweisung PID_Temp bietet zwei Betriebsarten für die Selbsteinstellung:

- “Erstoptimierung” (Parameter “Mode” = 1)
- “Feineinstellung” (Parameter “Mode” = 2)

Je nach Reglerkonfiguration stehen verschiedene Varianten dieser Einstellmethoden zur Verfügung:

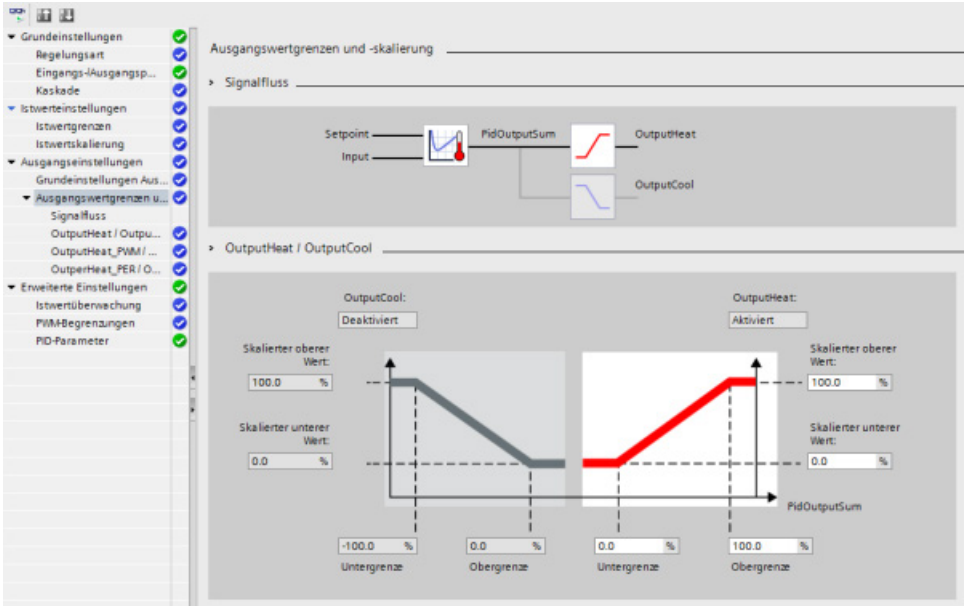
Konfiguration	Regler mit Heizausgang	Regler mit Heiz- und Kühlausgang und Verwendung des Kühlfaktors	Regler mit Heiz- und Kühlausgang und Verwendung von zwei PID-Parametersätzen
Zugehörige TO-DB-Werte	• Config.ActivateCooling = FALSCH • Config.AdvancedCooling = irrelevant	• Config.ActivateCooling = WAHR • Config.AdvancedCooling = FALSCH	• Config.ActivateCooling = WAHR • Config.AdvancedCooling = WAHR
Verfügbare Einstellmethoden	• “Erstoptimierung Heizung” • “Feineinstellung Heizung" (Kühlversatz kann nicht verwendet werden)	• “Erstoptimierung Heizung” • “Feineinstellung Heizung" (Kühlversatz kann verwendet werden)	• “Erstoptimierung Heizung und Kühlung” • “Erstoptimierung Heizung” • “Erstoptimierung Kühlung” • “Feineinstellung Heizung" (Kühlversatz kann verwendet werden) • “Feineinstellung Kühlung" (Heizversatz kann verwendet werden)

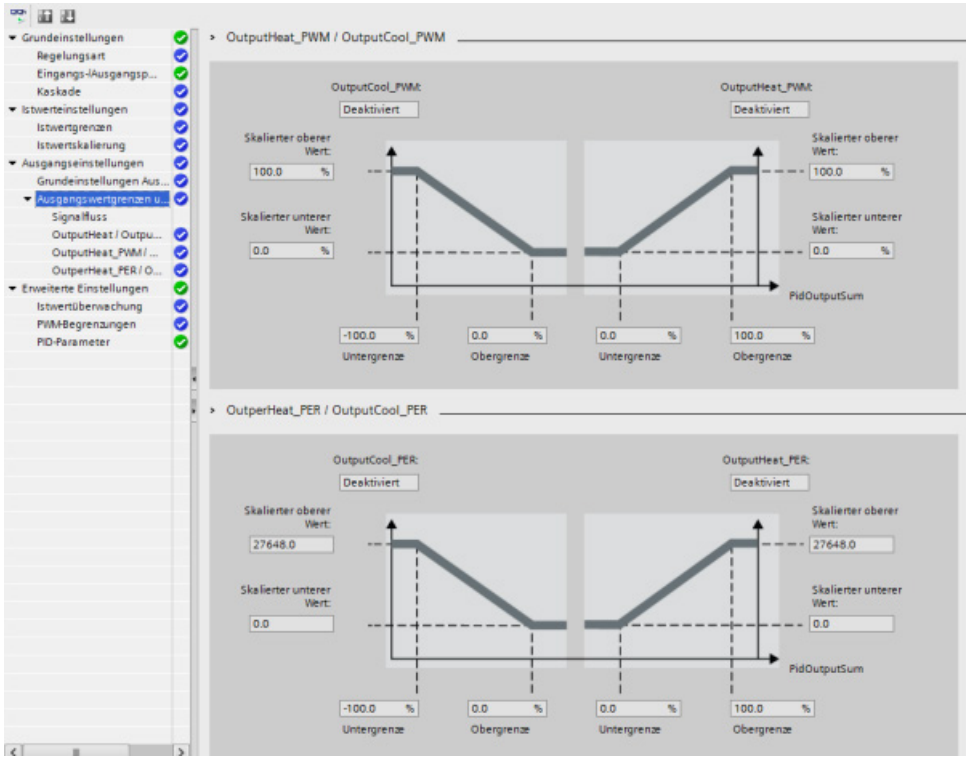
Ausgangswertgrenzen und -skalierung

Kühlaktivierung deaktiviert

Wenn Sie die Anweisung PID_Temp als Master für eine Kaskade konfigurieren, ist das Kontrollkästchen “Ausgang aktivieren (Kühlung)” in der Ansicht "Grundeinstellungen" nicht selektiert und deaktiviert, und auch alle Einstellungen in der Ansicht "Ausgangseinstellungen", die von der Aktivierung der Kühlung abhängig sind, sind deaktiviert.

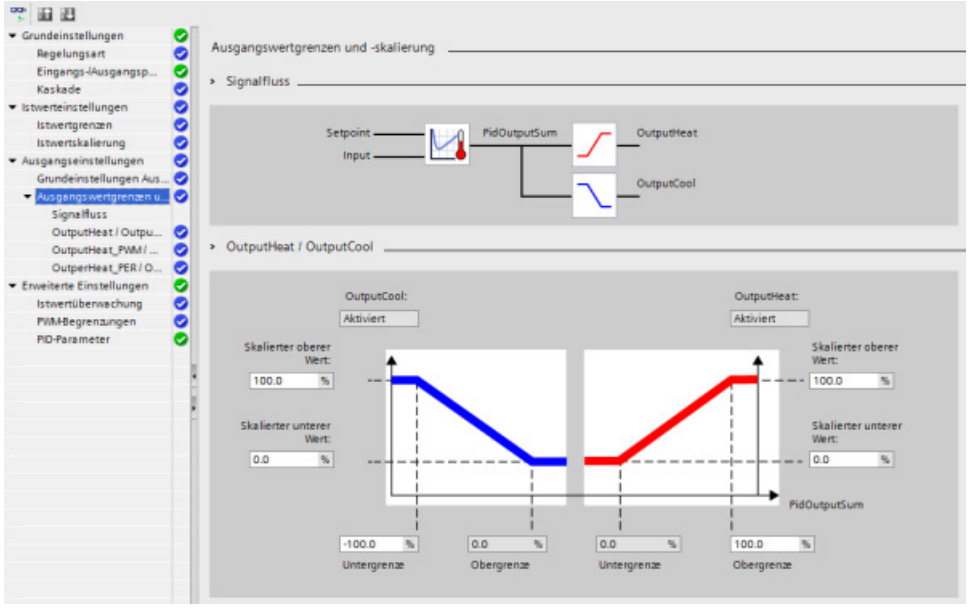
Die nachstehende Abbildung zeigt den Abschnitt "Ausgangswertgrenzen und -skalierung” in der Ansicht “Ausgangseinstellungen” mit deaktivierter Kühlung (OutputHeat_PWM in der Ansicht “Eingangs-/Ausgangsparameter” ausgewählt und OutputHeat immer aktiviert):

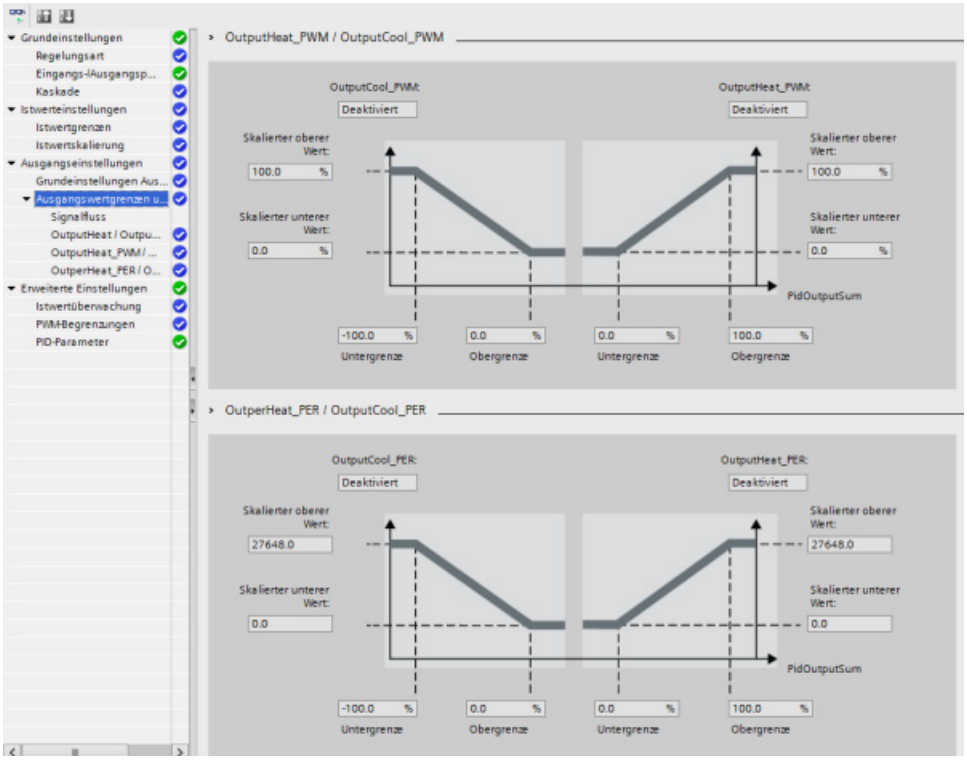




Kühlaktivierung aktiviert

Die nachstehende Abbildung zeigt den Abschnitt "Ausgangswertgrenzen und -skalierung" in der Ansicht "Ausgansgeinstellungen" mit aktivierter Kühlung (OutputCool_PER und OutputHeat_PWM in der Ansicht "Eingangs-/Ausgangsparameter" ausgewählt; OutputCool und OutputHeat immer aktiviert):





Betriebsarten

Um die Betriebsart manuell zu ändern, müssen Sie den Parameter "Mode" des Reglers festlegen und ihn aktivieren, indem Sie “ModeActivate” von FALSCH nach WAHR setzen (ausgelöst durch eine steigende Flanke). Sie müssen “ModeActivate” vor dem nächsten Betriebsartenwechsel zurücksetzen; der Parameter wird nicht automatisch zurückgesetzt.

Der Ausgangsparameter “State” zeigt die aktuelle Betriebsart und wird, wenn möglich, in den gewünschten “Mode” gesetzt. Der Parameter “State” kann nicht direkt geändert werden; er wird nur über den Parameter “Mode” oder durch automatische Betriebsartenwechsel des Reglers geändert.

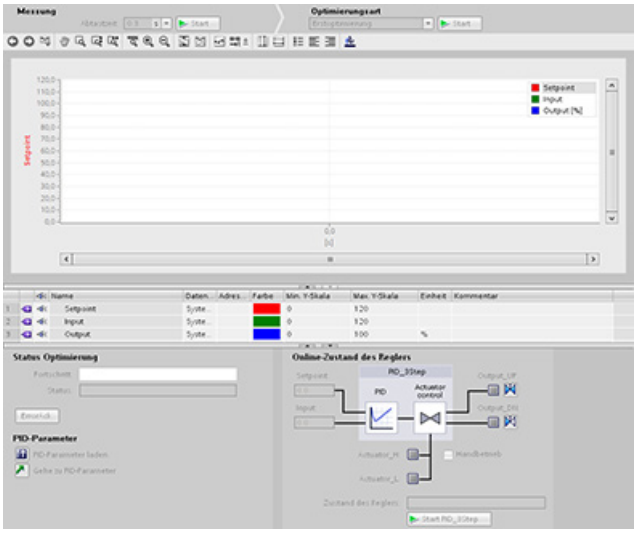
"Mode" / "State"	Name	Beschreibung
0	Inaktiv	Die Anweisung PID_Temp: • Deaktiviert den PID-Algorithmus und die Impulsdauermodulation • Setzt alle Reglerausgänge (OutputHeat, OutputCool, OutputHeat_PWM, OutputCool_PWM, OutputHeat_PER, OutputCool_PER) unabhängig von den konfigurierten Ausgangsgrenzen oder -versatzwerten auf "0" (FALSCH). Sie erreichen diesen Modus durch Einstellung von “Mode” = 0, “Reset” = WAHR oder durch einen Fehler.
1	Erstoptimierung (Anlaufeinstellung/SUT)	Diese Betriebsart legt die Parameter während des ersten Anlaufs des Reglers fest. Im Gegensatz zu PID_Compact müssen Sie bei PID_Temp mit den Parametern “Heat.EnableTuning” und “Cool.EnableTuning” auswählen, ob Sie Heizungseinstellung, Kühlungseinstellung oder beides benötigen. Sie können die "Erstoptimierung" im inaktiven Modus, im Automatikbetrieb oder Handbetrieb aktivieren. Ist die Einstellung erfolgreich, wechselt PID_Temp in den Automatikbetrieb. Ist die Einstellung nicht erfolgreich, ist der Wechsel der Betriebsart von “ActivateRecoverMode” abhängig.
2	Feineinstellung (Einstellung in Run/TIR)	Diese Betriebsart legt die optimale Parametrierung des PID-Reglers am Sollwert fest. Im Gegensatz zu PID_Compact müssen Sie bei PID_Temp mit den Parametern “Heat.EnableTuning” und “Cool.EnableTuning” auswählen, ob Sie Heizungseinstellung oder Kühlungseinstellung benötigen. Sie können die "Feineinstellung" im inaktiven Modus, im Automatikbetrieb oder Handbetrieb aktivieren. Ist die Einstellung erfolgreich, wechselt PID_Temp in den Automatikbetrieb. Ist die Einstellung nicht erfolgreich, ist der Wechsel der Betriebsart von “ActivateRecoverMode” abhängig.
3	Automatikbetrieb	Im Automatikbetrieb (standardmäßiger PID-Regelbetrieb) bestimmt das Ergebnis des PID-Algorithmus die Ausgangswerte. PID_Temp wechselt in den inaktiven Modus, wenn ein Fehler auftritt und “ActivateRecoverMode” = FALSCH ist. Tritt ein Fehler auf und ist “ActivateRecoverMode” = WAHR, ist der Wechsel der Betriebsart von dem Fehler abhängig. Weitere Informationen finden Sie bei den ErrorBit-Parametern (Seite 513) der Anweisung PID_Temp.

"Mode" / "State"	Name	Beschreibung
4	Handbetrieb	In dieser Betriebsart skaliert, begrenzt und überträgt der PID-Regler den Wert des Parameters “ManualValue” an die Ausgänge. Der PID-Regler weist den “ManualValue” in der Skalierung des PID-Algorithmus zu (wie “PidOutputSum”), deshalb entscheidet dessen Wert, ob er an den Heiz- oder Kühlausgängen wirksam ist. Sie erreichen diesen Modus durch Einstellung von “Mode” = 4 oder “ManualEnable” = WAHR.
5	Ersatzausgangswert mit Fehlerüberwachung (Wiederherstellungsmodus)	Sie können diese Betriebsart durch Einstellung von "Mode" = 5 aktivieren. Die Betriebsart ist eine automatische Fehlerreaktion des Reglers, wenn der Automatikbetrieb zum Zeitpunkt des Auftretens des Fehlers aktiv ist: • SetSubstituteOutput = FALSCH (letzter gültiger Ausgangswert) • SetSubstituteOutput = WAHR (im Parameter "SubstituteOutput" gespeicherter Wert) Wenn PID_Temp im “Automatikbetrieb” ist und der Parameter “ActivateRecoverMode” = WAHR ist, wechselt PID_Temp in diese Betriebsart, wenn die folgenden Fehler auftreten: • "Ungültiger Wert am Parameter "Input_PER". Prüfen Sie auf einen Fehler am Analogeingang (z. B. Drahtbruch)." (ErrorBits = DW#16#0002) • "Ungültiger Wert am Parameter "Input". Wert ist keine Zahl.” (ErrorBits = DW#16#0200) • "Berechnung des Ausgangswerts fehlgeschlagen. Prüfen Sie die PID-Parameter." (ErrorBits = DW#16#0400) • "Ungültiger Wert am Parameter "Setpoint". Wert ist keine Zahl.” (ErrorBits = DW#16#1000) Steht der Fehler nicht mehr an, wechselt PID_Temp automatisch wieder in den Automatikbetrieb.

10.2.13 PID_Compact- und PID_3Step-Regler in Betrieb nehmen

Sie konfigurieren den PID-Regler im Inbetriebnahme-Editor für die Selbsteinstellung beim Anlauf und für die Selbsteinstellung während des Betriebs. Um den Inbetriebnahme-Editor aufzurufen, klicken Sie im Anweisungsverzeichnis oder in der Projektnavigation auf das entsprechende Symbol.

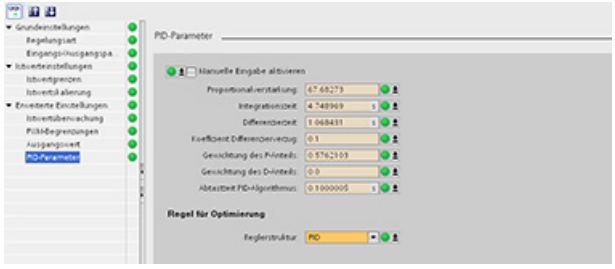
Tabelle 10- 33 Beispiel für einen Inbetriebnahmebildschirm (PID_3Step)

	• Messung: Um den Sollwert, den Prozesswert (Eingangswert) und den Ausgangswert in einer Echtzeitkurve anzuzeigen, geben Sie die Abtastzeit ein und klicken auf die Schaltfläche "Start". • Optimierungsart: Um den PID-Regler einzustellen, wählen Sie entweder "Erstoptimierung" oder "Feineinstellung" (manuell) aus und klicken auf die Schaltfläche "Start". Der PID-Regler durchläuft mehrere Phasen, um die Zeiten für Systemantwort und Aktualisierung zu berechnen. Anhand dieser Werte werden die entsprechenden Optimierungsparameter berechnet. Nach Abschluss des Einstellungsprozesses können Sie die neuen Parameter speichern, indem Sie im Inbetriebnahme-Editor im Bereich "PID-Parameter" auf die Schaltfläche "PID-Parameter laden" klicken. Tritt während der Einstellung ein Fehler auf, geht der Ausgangswert des PID-Reglers auf 0. Der PID-Regler wird in die "inaktive" Betriebsart versetzt. Der Status zeigt den Fehler an.

PID-Startwertsteuerung

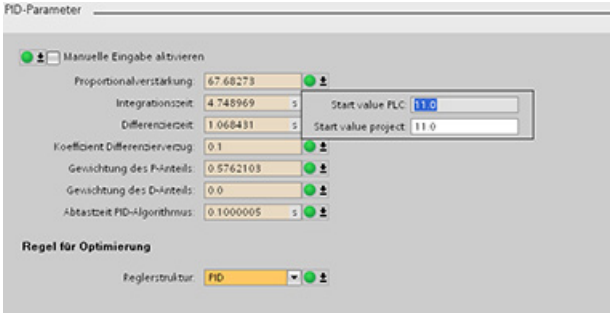
Sie können die Istwerte der PID-Konfigurationsparameter ändern, so dass das Verhalten des PID-Reglers im Online-Modus optimiert werden kann.

Öffnen Sie die Technologieobjekte für Ihren PID-Regler und dessen Konfigurationsobjekt. Um die Startwertsteuerung aufzurufen, klicken Sie auf das Brillensymbol in der linken oberen Ecke des Dialogfelds:



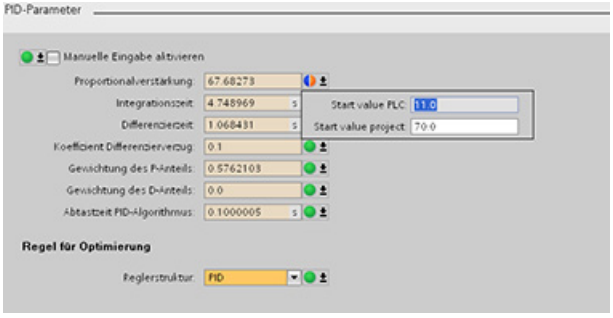
Nun können Sie die Werte beliebiger Konfigurationsparameter Ihres PID-Reglers wie in nachstehender Abbildung gezeigt ändern.

Sie können für jeden Parameter den Istwert mit dem Startwert des Projekts (offline) und dem Startwert des PLCs (online) vergleichen. Dies ist erforderlich, um die Online-/Offline-Unterschiede des Technologieobjekt-Datenbausteins (TO-DB) zu vergleichen und die Werte zu kennen, die beim nächsten Wechsel von STOP nach START des PLCs als aktuelle Werte verwendet werden. Außerdem erhalten Sie durch ein Vergleichssymbol einen visuellen Hinweis, um Online-/Offline-Unterschiede schnell zu erkennen:



Die obige Abbildung zeigt den Bildschirm der PID-Parameter mit Vergleichssymbolen. Die Symbole weisen darauf hin, welche Werte sich zwischen Online- und Offline-Projekt unterscheiden. Ein grünes Symbol kennzeichnet die Werte, die gleich sind. Ein blau-orangefarbenes Symbol zeigt an, dass die Werte voneinander abweichen.

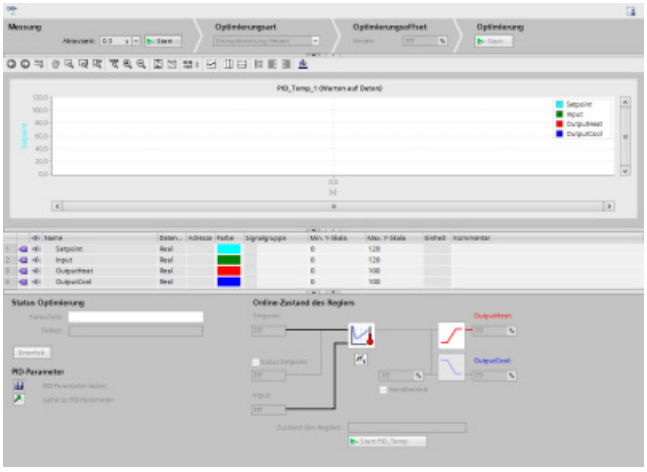
Außerdem können Sie auf die Parameterschaltfläche mit dem Abwärtspfeil klicken, um ein kleines Fenster anzuzeigen, in dem für jeden Parameter der Startwert des Projekts (offline) und der Startwert des PLCs (online) angezeigt werden:



10.2.14 PID_Temp-Regler in Betrieb nehmen

Sie konfigurieren den PID-Regler im Inbetriebnahme-Editor für die Selbsteinstellung beim Anlauf und für die Selbsteinstellung während des Betriebs. Um den Inbetriebnahme-Editor aufzurufen, klicken Sie im Anweisungsverzeichnis oder in der Projektnavigation auf das entsprechende Symbol.

Tabelle 10- 34 Beispiel für einen Inbetriebnahmebildschirm (PID_Temp)

	Messung: Um den Sollwert, den Prozesswert (Eingangswert) und den Ausgangswert in einer Echtzeitkurve anzuzeigen, geben Sie die Abtastzeit ein und klicken auf die Schaltfläche "Start". Optimierungsart: Um den PID_Temp-Regler einzustellen, wählen Sie entweder "Erstoptimierung" oder "Feineinstellung" (manuell) aus und klicken auf die Schaltfläche "Start". Der PID-Regler durchläuft mehrere Phasen, um die Zeiten für Systemantwort und Aktualisierung zu berechnen. Anhand dieser Werte werden die entsprechenden Optimierungsparameter berechnet. Nach Abschluss des Einstellungsprozesses können Sie die neuen Parameter speichern, indem Sie im Inbetriebnahme-Editor im Bereich "PID-Parameter" auf die Schaltfläche "PID-Parameter laden" klicken. Tritt während der Einstellung ein Fehler auf, geht der Ausgangswert des PID-Reglers auf "0". Der PID-Regler wird in die "inaktive" Betriebsart versetzt. Der Status zeigt den Fehler an.

PWM-Grenzwerte

Stellglieder, die über die PWM-Funktion der Software des PID_Temp-Reglers geregelt werden, müssen möglicherweise vor einer zu kurzen Impulsdauer geschützt werden (z. B. muss ein Thyristorrelais mehr als 20 ms eingeschaltet sein, damit es überhaupt reagieren kann). Hierfür weisen Sie eine minimale Einschaltzeit zu. Das Stellglied kann auch kurze Impulse vernachlässigen und somit die Regelqualität beeinträchtigen. Hierfür kann eine minimale Ausschaltzeit erforderlich sein (z. B. zum Verhindern von Überhitzung).

Um die Ansicht "PWM-Begrenzungen" aufzurufen, müssen Sie die funktionale Ansicht in der Konfiguration der Technologieobjekte (TO) öffnen und im Navigationsverzeichnis unter "Erweiterte Einstellungen" die "PWM-Begrenzungen" auswählen.

Wenn Sie die Ansicht “PWM-Begrenzungen in der funktionalen Ansicht öffnen und die Überwachung aktivieren (Brillensymbol), zeigen alle Bedienelemente den Online-Überwachungswert aus dem TO-DB mit orangefarbenem Hintergrund und Regelung mehrerer Werte an, und Sie können die Werte bearbeiten (sofern die Konfigurationsbedingungen erfüllt sind, siehe hierzu nachstehende Tabelle).

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Minimale Einschaltzeit (Heizen) [1,2]	"Config.Output.Heat.MinimumOnTime"	Real	100000.0 >= "Config.Output.Heat.MinimumOnTime >= 0,0	Ein Impuls an OutputHeat_PWM ist niemals kürzer als dieser Wert.
Minimale Ausschaltzeit (Heizen) [1,2]	"Config.Output.Heat.MinimumOffTime"	Real	100000.0 >= "Config.Output.Heat.MinimumOffTime >= 0,0	Eine Unterbrechung an OutputHeat_PWM ist niemals kürzer als dieser Wert.
Minimale Einschaltzeit (Kühlen) [1,3,4]	"Config.Output.Cool.MinimumOnTime"	Real	100000.0 >= Config.Output.Cool.MinimumOnTime >= 0,0	Ein Impuls an OutputCool_PWM ist niemals kürzer als dieser Wert.
Minimale Ausschaltzeit (Kühlen) [1,3,4]	"Config.Output.Cool.MinimumOffTime"	Real	100000.0 >= Config.Output.Cool.MinimumOffTime >= 0,0	Eine Unterbrechung an OutputCool_PWM ist niemals kürzer als dieser Wert.

1 Das Feld zeigt den Wert in der Zeiteinheit "s" (Sekunden) an.

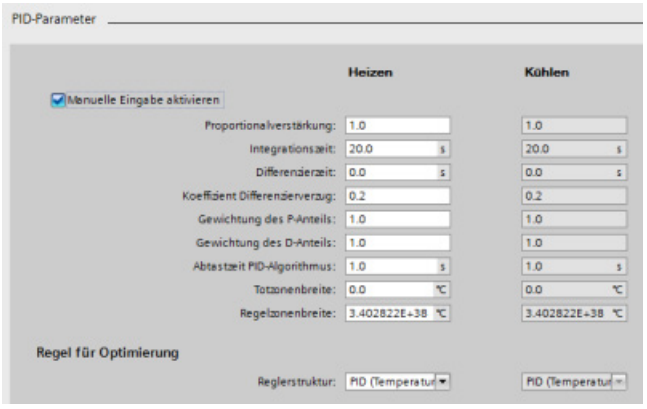
2 Wenn für "Auswahl Ausgang (Heizung)" in der Ansicht “Grundeinstellungen” nicht “OutputHeat_PWM” (Config.Output.Heat.Select = WAHR) eingestellt ist, müssen Sie diesen Wert auf "0,0" setzen.

3 Wenn für "Auswahl Ausgang (Kühlung)" in der Ansicht “Grundeinstellungen” nicht “OutputCool_PWM” (Config.Output.Cool.Select = WAHR) eingestellt ist, müssen Sie diesen Wert auf "0,0" setzen.

4 Nur verfügbar, wenn Sie in der Ansicht "Grundeinstellungen" "Ausgang aktivieren (Kühlung)" aktivieren (Config.ActivateCooling = WAHR).

PID-Parameter

Nachstehend wird der Abschnitt "PID-Parameter" der Ansicht “Erweiterte Einstellungen” mit der Kühlung und/oder der Funktion “PID-Parameterumschaltung” deaktiviert gezeigt.



Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Manuelle Eingabe aktivieren	"Retain.CtrlParams. SetByUser"	Bool	Bool	Sie müssen dieses Kontrollkästchen aktivieren, um PID-Parameter manuell eingeben zu können.
Proportionalverstärkung (Heizen) [2]	"Retain.CtrlParams. Heat.Gain"	Real	Verstärkung >= 0,0	PID-Proportionalverstärkung für die Heizung.
Integrationszeit (Heizen) [1,2]	"Retain.CtrlParams. Heat.Ti"	Real	100000.0 >= Ti >= 0,0	PID-Integrationszeit für die Heizung.
Differenzierzeit (Heizen) [1,2]	"Retain.CtrlParams. Heat.Td"	Real	100000.0 >= Td >= 0,0	PID-Differenzierzeit für die Heizung.
Koeffizient Differenzierverzug (Heizen) [2]	"Retain.CtrlParams. Heat.TdFiltRatio"	Real	TdFiltRatio >= 0,0	Der Koeffizient des PID-Differenzierverzugs für die Heizung, der den Differenzierverzug als Koeffizienten aus der PID-Differenzierzeit festlegt.
Gewichtung des P-Anteils (Heizen) [2]	"Retain.CtrlParams. Heat.PWeighting"	Real	1,0 >=PWeighting >= 0,0	Gewichtung der PID-Proportionalverstärkung für die Heizung entweder im direkten oder im zurückgeführten Regelpfad.

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Gewichtung des D-Anteils (Heizen) [2]	"Retain.CtrlParams. Heat.DWeighting"	Real	1,0 >=DWeighting >= 0,0	Gewichtung des PID-Differenzialanteils für die Heizung entweder im direkten oder im zurückgeführten Regelpfad.
Abtastzeit des PID-Algorithmus (Heizen) [1,2]	"Retain.CtrlParams. Heat.Cycle"	Real	100000,0 >=Zyklus > 0,0	Interner Aufrufzyklus des PID-Reglers für die Heizung. Gerundet auf ein ganzzahliges Vielfaches der Zykluszeit des FB-Aufrufs.
Totzonenbreite (Heizen) [2,3]	"Retain.CtrlParams. Heat.DeadZone"	Real	DeadZone >= 0,0	Breite der Totzone für die Regelabweichung der Heizung.
Regelzone (Heizen) [2,3]	"Retain.CtrlParams. Heat.ControlZone"	Real	ControlZone > 0,0	Breite der Regelabweichungszone für die Heizung, wo der PID-Regler aktiv ist. Wenn die Regelabweichung diesen Bereich verlässt, wird der Ausgang auf maximale Ausgangswerte umgeschaltet. Der Standardwert ist "MaxReal", also ist die Regelzone so lange deaktiviert, wie die Selbsteinstellung nicht ausgeführt wird. Der Wert "0,0" ist für die Regelzone nicht zulässig. Beim Wert "0,0" verhält sich PID_Temp wie ein Zweistellungsregler, der stets mit voller Leistung heizt oder kühlt.

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Reglerstruktur (Heizen)	"PIDSelfTune.SUT.TuneRuleHeat", "PIDSelfTune.TIR.TuneRuleHeat"	Int	"PIDSelfTune.SUT.TuneRuleHeat" = 0..2, "PIDSelfTune.TIR.TuneRuleHeat" = 0..5	Sie können den Einstellalgorithmus für die Heizung auswählen. Mögliche Auswahl: • PID (Temperatur) (= Standard) ("PIDSelfTune.SUT.TuneRuleHeat" = 2) ("PIDSelfTune.TIR.TuneRuleHeat" = 0) • PID ("PIDSelfTune.SUT.TuneRuleHeat" = 0) ("PIDSelfTune.TIR.TuneRuleHeat" = 0) • PI ("PIDSelfTune.SUT.TuneRuleHeat" = 1) ("PIDSelfTune.TIR.TuneRuleHeat" = 4) Alle anderen Kombinationen zeigen “Benutzerdefiniert” an, doch “Benutzerdefiniert” ist standardmäßig nicht vorhanden. “PID (Temperatur)” ist für PID_Temp neu, mit einer spezifischen Erstoptimierungsmethode (SUT) für Temperaturprozesse.
Proportionalverstärkung (Kühlen) [4]	"Retain.CtrlParams.Cool.Gain"	Real	Verstärkung >= 0,0	PID-Proportionalverstärkung für die Kühlung.
Integrationszeit (Kühlen) [1,4]	"Retain.CtrlParams.Cool.Ti"	Real	100000,0 >=Ti >= 0,0	PID-Integrationszeit für die Kühlung.
Differenzierzeit (Kühlen) [1,4]	"Retain.CtrlParams.Cool.Td"	Real	100000,0 >=Td >= 0,0	PID-Differenzierzeit für die Kühlung.

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Koeffizient Differenzierverzug (Kühlen) [4]	Retain.CtrlParams. Cool.TdFiltRatio"	Real	TdFiltRatio >= 0,0	Der Koeffizient des PID-Differenzierverzugs für die Kühlung, der den Differenzierverzug als Koeffizienten aus der PID-Differenzierzeit festlegt.
Gewichtung des P-Anteils (Kühlen) [4]	"Retain.CtrlParams. Cool.PWeighting"	Real	1,0 >=PWeighting >= 0,0	Gewichtung der PID-Proportionalverstärkung für die Kühlung entweder im direkten oder im zurückgeführten Regelpfad.
Gewichtung des D-Anteils (Kühlen) [4]	Retain.CtrlParams. Cool.DWeighting"	Real	1,0 >=DWeighting >= 0,0	Gewichtung des PID-Differenzialanteils für die Kühlung entweder im direkten oder im zurückgeführten Regelpfad.
Abtastzeit des PID-Algorithmus (Kühlen) [1,4]	"Retain.CtrlParams. Cool.Cycle"	Real	100000,0 >=Zyklus > 0,0	Interner Aufrufzyklus des PID-Reglers für die Kühlung. Gerundet auf ein ganzzahliges Vielfaches der Zykluszeit des FB-Aufrufs.
Totzonenbreite (Kühlen) [3,4]	"Retain.CtrlParams. Cool.DeadZone"	Real	DeadZone >= 0,0	Breite der Totzone für die Regelabweichung der Kühlung.

Einstellung	TO-DB-Parameter	Da-tentyp	Wertebereich	Beschreibung
Regelzone (Kühlen) [3,4]	"Retain.CtrlParams. Cool.ControlZone"	Real	ControlZone > 0,0	Breite der Regelabweichungszone für die Kühlung, wo der PID-Regler aktiv ist. Wenn die Regelabweichung diesen Bereich verlässt, wird der Ausgang auf maximale Ausgangswerte umgeschaltet. Der Standardwert ist "MaxReal", also ist die Regelzone so lange deaktiviert, wie die Selbsteinstellung nicht ausgeführt wird. Der Wert "0,0" ist für die Regelzone nicht zulässig. Beim Wert "0,0" verhält sich PID_Temp wie ein Zweistellungsregler, der stets mit voller Leistung heizt oder kühlt.

Einstellung	TO-DB-Parameter	Datentyp	Wertebereich	Beschreibung
Reglerstruktur (Kühlen)	"PIDSelfTune.SUT.TuneRuleCool", "PIDSelfTune.TIR.TuneRuleCool"	Int	"PIDSelfTune.SUT.TuneRuleHeat" = 0..2, "PIDSelfTune.TIR.TuneRuleHeat" = 0..5	Sie können den Einstellalgorithmus für die Kühlung auswählen. Mögliche Auswahl: • PID (Temperatur) (= Standard) ("PIDSelfTune.SUT.TuneRuleCool" = 2) ("PIDSelfTune.TIR.TuneRuleCool = 0) • PID ("PIDSelfTune.SUT.TuneRuleCool" = 0) ("PIDSelfTune.TIR.TuneRuleCool" = 0) • PI ("PIDSelfTune.SUT.TuneRuleCool" = 1) ("PIDSelfTune.TIR.TuneRuleCool" = 4) Alle anderen Kombinationen zeigen “Benutzerdefiniert” an, doch “Benutzerdefiniert” ist standardmäßig nicht vorhanden. “PID (Temperatur)” ist für PID_Temp neu, mit einer spezifischen Erstoptimierungsmethode (SUT) für Temperaturprozesse. Nur verfügbar, wenn Sie die folgenden Elemente auswählen: “Ausgang aktivieren (Kühlung)” in der Ansicht “Grundeinstellungen” ("Config.ActivateCooling" = WAHR) und “PID-Parameterumschaltung” in der Ansicht “Ausgangseinstellungen” (Config.AdvancedCooling = WAHR).

Einstellung	TO-DB-Parameter	Da-tentyp	Wertebereich	Beschreibung

1 Das Feld zeigt den Wert in der Zeiteinheit "s" (Sekunden) an.

2 Nur verfügbar, wenn Sie in den PID-Parametern "Manuelle Eingabe aktivieren” aktivieren ("Retain.CtrlParams.SetByUser" = WAHR).

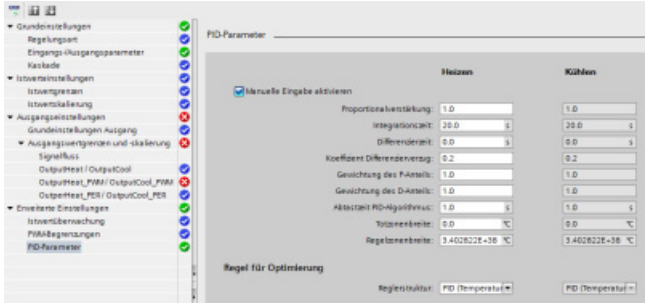
3 Die Maßeinheit wird am Ende des Felds wie in der Ansicht "Grundeinstellungen" ausgewählt angezeigt.

4 Nur verfügbar, wenn Sie die folgenden Elemente auswählen: "Manuelle Eingabe aktivieren” in den PID-Parametern ("Retain.CtrlParams.SetByUser" = WAHR), “Ausgang aktivieren (Kühlung)” in der Ansicht “Grundeinstellungen” ("Config.ActivateCooling" = WAHR) und “PID-Parameterumschaltung” in der Ansicht “Ausgangseinstellungen” (Config.AdvancedCooling = WAHR).

PID-Startwertsteuerung

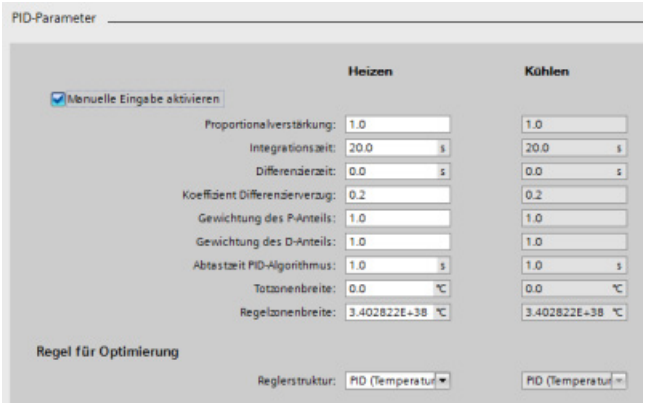
Sie können die Istwerte der PID-Konfigurationsparameter ändern, so dass das Verhalten des PID-Reglers im Online-Modus optimiert werden kann.

Öffnen Sie die Technologieobjekte für Ihren PID-Regler und dessen Konfigurationsobjekt. Um die Startwertsteuerung aufzurufen, klicken Sie auf das Brillensymbol in der linken oberen Ecke des Dialogfelds:



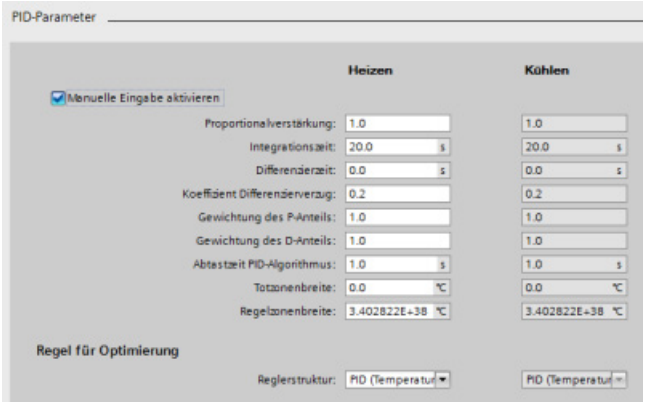
Nun können Sie die Werte beliebiger Konfigurationsparameter Ihres PID-Reglers wie in nachstehender Abbildung gezeigt ändern.

Sie können für jeden Parameter den Istwert mit dem Startwert des Projekts (offline) und dem Startwert des PLCs (online) vergleichen. Dies ist erforderlich, um die Online-/Offline-Unterschiede des Technologieobjekt-Datenbausteins (TO-DB) zu vergleichen und die Werte zu kennen, die beim nächsten Wechsel von STOP nach START des PLCs als aktuelle Werte verwendet werden. Außerdem erhalten Sie durch ein Vergleichssymbol einen visuellen Hinweis, um Online-/Offline-Unterschiede schnell zu erkennen:



Die obige Abbildung zeigt den Bildschirm der PID-Parameter mit Vergleichssymbolen. Die Symbole weisen darauf hin, welche Werte sich zwischen Online- und Offline-Projekt unterscheiden. Ein grünes Symbol kennzeichnet die Werte, die gleich sind. Ein blau-orangefarbenes Symbol zeigt an, dass die Werte voneinander abweichen.

Außerdem können Sie auf die Parameterschaltfläche mit dem Abwärtspfeil klicken, um ein kleines Fenster anzuzeigen, in dem für jeden Parameter der Startwert des Projekts (offline) und der Startwert des PLCs (online) angezeigt werden:



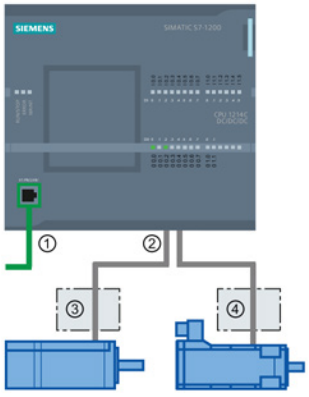
10.3 Bewegungssteuerung

Die CPU bietet Bewegungssteuerungsfunktionen für den Betrieb von Schrittmotoren und Servomotoren mit Impulsschnittstelle. Die Bewegungssteuerungsfunktion übernimmt die Steuerung und Überwachung der Antriebe.

- Das Technologieobjekt "Achse" konfiguriert die Daten des mechanischen Antriebs, die Antriebsschnittstelle, die dynamischen Parameter und andere Eigenschaften des Antriebs.
- Sie konfigurieren die Impuls- und Richtungsausgänge der CPU zum Steuern des Antriebs.
- Ihr Anwenderprogramm nutzt die Bewegungssteuerungsanweisungen zum Steuern der Achse und zum Auslösen von Bewegungsaufgaben.
- Stellen Sie über die PROFINET-Schnittstelle eine Online-Verbindung zwischen der CPU und dem Programmiergerät her. Neben den Online-Funktionen der CPU stehen für die Bewegungssteuerung weitere Inbetriebnahme- und Diagnosefunktionen zur Verfügung.

Hinweis

Änderungen, die Sie an der Bewegungssteuerungskonfiguration vornehmen und im Betriebszustand RUN ins Zielsystem laden, werden erst dann wirksam, wenn die CPU von STOP in RUN geht.



① PROFINET

② Impuls- und Richtungsausgänge

③ Leistungsteil Schrittmotor

④ Leistungsteil Servomotor

Die DC/DC/DC-Varianten der S7-1200 CPU haben integrierte Ausgänge für die direkte Steuerung von Antrieben. Die Relaisvarianten der CPU benötigen das Signalboard mit DC-Ausgängen für die Antriebssteuerung.

Ein Signalboard (SB) erweitert die integrierten E/A um einige zusätzliche Ein- und Ausgänge. Ein SB mit zwei Digitalausgängen kann als Impuls- und Richtungsausgang zur Steuerung eines Motors verwendet werden. Ein SB mit vier Digitalausgängen kann für Impuls- und Richtungsausgänge zur Steuerung zweier Motoren verwendet werden. Integrierte Relaisausgänge können nicht als Impulsausgänge zur Motorsteuerung verwendet werden. Unabhängig davon, ob Sie integrierte E/A oder SB-E/A oder eine Kombination aus beiden verwenden, können Sie maximal vier Impulsgeneratoren definieren.

Die vier Impulsgeneratoren verfügen über Standard-E/A-Zuweisungen. Sie können die Impulsgeneratoren jedoch auch für jeden digitalen Ausgang an der CPU oder am SB konfigurieren. Impulsgeneratoren an der CPU können nicht SMs oder dezentralen E/A zugewiesen werden.

Hinweis

Impulsfolgen können von anderen Anweisungen im Anwenderprogramm nicht verwendet werden

Wenn Sie die Ausgänge der CPU oder des Signalboards als Impulsgeneratoren (für PWM oder Bewegungssteuerungsanweisungen) konfigurieren, steuern die entsprechenden Ausgangsadressen nicht mehr die Ausgänge. Wenn Ihr Anwenderprogramm einen Wert in einen Ausgang schreibt, der als Impulsgenerator genutzt wird, schreibt die CPU diesen Wert nicht in den physischen Ausgang.

Tabelle 10- 35 Maximale Anzahl von steuerbaren Antrieben

Typ der CPU		Integrierte E/A; Kein SB installiert		Mit einem SB (2 x DC-Ausgänge)		Mit einem SB (4 x DC-Ausgänge)	
		Mit Richtung	Ohne Richtung	Mit Richtung	Ohne Richtung	Mit Richtung	Ohne Richtung
CPU 1211C	DC/DC/DC	2	4	3	4	4	4
	AC/DC/RLS	0	0	1	2	2	4
	DC/DC/RLS	0	0	1	2	2	4
CPU 1212C	DC/DC/DC	3	4	3	4	4	4
	AC/DC/RLS	0	0	1	2	2	4
	DC/DC/RLS	0	0	1	2	2	4
CPU 1214C	DC/DC/DC	4	4	4	4	4	4
	AC/DC/RLS	0	0	1	2	2	4
	DC/DC/RLS	0	0	1	2	2	4
CPU 1215C	DC/DC/DC	4	4	4	4	4	4
	AC/DC/RLS	0	0	1	2	2	4
	DC/DC/RLS	0	0	1	2	2	4
CPU 1217C	DC/DC/DC	4	4	4	4	4	4

Hinweis

Maximal sind vier Impulsgeneratoren möglich.

Unabhängig davon, ob Sie integrierte E/A, SB-E/A oder eine Kombination aus beiden verwenden, können Sie maximal vier Impulsgeneratoren definieren.

Tabelle 10- 36 CPU-Ausgang: Höchstfrequenz

CPU	CPU-Ausgangskanal:	Impuls- und Richtungsausgang	A/B, vorwärts/rückwärts und Impuls/Richtung
1211C	Aa.0 bis Aa.3	100 kHz	100 kHz
1212C	Aa.0 bis Aa.3	100 kHz	100 kHz
	Aa.4, Aa.5	20 kHz	20 kHz
1214C und 1215C	Aa.0 bis Aa.3	100 kHz	100 kHz
	Aa.4 bis Ab.1	20 kHz	20 kHz
1217C	DAa.0 bis DAa.3 (.0+, .0- bis .3+, .3-)	1 MHz	1 MHz
	DAa.4 bis DAb.1	100 kHz	100 kHz

Tabelle 10- 37 Signalboard-Ausgang: Höchstfrequenz (optionales Board)

Signalboard (SB)	SB-Ausgangskanal	Impuls- und Richtungsausgang	A/B, vorwärts/rückwärts und Impuls/Richtung
SB 1222, 200 kHz	DAe.0 bis DAe.3	200 kHz	200 kHz
SB 1223, 200 kHz	DAe.0, DAe.1	200 kHz	200 kHz
SB 1223	DAe.0, DAe.1	20 kHz	20 kHz

Tabelle 10- 38 Grenzfrequenzen von Impulsausgängen

Impulsausgang	Frequenz
Integriert	4 PTO: 2 Hz ≤ f ≤ 1 MHz, 4 PTO: 2 Hz ≤ f ≤ 100 kHz oder jede Kombination dieser Werte für 4 PTOs.[1] [2]
Standard-SB	2 Hz ≤ f ≤ 20 kHz
Schnelle SBs	2 Hz ≤ f ≤ 200 kHz

[1] In der nachstehenden Tabelle finden Sie vier mögliche Kombinationen der Ausgangsgeschwindigkeit bei der CPU 1217C.

[2] In der nachstehenden Tabelle finden Sie vier mögliche Kombinationen der Ausgangsgeschwindigkeit bei der CPU 1211C, CPU 1212C, CPU 1214C oder CPU 1215C.

Beispiel: Konfiguration der Impulsausgangsgeschwindigkeit der CPU 1217C

Hinweis

Mit den integrierten Differentialausgängen kann die CPU 1217C vier Impulsausgänge bis 1 MHz generieren.

Die folgenden Beispiele zeigen vier mögliche Kombinationen von Ausgangsgeschwindigkeiten:

- Beispiel 1: 4 x 1-MHz-PTOs, kein Richtungsausgang
- Beispiel 2: 1 x 1-MHz-, 2 x 100-kHz- und 1 x 20-kHz-PTOs, alle mit Richtungsausgang
- Beispiel 3: 4 x 200-kHz-PTOs, kein Richtungsausgang
- Beispiel 4: 2 x 100-kHz-PTOs und 2 x 200-kHz-PTOs, alle mit Richtungsausgang

P = Impuls D = Richtung		**Integrierte CPU-Ausgänge**										**Ausgänge schnelles SB**				**Ausgänge Standard-SB**	
		1-MHz-Ausgänge (A)				100-kHz-Ausgänge (A)						200-kHz-Ausgänge (A)				20-kHz-Ausgänge (A)	
		0.0+	0.1+	0.2+	0.3+	0.4	0.5	0.6	0.7	1.0	1.1	4.0	4.1	4.2	4.3	4.0	4.1
		0.0-	0.1-	0.2-	0.3-												
Beispiel 1: 4 x 1 MHz (kein Richtungsausgang)	PTO1	P															
	PTO2		P														
	PTO3			P													
	PTO4				P												
Beispiel 2: 1 x 1 MHz, 2 x 100 und 1 x 20 kHz (alle mit Richtungsausgang)	PTO1	P	D														
	PTO2					P	D										
	PTO3							P	D								
	PTO4															P	D
Beispiel 3: 4 x 200 kHz (kein Richtungsausgang)	PTO1											P					
	PTO2												P				
	PTO3													P			
	PTO4														P		

P = Impuls D = Richtung		Integrierte CPU-Ausgänge										Ausgänge schnelles SB				Ausgänge Standard-SB	
Beispiel 4: 2 x 100 kHz, 2 x 200 kHz (alle mit Richtungsausgang)	PTO1					P	D										
	PTO2							P	D								
	PTO3											P	D				
	PTO4													P	D		

Beispiel: Konfiguration der Impulsausgangsgeschwindigkeit der CPU 1211C, CPU 1212C, CPU 1214C und CPU 1215C

Die folgenden Beispiele zeigen vier mögliche Kombinationen von Ausgangsgeschwindigkeiten:

- Beispiel 1: 4 x 100-kHz-PTOs, kein Richtungsausgang
- Beispiel 2: 2 x 100-kHz-PTOs und 2 x 20-kHz-PTOs, alle mit Richtungsausgang
- Beispiel 3: 4 x 200-kHz-PTOs, kein Richtungsausgang
- Beispiel 4: 2 x 100-kHz-PTOs und 2 x 200-kHz-PTOs, alle mit Richtungsausgang

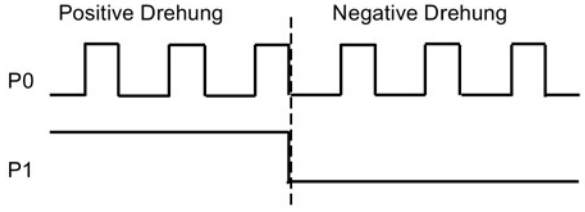
P = Impuls D = Richtung		Integrierte CPU-Ausgänge										Ausgänge schnelles SB				Ausgänge Standard-SB	
		100-kHz-Ausgänge (A)				20-kHz-Ausgänge (A)						200-kHz-Ausgänge (A)				20-kHz-Ausgänge (A)	
		0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	1.0	1.1	4.0	4.1	4.2	4.3	4.0	4.1
		CPU 1211C															
		CPU 1212C				CPU 1212C											
		CPU 1214C				CPU 1214C		CPU 1214C									
		CPU 1215C				CPU 1215C		CPU 1215C									
Beispiel 1: 4 x 100 kHz (kein Richtungsausgang)	PTO1	P															
	PTO2		P														
	PTO3			P													
	PTO4				P												
Beispiel 2: 2 x 100 kHz,	PTO1	P	D														
	PTO2			P	D												
	PTO3					P	D										

P = Impuls D = Richtung		Integrierte CPU-Ausgänge										Ausgänge schnelles SB				Ausgänge Standard-SB	
2 x 20 kHz (alle mit Richtungsausgang)	PTO4							P	D								
Beispiel 3: 4 x 200 kHz (kein Richtungsausgang)	PTO1											P					
	PTO2												P				
	PTO3													P			
	PTO4														P		
Beisp. 4: 2 x 100 kHz, 2 x 200 kHz (alle mit Richtungsausgang)	PTO1	P	D														
	PTO2			P	D												
	PTO3											P	D				
	PTO4													P	D		

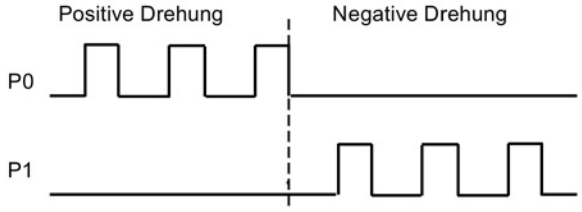
10.3.1 Phasenlage

Die Phasenlage-Schnittstelle zum Schrittmotor/Servoantrieb bietet die folgenden vier Optionen :

- PTO (Impuls A und Richtung B): Wenn Sie eine PTO-Option (Impuls A und Richtung B) auswählen, dann steuert ein Ausgang (P0) die Impulse und ein Ausgang (P1) steuert die Richtung. P1 ist high (aktiv), wenn Impulse in positiver Richtung erzeugt werden. P1 ist low (inaktiv), wenn Impulse in negativer Richtung erzeugt werden:

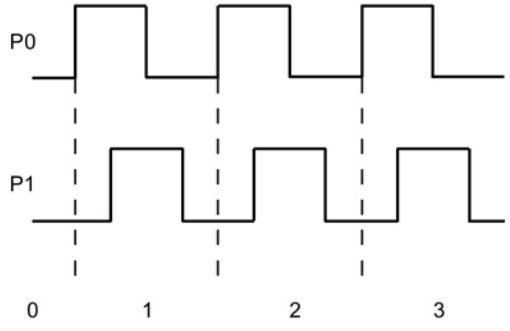
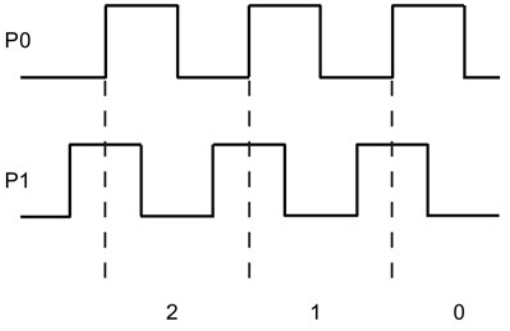


- PTO (Vorwärtszählen A und Rückwärtszählen B): Wenn Sie eine PTO-Option (Vorwärtszählen A und Rückwärtszählen B) auswählen, dann gibt ein Ausgang (P0) Impulse für positive Richtungen und ein anderer Ausgang (P1) Impulse für negative Richtungen aus.



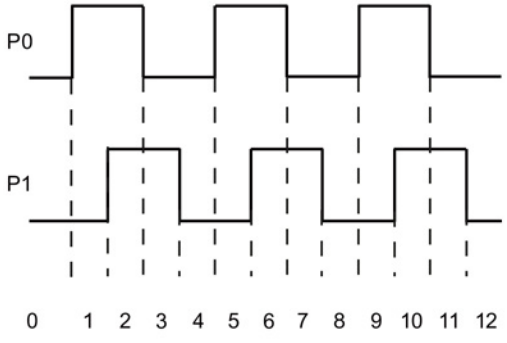
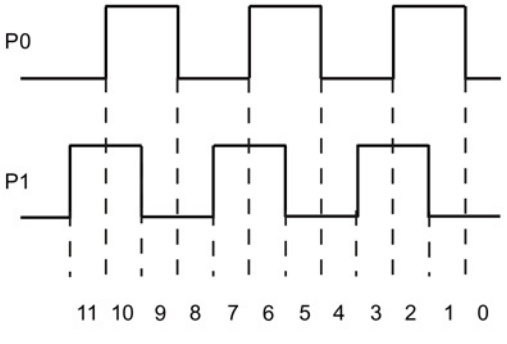
- PTO (A/B phasenverschoben): Wenn Sie eine PTO-Option (A/B phasenverschoben) auswählen, dann geben beide Ausgänge Impulse mit der angegebenen Geschwindigkeit, doch um 90 Grad phasenverschoben aus. Hierbei handelt es sich um eine 1X-Konfiguration, bei der ein Impuls die Zeitdauer zwischen positiven Übergängen von P0 aufweist. In diesem Fall wird die Richtung anhand des Ausgangs ermittelt, der zuerst von 0 nach 1 wechselt. Bei positiver Richtung geht P0 P1 voraus. Bei negativer Richtung geht P1 P0 voraus.

  Die Anzahl der erzeugten Impulse basiert auf der Anzahl der 0-nach-1-Übergänge von Phase A. Das Phasenverhältnis legt die Richtung der Bewegung fest:

PTO (A/B phasenverschoben)	
Phase A geht Phase B voraus (positive Bewegung)	Phase A folgt Phase B nach (negative Bewegung)
	
Anzahl Impulse	Anzahl Impulse

- PTO (A/B phasenverschoben - vierfach): Wenn Sie eine PTO-Option (A/B phasenverschoben - vierfach) auswählen, dann geben beide Ausgänge Impulse mit der angegebenen Geschwindigkeit, aber um 90 Grad phasenverschoben aus. Bei der vierfachen Option handelt es sich um eine 4X-Konfiguration, bei der ein Impuls dem Übergang aller Ausgänge entspricht (positiv und negativ). In diesem Fall wird die Richtung anhand des Ausgangs ermittelt, der zuerst von 0 nach 1 wechselt. Bei positiver Richtung geht P0 P1 voraus. Bei negativer Richtung geht P1 P0 voraus.

  Die vierfache Option basiert auf den positiven und negativen Übergängen sowohl von Phase A und Phase B. Die Anzahl der Übergänge wird von Ihnen konfiguriert. Das Phasenverhältnis (A geht B voraus oder B geht A voraus) legt die Richtung der Bewegung fest.

PTO (A/B phasenverschoben - vierfach)	
Phase A geht Phase B voraus (positive Bewegung)	Phase A folgt Phase B nach (negative Bewegung)
	
Anzahl Impulse	Anzahl Impulse

- PTO (Impuls und Richtung (Richtung abgewählt)): Wenn Sie den Richtungsausgang in einem PTO deaktivieren (Impuls und Richtung (Richtung abgewählt)), dann steuert der Ausgang (P0) die Impulse. Ausgang P1 wird nicht verwendet und steht für andere Zwecke im Programm zur Verfügung. In dieser Betriebsart werden von der CPU nur positive Bewegungssteuerungsbefehle akzeptiert. Wenn Sie diese Betriebsart wählen, erlaubt die Bewegungssteuerung keine unzulässigen negativen Konfigurationen. Sie können einen Ausgang einsparen, wenn Ihre Bewegungssteuerungsanwendung nur eine Richtung aufweist. Die einphasige Option (ein Ausgang) wird in der folgenden Abbildung gezeigt (dabei wird von positiver Polarität ausgegangen):

10.3.2 Konfigurieren eines Impulsgenerators

1. Hinzufügen eines Technologieobjekts:
   - Erweitern Sie in der Projektnavigation den Knoten "Technologieobjekte" und wählen Sie "Neues Objekt hinzufügen" aus.
   - Wählen Sie das Symbol "Achse" (benennen Sie es ggf. um) und klicken Sie auf "OK", um den Konfigurationseditor für das Achsenobjekt zu öffnen.
   - Rufen Sie die Eigenschaften "PTO für Achssteuerung auswählen" unter "Grundparameter" auf und wählen Sie den gewünschten Impuls aus.

     Hinweis

     Wenn PTO zuvor noch nicht in den CPU-Eigenschaften konfiguriert wurde, ist PTO so konfiguriert, dass einer der integrierten Ausgänge verwendet wird.

     Wenn Sie ein Ausgangs-Signalboard verwenden, wählen Sie die Schaltfläche "Gerätekonfiguration", um die CPU-Eigenschaften aufzurufen. In den "Impulsoptionen" können Sie unter "Parametrierung" die Ausgangsquelle für einen Signalboard-Ausgang konfigurieren.

   - Konfigurieren Sie die restlichen Grund- und erweiterten Parameter.
2. Programmieren Ihrer Anwendung: Fügen Sie die Anweisung MC_Power in einen Codebaustein ein.
   - Für den Achseneingang wählen Sie das Technologieobjekt "Achse", das Sie angelegt und konfiguriert haben.
   - Wird der Eingang Enable auf WAHR gesetzt, können die anderen Bewegungssteuerungsanweisungen funktionieren.
   - Wird der Eingang Enable auf FALSCH gesetzt, werden die anderen Bewegungssteuerungsanweisungen storniert.

     Hinweis

     Fügen Sie nur eine Anweisung MC_Power pro Achse ein.

3. Fügen Sie die anderen Bewegungssteuerungsanweisungen ein, um die erforderliche Bewegung zu erzeugen.

Hinweis

Konfigurieren eines Impulsgenerators für Signalboard-Ausgänge: Wählen Sie die Eigenschaften "Impulsgeneratoren (PTO/PWM)" für eine CPU (in der Gerätekonfiguration) und aktivieren Sie einen Impulsgenerator. Für die S7-1200 CPU V1.0, V2.0, V2.1 und V2.2 sind zwei Impulsgeneratoren verfügbar, während bei der S7-1200 CPU V3.0 und V4.0 vier Impulsgeneratoren zur Verfügung stehen. In demselben Konfigurationsbereich wählen Sie unter "Impulsoptionen" für den Impulsgenerator die Verwendung: "PTO" aus.

Hinweis

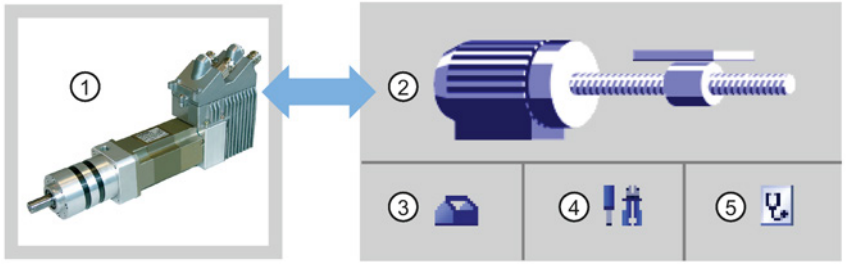
Die CPU berechnet Bewegungssteuerungsaufgaben in "Scheiben" bzw. Segmenten von 10 ms. Wenn eine Scheibe ausgeführt wird, wartet die nächste Scheibe in der Warteschlange auf die Ausführung. Wenn Sie die Bewegungssteuerungsaufgabe einer Achse unterbrechen (indem Sie eine andere neue Bewegungssteuerungsaufgabe für die Achse ausführen), kann die neue Bewegungssteuerungsaufgabe maximal 20 ms lang nicht ausgeführt werden (die Restdauer der aktuellen Scheibe plus die Scheibe in der Warteschlange).

10.3.3 Ungeregelte Bewegungssteuerung

10.3.3.1 Achse konfigurieren

Sie verbinden die ungeregelte Achse mit dem PLC und dem Antrieb über eine Impulsfolge (PTO).

STEP 7 bietet die Konfigurationswerkzeuge, Inbetriebnahmewerkzeuge und Diagnosewerkzeuge für das Technologieobjekt "Achse".



① Antrieb
② Technologieobjekt
③ Konfiguration
④ Inbetriebnahme
⑤ Diagnose

Hinweis

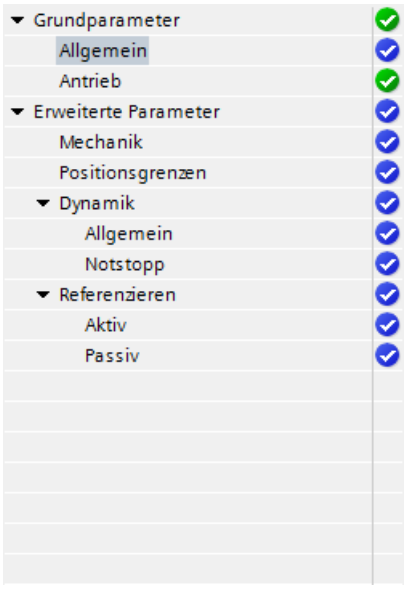
Bei CPUs mit einer Firmwareversion bis V2.2 benötigt die PTO die interne Funktionalität eines schnellen Zählers (HSC). Das bedeutet, dass der jeweilige HSC nicht anderweitig verwendet werden kann.

Die Zuordnung zwischen PTO und HSC ist fest vorgegeben. Wenn PTO1 aktiviert wird, erfolgt die Verbindung mit HSC1. Wenn PTO2 aktiviert wird, erfolgt die Verbindung mit HSC2. Sie können den aktuellen Wert (z. B. in ID1000) nicht überwachen, wenn Impulse auftreten.

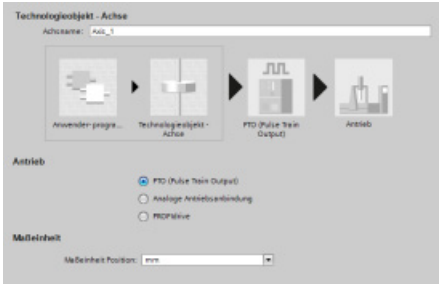
CPUs ab S7-1200 V3.0 haben diese Einschränkung nicht. Alle HSCs bleiben, wenn in diesen CPUs Impulsausgänge konfiguriert sind, für die Verwendung im Programm verfügbar.

Tabelle 10- 39 STEP 7-Werkzeuge für die Bewegungssteuerung

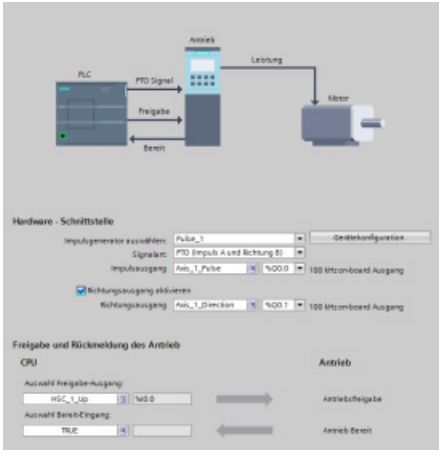
Werkzeug	Beschreibung
Konfiguration	Konfiguriert die folgenden Eigenschaften des Technologieobjekts "Achse": • Auswahl des gewünschten PTO und Konfiguration der Antriebsschnittstelle • Eigenschaften der Mechanik und des Übertragungsverhältnisses des Antriebs (bzw. der Maschine oder Anlage) • Eigenschaften der Positionsgrenzwerte, Dynamik und Referenzpunktfahrt Speichern Sie die Konfiguration im Datenbaustein des Technologieobjekts.
Inbetriebnahme	Testet die Funktion Ihrer Achse, ohne dass ein Anwenderprogramm erstellt werden muss. Wenn das Werkzeug gestartet wird, wird das Steuerpanel angezeigt. Die folgenden Befehle sind im Steuerpanel verfügbar: • Achse freigeben und deaktivieren • Achse im Tippbetrieb bewegen • Achse absolut und relativ bewegen • Referenzpunktfahrt der Achse durchführen • Fehler quittieren Die Geschwindigkeit und die Beschleunigung/Verzögerung kann für die Bewegungssteuerungsbefehle angegeben werden. Das Steuerpanel zeigt auch den aktuellen Achsenzustand.
Diagnose	Überwachung des aktuellen Zustands und der Fehlerinformationen für die Achse und den Antrieb.



Im Auswahlverzeichnis für die PTO-Achse sind die Konfigurationsmenüs für Geber, Modulo, Positionsüberwachung und Regelung nicht enthalten.



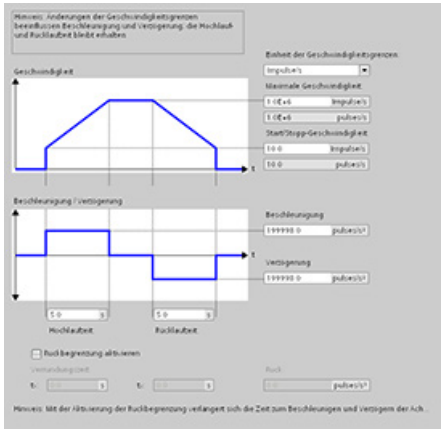
Nachdem Sie das Technologieobjekt für die Achse angelegt haben, konfigurieren Sie die Achse, indem Sie die grundlegenden Parameter definieren, z. B. die PTO und die Konfiguration der Antriebsschnittstelle. Sie konfigurieren auch die anderen Eigenschaften der Achse wie Positionsgrenzwerte, Dynamik und Referenzpunktfahrt.



Hinweis

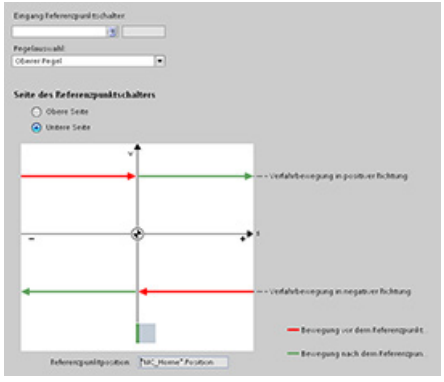
Sie müssen möglicherweise die Werte der Eingangsparameter der Bewegungssteuerungsanweisungen im Anwenderprogramm an die neue Einheit anpassen.

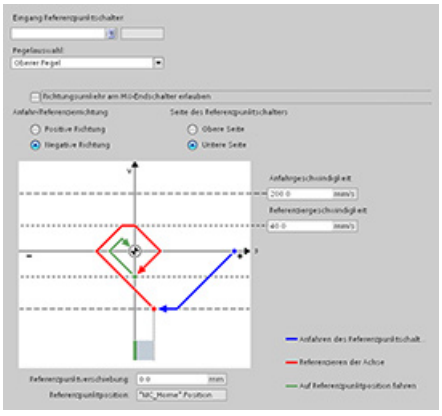
Konfigurieren Sie die Eigenschaften für die Antriebssignale, Antriebsmechanik und Positionsüberwachung (Hardware- und Software-Endschalter).



Sie konfigurieren die Bewegungssteuerungsdynamik und das Verhalten des Not-Aus-Befehls.

Ferner konfigurieren Sie das Verhalten der Referenzpunktfahrt (passiv und aktiv).





Im Steuerpanel "Inbetriebnahme" können Sie die Funktionalität unabhängig von Ihrem Anwenderprogramm testen.

Klicken Sie auf das Symbol "Anlauf", um die Achse in Betrieb zu nehmen.

Das Steuerpanel zeigt den aktuellen Zustand der Achse. Sie können nicht nur die Achse aktivieren und deaktivieren, Sie können auch die Positionierung der Achse prüfen (sowohl absolut als auch relativ) und Sie können Geschwindigkeit, Beschleunigung und Verzögerung vorgeben. Weiterhin können Sie die Referenzpunktfahrt und den Tippbetrieb testen. Im Steuerpanel können Sie auch Fehler quittieren.

10.3.3.2 Inbetriebnahme

Diagnosefunktion "Status- und Fehlerbits"

Mit der Diagnosefunktion "Status- und Fehlerbits" können Sie die wichtigsten Status- und Fehlermeldungen der Achse überwachen. Die Diagnosefunktionsanzeige ist im Online-Modus bei aktiver Achse in der Betriebsart "Manuelle Steuerung" und "Automatiksteuerung" verfügbar.

Tabelle 10- 40 Status der Achse

Status	Beschreibung
Freigegeben	Die Achse ist freigegeben und bereit, über Bewegungssteuerungsaufgaben gesteuert zu werden. (Variable des Technologieobjekts: <Achsenname>.StatusBits.Enable)
Referenzpunktfahrt durchgeführt	Die Achse ist am Referenzpunkt und kann absolute Positionierungsaufgaben der Bewegungssteuerungsanweisung MC_MoveAbsolute durchführen. Die Achse muss sich für die relative Referenzpunktfahrt nicht am Referenzpunkt befinden. Besondere Situationen: • Während der aktiven Referenzpunktfahrt ist der Status FALSCH. • Wenn für eine Achse, die sich am Referenzpunkt befindet, eine passive Referenzpunktfahrt durchgeführt wird, wird der Status während der passiven Referenzpunktfahrt auf WAHR gesetzt. (Variable des Technologieobjekts: <Achsenname>.StatusBits.HomingDone)
Fehler	Im Technologieobjekt "Achse" ist ein Fehler aufgetreten. Weitere Informationen zum Fehler sind in der Automatiksteuerung an den Parametern ErrorID und ErrorInfo der Bewegungssteuerungsanweisungen verfügbar. Im Handbetrieb zeigt das Feld "Letzter Fehler" im Steuerpanel ausführliche Informationen zur Fehlerursache an. (Variable des Technologieobjekts: <Achsenname>.StatusBits.Error)
Steuerpanel aktiv	Der Modus "Manuelle Steuerung" wurde im Steuerpanel aktiviert. Das Steuerpanel hat die Steuerpriorität über das Technologieobjekt "Achse". Die Achse kann nicht über das Anwenderprogramm gesteuert werden. (Variable des Technologieobjekts: <Achsenname>.StatusBits.ControlPanelActive)

Tabelle 10- 41 Antriebsstatus

Status	Beschreibung
Antrieb bereit	Der Antrieb ist betriebsbereit. (Variable des Technologieobjekts: <Achsenname>.StatusBits.DriveReady)
Fehler	Der Antrieb hat nach Ausfall des Signals "Bereit" einen Fehler gemeldet. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.DriveFault)

Tabelle 10- 42 Status der Achsbewegung

Status	Beschreibung
Stillstand	Die Achse ist im Stillstand. (Variable des Technologieobjekts: <Achsenname>.StatusBits.StandStill)
Beschleunigung	Die Achse beschleunigt. (Variable des Technologieobjekts: <Achsenname>.StatusBits.Acceleration)
Konstante Geschwindigkeit	Die Achse fährt mit konstanter Geschwindigkeit. (Variable des Technologieobjekts: <Achsenname>.StatusBits.ConstantVelocity)
Verzögerung	Die Achse verzögert (verlangsamt die Fahrt). (Variable des Technologieobjekts: <Achsenname>.StatusBits.Deceleration)

Tabelle 10- 43 Status des Bewegungsmodus

Status	Beschreibung
Positionierung	Die Achse führt eine Positionierungsaufgabe der Bewegungssteuerungsanweisung "MC_MoveAbsolute" oder "MC_MoveRelative" oder des Steuerpanels aus. (Variable des Technologieobjekts: <Achsenname>.StatusBits.PositioningCommand)
Geschwindigkeitsbefehl	Die Achse führt eine Aufgabe mit festgelegter Geschwindigkeit der Bewegungssteuerungsanweisung "MC_MoveVelocity" oder "MC_MoveJog" oder des Steuerpanels aus. (Variable des Technologieobjekts: <Achsenname>.StatusBits.SpeedCommand)
Referenzpunktfahrt	Die Achse führt eine Referenzpunktfahrtaufgabe der Bewegungssteuerungsanweisung "MC_Home" oder des Steuerpanels aus. (Variable des Technologieobjekts: <Achsenname>.StatusBits.Homing)

Tabelle 10- 44 Fehlerbits

Fehler	Beschreibung
Min. Softwaregrenze erreicht	Der untere Software-Endschalter wurde erreicht. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.SwLimitMinReached)
Min. Softwaregrenze überschritten	Der untere Software-Endschalter wurde überschritten. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.SwLimitMinExceeded)
Max. Softwaregrenze erreicht	Der obere Software-Endschalter wurde erreicht. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.SwLimitMaxReached)
Max. Softwaregrenze überschritten	Der obere Software-Endschalter wurde überschritten. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.SwLimitMaxExceeded)
Negative Hardwaregrenze	Der untere Hardware-Endschalter wurde angefahren. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.HwLimitMin)
Positive Hardwaregrenze	Der obere Hardware-Endschalter wurde angefahren. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.HwLimitMax)
PTO wird bereits verwendet	Eine zweite Achse verwendet die gleiche PTO und ist mit "MC_Power" freigegeben. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.HwUsed)

Fehler	Beschreibung
Konfigurationsfehler	Das Technologieobjekt "Achse" wurde falsch konfiguriert oder änderbare Konfigurationsdaten wurden während der Laufzeit des Anwenderprogramms fehlerhaft geändert. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.ConfigFault)
Allgemeiner Fehler	Ein interner Fehler ist aufgetreten. (Variable des Technologieobjekts: <Achsenname>.ErrorBits.SystemFault)

Diagnosefunktion "Bewegungsstatus"

Mit der Diagnosefunktion "Bewegungsstatus" überwachen Sie den Bewegungsstatus der Achse. Die Diagnosefunktionsanzeige ist im Online-Modus bei aktiver Achse in der Betriebsart "Manuelle Steuerung" und "Automatiksteuerung" verfügbar.

Tabelle 10- 45 Bewegungsstatus

Status	Beschreibung
Zielposition	Das Feld "Zielposition" zeigt die aktuelle Zielposition einer aktiven Positionierungsaufgabe der Bewegungssteuerungsanweisung "MC_MoveAbsolute" oder "MC_MoveRelative" oder des Steuerpanels an. Der Wert der "Zielposition" ist nur während der Ausführung einer Positionierungsaufgabe gültig. (Variable des Technologieobjekts: <Achsenname>.MotionStatus.TargetPosition)
Aktuelle Position	Das Feld "Aktuelle Position" zeigt die aktuelle Position der Achse an. Wenn sich die Achse nicht am Referenzpunkt befindet, zeigt der Wert den Positionswert relativ zur Freigabeposition der Achse an. (Variable des Technologieobjekts: <Achsenname>.MotionStatus.Position)
Aktuelle Geschwindigkeit	Das Feld "Aktuelle Geschwindigkeit" zeigt die aktuelle Geschwindigkeit der Achse an. (Variable des Technologieobjekts: <Achsenname>.MotionStatus.Velocity)

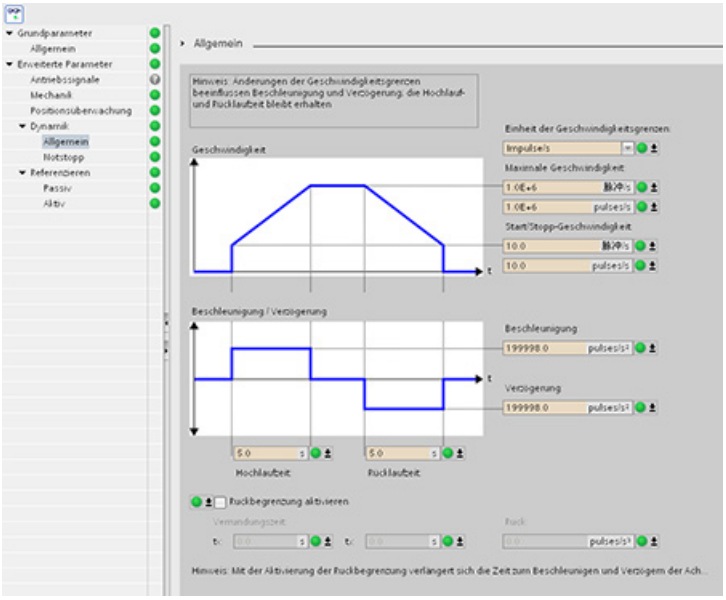
Tabelle 10- 46 Dynamikgrenzen

Dynamikgrenze	Beschreibung
Geschwindigkeit	Das Feld "Geschwindigkeit" zeigt die konfigurierte maximale Geschwindigkeit der Achse an. (Variable des Technologieobjekts: <Achsenname>.Config.DynamicLimits.MaxVelocity)
Beschleunigung	Das Feld "Beschleunigung" zeigt die aktuell konfigurierte Beschleunigung der Achse an. (Variable des Technologieobjekts: <Achsenname>.Config.DynamicDefaults.Acceleration)
Verzögerung	Das Feld "Verzögerung" zeigt die aktuell konfigurierte Verzögerung der Achse an. (Variable des Technologieobjekts: <Achsenname>.Config.DynamicDefaults.Deceleration)

Startwertsteuerung bei der Bewegungssteuerung

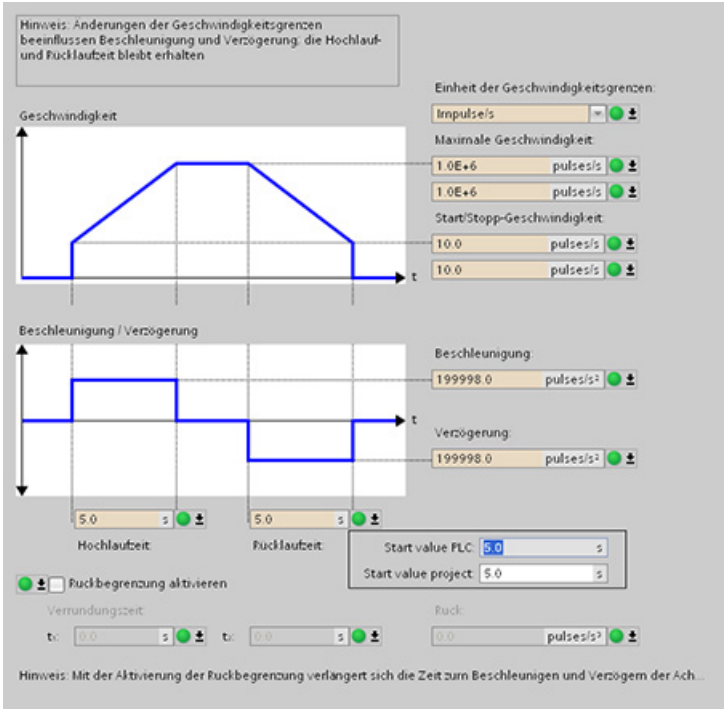
Sie können die Istwerte der Bewegungssteuerungs-Konfigurationsparameter so ändern, dass das Verhalten des Prozesses im Online-Modus optimiert werden kann.

Öffnen Sie die Technologieobjekte für die Bewegungssteuerung und deren Konfigurationsobjekt. Um die Startwertsteuerung aufzurufen, klicken Sie auf das Brillensymbol in der linken oberen Ecke des Dialogfelds:



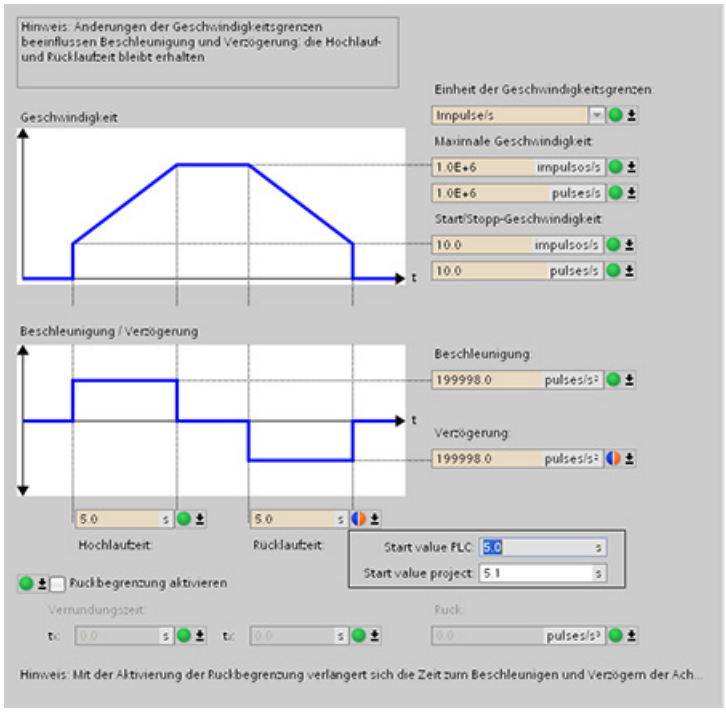
Nun können Sie die Werte beliebiger Konfigurationsparameter Ihrer Bewegungssteuerung wie in nachstehender Abbildung gezeigt ändern.

Sie können für jeden Parameter den Istwert mit dem Startwert des Projekts (offline) und dem Startwert des PLCs (online) vergleichen. Dies ist erforderlich, um die Online-/Offline-Unterschiede des Technologieobjekt-Datenbausteins (TO-DB) zu vergleichen und die Werte zu kennen, die beim nächsten Wechsel von STOP nach START des PLCs als aktuelle Werte verwendet werden. Außerdem erhalten Sie durch ein Vergleichssymbol einen visuellen Hinweis, um Online-/Offline-Unterschiede schnell zu erkennen:



Die Abbildung oben zeigt den Bildschirm der Bewegungssteuerungsparameter mit Vergleichssymbolen. Die Symbole weisen darauf hin, welche Werte sich zwischen Online- und Offline-Projekt unterscheiden. Ein grünes Symbol kennzeichnet die Werte, die gleich sind. Ein blau-orangefarbenes Symbol zeigt an, dass die Werte voneinander abweichen.

Außerdem können Sie auf die Parameterschaltfläche mit dem Abwärtspfeil klicken, um ein kleines Fenster anzuzeigen, in dem für jeden Parameter der Startwert des Projekts (offline) und der Startwert des PLCs (online) angezeigt werden.

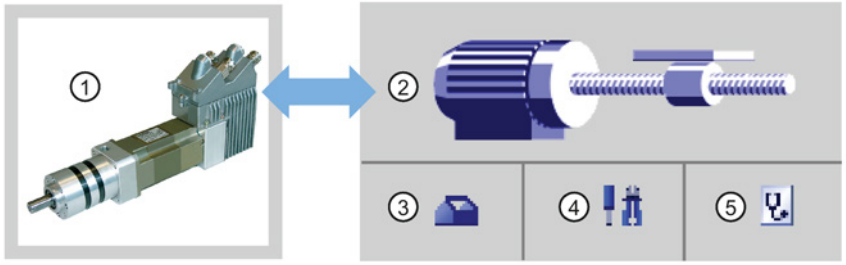


10.3.4 Geregelte Bewegungssteuerung

10.3.4.1 Achse konfigurieren

Sie verbinden die geregelte Achse mit dem PLC und dem Antrieb über den Analogantrieb oder PROFIdrive. Die geregelte Achse benötigt ebenfalls einen Geber.

STEP 7 bietet die Konfigurationswerkzeuge, Inbetriebnahmewerkzeuge und Diagnosewerkzeuge für das Technologieobjekt "Achse".



① Antrieb
② Technologieobjekt
③ Konfiguration
④ Inbetriebnahme
⑤ Diagnose

Tabelle 10- 47 STEP 7-Werkzeuge für die geregelte Bewegungssteuerung

Werkzeug	Beschreibung
Konfiguration	Konfiguriert die folgenden Eigenschaften des Technologieobjekts "Achse": • Auswahl der gewünschten Analogantriebsverbindung oder von PROFIdrive und Konfiguration von Antriebs- und Geberschnittstelle • Eigenschaften der Mechanik und des Übertragungsverhältnisses von Antrieb und Geber (bzw. von Maschine oder Anlage) • Eigenschaften der Positionsgrenzwerte, Dynamik und Referenzpunktfahrt Speichern Sie die Konfiguration im Datenbaustein des Technologieobjekts.
Inbetriebnahme	Testet die Funktion Ihrer Achse, ohne dass ein Anwenderprogramm erstellt werden muss. Wenn das Werkzeug gestartet wird, wird das Steuerpanel angezeigt. Die folgenden Befehle sind im Steuerpanel verfügbar: • Achse freigeben und deaktivieren • Achse im Tippbetrieb bewegen • Achse absolut und relativ bewegen • Referenzpunktfahrt der Achse durchführen • Fehler quittieren Die Geschwindigkeit und die Beschleunigung/Verzögerung kann für die Bewegungssteuerungsbefehle angegeben werden. Das Steuerpanel zeigt auch den aktuellen Achsenzustand.
Diagnose	Überwachung des aktuellen Zustands und der Fehlerinformationen für die Achse und den Antrieb.

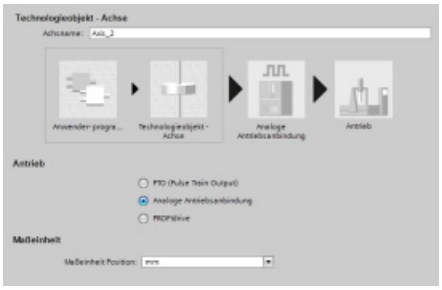
Hinweis

Sie müssen möglicherweise die Werte der Eingangsparameter der Bewegungssteuerungsanweisungen im Anwenderprogramm an die neue Einheit anpassen.

Nachdem Sie das Technologieobjekt für die Achse angelegt haben, konfigurieren Sie die Achse, indem Sie die grundlegenden Parameter definieren, entweder die Analogantriebs- oder die PROFIdrive-Verbindung und die Konfiguration von Antrieb und Geber.

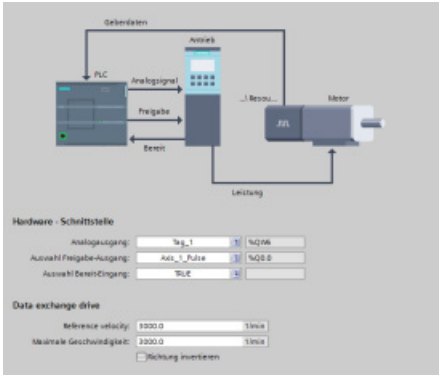
Im Auswahlverzeichnis für die Analogantriebs- oder PROFIdrive-Verbindung sind die Konfigurationsmenüs für Geber, Modulo, Positionsüberwachung und Regelung enthalten.

Konfiguration der Analogantriebsverbindung



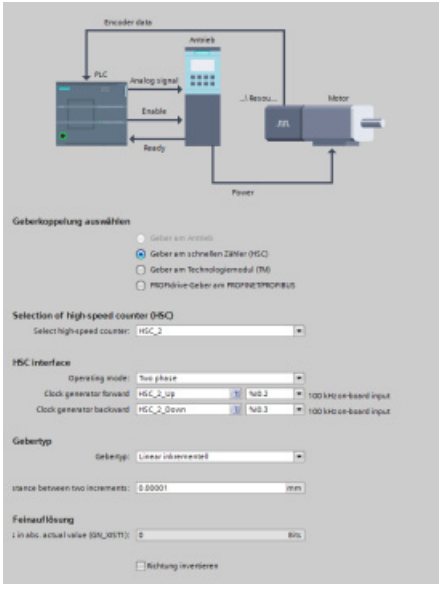
Im Konfigurationsdialog "Allgemein" wählen Sie die folgenden Parameter aus:

- Optionsfeld "Analogantriebsverbindung"
- Maßeinheit



Im Konfigurationsdialog "Antrieb" wählen Sie die folgenden Parameter aus:

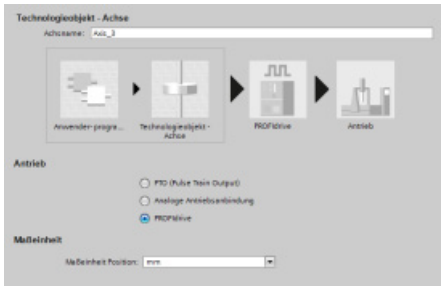
- Hardwareausgänge des Analogantriebs
- Datenaustausch Antriebsgeschwindigkeiten



Im Konfigurationsdialog "Geber" wählen Sie die folgenden Parameter aus:

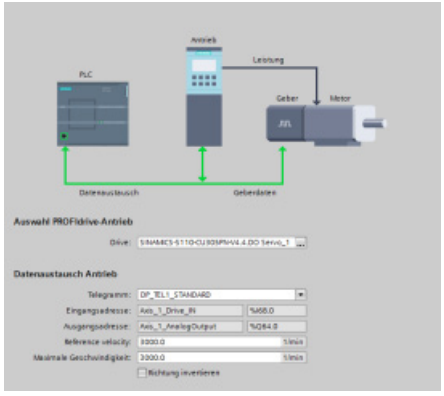
- Geberkopplung Analogantrieb (z. B. schneller Zähler (HSC))
- HSC-Schnittstelle
- Gebertyp
- Feinauflösung

PROFIdrive-Konfiguration



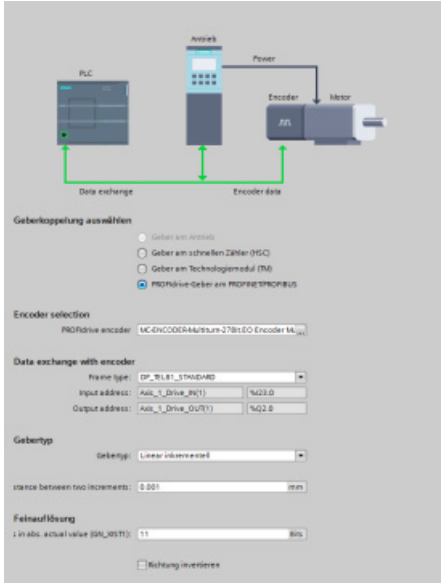
Im Konfigurationsdialog "Allgemein" wählen Sie die folgenden Parameter aus:

- Optionsfeld "PROFIdrive"
- Maßeinheit



Im Konfigurationsdialog "Antrieb" wählen Sie die folgenden Parameter aus:

- PROFIdrive-Antrieb
- Datenaustausch mit dem Antrieb



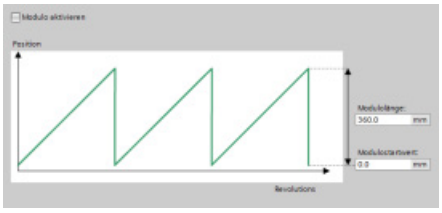
Im Konfigurationsdialog "Geber" wählen Sie die folgenden Parameter aus:

- PROFIdrive-Geberkopplung (z. B. PROFIdrive-Geber auf PROFINET)
- PROFIdrive-Geber
- Datenaustausch mit dem Geber
- Gebertyp
- Feinauflösung

Erweiterte Parameter

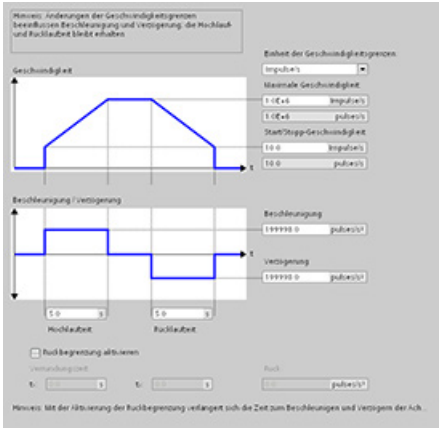
Sie können außerdem die folgenden Eigenschaften der geregelten Achse festlegen:

- Modulo
- Positionsgrenzen
- Dynamik
- Referenzpunktfahrt
- Positionsüberwachung
- Schleppfehler
- Stillstandsignal
- Regelung



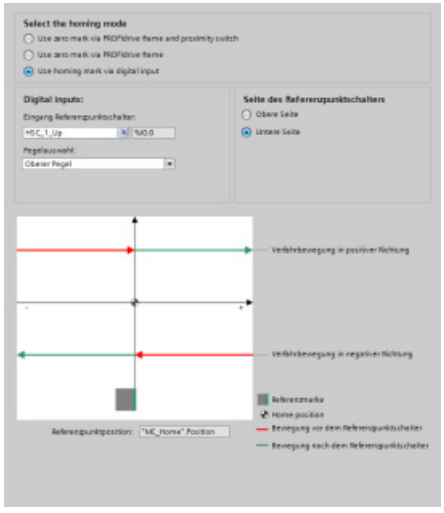
Modulo: Sie können eine "Modulo"-Achse konfigurieren, um die Last in einem zyklischen Bereich zu bewegen, der über einen Startwert bzw. eine Startposition und eine vorgegebene Länge verfügt. Wenn die Position der Last das Ende dieses Bereichs erreicht, wird sie automatisch wieder auf den Startwert gesetzt. Sie aktivieren die Felder "Länge" und "Modulo-Startwert", wenn Sie das Kontrollkästchen "Modul aktivieren" auswählen.

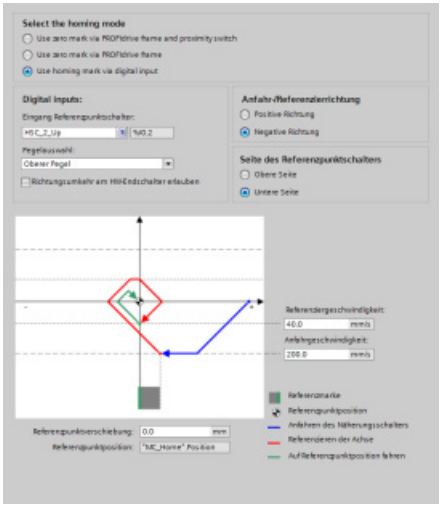
Positionsgrenzen: Sie können die Eigenschaften für die Antriebssignale, Antriebsmechanik und Positionsüberwachung (Hardware- und Software-Endschalter) konfigurieren.

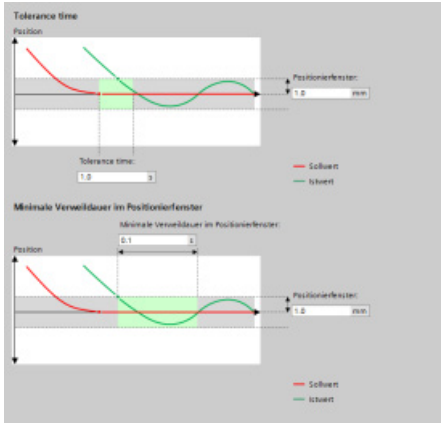


Dynamik: Sie können die Bewegungssteuerungsdynamik und das Verhalten des Not-Aus-Befehls konfigurieren.

Referenzpunktfahrt: Ferner können Sie das Verhalten der Referenzpunktfahrt (passiv und aktiv) konfigurieren.



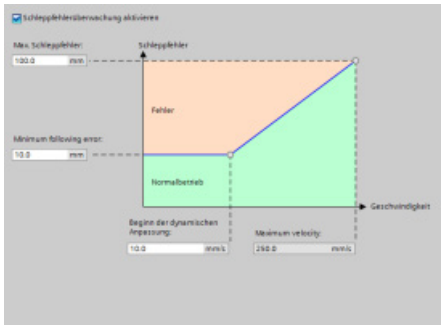




"Positionsüberwachung": Sie können die Toleranzzeit sowie eine minimale Verweilzeit im Positionierfenster konfigurieren.

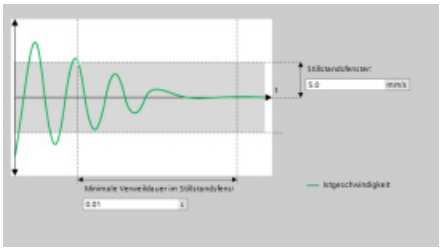
Das System verbindet die folgenden drei Parameter direkt mit dem TO-DB der Achse:

- Positionierfenster
- Toleranzzeit
- Kleinste Verweilzeit im Positionierfenster



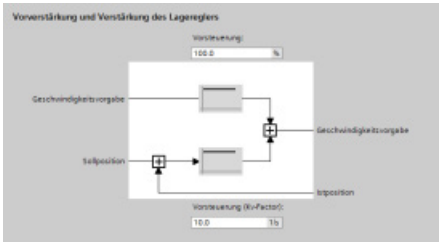
"Schleppfehler": Sie können die Differenz der zulässigen Fehlerdistanz über einen Geschwindigkeitsbereich konfigurieren. Um den Schleppfehler zu aktivieren, wählen Sie das Kontrollkästchen "Schleppfehlerüberwachung aktivieren" aus. Sie können die folgenden Parameter konfigurieren:

- Maximaler Schleppfehler
- Schleppfehler
- Beginn der dynamischen Anpassung
- Maximale Geschwindigkeit



"Stillstandsignal": Sie können die folgenden Parameter konfigurieren:

- Kleinste Verweilzeit im Stillstandsfenster
- Stillstandsfenster.



"Regelung": Sie können die Geschwindigkeitsverstärkung "Vorverstärkung (Kv-Faktor)" konfigurieren.

Im Steuerpanel "Inbetriebnahme" können Sie die Funktionalität unabhängig von Ihrem Anwenderprogramm testen.

Klicken Sie auf das Symbol "Anlauf", um die Achse in Betrieb zu nehmen.

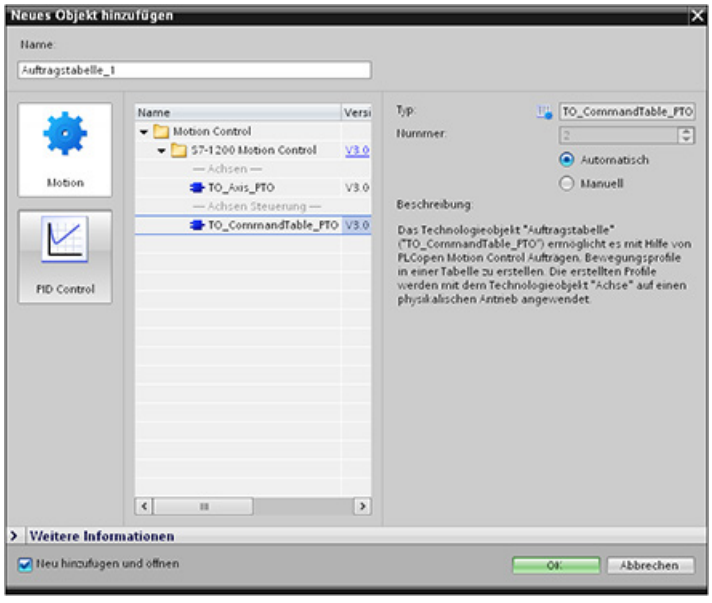
Das Steuerpanel zeigt den aktuellen Zustand der Achse. Sie können nicht nur die Achse aktivieren und deaktivieren, Sie können auch die Positionierung der Achse prüfen (sowohl absolut als auch relativ) und Sie können Geschwindigkeit, Beschleunigung und Verzögerung vorgeben. Weiterhin können Sie die Referenzpunktfahrt und den Tippbetrieb testen. Im Steuerpanel können Sie auch Fehler quittieren.

10.3.5 TO-Befehlstabelle PTO konfigurieren

Mit Hilfe der Technologieobjekte können Sie eine Anweisung MC_CommandTable konfigurieren. Im folgenden Beispiels wird gezeigt, wie dies durchgeführt wird.

Hinzufügen eines Technologieobjekts

1. Erweitern Sie in der Projektnavigation den Knoten "Technologieobjekte" und wählen Sie "Neues Objekt hinzufügen" aus.
2. Wählen Sie das Symbol "Befehlstabelle" (benennen Sie es ggf. um) und klicken Sie auf "OK", um den Konfigurationseditor für das Befehlstabellenobjekt zu öffnen.



Schritte für Ihre Anwendung planen

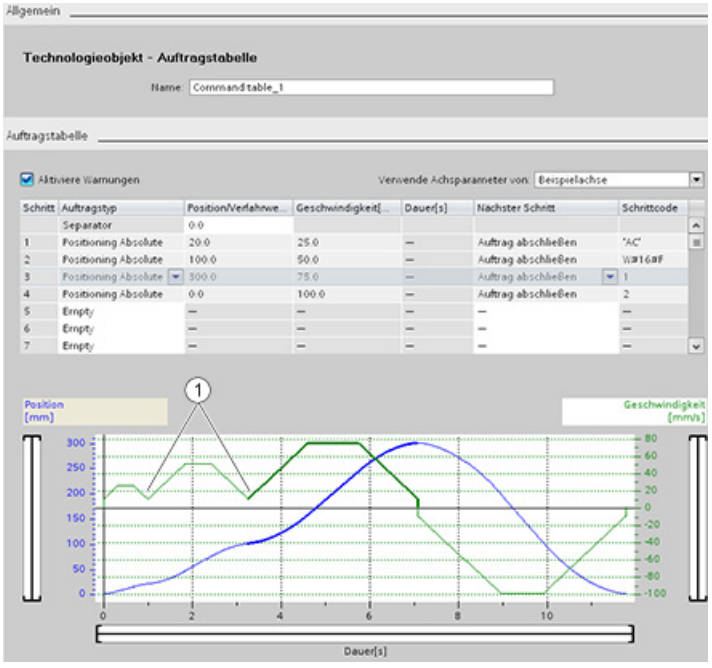
Die gewünschte Bewegungsfolge können Sie im Konfigurationsfenster "Befehlstabelle" erstellen und das Ergebnis in der grafischen Darstellung im Kurvendiagramm prüfen.

Sie können die Befehlstypen auswählen, die für die Verarbeitung der Befehlstabelle verwendet werden sollen. Bis zu 32 Schritte können eingegeben werden. Die Befehle werden der Reihe nach verarbeitet und erzeugen mühelos ein komplexes Bewegungsprofil.

Tabelle 10- 48 MC_CommandTable-Befehlstypen

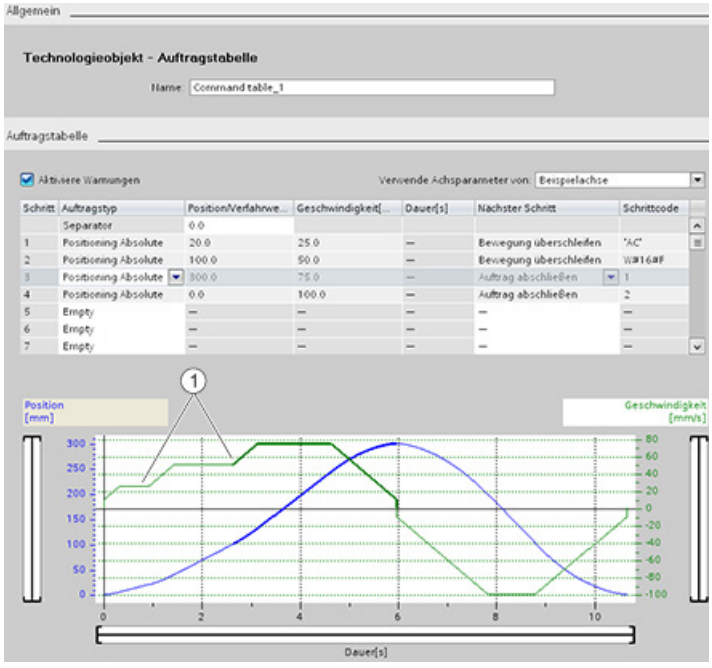
Befehlstyp	Beschreibung
Empty	Dieser Befehl dient als Platzhalter für jeden hinzuzufügenden Befehl. Der leere Eintrag wird bei der Verarbeitung der Befehlstabelle ignoriert.
Halt	Der Befehl hält die Achse an. Hinweis: Der Befehl wird nur nach einem Befehl "Velocity setpoint" durchgeführt.
Positioning Relative	Der Befehl positioniert die Achse basierend auf der Distanz. Er bewegt die Achse um die angegebene Distanz mit der angegebenen Geschwindigkeit.
Positioning Absolute	Der Befehl positioniert die Achse basierend auf der Lage. Er bewegt die Achse mit der vorgegebenen Geschwindigkeit an die angegebene Position.
Velocity setpoint	Der Befehl bewegt die Achse mit der angegebenen Geschwindigkeit.
Wait	Der Befehl wartet, bis der angegebene Zeitraum abgelaufen ist. "Wait" stoppt keine aktive Verfahrbewegung.
Separator	Der Befehl fügt eine Trennlinie ("Separator") oberhalb der ausgewählten Linie ein. Die Trennlinie ermöglicht die Definition mehrerer Profile in einer Befehlstabelle.

In der folgenden Abbildung wird "Command complete" als Übergang zum nächsten Schritt verwendet. Diese Art des Übergangs ermöglicht es Ihrem Gerät, auf die Start-/Stoppgeschwindigkeit zu verzögern und dann zu Beginn des nächsten Schritts erneut zu beschleunigen.



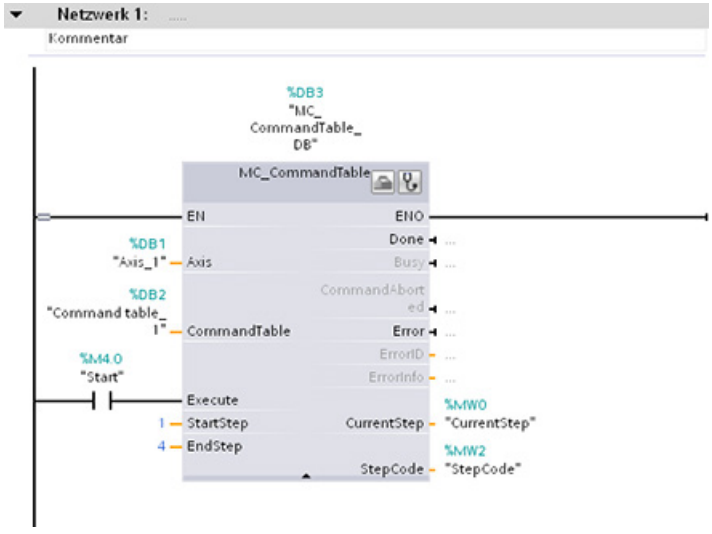
① Die Achse verzögert zwischen den Schritten auf die Start-/Stoppgeschwindigkeit.

In der folgenden Abbildung wird "Blending motion" als Übergang zum nächsten Schritt verwendet. Diese Art des Übergangs ermöglicht es Ihrem Gerät, die Geschwindigkeit bis zum Start des nächsten Schritts beizubehalten, was zu einem glatten Übergang des Geräts von einem Schritt zum nächsten führt. Bei Verwendung von "Blending motion" kann sich die Gesamtdauer für die vollständige Ausführung eines Profils verkürzen. Ohne Blending dauert die Ausführung des Beispiels sieben Sekunden. Mit Blending wird die Ausführungszeit um eine Sekunde auf insgesamt sechs Sekunden reduziert.



① Die Achse bewegt sich ununterbrochen weiter und beschleunigt oder verzögert auf die Geschwindigkeit des nächsten Schritts, wodurch Zeit gespart und mechanischer Verschleiß verringert wird.

Die Funktionsweise Ihrer Befehlstabelle wird, wie im Folgenden gezeigt, von der Anweisung MC_CommandTable gesteuert:



10.3.6 Funktionsweise der Bewegungssteuerung bei der S7-1200

10.3.6.1 Für die Bewegungssteuerung verwendete CPU-Ausgänge

Die CPU bietet vier Impulsgeneratoren. Jeder Impulsgenerator bietet einen Impulsausgang und einen Richtungsausgang zum Steuern eines Schrittmotorantriebs oder eines Servomotorantriebs mit Impulsschnittstelle. Der Impulsausgang bietet den Antrieb mit den für die Motorbewegung erforderlichen Impulsen. Der Richtungsausgang steuert die Verfahrrichtung des Antriebs.

Der PTO-Ausgang erzeugt einen Rechteckausgang mit variabler Frequenz. Die Impulserzeugung wird über Konfigurations- und Ausführungsinformationen gesteuert, die über die H/W-Konfiguration und/oder SFCs/SFBs vorgegeben werden.

Während sich die CPU im Betriebszustand RUN befindet, steuern je nach Auswahl des Anwenders entweder die Werte im Prozessabbild oder die Ausgänge des Impulsgenerators die digitalen Ausgänge. Im Betriebszustand STOP steuert der PTO-Generator die Ausgänge nicht.

Integrierte CPU-Ausgänge und Ausgänge eines Signalboards können als Impuls- und Richtungsausgänge verwendet werden. Sie wählen zwischen integrierten CPU-Ausgängen und Ausgängen des Signalboards während der Gerätekonfiguration der Impulsgeneratoren (PTO/PWM) im Register "Eigenschaften". Nur PTO (Pulse Train Output) gilt für die Bewegungssteuerung.

Die folgende Tabelle zeigt die E/A-Standardbelegungen; die vier Impulsgeneratoren können jedoch für jeden Digitalausgang konfiguriert werden.

Hinweis

Impulsfolgen können nicht von anderen Operationen im Anwenderprogramm verwendet werden.

Wenn Sie die Ausgänge der CPU oder des Signalboards als Impulsgeneratoren (für PWM oder Bewegungssteuerungsanweisungen) konfigurieren, steuern die entsprechenden Ausgangsadressen nicht mehr die Ausgänge. Wenn Ihr Anwenderprogramm einen Wert in einen Ausgang schreibt, der als Impulsgenerator genutzt wird, schreibt die CPU diesen Wert nicht in den physischen Ausgang.

Hinweis

PTO-Richtungsausgänge können für eine anderweitige Nutzung in Ihrem Programm freigegeben werden.

Jeder PTO müssen zwei Ausgänge zugewiesen werden: ein Impulsausgang und ein Richtungsausgang. Es ist möglich, nur den Impulsausgang, aber nicht den Richtungsausgang zu verwenden. Dadurch wird der Richtungsausgang für andere Funktionen im Anwenderprogramm freigegeben. Der Ausgang kann nicht gleichzeitig als PTO-Richtungsausgang und im Anwenderprogramm benutzt werden.

Tabelle 10- 49 Standardadresszuweisungen für die Impuls- und Richtungsausgänge

Verwendung von Ausgängen für die Bewegungssteuerung		
	Impuls	Richtung
PTO1		
Integrierte E/A	A0.0	A0.1
SB-E/A	A4.0	A4.1
PTO2		
Integrierte E/A	A0.2	A0.3
SB-E/A	A4.2 [1]	A4.3 [1]
PTO3		
Integrierte E/A	A0.4[2]	A0.5[2]
SB-E/A	A4.0	A4.1
PTO4		
Integrierte E/A	A0.6[3]	A0.7[3]
SB-E/A	A4.2	A4.3

1 Ausgänge A4.2 und A4.3 stehen nur am SB1222 DA4 zur Verfügung.

2 Die CPU 1211C hat keine Ausgänge A0.4, A0.5, A0.6 und A0.7. Deshalb können diese Ausgänge bei der CPU 1211C nicht verwendet werden.

3 Die CPU 1212C hat keine Ausgänge A0.6 und A0.7. Deshalb können diese Ausgänge bei der CPU 1212C nicht verwendet werden.

4 Diese Tabelle gilt für die PTO-Funktionen der CPU 1211C, CPU 1212C, CPU 1214C, CPU 1215C und CPU 1217C.

Antriebsschnittstelle

Für die Bewegungssteuerung können Sie optional für eine Antriebsschnittstelle die Zustände "Antrieb freigegeben" und "Antrieb bereit" konfigurieren. Wenn Sie die Antriebsschnittstelle verwenden, können der Digitalausgang für "Antrieb freigegeben" und der Digitaleingang für "Antrieb bereit" frei ausgewählt werden.

Hinweis

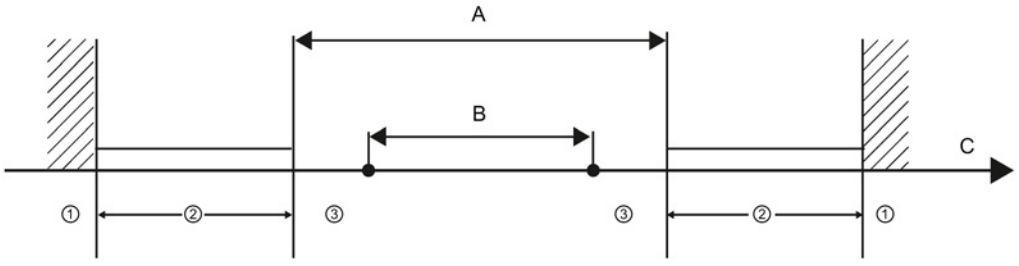
Die Firmware übernimmt die Steuerung über die entsprechenden Impuls- und Richtungsausgänge, wenn die PTO (Pulse Train Output) ausgewählt und einer Achse zugewiesen wurde.

Mit dieser Übernahme der Steuerungsfunktion wird auch die Verbindung zwischen dem Prozessabbild und dem E/A-Ausgang getrennt. Der Anwender hat zwar die Möglichkeit, über das Anwenderprogramm oder die Beobachtungstabelle in das Prozessabbild der Impuls- und Richtungsausgänge zu schreiben, diese Daten werden jedoch nicht an die Ausgänge übertragen. Entsprechend ist es auch nicht möglich, die Ausgänge über das Anwenderprogramm oder die Beobachtungstabelle zu überwachen. Die ausgelesenen Informationen geben lediglich den Wert des Prozessabbilds wieder und entsprechen in keiner Weise dem tatsächlichen Zustand der Ausgänge.

Bei allen anderen CPU-Ausgängen, die nicht permanent von der CPU-Firmware verwendet werden, kann der Zustand der Ausgänge wie üblich über das Prozessabbild gesteuert und überwacht werden.

10.3.6.2 Hardware- und Software-Endschalter für die Bewegungssteuerung

Mit den Hardware- und Software-Endschaltern begrenzen Sie den zulässigen Verfahrbereich und den Arbeitsbereich Ihrer Achse.



①	Mechanischer Stopp	A	Zulässiger Verfahrbereich der Achse
②	Untere und obere Hardware-Grenzwerte	B	Arbeitsbereich der Achse
③	Untere und obere Software-Grenzwerte	C	Entfernung

Hardware- und Software-Endschalter müssen vor der Verwendung in der Konfiguration oder im Anwenderprogramm aktiviert werden. Software-Endschalter sind erst nach der Referenzpunktfahrt der Achse aktiv.

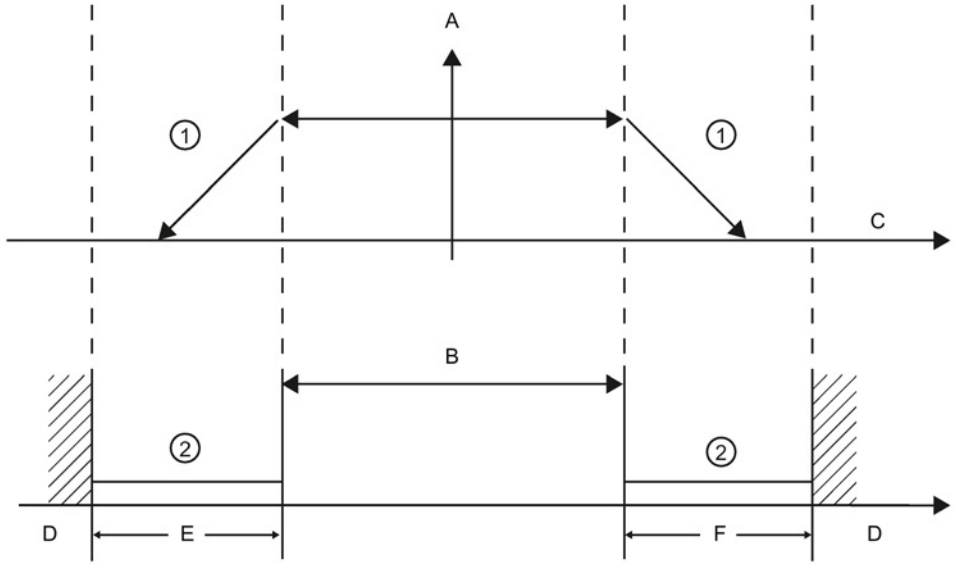
Hardware-Endschalter

Hardware-Endschalter geben den maximalen Verfahrbereich der Achse vor. Hardware-Endschalter sind physikalische Schaltelemente, die an alarmfähige Eingänge der CPU angeschlossen werden müssen. Verwenden Sie nur Hardware-Endschalter, die nach der Anfahrt dauerhaft geschaltet bleiben. Der Schaltzustand kann erst nach der Rückkehr in den zulässigen Verfahrbereich rückgängig gemacht werden.

Tabelle 10- 50 Verfügbare Eingänge für Hardwaregrenzen

Beschreibung	RPS	LIM-	LIM+
Integrierte E/A	E0.0 - E1.5		
SB-E/A	E4.0 - E4.3		

Wenn die Hardware-Endschalter angefahren werden, bremst die Achse mit der konfigurierten Notfallverzögerung bis zum Stillstand. Die angegebene Notfallverzögerung muss ausreichend sein, um die Achse vor dem mechanischen Stopp zuverlässig zu stoppen. Das folgende Diagramm zeigt das Verhalten der Achse nach Anfahrt der Hardware-Endschalter.



① Die Achse bremst mit der konfigurierten Notfallverzögerung bis zum Stillstand.
② Bereich, in dem die Hardware-Endschalter den Zustand "angefahren" signalisieren.
A [Geschwindigkeit]
B Zulässiger Verfahrbereich
C Entfernung
D Mechanischer Stopp
E Unterer Hardware-Endschalter
F Oberer Hardware-Endschalter

⚠WARNUNG

Gefahren beim Ändern der Filterzeit für digitale Eingangskanäle

Wenn die bisherige Einstellung der Filterzeit eines Digitaleingangskanals verändert wird, muss möglicherweise bis zu 20,0 ms lang ein neuer Eingangswert mit Pegel "0" vorhanden sein, damit der Filter vollständig auf neue Eingänge reagiert. Während dieses Zeitraums werden kurze Impulse mit Pegel "0", die kürzer als 20,0 ms sind, möglicherweise nicht erkannt oder gezählt.

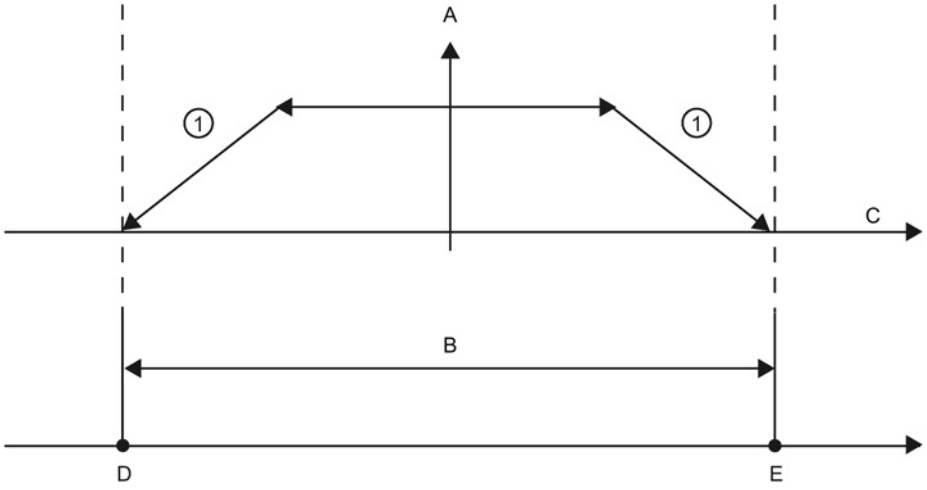
Das Verändern der Filterzeiten kann unerwarteten Betrieb der Maschine bzw. des Prozesses verursachen, was zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen kann.

Um sicherzustellen, dass eine neue Filterzeit sofort wirksam wird, müssen Sie die CPU aus- und wieder einschalten.

Software-Endschalter

Software-Endschalter begrenzen den Arbeitsbereich der Achse. Sie sollten sich relativ zum Verfahrbereich innerhalb der Hardware-Endschalter befinden. Weil die Positionen der Software-Endschalter flexibel eingerichtet werden können, lässt sich der Arbeitsbereich der Achse individuell je nach aktuellem Verfahrprofil einschränken. Im Gegensatz zu Hardware-Endschaltern werden Software-Endschalter ausschließlich über die Software implementiert und benötigen keine eigenen Schaltelemente.

Wenn Software-Endschalter aktiviert werden, wird eine aktive Bewegung an der Position des Software-Endschalters gestoppt. Die Achse wird mit der konfigurierten Verzögerung gebremst. Das folgende Diagramm zeigt das Verhalten der Achse bis zum Erreichen des Software-Endschalters.



① Die Achse bremst mit der konfigurierten Verzögerung bis zum Stillstand.

A [Geschwindigkeit]

B Arbeitsbereich

C Entfernung

D Unterer Software-Endschalter

E Oberer Software-Endschalter

Verwenden Sie zusätzlich Hardware-Endschalter, wenn sich ein mechanischer Endstopp nach den Software-Endschaltern befindet und die Gefahr mechanischer Schäden besteht.

Weitere Informationen

Ihr Anwenderprogramm kann die Grenzwerte der Hardware- oder Softwareposition durch Aktivieren oder Deaktivieren der Hardware- und Software-Grenzwertfunktionen übersteuern. Die Auswahl erfolgt über den Achsen-DB.

- Um die Hardware-Grenzwertfunktion zu aktivieren, rufen Sie die Variable "Active" (Bool) im DB-Pfad "<Achsenname>/Config/**PositonLimits_HW**" auf. Der Zustand der Variable "Active" aktiviert oder deaktiviert die Verwendung von Hardware-Positionsgrenzwerten.
- Um die Software-Grenzwertfunktion zu aktivieren, rufen Sie die Variable "Active" (Bool) im DB-Pfad "<Achsenname>/Config/**Position Limits_SW**" auf. Der Zustand der Variable "Active" aktiviert oder deaktiviert die Verwendung von Software-Positionsgrenzwerten.

Sie können die Software-Positionsgrenzwerte auch mit Ihrem Anwenderprogramm ändern (beispielsweise um die Maschineneinrichtung flexibler zu gestalten oder um die Umrüstzeiten von Maschinen zu verkürzen). Ihr Anwenderprogramm kann neue Werte in die Variablen " MinPosition" und " MaxPosition" (physikalische Einheiten im Format Real) im DB "<Achsenname>/Config/**PositionLimits_SW**" schreiben.

10.3.6.3 Referenzpunktfahrt

Bei der Referenzpunktfahrt werden die Achsenkoordinaten an die reale, physikalische Position des Antriebs angepasst. (Befindet sich der Antrieb gegenwärtig an Position x, wird die Achse in Position x gebracht.) Bei positionsgesteuerten Achsen beziehen sich die Einträge und Anzeigen für die Position exakt auf diese Achsenkoordinaten.

Hinweis

Die Übereinstimmung zwischen den Achsenkoordinaten und der realen Situation ist äußerst wichtig. Dieser Schritt ist erforderlich, um sicherzustellen, dass die absolute Zielposition der Achse auch exakt mit dem Antrieb erreicht wird.

Die Anweisung MC_Home löst die Referenzpunktfahrt der Achse aus.

Es gibt vier verschiedene Funktionen für die Referenzpunktfahrt. Die ersten beiden Funktionen ermöglichen es dem Anwender, die aktuelle Position der Achse einzustellen, und die beiden zweiten positionieren die Achse in Bezug auf einen Referenzpunktsensor.

- Betriebsart 0 - Direkte Referenzpunktfahrt absolut: Wenn diese Betriebsart ausgeführt wird, wird der Achse genau mitgeteilt, wo sie sich befindet. Die interne Positionsvariable wird auf den Wert des Positionseingangs der Anweisung für die Referenzpunktfahrt gesetzt. Dies wird bei der Kalibrierung und Einrichtung von Maschinen verwendet.

  Die Achsenposition wird unabhängig vom Referenzpunktschalter gesetzt. Aktive Verfahrbewegungen werden nicht abgebrochen. Der Wert des Eingangsparameters Position der Anweisung MC_Home wird sofort als Referenzpunkt der Achse gesetzt. Um den Referenzpunkt einer genauen mechanischen Position zuzuweisen, muss sich die Achse zum Zeitpunkt der Referenzpunkteinstellung an dieser Position im Stillstand befinden.

- Betriebsart 1 - Direkte Referenzpunktfahrt relativ: Diese Betriebsart nutzt bei der Ausführung die interne Positionsvariable und fügt den Wert des Positionseingangs der Anweisung für die Referenzpunktfahrt in diese Variable ein. Dies wird typischerweise genutzt, um den Maschinenversatz zu berücksichtigen.

  Die Achsenposition wird unabhängig vom Referenzpunktschalter gesetzt. Aktive Verfahrbewegungen werden nicht abgebrochen. Die folgende Aussage gilt für die Achsenposition nach der Referenzpunktfahrt: Neue Achsenposition = aktuelle Achsenposition + Wert des Parameters Position der Anweisung MC_Home.

- Betriebsart 2 - Passive Referenzpunktfahrt: Wenn sich die Achse bewegt und den Referenzpunktschalter überfährt, wird die aktuelle Position als Referenzpunkt gesetzt. Bei dieser Funktion wird der normale Maschinenverschleiß und das Zahnflankenspiel berücksichtigt, um den Bedarf an manuellem Verschleißausgleich zu verhindern. Der Positionseingang der Anweisung für die Referenzpunktfahrt wird wie zuvor zur vom Referenzpunktschalter angegebenen Position addiert, um den Referenzpunkt mühelos zu versetzen.

  Während der passiven Referenzpunktfahrt führt die Anweisung MC_Home keine Referenzpunktfahrtbewegung durch. Die für diesen Schritt erforderliche Verfahrbewegung müssen Sie über andere Bewegungssteuerungsanweisungen implementieren. Wenn der Referenzpunktschalter erkannt wird, wird die Achse entsprechend der Konfiguration an den Referenzpunkt gefahren. Aktive Verfahrbewegungen werden beim Start der passiven Referenzpunktfahrt nicht abgebrochen.

- Betriebsart 3 - Aktive Referenzpunktfahrt: Diese Betriebsart ist das präziseste Verfahren für die Referenzpunktfahrt der Achse. Die anfängliche Richtung und Geschwindigkeit der Bewegung werden in den erweiterten Parametern in der Konfiguration des Technologieobjekts unter Referenzpunktfahrt konfiguriert. Dies ist abhängig von der Maschinenkonfiguration. Zudem kann festgelegt werden, ob die steigende oder fallende Flanke des Signals des Referenzpunktschalters der Referenzpunkt ist. Praktisch alle Sensoren haben einen aktiven Bereich, und wenn die Position "Steady State On" als Referenzpunktsignal verwendet wurde, besteht die Möglichkeit eines Fehlers beim Referenzpunkt, weil der aktive Bereich des EIN-Signals einen Entfernungsbereich abdeckt. Durch Verwendung der steigenden oder fallenden Flanke dieses Signals resultiert ein sehr viel präziserer Referenzpunkt. Wie bei allen anderen Betriebsarten wird der Wert des Positionseingangs der Anweisung für die Referenzpunktfahrt zum Hardware-Referenzpunkt addiert.

  Bei der aktiven Referenzpunktanfahrt führt die Anweisung MC_Home die erforderliche Referenzpunktanfahrt durch. Wenn der Referenzpunktschalter erkannt wird, wird die Achse entsprechend der Konfiguration an den Referenzpunkt gefahren. Aktive Verfahrbewegungen werden abgebrochen.

Bei den Betriebsarten 0 und 1 muss die Achse überhaupt nicht bewegt werden. Sie werden typischerweise bei der Einrichtung und Kalibrierung verwendet. Bei den Betriebsarten 2 und 3 muss die Achse bewegt werden und einen Sensor überfahren, der im Technologieobjekt "Achse" als Referenzpunktschalter konfiguriert ist. Der Referenzpunkt kann im Arbeitsbereich der Achse oder außerhalb des normalen Arbeitsbereichs der Achse, muss jedoch innerhalb des Bewegungsbereichs platziert werden.

Konfiguration der Parameter für die Referenzpunktfahrt

Sie konfigurieren die Parameter für aktive und passive Referenzpunktfahrt im Konfigurationsfenster "Referenzpunktfahrt". Das Verfahren der Referenzpunktfahrt wird über den Eingangsparameter "Mode" der Bewegungssteuerungsanweisung festgelegt. Hier bedeutet Mode = 2 passive Referenzpunktfahrt und Mode = 3 bedeutet aktive Referenzpunktfahrt.

Hinweis

Stellen Sie anhand einer der folgenden Maßnahmen sicher, dass die Maschine bei Richtungsumkehr nicht zu einem mechanischen Endstopp fährt:

- Halten Sie die Anfahrgeschwindigkeit gering
- Vergrößern Sie die konfigurierte Beschleunigung/Verzögerung
- Vergrößern Sie die Entfernung zwischen Hardware-Endschalter und mechanischem Stopp

Tabelle 10- 51 Konfigurationsparameter für die Referenzpunktfahrt der Achse

Parameter	Beschreibung
Eingang Referenzpunktschalter (Aktive und passive Referenzpunktfahrt)	Wählen Sie den digitalen Eingang für den Referenzpunktschalter in der Klappliste aus. Der Eingang muss alarmfähig sein. Die integrierten Eingänge der CPU und die Eingänge eines gesteckten Signalboards können als Eingänge für den Referenzpunktschalter ausgewählt werden. Die Standardfilterzeit für die digitalen Eingänge beträgt 6,4 ms. Wenn die digitalen Eingänge als Referenzpunktschalter verwendet werden, kann dies zu unerwünschten Verzögerungen und damit zu Ungenauigkeiten führen. Abhängig von der verringerten Geschwindigkeit und dem Ausmaß des Referenzpunktschalters wird der Referenzpunkt möglicherweise nicht erkannt. Die Filterzeit kann in der Gerätekonfiguration der digitalen Eingänge unter "Eingangsfilter" eingestellt werden. Die angegebene Filterzeit muss kleiner als die Dauer des Eingangssignals am Referenzpunktschalter sein.
Automatische Umkehr nach Erreichen der Hardware-Endschalter (Nur bei aktiver Referenzpunktfahrt)	Aktivieren Sie das Kontrollkästchen, um den Hardware-Endschalter als Umkehrnocken für die Referenzpunktanfahrt zu nutzen. Die Hardware-Endschalter müssen für die Richtungsumkehr konfiguriert und aktiviert sein. Wenn der Hardware-Endschalter während der aktiven Referenzpunktfahrt erreicht wird, bremst die Achse mit der konfigurierten Verzögerung (nicht mit der Notfallverzögerung) und kehrt die Richtung um. Der Referenzpunktschalter wird dann in umgekehrter Richtung erfasst. Wenn die Richtungsumkehr nicht aktiv ist und die Achse den Hardware-Endschalter während der aktiven Referenzpunktfahrt erreicht, wird die Referenzpunktanfahrt mit einem Fehler abgebrochen und die Achse mit der Notfallverzögerung gebremst.
Anfahrrichtung (Aktive und passive Referenzpunktfahrt)	Bei der Richtungsauswahl legen Sie die "Anfahrrichtung" für die aktive Referenzpunktfahrt zum Suchen des Referenzpunktschalters sowie die Richtung der Referenzpunktfahrt fest. Die Richtung der Referenzpunktfahrt legt die Verfahrrichtung der Achse zum Anfahren der konfigurierten Seite des Referenzpunktschalters fest, um die Referenzpunktfahrt durchzuführen.
Referenzpunktschalter (Aktive und passive Referenzpunktfahrt)	• Aktive Referenzpunktfahrt: Geben Sie an, ob die Achse auf der linken oder rechten Seite des Referenzpunktschalters referenziert werden soll. Je nach Startposition der Achse und der Konfiguration der Parameter für die Referenzpunktfahrt kann sich die Abfolge der Referenzpunktanfahrt vom Diagramm im Konfigurationsfenster unterscheiden. • Passive Referenzpunktfahrt: Bei der passiven Referenzpunktfahrt müssen die Verfahrbewegungen zum Zweck der Referenzpunktfahrt vom Anwender über Bewegungssteuerungsbefehle implementiert werden. Die Seite des Referenzpunktschalters, auf der die Referenzpunktfahrt erfolgt, ist von den folgenden Faktoren abhängig: – Konfigurierte Anfahrrichtung – Konfigurierter Referenzpunktschalter – Aktuelle Verfahrrichtung während der passiven Referenzpunktfahrt
Anfahrgeschwindigkeit (Nur bei aktiver Referenzpunktfahrt)	Geben Sie die Geschwindigkeit an, mit der der Referenzpunktschalter während der Referenzpunktanfahrt gesucht werden soll. Grenzwerte (unabhängig von der ausgewählten Anwendereinheit): Start-/Stoppgeschwindigkeit ≤ Anfahrgeschwindigkeit ≤ Maximalgeschwindigkeit

Parameter	Beschreibung
Verringerte Geschwindigkeit (Nur bei aktiver Referenzpunktfahrt)	Geben Sie die Geschwindigkeit an, mit der die Achse den Referenzpunktschalter für die Referenzpunktfahrt anfährt. Grenzwerte (unabhängig von der ausgewählten Anwendereinheit): Start-/Stoppgeschwindigkeit ≤ verringerte Geschwindigkeit ≤ Maximalgeschwindigkeit
Referenzpunktversatz (Nur bei aktiver Referenzpunktfahrt)	Wenn der gewünschte Referenzpunkt von der Position des Referenzpunktschalters abweicht, kann der Referenzpunktversatz in diesem Feld angegeben werden. Ist der Wert ungleich 0, führt die Achse die folgenden Vorgänge nach der Referenzpunktfahrt am Referenzpunktschalter durch: 1. Bewegen der Achse mit verringerter Geschwindigkeit um den Wert des Referenzpunktversatzes. 2. Wenn die Position des Referenzpunkts erreicht ist, wird die Achsenposition auf den absoluten Referenzpunkt gesetzt. Der absolute Referenzpunkt wird über den Parameter "Position" der Bewegungssteuerungsanweisung "MC_Home" angegeben. Grenzwerte (unabhängig von der ausgewählten Anwendereinheit): -1,0e12 ≤ Referenzpunktversatz ≤ 1,0e12

Tabelle 10- 52 Die Referenzpunktfahrt beeinflussende Faktoren

Beeinflussende Faktoren:			Ergebnis:
Konfiguration Anfahrrichtung	Konfiguration Referenzpunktschalter	Aktuelle Verfahrrichtung	Referenzpunktfahrt zu Referenzpunktschalter
Positiv	"Linke (negative) Seite"	Positive Richtung	Links
		Negative Richtung	Rechts
Positiv	"Rechte (positive) Seite"	Positive Richtung	Rechts
		Negative Richtung	Links
Negativ	"Linke (negative) Seite"	Positive Richtung	Rechts
		Negative Richtung	Links
Negativ	"Rechte (positive) Seite"	Positive Richtung	Links
		Negative Richtung	Rechts

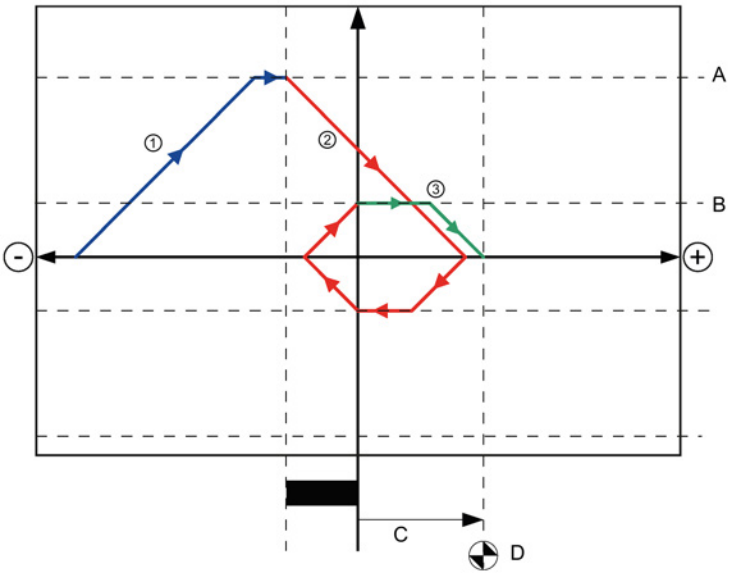
Abfolge bei der aktiven Referenzpunktfahrt

Sie aktive Referenzpunktfahrt starten Sie mit der Bewegungssteuerungsanweisung "MC_Home" (Eingangsparameter Mode = 3). Der Eingangsparameter "Position" gibt in diesem Fall die absoluten Koordinaten des Referenzpunkts an. Alternativ können Sie die aktive Referenzpunktfahrt zu Testzwecken im Steuerpanel starten.

Das folgende Diagramm zeigt ein Beispiel einer Kennlinie für eine aktive Referenzpunktanfahrt mit den folgenden Konfigurationsparametern:

- "Anfahrrichtung" = "Positive Anfahrrichtung"
- "Referenzpunktschalter" = "Rechte (positive) Seite"
- Wert des "Referenzpunktversatzes" > 0

Tabelle 10- 53 Geschwindigkeitskennlinie für die Referenzpunktfahrt bei der Bewegungssteuerung

Funktionsweise	Hinweise	
	A	Anfahrgeschwindigkeit
	B	Verringerte Geschwindigkeit
	C	Referenzpunktkoordinaten
	D	Referenzpunktversatz
①	Suchphase (blaues Kennliniensegment): Wenn die aktive Referenzpunktfahrt startet, beschleunigt die Achse auf die konfigurierte "Anfahrgeschwindigkeit" und sucht bei dieser Geschwindigkeit nach dem Referenzpunktschalter.	
②	Referenzpunktanfahrt (roter Kennlinienabschnitt): Wenn der Referenzpunktschalter erkannt ist, bremst die Achse in diesem Beispiel und kehrt um, um auf der konfigurierten Seite des Referenzpunktschalters bei der konfigurierten "verringerten Geschwindigkeit" an den Referenzpunkt zu fahren.	
③	Verfahren zum Referenzpunkt (grünes Kennliniensegment): Nach der Referenzpunktfahrt am Referenzpunktschalter verfährt die Achse bei "verringerter Geschwindigkeit" zu den "Referenzpunktkoordinaten". Beim Erreichen der "Referenzpunktkoordinaten" wird die Achse an dem Positionswert gestoppt, der im Eingangsparameter Position der Anweisung MC_Home angegeben wurde.	

Hinweis

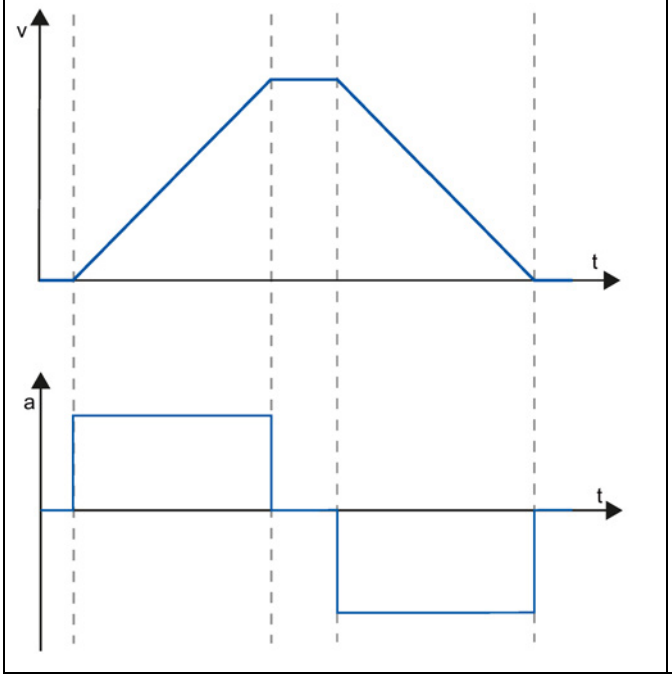
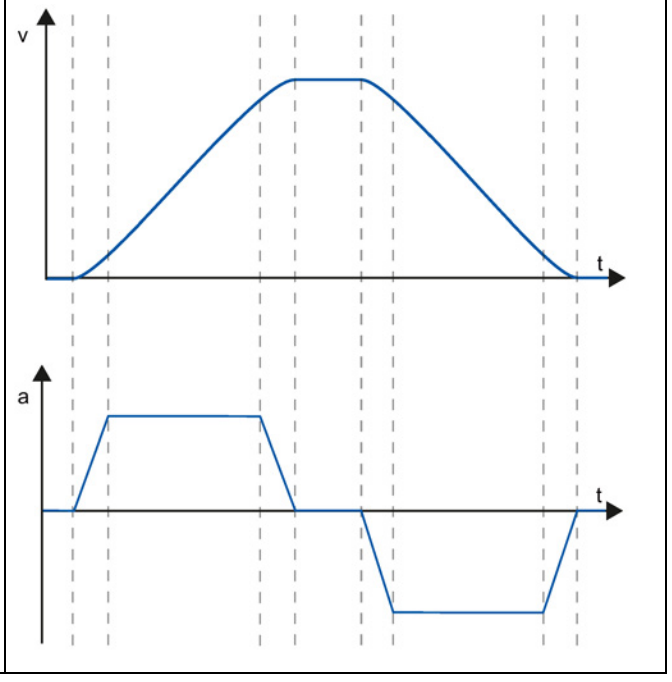
Wenn die Referenzpunktsuche nicht wie erwartet funktioniert, prüfen Sie die den Hardware-Grenzwerten oder dem Referenzpunkt zugewiesenen Eingänge. Möglicherweise wurden die Flankenalarme für diese Eingänge in der Gerätekonfiguration deaktiviert.

Prüfen Sie die Konfigurationsdaten des betroffenen Technologieobjekts "Achse", um festzustellen, welche Eingänge ggf. zu "HW Low Limit Switch Input", "HW High Limit Switch Input" und "Input reference point switch" zugewiesen sind. Öffnen Sie dann die Gerätekonfiguration der CPU und untersuchen Sie jeden der zugewiesenen Eingänge. Prüfen Sie, ob "Erkennung steigende Flanke aktivieren" und "Erkennung fallende Flanke aktivieren" beide ausgewählt sind. Sind diese Eigenschaften nicht ausgewählt, löschen Sie die angegebenen Eingänge in der Achsenkonfiguration und wählen sie erneut aus.

10.3.6.4 Ruckbegrenzung

Mit der Ruckbegrenzung können Sie die mechanischen Beanspruchungen während der Beschleunigungs- und Verzögerungsphase verringern. Der Wert für die Beschleunigung und Verzögerung wird nicht abrupt verändert, wenn die Schrittbegrenzung aktiv ist, der Wert wird stattdessen während einer Übergangsphase angepasst. Die folgende Abbildung zeigt die Geschwindigkeits- und Beschleunigungskurve ohne und mit Ruckbegrenzung.

Tabelle 10- 54 Ruckbegrenzung

Fahrt ohne Schrittbegrenzung	Fahrt mit Schrittbegrenzung
	

Die Ruckbegrenzung sorgt für ein "geglättetes" Geschwindigkeitsprofil der Achsbewegung. Dadurch wird ein weiches Anfahren und Abbremsen beispielsweise eines Förderbands gewährleistet.

10.3.7 Bewegungssteuerungsanweisungen

10.3.7.1 Übersicht MC-Anweisungen

Die Bewegungssteuerungsanweisungen verwenden einen zugehörigen Technologie-Datenbaustein und die dafür zugewiesene PTO (Impulsfolge) der CPU, um die Bewegung einer Achse zu steuern.

- MC_Power (Seite 590) aktiviert und deaktiviert eine Achse für die Bewegungssteuerung.
- MC_Reset (Seite 593) setzt alle Bewegungssteuerungsfehler zurück. Alle Bewegungssteuerungsfehler, die quittiert werden können, werden quittiert.
- MC_Home (Seite 594) stellt die Beziehung zwischen dem Achsensteuerungsprogramm und dem mechanischen Positionierungssystem der Achse her.
- MC_Halt (Seite 597) storniert alle Bewegungsvorgänge und bewirkt den Stopp der Achsenbewegung. Die Halteposition ist nicht definiert.
- MC_MoveAbsolute (Seite 599) startet die Bewegung zu einer absoluten Position. Der Auftrag ist beendet, wenn die Zielposition erreicht ist.
- MC_MoveRelative (Seite 601) startet eine Positionierbewegung relativ zur Startposition.
- MC_MoveVelocity (Seite 603) bewirkt, dass sich die Achse mit der angegebenen Geschwindigkeit bewegt.
- MC_MoveJog (Seite 606) führt den Tippbetrieb zu Test- und Anlaufzwecken aus.
- MC_CommandTable (Seite 609) führt Achssteuerungsbefehle als Bewegungsfolge aus.
- MC_ChangeDynamic (Seite 612) ändert die Dynamikeinstellungen der Achse.
- MC_WriteParam (Seite 614) schreibt bestimmte Parameter, um die Funktionalität der Achse im Anwenderprogramm zu ändern.
- MC_ReadParam (Seite 616) liest bestimmte Parameter, die die aktuelle Position, Geschwindigkeit usw. der im Achseneingang definierten Achse angeben.

CPU-Firmwarestände

Wenn Sie mit einer S7-1200 CPU mit Firmware V4.1 arbeiten, wählen Sie Version V5.0 der Bewegungsanweisungen.

Wenn Sie mit einer S7-1200 CPU mit Firmwareversion V4.0 oder niedriger arbeiten, wählen Sie die entsprechende Version V4.0, V3.0, V2.0 oder V1.0 der Bewegungsanweisungen.

Hinweis

Die CPU berechnet Bewegungssteuerungsaufgaben in "Scheiben" bzw. Segmenten von 10 ms. Wenn eine Scheibe ausgeführt wird, wartet die nächste Scheibe in der Warteschlange auf die Ausführung. Wenn Sie die Bewegungssteuerungsaufgabe einer Achse unterbrechen (indem Sie eine andere neue Bewegungssteuerungsaufgabe für die Achse ausführen), kann die neue Bewegungssteuerungsaufgabe maximal 20 ms lang nicht ausgeführt werden (die Restdauer der aktuellen Scheibe plus die Scheibe in der Warteschlange).

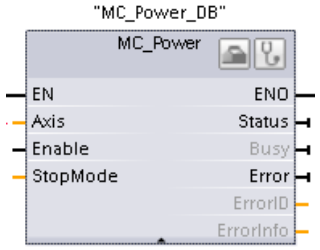
10.3.7.2 Anweisung MC_Power (Achse freigeben/sperren)

Hinweis

Wenn die Achse wegen eines Fehlers ausgeschaltet wird, wird sie nach Behebung und Quittierung des Fehlers automatisch wieder aktiviert. Hierfür ist erforderlich, dass der Eingangsparameter Enable den Wert WAHR während dieses Vorgangs gespeichert hat.

Tabelle 10- 55 Anweisung MC_Power

KOP/FUP	SCL	Beschreibung

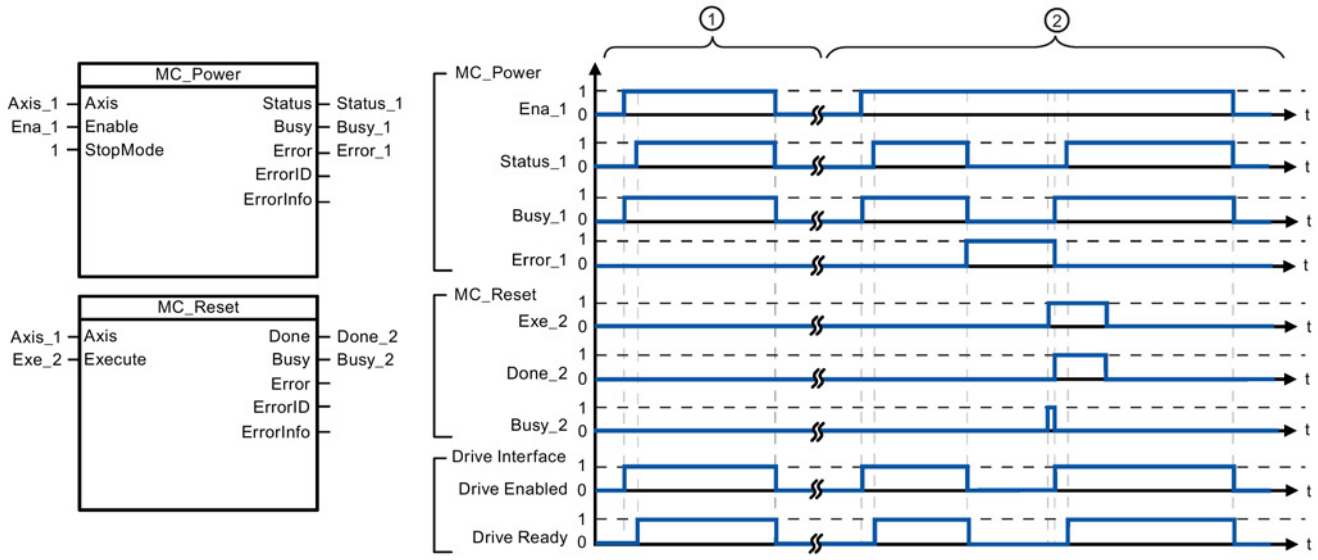
	`"MC_Power_DB"(`
	`Axis:=_multi_fb_in_,`
	`Enable:=_bool_in_,`
	`StopMode:=_int_in_,`
	`Status=>_bool_out_,`
	`Busy=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorIn-`
	`fo=>_word_out_);`
		Die Bewegungssteuerungsanweisung MC_Power aktiviert oder deaktiviert eine Achse. Bevor Sie die Achse aktivieren oder deaktivieren können, prüfen Sie die folgenden Bedingungen:
		• Das Technologieobjekt wurde korrekt konfiguriert.
		• Es steht kein Freigabe verhindernder Fehler an.
		Die Ausführung von MC_Power kann nicht von einer Bewegungssteuerungsaufgabe abgebrochen werden. Durch Deaktivieren der Achse (Eingangsparameter Enable = FALSCH) werden alle Bewegungssteuerungsaufgaben für das zugehörige Technologieobjekt abgebrochen.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_Power_DB" der Name des Instanz-DBs.

Tabelle 10- 56 Parameter für die MC_Power-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_1	Technologieobjekt "Achse"
Enable	IN	Bool	• FALSE (Standard): Alle aktiven Aufgaben werden entsprechend dem parametrierten "StopMode" abgebrochen und die Achse wird gestoppt. • TRUE: Die Bewegungssteuerung versucht, die Achse zu aktivieren.
StopMode	IN	Int	• 0: Not-Aus: Steht eine Anforderung zur Deaktivierung der Achse an, bremst die Achse mit der konfigurierten Notfallverzögerung. Die Achse wird nach Erreichen des Stillstands deaktiviert. • 1: Sofortiger Stopp: Steht eine Anforderung zur Deaktivierung der Achse an, wird die Achse ohne Verzögerung deaktiviert. Der Impulsausgang wird sofort gestoppt. • 2: Not-Aus mit Rucksteuerung: Steht eine Anforderung zur Deaktivierung der Achse an, bremst die Achse mit der konfigurierten Not-Aus-Verzögerung. Wenn die Rucksteuerung aktiviert ist, wird der konfigurierte Ruck berücksichtigt. Die Achse wird nach Erreichen des Stillstands deaktiviert.
Status	OUT	Bool	Status der Achsenfreigabe: • FALSE: Die Achse ist deaktiviert: – Die Achse führt keine Bewegungssteuerungsaufgaben aus und nimmt keine neuen Aufgaben an (Ausnahme: Aufgabe MC_Reset). – Die Achse ist nicht am Referenzpunkt. – Beim Deaktivieren wechselt der Zustand erst dann nach FALSCH, wenn die Achse den Stillstand erreicht. • TRUE: Die Achse ist aktiviert: – Die Achse ist bereit, Bewegungssteuerungsaufgaben auszuführen. – Beim Aktivieren der Achse wechselt der Zustand erst dann nach WAHR, wenn das Signal "Antrieb bereit" ansteht. Wurde die Antriebsschnittstelle "Antrieb bereit" während der Achsenkonfiguration nicht eingerichtet, wechselt der Zustand sofort nach WAHR.
Busy	OUT	Bool	FALSE: MC_Power ist nicht aktiv. TRUE: MC_Power ist aktiv.
Error	OUT	Bool	FALSE: Kein Fehler TRUE: In der Bewegungssteuerungsanweisung "MC_Power" oder im zugehörigen Technologieobjekt ist ein Fehler aufgetreten. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt.
ErrorID	OUT	Word	Fehler-ID für Parameter "Error"
ErrorInfo	OUT	Word	Fehlerinfo-ID für Parameter ErrorID



① Eine Achse wird aktiviert und dann wieder deaktiviert. Nachdem der Antrieb das Signal "Antrieb bereit" an die CPU zurückgemeldet hat, kann die erfolgreiche Aktivierung über "Status_1" ausgelesen werden.

② Nach einer Achsenfreigabe ist ein Fehler aufgetreten, der verursacht hat, dass die Achse deaktiviert wurde. Der Fehler wird behoben und mit "MC_Reset" quittiert. Die Achse wird dann wieder aktiviert.

Um eine Achse mit konfigurierter Antriebsschnittstelle zu aktivieren, gehen Sie wie folgt vor:

1. Prüfen Sie die oben angegebenen Voraussetzungen.
2. Initialisieren Sie den Eingangsparameter "StopMode" mit dem gewünschten Wert. Setzen Sie den Eingangsparameter "Enable" auf WAHR.

   Der Freigabeausgang für "Antrieb freigegeben" wechselt nach WAHR, um die Spannung für den Antrieb zu aktivieren. Die CPU wartet auf das Signal "Antrieb bereit" des Antriebs.

   Wenn das Signal "Antrieb bereit" am konfigurierten Bereitschaftseingang der CPU verfügbar ist, wird die Achse freigegeben. Der Ausgangsparameter "Status" und die Technologieobjektvariable <Achsenname>.StatusBits.Enable geben den Wert WAHR an.

Um eine Achse ohne konfigurierte Antriebsschnittstelle zu aktivieren, gehen Sie wie folgt vor:

1. Prüfen Sie die oben angegebenen Voraussetzungen.
2. Initialisieren Sie den Eingangsparameter "StopMode" mit dem gewünschten Wert. Setzen Sie den Eingangsparameter "Enable" auf WAHR. Die Achse ist aktiviert. Der Ausgangsparameter "Status" und die Technologieobjektvariable <Achsenname>.StatusBits.Enable geben den Wert WAHR an.

Um eine Achse zu deaktivieren, gehen Sie wie folgt vor:

1. Bringen Sie die Achse zum Stillstand.

   Anhand der Technologieobjektvariable <Achsenname>.StatusBits.StandStill können Sie erkennen, wann sich die Achse im Stillstand befindet.
2. Setzen Sie den Eingangsparameter "Enable" nach Erreichen des Stillstands auf FALSCH.
3. Wenn die Ausgangsparameter "Busy" und "Status" und die Technologieobjektvariable <Achsenname>.StatusBits.Enable den Wert FALSCH anzeigen, ist die Deaktivierung der Achse beendet.

10.3.7.3 Anweisung MC_Reset (Fehler bestätigen)

Tabelle 10- 57 Anweisung MC_Reset

KOP/FUP	SCL	Beschreibung

	`"MC_Reset_DB"(`
	`Axis:=_multi_fb_in_,`
	`Execute:=_bool_in_,`
	`Restart:=_bool_in_,`
	`Done=>_bool_out_,`
	`Busy=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		Mit der Anweisung MC_Reset quittieren Sie "Betriebsfehler mit Achsenstopp" und "Konfigurationsfehler". Die Fehler, die quittiert werden müssen, finden Sie in der "Liste von ErrorIDs und ErrorInfos" unter "Abhilfe".
		Bevor Sie die Anweisung MC_Reset verwenden, müssen Sie die Ursache eines anstehenden zu quittierenden Konfigurationsfehlers behoben haben (indem Sie beispielsweise einen ungültige Beschleunigungswert im Technologieobjekt "Achse" in einen gültigen Wert ändern).
		Ab Version 3.0 ermöglicht der Befehl Restart das Laden der Achsenkonfiguration in den Arbeitsspeicher im Betriebszustand RUN.

1 STEP 7 erstellt den DB automatisch, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_Reset_DB" der Name des Instanz-DBs.

Die Aufgabe MC_Reset kann von keiner anderen Bewegungssteuerungsaufgabe abgebrochen werden. Die neue Aufgabe MC_Reset bricht keine anderen aktiven Bewegungssteuerungsaufgaben ab.

Tabelle 10- 58 Parameter der Anweisung MC_Reset

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_1	Technologieobjekt "Achse"
Execute	IN	Bool	Starten der Aufgabe bei einer positiven Flanke
Restart	IN	Bool	TRUE = Laden der Achsenkonfiguration aus dem Ladespeicher in den Arbeitsspeicher. Der Befehl kann nur bei deaktivierter Achse ausgeführt werden.
			FALSE = Anstehende Fehler werden quittiert
Done	OUT	Bool	WAHR = Fehler wurde quittiert.
Busy	OUT	Bool	WAHR = Die Aufgabe wird ausgeführt.
Error	OUT	Bool	WAHR = Während der Ausführung der Aufgabe trat ein Fehler auf. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt.
ErrorID	OUTP	Word	Fehler-ID für Parameter "Error"
ErrorInfo	OUT	Word	Fehlerinfo-ID für Parameter ErrorID

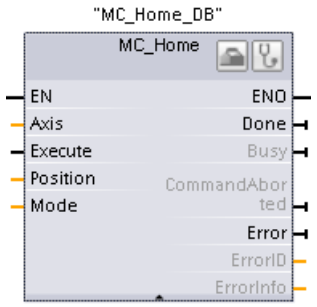
Um einen Fehler mit MC_Reset zu quittieren, gehen Sie folgendermaßen vor:

1. Prüfen Sie die oben angegebenen Voraussetzungen.
2. Starten Sie die Quittierung des Fehlers mit einer steigenden Flanke am Eingangsparameter Execute.
3. Der Fehler wurde quittiert, wenn Done gleich WAHR ist und die Technologieobjektvariable <Achsenname>.StatusBits.Error gleich FALSCH ist.

10.3.7.4 Anweisung MC_Home (Referenzpunktfahrt der Achse durchführen)

Tabelle 10- 59 Anweisung MC_Home

KOP/FUP	SCL	Beschreibung

	`"MC_Home_DB"(`
	`Axis:=_multi_fb_in_,`
	`Execute:=_bool_in_,`
	`Position:=_real_in_,`
	`Mode:=_int_in_,`
	`Done=>_bool_out_,`
	`Busy=>_bool_out_,`
	`CommandAborted=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		Mit der Anweisung MC_Home passen Sie die Achsenkoordinaten an die reale, physikalische Position des Antriebs an. Für die absolute Positionierung der Achse ist eine Referenzpunktfahrt erforderlich:
		Um die Anweisung MC_Home zu verwenden, muss die Achse zunächst freigegeben werden.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

[2] Im SCL-Beispiel ist "MC_Home_DB" der Name des Instanz-DBs.

Die folgenden Arten von Referenzpunktfahrten stehen zur Verfügung:

- Direkte Referenzpunktfahrt absolut (Mode = 0): Die aktuelle Achsenposition wird auf den Wert des Parameters "Position" gesetzt.
- Direkte Referenzpunktfahrt relativ (Mode = 1): Die aktuelle Achsenposition wird um den Wert des Parameters "Position" versetzt.
- Passive Referenzpunktfahrt (Mode = 2): Während der passiven Referenzpunktfahrt führt die Anweisung MC_Home keine Referenzpunktfahrtbewegung durch. Die für diesen Schritt erforderliche Verfahrbewegung müssen Sie über andere Bewegungssteuerungsanweisungen implementieren. Wenn der Referenzpunktwechsel erkannt wird, wird die Achse an den Referenzpunkt gefahren.
- Aktive Referenzpunktfahrt (Mode = 3): Die Referenzpunktfahrt wird automatisch durchgeführt.

Tabelle 10- 60 Parameter für die MC_Home-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_PTO	Technologieobjekt "Achse"
Execute	IN	Bool	Starten der Aufgabe bei einer positiven Flanke
Position	IN	Real	• Mode = 0, 2 und 3 (Absolute Position der Achse nach Beendigung der Referenzpunktfahrt) • Mode = 1 (Korrekturwert für die aktuelle Achsenposition) Grenzwerte: $-1{,}0e^{12} \leq$ Position $\leq 1{,}0e^{12}$
Mode	IN	Int	Art der Referenzpunktfahrt • 0: Direkte Referenzpunktfahrt absolut Die neue Achsenposition ist der Positionswert des Parameters "Position". • 1: Direkte Referenzpunktfahrt relativ Die neue Achsenposition ist die aktuelle Achsenposition + Positionswert des Parameters "Position". • 2: Passive Referenzpunktfahrt Referenzpunktfahrt entsprechend der Achsenkonfiguration. Nach der Referenzpunktfahrt wird der Wert des Parameters "Position" als neue Achsenposition eingestellt. • 3: Aktive Referenzpunktfahrt Referenzpunktannäherung entsprechend der Achsenkonfiguration. Nach der Referenzpunktfahrt wird der Wert des Parameters "Position" als neue Achsenposition eingestellt.
Done	OUT	Bool	WAHR = Aufgabe durchgeführt
Busy	OUT	Bool	WAHR = Die Aufgabe wird ausgeführt.
CommandAborted	OUT	Bool	WAHR = Während der Ausführung wurde die Aufgabe von einer anderen Aufgabe abgebrochen.
Error	OUT	Bool	WAHR = Während der Ausführung der Aufgabe trat ein Fehler auf. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt.
ErrorID	OUT	Word	Fehler-ID für Parameter "Error"
ErrorInfo	OUT	Word	Fehlerinfo-ID für Parameter ErrorID

Hinweis

Die Referenzpunktfahrt der Achse geht unter den folgenden Voraussetzungen verloren

- Deaktivierung der Achse durch die Anweisung MC_Power
- Umschalten zwischen Automatik- und Handbetrieb
- Nach dem Start der aktiven Referenzpunktfahrt (nach erfolgreicher Durchführung der Referenzpunktfahrt steht die Referenzpunktfahrt der Achse erneut zur Verfügung.)
- Nach Aus- und wieder Einschalten der CPU
- Nach Neustart der CPU (RUN-in-STOP oder STOP-in-RUN)

Um die Achse an den Referenzpunkt zu fahren, gehen Sie folgendermaßen vor:

1. Prüfen Sie die oben angegebenen Voraussetzungen.
2. Initialisieren Sie die erforderlichen Eingangsparameter mit Werten und starten Sie die Referenzpunktfahrt mit einer steigenden Flanke am Eingangsparameter "Execute".
3. Wenn der Ausgangsparameter "Done" und die Technologieobjektvariable <Achsenname>.StatusBits.HomingDone den Wert WAHR angeben, ist die Referenzpunktfahrt beendet.

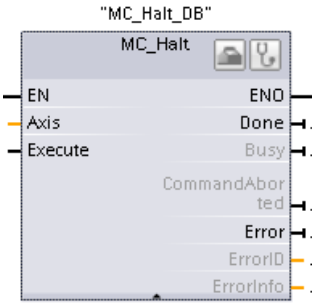
Tabelle 10- 61 Übersteuerungsantwort

Betriebsart	Beschreibung	
0 oder 1	Die Aufgabe MC_Home kann von keiner anderen Bewegungssteuerungsaufgabe abgebrochen werden. Die neue Aufgabe MC_Home bricht keine aktiven Bewegungssteuerungsaufgaben ab. Positionsbezogene Bewegungssteuerungsaufgaben werden nach der Referenzpunktfahrt entsprechend der neuen Referenzpunktposition (Wert am Eingangsparameter Position) wieder aufgenommen.	
2	Die Aufgabe MC_Home kann von den folgenden Bewegungssteuerungsaufgaben abgebrochen werden: Mode der Aufgabe MC_Home = 2, 3: Die neue Aufgabe MC_Home bricht die folgende aktive Bewegungssteuerungsaufgabe ab. Mode der Aufgabe MC_Home = 2: Positionsbezogene Bewegungssteuerungsaufgaben werden nach der Referenzpunktfahrt entsprechend der neuen Referenzpunktposition (Wert am Eingangsparameter Position) wieder aufgenommen.	
3	Die Aufgabe MC_Home kann von den folgenden Bewegungssteuerungsaufgaben abgebrochen werden: • MC_Home Mode = 3 • MC_Halt • MC_MoveAbsolute • MC_MoveRelative • MC_MoveVelocity • MC_MoveJog	Die neue Aufgabe MC_Home bricht die folgenden aktiven Bewegungssteuerungsaufgaben ab: • MC_Home-Modus = 2, 3 • MC_Halt • MC_MoveAbsolute • MC_MoveRelative • MC_MoveVelocity • MC_MoveJog

10.3.7.5 Anweisung MC_Halt (Achse pausieren)

Tabelle 10- 62 Anweisung MC_Halt

KOP/FUP	SCL	Beschreibung

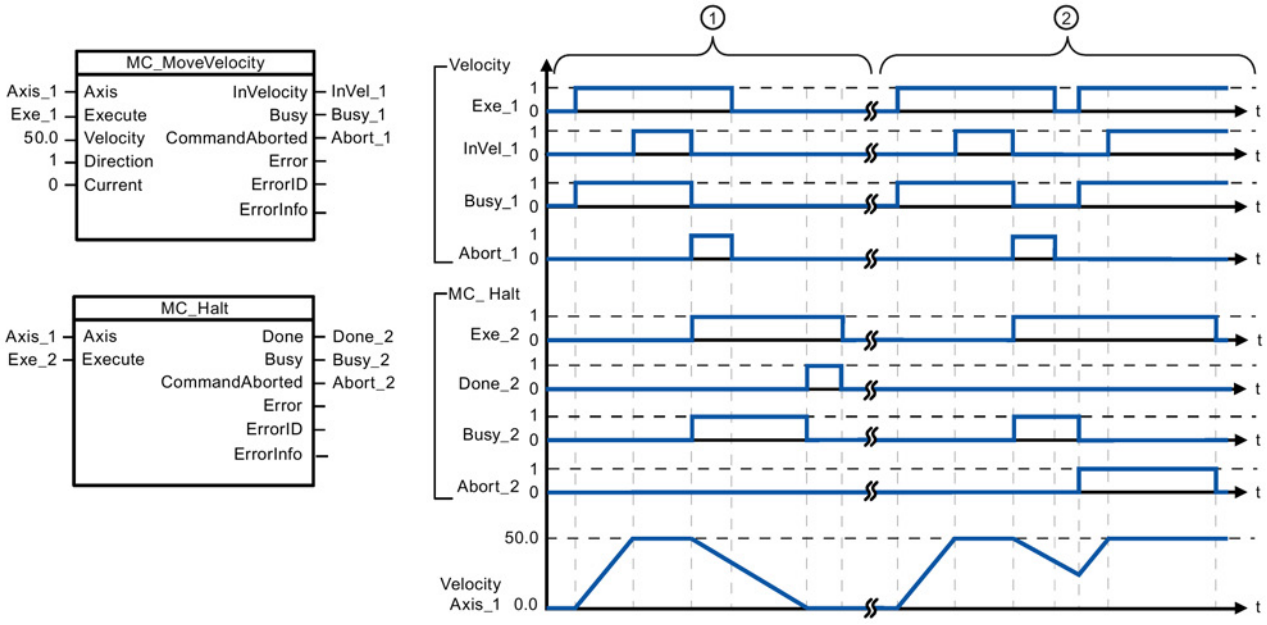
	`"MC_Halt_DB"(`
	`  Axis:=_multi_fb_in_,`
	`  Execute:=_bool_in_,`
	`  Done=>_bool_out_,`
	`  Busy=>_bool_out_,`
	`  CommandAborted=>_bool_out_,`
	`  Error=>_bool_out_,`
	`  ErrorID=>_word_out_,`
	`  ErrorInfo=>_word_out_);`
		Mit der Anweisung MC_Halt stoppen Sie alle Bewegungen und bringen die Achse zum Stillstand. Die Stillstandposition ist nicht definiert.
		Um die Anweisung MC_Halt zu verwenden, muss die Achse zunächst freigegeben werden.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_Halt_DB" der Name des Instanz-DBs.

Tabelle 10- 63 Parameter für die MC_Halt-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_1	Technologieobjekt "Achse"
Execute	IN	Bool	Starten der Aufgabe bei einer positiven Flanke
Done	OUT	Bool	WAHR = Nullgeschwindigkeit erreicht
Busy	OUT	Bool	WAHR = Die Aufgabe wird ausgeführt.
CommandAborted	OUT	Bool	WAHR = Während der Ausführung wurde die Aufgabe von einer anderen Aufgabe abgebrochen.
Error	OUT	Bool	WAHR = Während der Ausführung der Aufgabe trat ein Fehler auf. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt.
ErrorID	OUT	Word	Fehler-ID für Parameter "Error"
ErrorInfo	OUT	Word	Fehlerinfo-ID für Parameter ErrorID



Die folgenden Werte wurden im Konfigurationsfenster "Dynamik > Allgemein" konfiguriert: Beschleunigung = 10,0 und Verzögerung = 5,0

① Die Achse wird von einer Aufgabe MC_Halt gebremst, bis sie zum Stillstand kommt. Der Stillstand der Achse wird über "Done_2" gemeldet.

② Während die Aufgabe MC_Halt die Achse bremst, wird die Aufgabe von einer anderen Bewegungssteuerungsaufgabe abgebrochen. Der Abbruch wird über "Abort_2" gemeldet.

Übersteuerungsantwort

Die Aufgabe MC_Halt kann von den folgenden Bewegungssteuerungsaufgaben abgebrochen werden:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

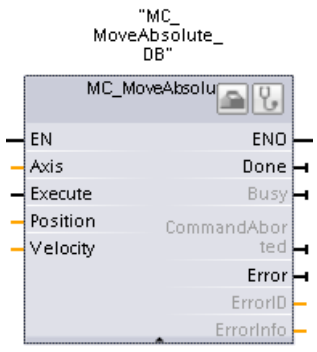
Die neue Aufgabe MC_Halt bricht die folgenden aktiven Bewegungssteuerungsaufgaben ab:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

10.3.7.6 Anweisung MC_MoveAbsolute (Achse absolut positionieren)

Tabelle 10- 64 Anweisung MC_MoveAbsolute

KOP/FUP	SCL	Beschreibung

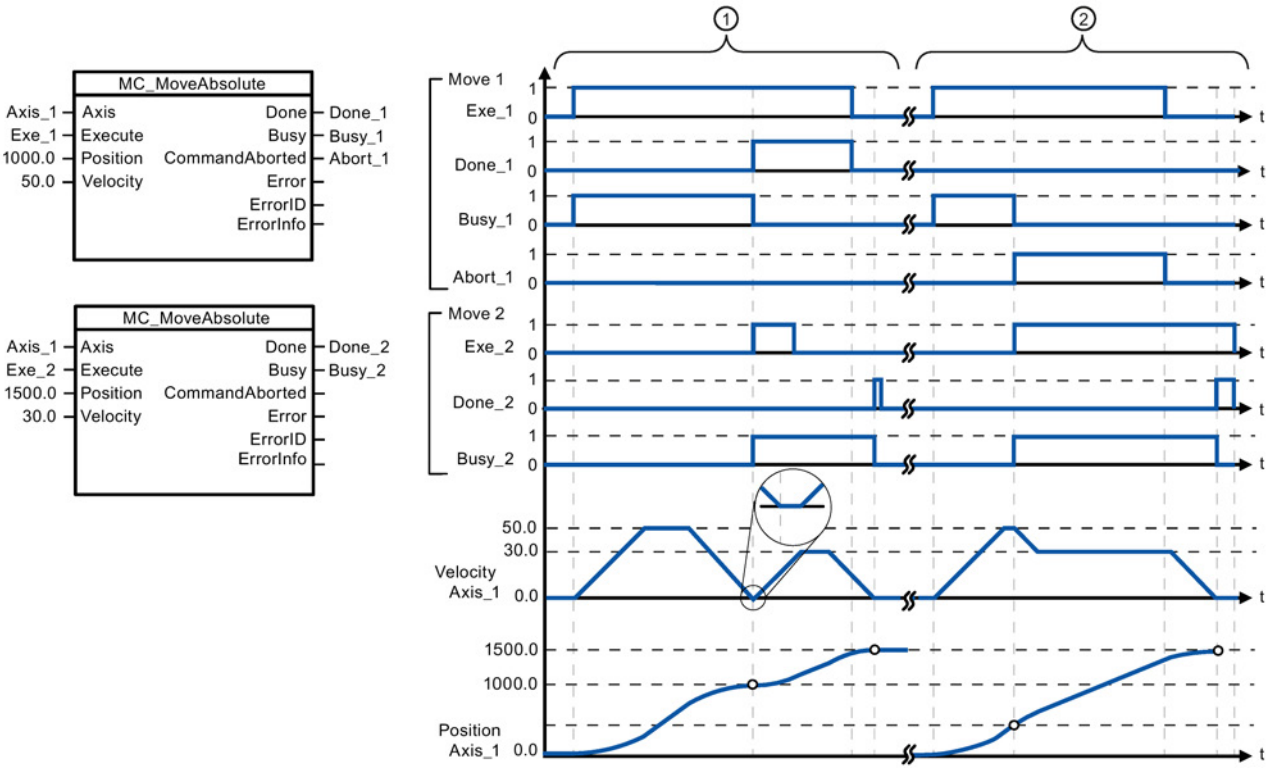
	`"MC_MoveAbsolute_DB"(`
	`Axis:=_multi_fb_in_,`
	`Execute:=_bool_in_,`
	`Position:=_real_in_,`
	`Velocity:=_real_in_,`
	`Done=>_bool_out_,`
	`Busy=>_bool_out_,`
	`CommandAborted=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		Mit der Anweisung MC_MoveAbsolute starten Sie eine Positionierbewegung der Achse zu einer absoluten Position.
		Um die Anweisung MC_MoveAbsolute zu verwenden, muss die Achse zunächst freigegeben und zum Referenzpunkt gefahren werden.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_MoveAbsolute_DB" der Name des Instanz-DBs.

Tabelle 10- 65 Parameter für die MC_MoveAbsolute-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_1	Technologieobjekt "Achse"
Execute	IN	Bool	Starten der Aufgabe bei einer positiven Flanke (Standardwert: Falsch)
Position	IN	Real	Absolute Zielposition (Standardwert: 0.0) Grenzwerte: $-1{,}0e^{12} \leq$ Position $\leq 1{,}0e^{12}$
Velocity	IN	Real	Achsengeschwindigkeit (Standardwert: 10.0) Diese Geschwindigkeit wird wegen der konfigurierten Beschleunigung und Verzögerung und der anzufahrenden Zielposition nicht immer erreicht. Grenzwerte: Start-/Stoppgeschwindigkeit ≤ Velocity ≤ Maximalgeschwindigkeit
Done	OUT	Bool	WAHR = Absolute Zielposition erreicht
Busy	OUT	Bool	WAHR = Die Aufgabe wird ausgeführt.
CommandAborted	OUT	Bool	WAHR = Während der Ausführung wurde die Aufgabe von einer anderen Aufgabe abgebrochen.
Error	OUT	Bool	WAHR = Während der Ausführung der Aufgabe trat ein Fehler auf. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt.
ErrorID	OUT	Word	Fehler-ID für Parameter "Error" (Standardwert: 0000)
ErrorInfo	OUT	Word	Fehlerinfo-ID für Parameter "ErrorID" (Standardwert: 0000)



Die folgenden Werte wurden im Konfigurationsfenster "Dynamik > Allgemein" konfiguriert: Beschleunigung = 10,0 und Verzögerung = 10,0

① Eine Achse wird mit der Aufgabe MC_MoveAbsolute zur absoluten Position 1000,0 gefahren. Wenn die Achse die Zielposition erreicht, wird dies über "Done_1" gemeldet. Wenn "Done_1" = WAHR ist, wird eine andere Aufgabe MC_MoveAbsolute mit der Zielposition 1500,0 gestartet. Wegen der Antwortzeiten (z. B. Zykluszeit des Anwenderprogramms usw.) kommt die Achse kurz zum Stillstand (siehe vergrößerter Ausschnitt). Wenn die Achse die neue Zielposition erreicht, wird dies über "Done_2" gemeldet.

② Eine aktive Aufgabe MC_MoveAbsolute wird von einer anderen Aufgabe MC_MoveAbsolute abgebrochen. Der Abbruch wird über "Abort_1" gemeldet. Die Achse wird dann mit der neuen Geschwindigkeit zur neuen Zielposition 1500,0 gefahren. Wenn die neue Zielposition erreicht ist, wird dies über "Done_2" gemeldet.

Übersteuerungsantwort

Die Aufgabe MC_MoveAbsolute kann von den folgenden Bewegungssteuerungsaufgaben abgebrochen werden:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

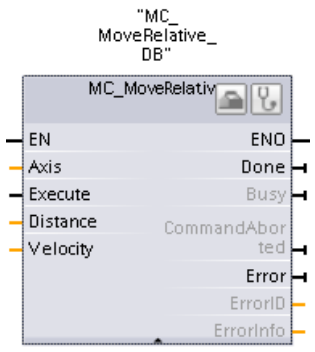
Die neue Aufgabe MC_MoveAbsolute bricht die folgenden aktiven Bewegungssteuerungsaufgaben ab:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

10.3.7.7 Anweisung MC_MoveRelative (Achse relativ positionieren)

Tabelle 10- 66 Anweisung MC_MoveRelative

KOP/FUP	SCL	Beschreibung

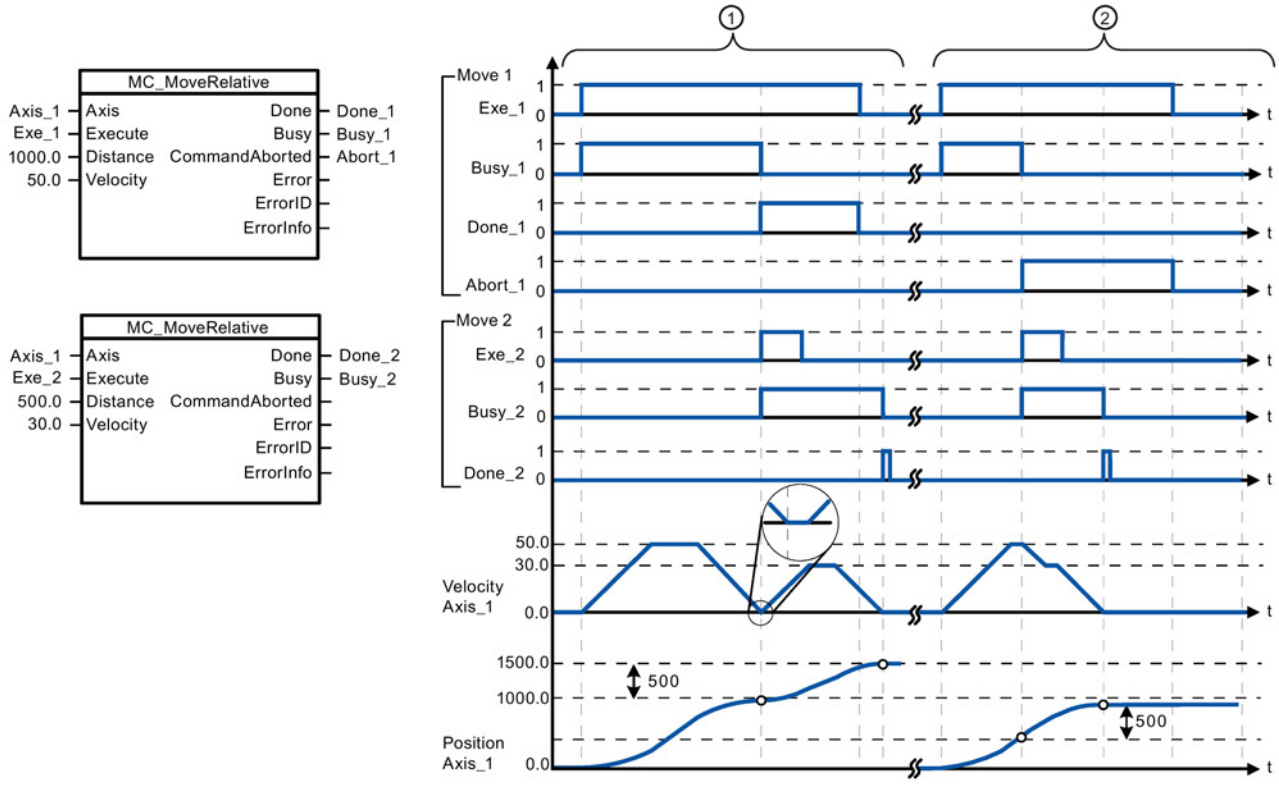
	`"MC_MoveRelative_DB"(`
	`Axis:=_multi_fb_in_,`
	`Execute:=_bool_in_,`
	`Distance:=_real_in_,`
	`Velocity:=_real_in_,`
	`Done=>_bool_out_,`
	`Busy=>_bool_out_,`
	`CommandAborted=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		Mit der Anweisung MC_MoveRelative starten Sie eine Positionierbewegung relativ zur Startposition.
		Um die Anweisung MC_MoveRelative zu verwenden, muss die Achse zunächst freigegeben werden.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_MoveRelative_DB" der Name des Instanz-DBs.

Tabelle 10- 67 Parameter für die MC_MoveRelative-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_1	Technologieobjekt "Achse"
Execute	IN	Bool	Starten der Aufgabe bei einer positiven Flanke (Standardwert: Falsch)
Distance	IN	Real	Verfahrweg für den Positioniervorgang (Standardwert: 0.0) Grenzwerte: $-1{,}0e^{12} \leq$ Distance $\leq 1{,}0e^{12}$
Velocity	IN	Real	Achsengeschwindigkeit (Standardwert: 10.0) Diese Geschwindigkeit wird wegen der konfigurierten Beschleunigung und Verzögerung und des zu fahrenden Wegs nicht immer erreicht. Grenzwerte: Start-/Stoppgeschwindigkeit ≤ Velocity ≤ Maximalgeschwindigkeit
Done	OUT	Bool	WAHR = Zielposition erreicht
Busy	OUT	Bool	WAHR = Die Aufgabe wird ausgeführt.
CommandAborted	OUT	Bool	WAHR = Während der Ausführung wurde die Aufgabe von einer anderen Aufgabe abgebrochen.
Error	OUT	Bool	WAHR = Während der Ausführung der Aufgabe trat ein Fehler auf. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt.
ErrorID	OUT	Word	Fehler-ID für Parameter "Error" (Standardwert: 0000)
ErrorInfo	OUT	Word	Fehlerinfo-ID für Parameter "ErrorID" (Standardwert: 0000)



Die folgenden Werte wurden im Konfigurationsfenster "Dynamik > Allgemein" konfiguriert: Beschleunigung = 10,0 und Verzögerung = 10,0

① Die Achse wird von einer Aufgabe MC_MoveRelative den Weg ("Distance") 1000,0 gefahren. Wenn die Achse die Zielposition erreicht, wird dies über "Done_1" gemeldet. Wenn "Done_1" = WAHR ist, wird eine andere Aufgabe MC_MoveRelative mit dem Verfahrweg 500,0 gestartet. Wegen der Antwortzeiten (z. B. Zykluszeit des Anwenderprogramms) kommt die Achse kurz zum Stillstand (siehe vergrößerter Ausschnitt). Wenn die Achse die neue Zielposition erreicht, wird dies über "Done_2" gemeldet.

② Eine aktive Aufgabe MC_MoveRelative wird von einer anderen Aufgabe MC_MoveRelative abgebrochen. Der Abbruch wird über "Abort_1" gemeldet. Die Achse wird dann mit der neuen Geschwindigkeit den neuen Weg ("Distance") 500,0 gefahren. Wenn die neue Zielposition erreicht ist, wird dies über "Done_2" gemeldet.

Übersteuerungsantwort

Die Aufgabe MC_MoveRelative kann von den folgenden Bewegungssteuerungsaufgaben abgebrochen werden:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

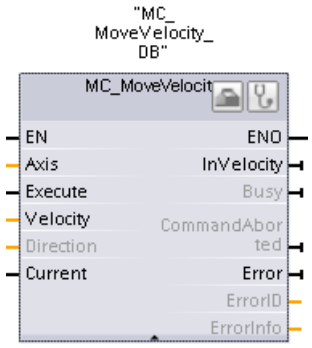
Die neue Aufgabe MC_MoveRelative bricht die folgenden aktiven Bewegungssteuerungsaufgaben ab:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

10.3.7.8 Anweisung MC_MoveVelocity (Achse mit vordefinierter Geschwindigkeit bewegen)

Tabelle 10- 68 Anweisung MC_MoveVelocity

KOP/FUP	SCL	Beschreibung

	`"MC_MoveVelocity_DB"(`
	`Axis:=_multi_fb_in_,`
	`Execute:=_bool_in_,`
	`Velocity:=_real_in_,`
	`Direction:=_int_in_,`
	`Current:=_bool_in_,`
	`InVelocity=>_bool_out_,`
	`Busy=>_bool_out_,`
	`CommandAborted=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		Mit der Anweisung MC_MoveVelocity bewegen Sie die Achse konstant mit der angegebenen Geschwindigkeit.
		Um die Anweisung MC_MoveVelocity zu verwenden, muss die Achse zunächst freigegeben werden.

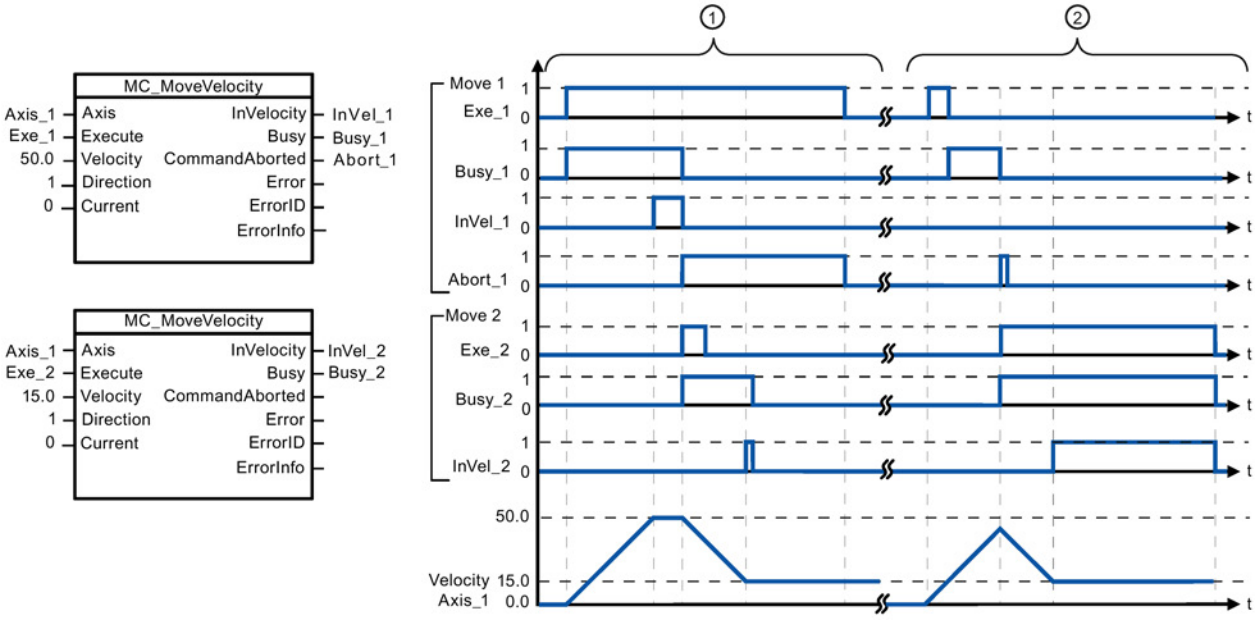
1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_MoveVelocity_DB" der Name des Instanz-DBs.

Tabelle 10- 69 Parameter für die MC_MoveVelocity-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_1	Technologieobjekt "Achse"
Execute	IN	Bool	Starten der Aufgabe bei einer positiven Flanke (Standardwert: Falsch)
Velocity	IN	Real	Geschwindigkeitsangabe für die Achsenbewegung (Standardwert: 10.0) Grenzwerte: Start-/Stoppgeschwindigkeit ≤ |Velocity| ≤ Maximalgeschwindigkeit (Velocity = 0,0 ist zulässig)
Direction	IN	Int	Richtungsangabe: • 0: Die Drehrichtung entspricht dem Vorzeichen des Werts im Parameter "Velocity" (Standardwert) • 1: Positive Drehrichtung (das Vorzeichen des Werts im Parameter "Velocity" wird ignoriert). • 2: Negative Drehrichtung (das Vorzeichen des Werts im Parameter "Velocity" wird ignoriert).

Parameter und Datentyp		Datentyp	Beschreibung
Current	IN	Bool	Aktuelle Geschwindigkeit beibehalten: • FALSCH: "Aktuelle Geschwindigkeit beibehalten" ist deaktiviert. Die Werte der Parameter "Velocity" und "Direction" werden verwendet. (Standardwert) • WAHR: "Aktuelle Geschwindigkeit beibehalten" ist aktiviert. Die Werte der Parameter "Velocity" und "Direction" werden nicht berücksichtigt. Wenn die Achse die Bewegung mit der aktuellen Geschwindigkeit wiederaufnimmt, gibt der Parameter "InVelocity" den Wert TRUE. aus.
InVelocity	OUT	Bool	WAHR: • Wenn "Current" = FALSE: Die im Parameter "Velocity" angegebene Geschwindigkeit wurde erreicht. • Wenn "Current" = TRUE: Die Achse fährt zur Startzeit mit der aktuellen Geschwindigkeit.
Busy	OUT	Bool	WAHR = Die Aufgabe wird ausgeführt.
CommandAborted	OUT	Bool	WAHR = Während der Ausführung wurde die Aufgabe von einer anderen Aufgabe abgebrochen.
Error	OUT	Bool	WAHR = Während der Ausführung der Aufgabe trat ein Fehler auf. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt.
ErrorID	OUT	Word	Fehler-ID für Parameter "Error" (Standardwert: 0000)
ErrorInfo	OUT	Word	Fehlerinfo-ID für Parameter "ErrorID" (Standardwert: 0000)



Die folgenden Werte wurden im Konfigurationsfenster "Dynamik > Allgemein" konfiguriert: Beschleunigung = 10,0 und Verzögerung = 10,0

① Eine aktive Aufgabe MC_MoveVelocity meldet über "InVel_1", dass die Zielgeschwindigkeit erreicht wurde. Sie wird dann von einer anderen Aufgabe MC_MoveVelocity abgebrochen. Der Abbruch wird über "Abort_1" gemeldet. Wenn die neue Zielgeschwindigkeit 15,0 erreicht ist, wird dies über "InVel_2" gemeldet. Die Achse bewegt sich dann mit der neuen konstanten Geschwindigkeit weiter.

② Eine aktive Aufgabe MC_MoveVelocity wird von einer anderen Aufgabe MC_MoveVelocity abgebrochen, bevor sie ihre Zielgeschwindigkeit erreicht. Der Abbruch wird über "Abort_1" gemeldet. Wenn die neue Zielgeschwindigkeit 15,0 erreicht ist, wird dies über "InVel_2" gemeldet. Die Achse bewegt sich dann mit der neuen konstanten Geschwindigkeit weiter.

Übersteuerungsantwort

Die Aufgabe MC_MoveVelocity kann von den folgenden Bewegungssteuerungsaufgaben abgebrochen werden:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

Die neue Aufgabe MC_MoveVelocity bricht die folgenden aktiven Bewegungssteuerungsaufgaben ab:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

Hinweis

Verhalten bei auf Null gesetzter Geschwindigkeit (Velocity = 0,0)

Eine Aufgabe MC_MoveVelocity mit "Velocity" = 0,0 (wie eine Aufgabe MC_Halt) bricht aktive Bewegungssteuerungsaufgaben ab und stoppt die Achse mit der konfigurierten Verzögerung. Wenn die Achse zum Stillstand kommt, gibt der Ausgangsparameter "InVelocity" mindestens einen Programmzyklus lang WAHR an.

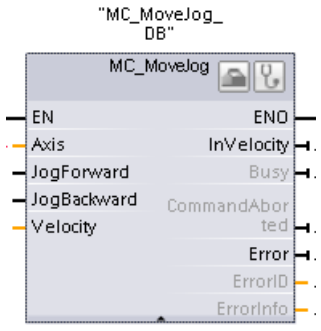
"Busy" gibt während des Verzögerungsvorgangs den Wert WAHR an und geht zusammen mit "InVelocity" nach FALSCH. Wenn der Parameter "Execute" = WAHR gesetzt ist, sind "InVelocity" und "Busy" als gespeichert gesetzt.

Wenn die Aufgabe MC_MoveVelocity gestartet wird, wird das Statusbit "SpeedCommand" im Technologieobjekt gesetzt. Das Statusbit "ConstantVelocity" wird beim Stillstand der Achse gesetzt. Beide Bits werden an die neue Situation angepasst, wenn eine neue Bewegungssteuerungsaufgabe gestartet wird.

10.3.7.9 Anweisung MC_MoveJog (Achse im Tippbetrieb bewegen)

Tabelle 10- 70 Anweisung MC_MoveJog

KOP/FUP	SCL	Beschreibung

	`"MC_MoveJog_DB"(`
	`  Axis:=_multi_fb_in_,`
	`  JogForward:=_bool_in_,`
	`  JogBackward:=_bool_in_,`
	`  Velocity:=_real_in_,`
	`  InVelocity=>_bool_out_,`
	`  Busy=>_bool_out_,`
	`  CommandAborted=>_bool_out_,`
	`  Error=>_bool_out_,`
	`  ErrorID=>_word_out_,`
	`  ErrorInfo=>_word_out_);`
		Mit der Anweisung MC_MoveJog bewegen Sie die Achse konstant mit der angegebenen Geschwindigkeit im Tippbetrieb. Diese Anweisung dient üblicherweise zu Test- und Inbetriebnahmezwecken.
		Um die Anweisung MC_MoveJog zu verwenden, muss die Achse zunächst freigegeben werden.

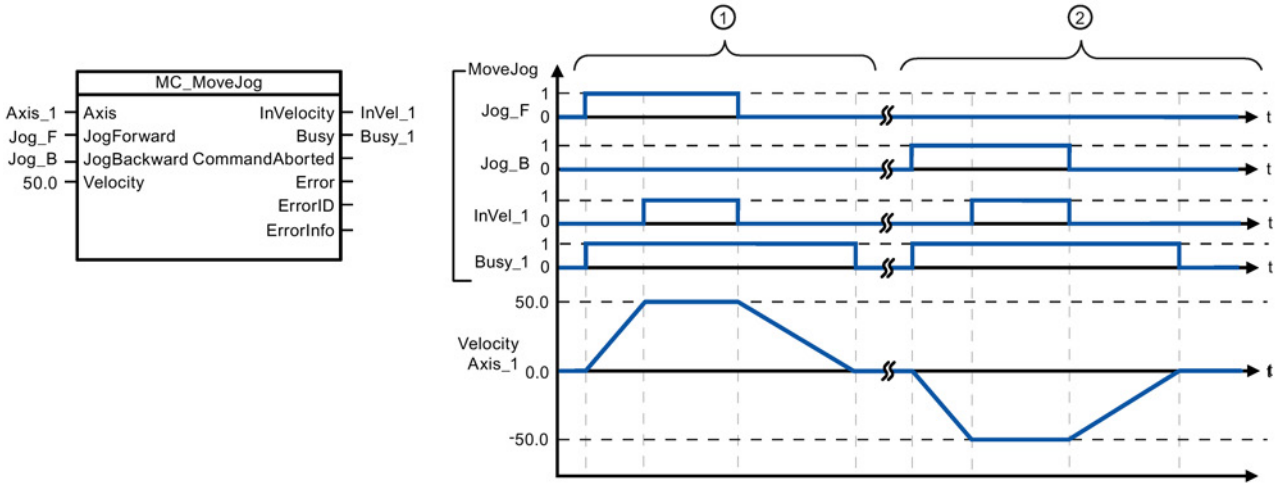
1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_MoveJog_DB" der Name des Instanz-DBs.

Tabelle 10- 71 Parameter für die MC_MoveJog-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_1	Technologieobjekt "Achse"
JogForward[1]	IN	Bool	Solange der Parameter WAHR ist, bewegt sich die Achse mit der im Parameter "Velocity" angegebenen Geschwindigkeit in positiver Richtung. Das Vorzeichen des Werts im Parameter "Velocity" wird ignoriert. (Standardwert: Falsch)
JogBackward[1]	IN	Bool	Solange der Parameter WAHR ist, bewegt sich die Achse mit der im Parameter "Velocity" angegebenen Geschwindigkeit in negativer Richtung. Das Vorzeichen des Werts im Parameter "Velocity" wird ignoriert. (Standardwert: Falsch)
Velocity	IN	Real	Voreingestellte Geschwindigkeit für den Tippbetrieb (Standardwert: 10.0) Grenzwerte: Start-/Stoppgeschwindigkeit ≤ |Velocity| ≤ Maximalgeschwindigkeit
InVelocity	OUT	Bool	WAHR = Die im Parameter "Velocity angegebene Geschwindigkeit wurde erreicht.
Busy	OUT	Bool	WAHR = Die Aufgabe wird ausgeführt.
CommandAborted	OUT	Bool	WAHR = Während der Ausführung wurde die Aufgabe von einer anderen Aufgabe abgebrochen.
Error	OUT	Bool	WAHR = Während der Ausführung der Aufgabe trat ein Fehler auf. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt.
ErrorID	OUT	Word	Fehler-ID für Parameter "Error" (Standardwert: 0000)
ErrorInfo	OUT	Word	Fehlerinfo-ID für Parameter "ErrorID" (Standardwert: 0000)

[1] Wenn beide Parameter JogForward und JogBackward gleichzeitig WAHR sind, stoppt die Achse mit der konfigurierten Verzögerung. Ein Fehler wird in den Parametern "Error", "ErrorID" und "ErrorInfo" gemeldet.



Die folgenden Werte wurden im Konfigurationsfenster "Dynamik > Allgemein" konfiguriert: Beschleunigung = 10,0 und Verzögerung = 5,0

① Die Achse wird im Tippbetrieb über "Jog_F" in positiver Richtung bewegt. Wenn die Zielgeschwindigkeit 50,0 erreicht ist, wird dies über "InVelo_1" gemeldet. Die Achse bremst erneut bis zum Stillstand, nachdem Jog_F zurückgesetzt wird.

② Die Achse wird im Tippbetrieb über "Jog_B" in negativer Richtung bewegt. Wenn die Zielgeschwindigkeit 50,0 erreicht ist, wird dies über "InVelo_1" gemeldet. Die Achse bremst erneut bis zum Stillstand, nachdem Jog_B zurückgesetzt wird.

Übersteuerungsantwort

Die Aufgabe MC_MoveJog kann von den folgenden Bewegungssteuerungsaufgaben abgebrochen werden:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

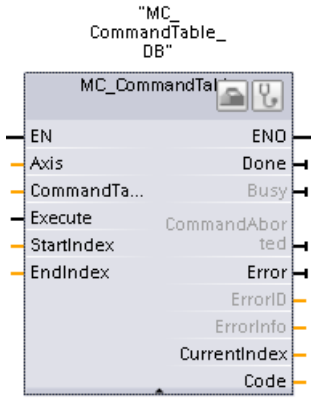
Die neue Aufgabe MC_MoveJog bricht die folgenden aktiven Bewegungssteuerungsaufgaben ab:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog

10.3.7.10 Anweisung MC_CommandTable (Achssteuerungsbefehle als Bewegungsfolge ausführen)

Tabelle 10- 72 Anweisung MC_CommandTable

KOP/FUP	SCL	Beschreibung

	`"MC_CommandTable_DB"(`
	`Axis:=_multi_fb_in_,`
	`CommandTable:=_multi_fb_in_,`
	`Execute:=_bool_in_,`
	`StartIndex:=_uint_in_,`
	`EndIndex:=_uint_in_,`
	`Done=>_bool_out_,`
	`Busy=>_bool_out_,`
	`CommandAborted=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_,`
	`CurrentIndex=>_uint_out_,`
	`Code=>_word_out_);`
		Führt eine Reihe einzelner Bewegungen für eine Motorsteuerungsachse aus, die zu einer Bewegungsfolge verbunden werden können.
		Einzelne Bewegungen werden in einer Befehlstabelle eines Technologieobjekts für die Impulsfolge konfiguriert (TO_CommandTable_PTO).

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

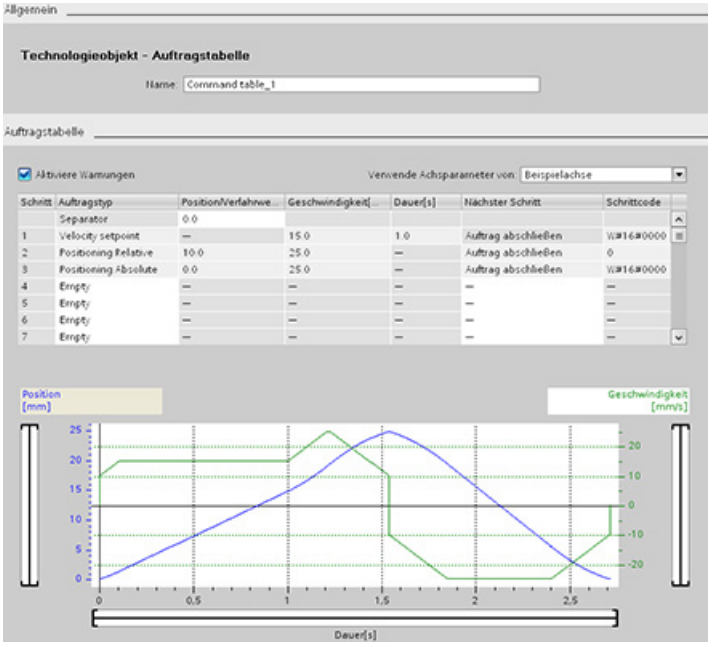
2 Im SCL-Beispiel ist "MC_CommandTable_DB" der Name des Instanz-DBs.

Tabelle 10- 73 Parameter für die MC_CommandTable-Anweisung

Parameter und Datentyp		Datentyp	Anfangswert	Beschreibung
Axis	IN	TO_Axis_1	-	Technologieobjekt "Achse"
Table	IN	TO_CommandTable_1	-	Technologieobjekt "Befehlstabelle"
Execute	IN	Bool	FALSCH	Auftrag bei steigender Flanke starten
StartIndex	IN	Int	1	Verarbeitung der Befehlstabelle mit diesem Schritt starten Grenzwerte: 1 ≤ **StartIndex** ≤ EndIndex
EndIndex	IN	Int	32	Verarbeitung der Befehlstabelle mit diesem Schritt beenden Grenzwerte: StartIndex ≤ **EndIndex** ≤ 32
Done	OUT	Bool	FALSCH	Verarbeitung von MC_CommandTable erfolgreich ausgeführt
Busy	OUT	Bool	FALSCH	Operation in Bearbeitung
CommandAborted	OUT	Bool	FALSCH	Die Aufgabe wurde während der Bearbeitung von einer anderen Aufgabe abgebrochen.
Error	OUT	Bool	FALSCH	Ein Fehler ist während der Bearbeitung aufgetreten. Die Ursache wird von den Parametern ErrorID und ErrorInfo. angegeben
ErrorID	OUT	Word	16#0000	Fehler-ID
ErrorInfo	OUT	Word	16#0000	Fehlerinformation

Parameter und Datentyp		Datentyp	Anfangs-wert	Beschreibung
Step	OUT	Int	0	Schritt wird gerade bearbeitet
Code	OUT	Word	16#0000	Anwenderdefinierte Kennung des in Bearbeitung befindlichen Schritts

Die gewünschte Bewegungsfolge können Sie im Konfigurationsfenster "Befehlstabelle" erstellen und das Ergebnis in der grafischen Darstellung im Kurvendiagramm prüfen.



Sie können die Befehlstypen auswählen, die für die Verarbeitung der Befehlstabelle verwendet werden sollen. Bis zu 32 Aufträge können eingegeben werden. Die Befehle werden der Reihe nach verarbeitet.

Tabelle 10- 74 MC_CommandTable-Befehlstypen

Befehlstyp	Beschreibung
Empty	Dieser Befehl dient als Platzhalter für jeden hinzuzufügenden Befehl. Der leere Eintrag wird bei der Verarbeitung der Befehlstabelle ignoriert.
Halt	Der Befehl hält die Achse an. Hinweis: Der Befehl wird nur nach einem Befehl "Velocity setpoint" durchgeführt.
Positioning Relative	Der Befehl positioniert die Achse basierend auf der Distanz. Er bewegt die Achse um die angegebene Distanz und Geschwindigkeit.
Positioning Absolute	Der Befehl positioniert die Achse basierend auf der Lage. Er bewegt die Achse mit der vorgegebenen Geschwindigkeit an die angegebene Position.
Velocity setpoint	Der Befehl bewegt die Achse mit der angegebenen Geschwindigkeit.
Wait	Der Befehl wartet, bis der angegebene Zeitraum abgelaufen ist. "Wait" stoppt keine aktive Verfahrbewegung.
Separator	Der Befehl fügt eine Trennlinie ("Separator") oberhalb der ausgewählten Linie ein. Die Trennlinie ermöglicht die Definition mehrerer Profile in einer Befehlstabelle.

Voraussetzungen für die Ausführung von MC_CommandTable:

- Das Technologieobjekt TO_Axis_PTO V2.0 muss ordnungsgemäß konfiguriert sein.
- Das Technologieobjekt TO_CommandTable_PTO muss korrekt konfiguriert sein.
- Die Achse muss freigegeben sein.

Übersteuerungsantwort

Die Aufgabe MC_CommandTable kann von den folgenden Bewegungssteuerungsaufgaben abgebrochen werden:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog
- MC_CommandTable

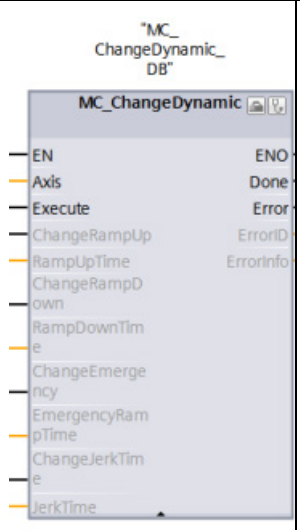
Die neue Aufgabe MC_CommandTable bricht die folgenden aktiven Bewegungssteuerungsaufgaben ab:

- MC_Home Mode = 3
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative
- MC_MoveVelocity
- MC_MoveJog
- MC_CommandTable
- Der aktuelle Bewegungssteuerungsauftrag mit dem Start des ersten Befehls "Positioning Relative", "Positioning Absolute", "Velocity setpoint" oder "Halt"

10.3.7.11 Anweisung MC_ChangeDynamic (Dynamikeinstellungen der Achse ändern)

Tabelle 10- 75 Anweisung MC_ChangeDynamic

KOP/FUP	SCL	Beschreibung

	`"MC_ChangeDynamic_DB"(`
	`Execute:=_bool_in_,`
	`ChangeRampUp:=_bool_in_,`
	`RampUpTime:=_real_in_,`
	`ChangeRampDown:=_bool_in_,`
	`RampDownTime:=_real_in_,`
	`ChangeEmergency:=_bool_in_,`
	`EmergencyRampTime:=_real_in_,`
	`ChangeJerkTime:=_bool_in_,`
	`JerkTime:=_real_in_,`
	`Done=>_bool_out_,`
	`Error=>_bool_out_,`
	`ErrorID=>_word_out_,`
	`ErrorInfo=>_word_out_);`
		Ändert die Dynamikeinstellungen einer Bewegungssteuerungsachse:
		• Wert der Hochlaufzeit (Beschleunigung) ändern
		• Wert der Rücklaufzeit (Verzögerung) ändern
		• Wert der Not-Aus-Rücklaufzeit (Not-Aus-Verzögerung) ändern
		• Wert der Glättungszeit (Ruck) ändern

1 STEP 7 erstellt den DB automatisch, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_ChangeDynamic_DB" der Name des Instanz-DBs.

Tabelle 10- 76 Parameter für die MC_ChangeDynamic-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
Axis	IN	TO_Axis_1	Technologieobjekt "Achse"
Execute	IN	Bool	Starten des Befehls bei einer positiven Flanke. Standardwert: FALSE
ChangeRampUp	IN	Bool	TRUE = Hochlaufzeit in Übereinstimmung mit dem Eingangsparameter "RampUpTime" ändern. Standardwert: FALSE
RampUpTime	IN	Real	Zeit (in Sekunden), während der ohne Ruckbegrenzung vom Stillstand zur konfigurierten Maximalgeschwindigkeit beschleunigt werden soll. Standardwert: 5.00 Die Änderung beeinflusst die Variable <Achsenname>. Config.DynamicDefaults.Acceleration. Die Wirksamkeit der Änderung wird in der Beschreibung dieser Variablen gezeigt.
ChangeRampDown	IN	Bool	TRUE = Rücklaufzeit in Übereinstimmung mit dem Eingangsparameter "RampDownTime" ändern. Standardwert: FALSE
RampDownTime	IN	Real	Zeit (in Sekunden), während der die Achse ohne Ruckbegrenzung von der konfigurierten Maximalgeschwindigkeit bis zum Stillstand verzögert werden soll. Standardwert: 5.00 Die Änderung beeinflusst die Variable <Achsenname>. Config.DynamicDefaults.Deceleration. Die Wirksamkeit der Änderung wird in der Beschreibung dieser Variablen gezeigt.

Parameter und Datentyp		Datentyp	Beschreibung
ChangeEmergency	IN	Bool	TRUE = Not-Aus-Rücklaufzeit in Übereinstimmung mit dem Eingangsparameter "EmergencyRampTime" ändern. Standardwert: FALSE
EmergencyRampTime	IN	Real	Zeit (in Sekunden), während der die Achse im Not-Aus-Modus ohne Ruckbegrenzung von der konfigurierten Maximalgeschwindigkeit bis zum Stillstand verzögert werden soll. Standardwert: 2.00 Die Änderung beeinflusst die Variable <Achsenname>. Config.DynamicDefaults.EmergencyDeceleration. Die Wirksamkeit der Änderung wird in der Beschreibung dieser Variablen gezeigt.
ChangeJerkTime	IN	Bool	TRUE = Glättungszeit in Übereinstimmung mit dem Eingangsparameter "JerkTime" ändern. Standardwert: FALSE
JerkTime	IN	Real	Glättungszeit (in Sekunden) für die Beschleunigungs- und Verzögerungsrampen der Achse. Standardwert: 0.25 Die Änderung beeinflusst die Variable <Achsenname>. Config.DynamicDefaults.Jerk. Die Wirksamkeit der Änderung wird in der Beschreibung dieser Variablen gezeigt.
Done	OUT	Bool	TRUE = Die geänderten Werte wurden in den Technologiedatenbaustein geschrieben. Die Beschreibung der Variablen wird angezeigt, wenn die Änderung wirksam wird. Standardwert: FALSE
Error	OUT	Bool	TRUE = Während der Ausführung des Befehls trat ein Fehler auf. Die Fehlerursache ist in den Parametern ErrorID und ErrorInfo hinterlegt. Standardwert: FALSE
ErrorID	OUT	Word	Fehler-ID Standardwert: 16#0000
ErrorInfo	IN	Word	Fehlerinformationen. Standardwert: 16#0000

Voraussetzungen für die Ausführung von MC_ ChangeDynamic:

- Das Technologieobjekt TO_Axis_PTO V2.0 muss ordnungsgemäß konfiguriert sein.
- Die Achse muss freigegeben sein.

Übersteuerungsantwort

Ein Befehl MC_ChangeDynamic kann von keinem anderen Bewegungssteuerungsbefehl abgebrochen werden.

Der neue Befehl MC_ChangeDynamic bricht keine aktiven Bewegungssteuerungsaufgaben ab.

Hinweis

Für die Eingangsparameter "RampUpTime", "RampDownTime", "EmergencyRampTime" und "RoundingOffTime" können Werte angegeben werden, die dazu führen, dass die resultierenden Achsenparameter "Beschleunigung", "Verzögerung", "Not-Aus-Verzögerung" und "Ruck" außerhalb der zulässigen Grenzwerte liegen.

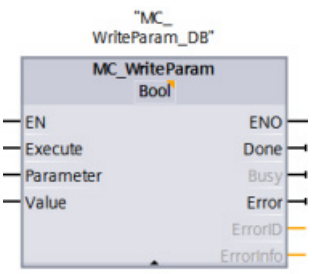
Stellen Sie sicher, dass Sie die Parameter MC_ChangeDynamic innerhalb der Grenzwerte der Dynamikkonfiguration des Technologieobjekts "Achse" halten.

10.3.7.12 Anweisung MC_WriteParam (Parameter des Technologieobjekts schreiben)

Mit der Anweisung MC_WriteParam können Sie bestimmte Parameter schreiben, um die Funktionalität der Achse im Anwenderprogramm zu ändern.

Tabelle 10- 77 Anweisung MC_WriteParam

KOP/FUP	SCL	Beschreibung

	`"MC_WriteParam_DB"(`
	`  Parameter:=_variant_in_,`
	`  Value:=_variant_in_,`
	`  Execute:=_bool_in_,`
	`  Done:=_bool_out_,`
	`  Error:=_real_out_,`
	`  ErrorID:=_word_out_,`
	`  ErrorInfo:=_word_out_);`
		Mit der Anweisung MC_WriteParam können Sie in öffentliche Parameter schreiben (z. B. Beschleunigung und Werte in Anwender-DBs).

1 STEP 7 erstellt den DB automatisch, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_WriteParam_DB" der Name des Instanz-DBs.

Sie können in öffentliche Parameter schreiben. Sie können nicht in "MotionStatus" und "StatusBits" schreiben. Die folgende Tabelle zeigt die gültigen Parameter:

Name des beschreibbaren Parameters	Name des beschreibbaren Parameters
Actor.InverseDirection	DynamicDefaults.Acceleration
Actor.DirectionMode	DynamicDefaults.Deceleration
Actor.DriveParameter.PulsesPerDriveRevolution	DynamicDefaults.Jerk
Sensor[1].ActiveHoming.Mode	DynamicDefaults.EmergencyDeceleration
Sensor[1].ActiveHoming.SideInput	PositionLimitsHW.Active
Sensor[1].ActiveHoming.Offset	PositionLimitsHW.MaxSwitchedLevel
Sensor[1].ActiveHoming.SwitchedLevel	PositionLimitsHW.MinSwitchedLevel
Sensor[1].PassiveHoming.Mode	PositionLimitsSW.Active
Sensor[1].PassiveHoming.SideInput	PositionLimitsSW.MinPosition
Sensor[1].PassiveHoming.SwitchedLevel	PositionLimitsSW.MaxPosition
Units.LengthUnit	Homing.AutoReversal
Mechanics.LeadScrew	Homing.ApproachDirection
DynamicLimits.MinVelocity	Homing.ApproachVelocity
DynamicLimits.MaxVelocity	Homing.ReferencingVelocity

Tabelle 10- 78 Parameter für die Anweisung MC_WriteParam

Parameter und Datentyp		Datentyp	Beschreibung
PARAMNAME	IN	Variant	Name des Parameters, in den der Wert geschrieben wird
VALUE	IN	Variant	Wert, der in den zugewiesenen Parameter geschrieben werden soll
EXECUTE	IN	Bool	Anweisung starten. Standardwert: FALSE
DONE	OUT	Bool	Wert wurde geschrieben. Standardwert: FALSE
BUSY	OUT	Bool	Bei TRUE wird die Anweisung gerade ausgeführt. Standardwert: FALSE
ERROR	OUT	Real	Bei TRUE ist ein Fehler aufgetreten. Standardwert: FALSE
ERRORID	OUT	Word	ID des Fehlers
ERRORINFO	OUT	Word	Zusatzinformation zur ERRORID

Tabelle 10- 79 Bedingungscodes für ERRORID und ERRORINFO

ERRORID (W#16#...)	ERRORINFO (W#16#...)	Beschreibung
0	0	Parameter TO-DB einer Achse wurde erfolgreich geändert
8410[1]	0028[1]	Ungültiger Parameter (Achsparameter TO-DB mit falscher Länge)
8410[1]	0029[1]	Ungültiger Parameter (kein Achsparameter TO-DB)
8410[1]	002B[1]	Ungültiger Parameter (Achsparameter TO-DB schreibgeschützt)
8410[1]	002C[1]	Gültiger Parameter, aber Achse nicht deaktiviert
Config Error[2]	Config Error[2]	Gültiger Parameter (öffentlicher Achsparameter TO-DB schreibgeschützt) nicht im zulässigen Bereich
Config Error[3]	Config Error[3]	Gültiger Parameter (öffentlicher Achsparameter TO-DB) nicht im zulässigen Bereich

[1] Fehler in MC_WriteParam

[2] Fehler in MC_Power

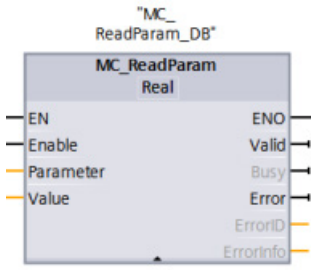
[3] Fehler in MC_Power und MC_MoveXXX oder MC_CommandTable

10.3.7.13 Anweisung MC_ReadParam (Parameter eines Technologieobjekts lesen)

Mit der Anweisung MC_ReadParam lesen Sie eine ausgewählte Anzahl von Parametern, die die aktuelle Position, Geschwindigkeit usw. der im Eingang "Axis" definierten Achse angeben.

Tabelle 10- 80 Anweisung MC_ReadParam

KOP/FUP	SCL	Beschreibung

	`"MC_ReadParam_DB"(`
	`Enable:=_bool_in_,`
	`Parameter:=_variant_in_,`
	`Value:=_variant_in_out_,`
	`Valid:=_bool_out_,`
	`Busy:=_bool_out_,`
	`Error:=_real_out_,`
	`ErrorID:=_word_out_,`
	`ErrorInfo:=_word_out_);`
		Mit der Anweisung MC_ReadParam lesen Sie einzelne Statuswerte unabhängig vom Steuerpunkt im Zyklus.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

2 Im SCL-Beispiel ist "MC_ReadParam_DB" der Name des Instanz-DBs.

Die Anweisung MC_ReadParam funktioniert bei Freigabeverhalten. Solange der Eingang "Enable" wahr ist, liest die Anweisung den angegebenen "Parameter" in den Speicherort "Value".

Der "Position"-Wert der Achse "MotionStatus" wird basierend auf dem aktuellen HSC-Wert an jedem Steuerpunkt im Zyklus aktualisiert.

Der "Velocity"-Wert der Achse "MotionStatus" ist die Befehlsgeschwindigkeit am Ende des aktuellen Segments (wird alle ~10 ms aktualisiert). Die Anweisung MC_ReadParam kann auch diesen Wert lesen.

Tritt ein Fehler auf, geht die Anweisung in einen Fehlerzustand, der nur durch eine neue steigende Flanke am Eingang "Enable" zurückgesetzt werden kann.

Tabelle 10- 81 Parameter für die MC_ReadParam-Anweisung

Parameter und Datentyp		Datentyp	Beschreibung
ENABLE	IN	Bool	Start der Anweisung. Standardwert: FALSE
PARAMETER	IN	Variant	Pointer auf den zu lesenden TO-Parameter
VALID	OUT	Bool	Wenn TRUE, wurde der Wert gelesen. Standardwert: FALSE
BUSY	OUT	Bool	Wenn TRUE, ist die Anweisung in Betrieb. Standardwert: FALSE
ERROR	OUT	Real	Wenn TRUE, ist ein Fehler aufgetreten. Standardwert: FALSE
ERRORID	OUT	Word	ID des Fehlers. Standardwert: 0
ERRORINFO	OUT	Word	Zugehörige Informationen zu ERRORID. Standardwert: 0
VALUE	INOUT	Variant	Pointer auf den Speicherort, an dem der gelesene Wert gespeichert ist

Tabelle 10- 82 Bedingungscodes für ERRORID und ERRORINFO

ERRORID (W#16#...)	ERRORINFO (W#16#...)	Beschreibung
0	0	Parameter erfolgreich gelesen
8410	0028	Ungültiger Parameter (falsche Länge)
8410	0029	Ungültiger Parameter (kein TO-DB)
8410	0030	Ungültiger Parameter (nicht lesbar)
8411	0032	Ungültiger Parameter (falscher Wert)

TO-Parameter

Die Achse "MotionStatus" besteht aus vier Werten. Überwachen Sie diese Werte auf Veränderungen, die während der Ausführung des Programms gelesen werden können:

Variablenname	Datentyp	Lesbar über MC_ReadParam
MotionStatus:	Struktur	Nein
• Position	REAL	Ja
• Geschwindigkeit	REAL	Ja
• Entfernung	REAL	Ja
• Zielposition	REAL	Ja

10.3.8 Aktive Befehle überwachen

10.3.8.1 Bewegungssteuerungsanweisungen mit einem Ausgangsparameter "Done" überwachen

Bewegungssteuerungsanweisungen mit dem Ausgangsparameter "Done" werden vom Eingangsparameter "Execute" gestartet und haben einen definierten Abschluss (z. B. bei der Bewegungssteuerungsanweisung "MC_Home": Referenzpunktfahrt war erfolgreich). Die Aufgabe ist beendet und die Achse ist im Stillstand.

- Der Ausgangsparameter "Done" meldet den Wert WAHR, wenn die Aufgabe erfolgreich durchgeführt wurde.
- Die Ausgangsparameter "Busy", "CommandAborted" und "Error" melden, dass die Aufgabe noch verarbeitet wird, abgebrochen wurde oder dass ein Fehler anliegt. Die Bewegungssteuerungsanweisung "MC_Reset" kann nicht abgebrochen werden, sie hat deshalb keinen Ausgangsparameter "CommandAborted".
  - Während der Verarbeitung der Bewegungssteuerungsaufgabe meldet der Ausgangsparameter "Busy" den Wert WAHR. Wurde die Aufgabe beendet, abgebrochen oder durch einen Fehler gestoppt, ändert sich der Wert des Ausgangsparameters "Busy" in FALSCH. Diese Änderung tritt unabhängig vom Signal am Eingangsparameter "Execute" ein.
  - Die Ausgangsparameter "Done", "CommandAborted" und "Error" melden mindestens einen Zyklus lang den Wert WAHR. Diese Statusmeldungen werden gespeichert gehalten, während der Eingangsparameter "Execute" auf WAHR ist.

Die Aufgaben der folgenden Bewegungssteuerungsanweisungen haben einen definierten Abschluss:

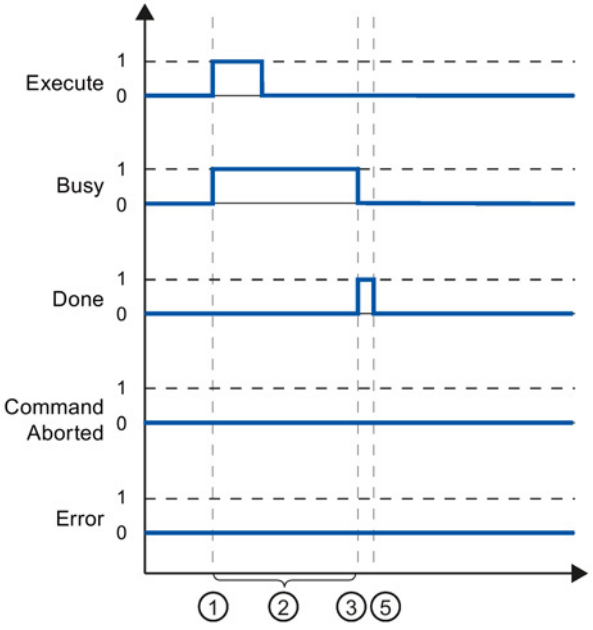
- MC_Reset
- MC_Home
- MC_Halt
- MC_MoveAbsolute
- MC_MoveRelative

Das Verhalten der Statusbits wird im Folgenden für verschiedene Beispielsituationen aufgeführt.

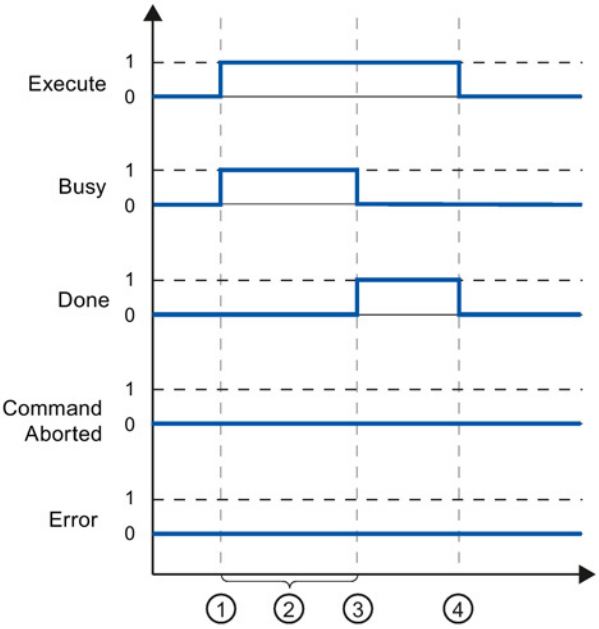
- Das erste Beispiel zeigt das Verhalten der Achse bei einer abgeschlossenen Aufgabe. Wenn die Bewegungssteuerungsaufgabe zum Zeitpunkt des Abschlusses vollständig durchgeführt wurde, wird dies durch den Wert WAHR am Ausgangsparameter "Done" gemeldet. Der Signalzustand des Eingangsparameters "Execute" beeinflusst die Anzeigedauer im Ausgangsparameter "Done".
- Das zweite Beispiel zeigt das Verhalten der Achse bei einer abgebrochenen Aufgabe. Wenn die Bewegungssteuerungsaufgabe während der Ausführung abgebrochen wird, wird dies durch den Wert WAHR am Ausgangsparameter "CommandAborted" gemeldet. Der Signalzustand des Eingangsparameters "Execute" beeinflusst die Anzeigedauer im Ausgangsparameter "CommandAborted".

- Das dritte Beispiel zeigt das Verhalten der Achse im Fehlerfall. Tritt während der Ausführung der Bewegungssteuerungsaufgabe ein Fehler auf, wird dies durch den Wert WAHR am Ausgangsparameter "Error" gemeldet. Der Signalzustand des Eingangsparameters "Execute" beeinflusst die Anzeigedauer im Ausgangsparameter "Error".

Tabelle 10- 83 Beispiel 1 - Vollständige Durchführung der Aufgabe



Wenn "Execute" = FALSCH während der Verarbeitung der Aufgabe



Wenn "Execute" = FALSCH nach Abschluss der Aufgabe

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "Execute" gestartet. Je nach Programmierung kann "Execute" während der Ausführung der Aufgabe auf den Wert FALSCH zurückgesetzt werden oder der Wert WAHR kann bis nach Abschluss der Aufgabe gehalten werden.

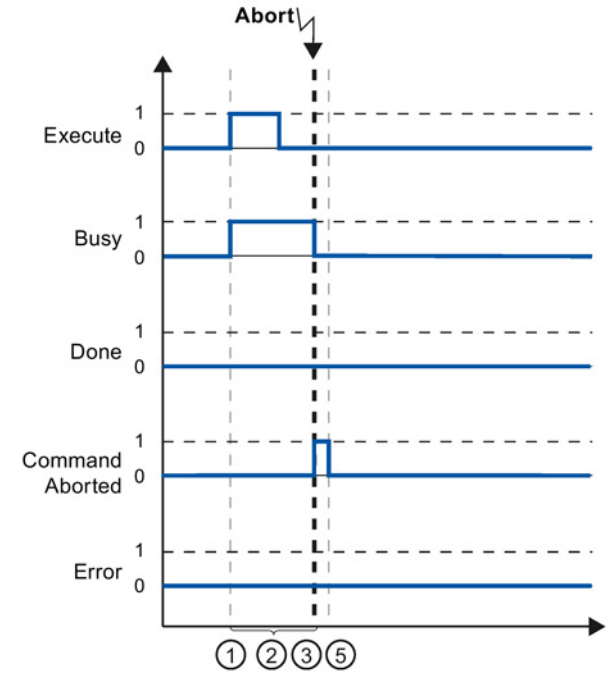
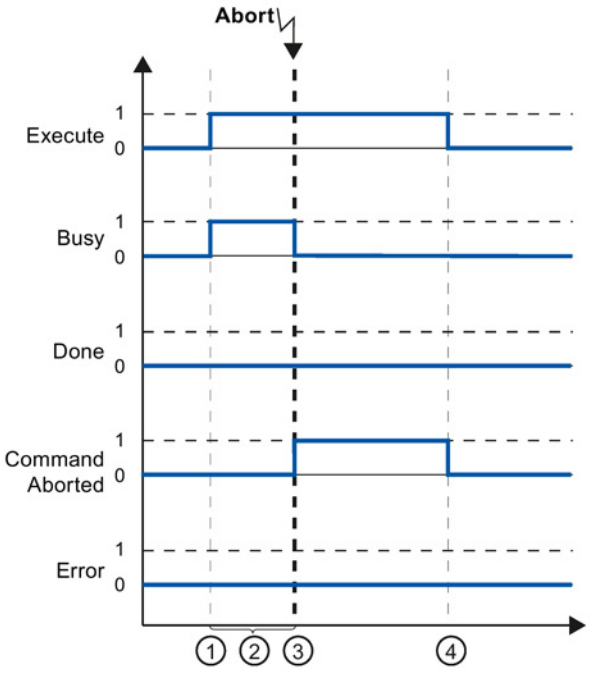
② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Bei Abschluss der Aufgabe (z. B. der Bewegungssteuerungsanweisung "MC_Home": Referenzpunktfahrt war erfolgreich), geht der Ausgangsparameter "Busy" auf FALSCH und "Done" auf WAHR.

④ Wenn "Execute" den Wert WAHR bis nach Abschluss der Aufgabe beibehält, dann bleibt "Done" ebenfalls WAHR und wechselt seinen Wert zusammen mit "Execute" in FALSCH.

⑤ Wenn "Execute" vor Abschluss der Aufgabe auf FALSCH gesetzt wurde, meldet "Done" den Wert WAHR nur einen Zyklus lang.

Tabelle 10- 84 Beispiel 2 - Abbruch der Aufgabe

	
Wenn "Execute" = FALSCH vor Abbruch der Aufgabe	Wenn "Execute" = FALSCH nach Abbruch der Aufgabe

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "Execute" gestartet. Je nach Programmierung kann "Execute" während der Ausführung der Aufgabe auf den Wert FALSCH zurückgesetzt werden oder der Wert WAHR kann bis nach Abschluss der Aufgabe gehalten werden.

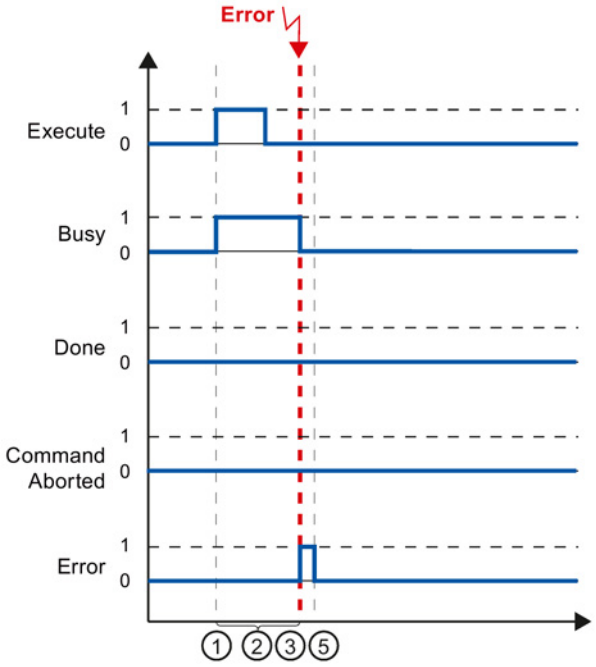
② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Die Aufgabe wird während ihrer Ausführung von einer anderen Bewegungssteuerungsaufgabe abgebrochen. Wenn die Aufgabe abgebrochen wird, geht der Ausgangsparameter "Busy" auf FALSCH und "CommandAborted" auf WAHR.

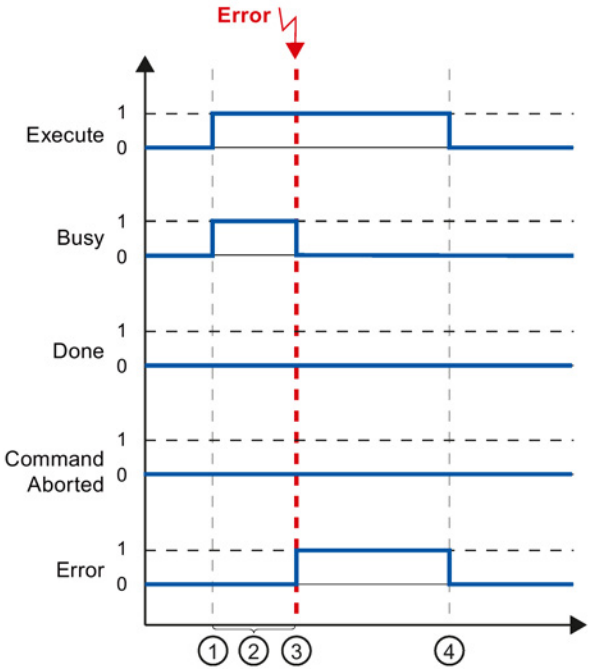
④ Wenn "Execute" den Wert WAHR bis nach Abbruch der Aufgabe beibehält, dann bleibt "CommandAborted" ebenfalls WAHR und wechselt seinen Wert zusammen mit "Execute" in FALSCH.

⑤ Wenn "Execute" vor Abbruch der Aufgabe auf FALSCH gesetzt wurde, meldet "CommandAborted" den Wert WAHR nur einen Zyklus lang.

Tabelle 10- 85 Beispiel 3 - Fehler während der Ausführung der Aufgabe



Wenn "Execute" = FALSCH vor Auftreten des Fehlers



Wenn "Execute" = FALSCH nach Auftreten des Fehlers

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "Execute" gestartet. Je nach Programmierung kann "Execute" während der Ausführung der Aufgabe auf den Wert FALSCH zurückgesetzt werden oder der Wert WAHR kann bis nach Abschluss der Aufgabe gehalten werden.

② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Während der Ausführung der Aufgabe ist ein Fehler aufgetreten. Wenn der Fehler auftritt, geht der Ausgangsparameter "Busy" auf FALSCH und "Error" auf WAHR.

④ Wenn "Execute" den Wert WAHR bis nach Auftreten des Fehlers beibehält, dann bleibt "Error" ebenfalls WAHR und wechselt seinen Wert erst zusammen mit "Execute" in FALSCH.

⑤ Wenn "Execute" vor Auftreten des Fehlers auf FALSCH gesetzt wurde, meldet "Error" den Wert WAHR nur einen Zyklus lang.

10.3.8.2 Anweisung MC_Velocity überwachen

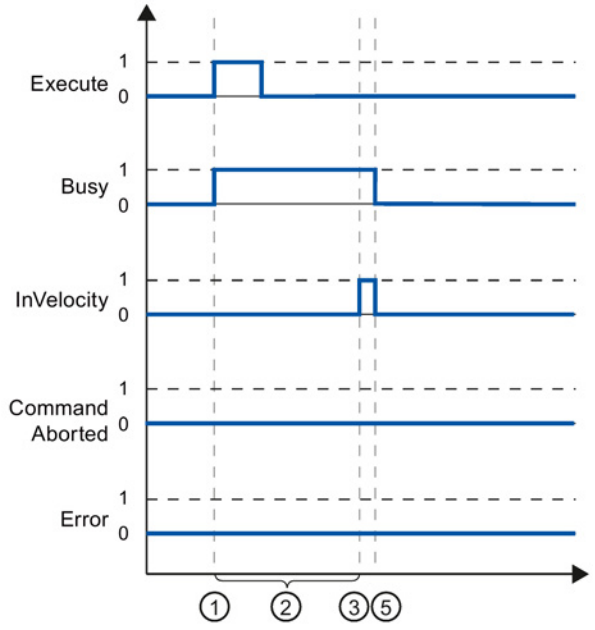
Die Aufgaben der Bewegungssteuerungsanweisung "MC_MoveVelocity" lösen eine Bewegung mit der vorgegebenen Geschwindigkeit aus.

- Die Aufgaben der Bewegungssteuerungsanweisung "MC_MoveVelocity" haben kein definiertes Ende. Das Ziel der Aufgabe ist erfüllt, wenn die parametrierte Geschwindigkeit zum ersten Mal erreicht ist und die Achse mit konstanter Geschwindigkeit fährt. Wenn die parametrierte Geschwindigkeit erreicht ist, wird dies durch den Wert WAHR am Ausgangsparameter "InVelocity" gemeldet.
- Die Aufgabe ist beendet, wenn die parametrierte Geschwindigkeit erreicht ist und der Eingangsparameter "Execute" auf den Wert FALSCH gesetzt wurde. Die Achsbewegung ist jedoch mit dem Abschluss der Aufgabe noch nicht beendet. Beispiel: Die Achsbewegung kann mit der Bewegungssteuerungsaufgabe "MC_Halt" gestoppt werden.
- Die Ausgangsparameter "Busy", "CommandAborted" und "Error" melden, dass die Aufgabe noch verarbeitet wird, abgebrochen wurde oder dass ein Fehler anliegt.
  - Während der Ausführung der Bewegungssteuerungsaufgabe meldet der Ausgangsparameter "Busy" den Wert WAHR. Wurde die Aufgabe beendet, abgebrochen oder durch einen Fehler gestoppt, ändert sich der Wert des Ausgangsparameters "Busy" in FALSCH. Diese Änderung tritt unabhängig vom Signal am Eingangsparameter "Execute" ein.
  - Die Ausgangsparameter "InVelocity", "CommandAborted" und "Error" melden mindestens einen Zyklus lang den Wert WAHR, wenn ihre Bedingungen erfüllt sind. Diese Statusmeldungen werden gespeichert gehalten, während der Eingangsparameter "Execute" auf WAHR ist.

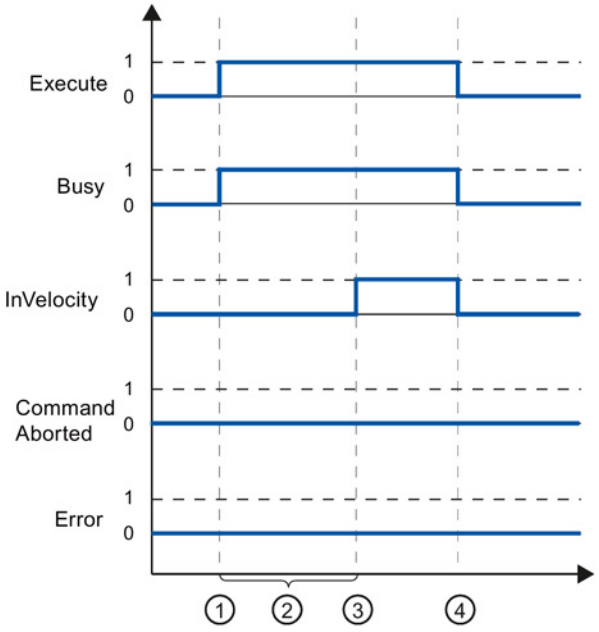
Das Verhalten der Statusbits wird im Folgenden für verschiedene Beispielsituationen aufgeführt:

- Das erste Beispiel zeigt das Verhalten, wenn die Achse die parametrierte Geschwindigkeit erreicht. Wenn die Bewegungssteuerungsaufgabe zu dem Zeitpunkt, zu dem die parametrierte Geschwindigkeit erreicht wird, ausgeführt wurde, wird dies durch den Wert WAHR am Ausgangsparameter "InVelocity" gemeldet. Der Signalzustand des Eingangsparameters "Execute" beeinflusst die Anzeigedauer im Ausgangsparameter "InVelocity".
- Das zweite Beispiel zeigt das Verhalten, wenn die Aufgabe vor Erreichen der parametrierten Geschwindigkeit abgebrochen wird. Wenn die Bewegungssteuerungsaufgabe vor Erreichen der parametrierten Geschwindigkeit abgebrochen wird, wird dies durch den Wert WAHR am Ausgangsparameter "CommandAborted" gemeldet. Der Signalzustand des Eingangsparameters "Execute" beeinflusst die Anzeigedauer im Ausgangsparameter "CommandAborted".
- Das dritte Beispiel zeigt das Verhalten der Achse, wenn vor Erreichen der parametrierten Geschwindigkeit ein Fehler auftritt. Tritt während der Ausführung der Bewegungssteuerungsaufgabe ein Fehler auf, bevor die parametrierte Geschwindigkeit erreicht wurde, wird dies durch den Wert WAHR am Ausgangsparameter "Error" gemeldet. Der Signalzustand des Eingangsparameters "Execute" beeinflusst die Anzeigedauer im Ausgangsparameter "Error".

Tabelle 10- 86 Beispiel 1 - Wenn die parametrierte Geschwindigkeit erreicht wird



Wenn "Execute" = FALSCH vor Erreichen der parametrierten Geschwindigkeit



Wenn "Execute" = FALSCH nach Erreichen der parametrierten Geschwindigkeit

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "Execute" gestartet. Je nach Programmierung kann "Execute" auf den Wert FALSCH zurückgesetzt werden, bevor die parametrierte Geschwindigkeit erreicht ist, oder alternativ erst nach Erreichen der parametrierten Geschwindigkeit.

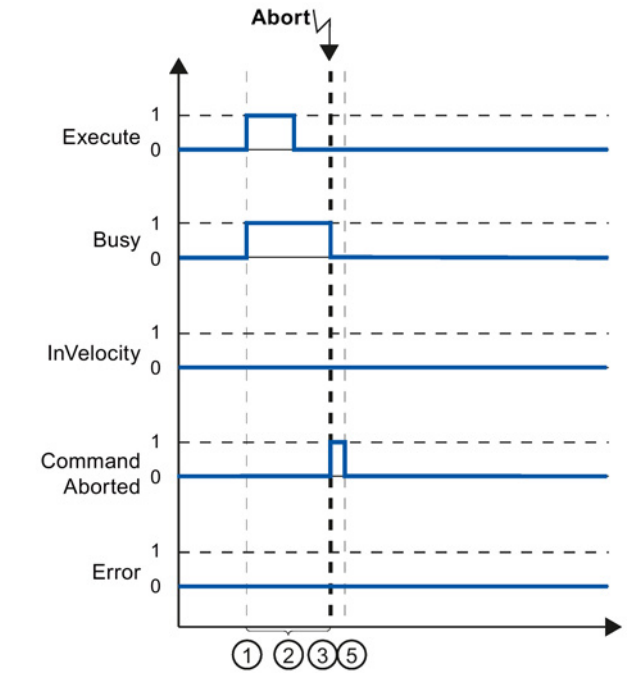
② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Wenn die parametrierte Geschwindigkeit erreicht ist, wechselt der Ausgangsparameter "InVelocity" auf WAHR.

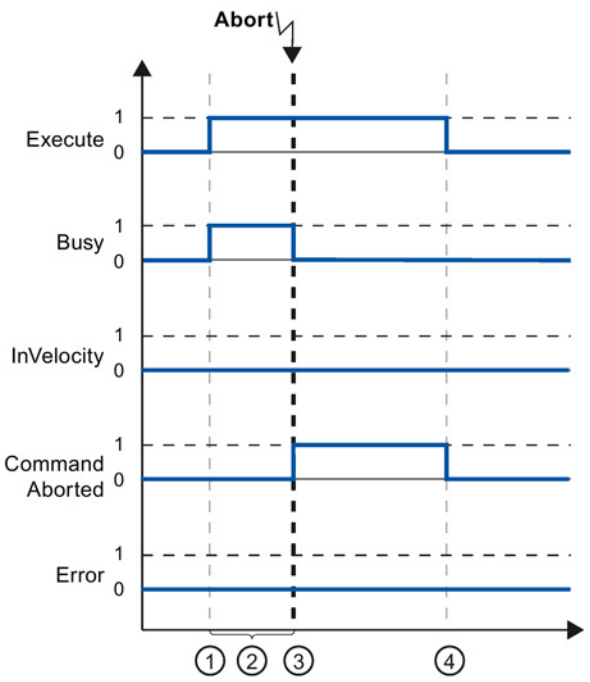
④ Wenn "Execute" den Wert WAHR auch nach Erreichen der parametrierten Geschwindigkeit beibehält, bleibt die Aufgabe aktiv. "InVelocity" und "Busy" behalten den Wert WAHR bei und gehen erst zusammen mit "Execute" nach FALSCH.

⑤ Wenn "Execute" vor Erreichen der parametrierten Geschwindigkeit auf FALSCH zurückgesetzt wurde, ist die Aufgabe beendet, wenn die parametrierte Geschwindigkeit erreicht wird. "InVelocity" meldet einen Zyklus lang den Wert WAHR und geht zusammen mit "Busy" nach FALSCH.

Tabelle 10- 87 Beispiel 2 - Wenn die Aufgabe vor Erreichen der parametrierten Geschwindigkeit abgebrochen wird



Wenn "Execute" = FALSCH vor Abbruch der Aufgabe



Wenn "Execute" = FALSCH nach Abbruch der Aufgabe

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "Execute" gestartet. Je nach Programmierung kann "Execute" während der Ausführung der Aufgabe auf den Wert FALSCH zurückgesetzt werden oder der Wert WAHR kann bis nach Abschluss der Aufgabe gehalten werden.

② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Die Aufgabe wird während ihrer Ausführung von einer anderen Bewegungssteuerungsaufgabe abgebrochen. Wenn die Aufgabe abgebrochen wird, geht der Ausgangsparameter "Busy" auf FALSCH und "CommandAborted" auf WAHR.

④ Wenn "Execute" den Wert WAHR bis nach Abbruch der Aufgabe beibehält, dann bleibt ""CommandAborted" ebenfalls WAHR und wechselt seinen Wert zusammen mit "Execute" in FALSCH.

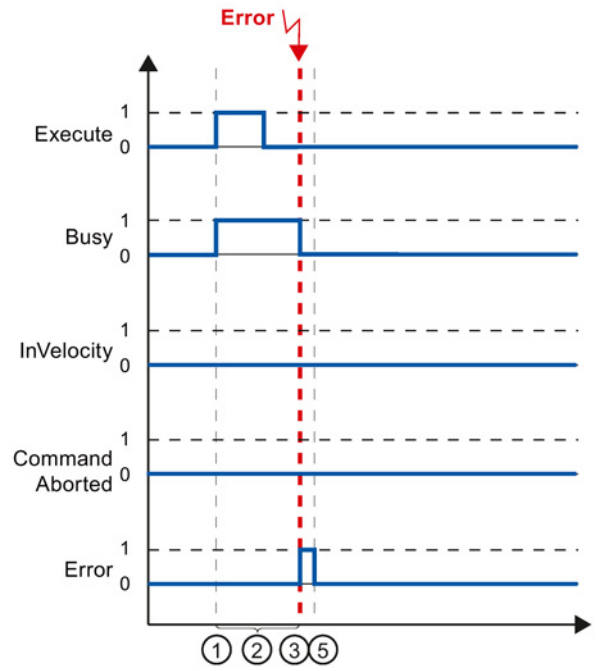
⑤ Wenn "Execute" vor Abbruch der Aufgabe auf FALSCH zurückgesetzt wurde, meldet "CommandAborted" den Wert WAHR nur einen Zyklus lang.

Hinweis

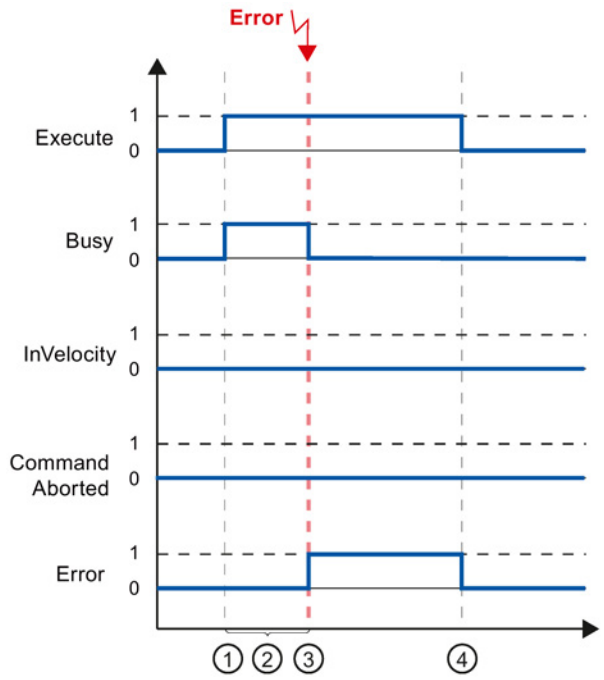
Unter den folgenden Bedingungen wird ein Abbruch nicht am Ausgangsparameter "CommandAborted" angezeigt:

- Die parametrierte Geschwindigkeit wurde erreicht, Eingangsparameter "Execute" hat den Wert FALSCH und eine neue Bewegungssteuerungsaufgabe wurde gestartet.
- Wenn die parametrierte Geschwindigkeit erreicht ist und Eingangsparameter "Execute" den Wert FALSCH hat, ist die Aufgabe beendet. Deshalb wird der Start einer neuen Aufgabe nicht als Abbruch gemeldet.

Tabelle 10- 88 Beispiel 3 - Wenn vor Erreichen der parametrierten Geschwindigkeit ein Fehler auftritt



Wenn "Execute" = FALSCH vor Auftreten des Fehlers



Wenn "Execute" = FALSCH nach Auftreten des Fehlers

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "Execute" gestartet. Je nach Programmierung kann "Execute" während der Ausführung der Aufgabe auf den Wert FALSCH zurückgesetzt werden oder der Wert WAHR kann bis nach Abschluss der Aufgabe gehalten werden.

② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Während der Ausführung der Aufgabe tritt ein Fehler auf. Wenn der Fehler auftritt, geht der Ausgangsparameter "Busy" auf FALSCH und "Error" auf WAHR.

④ Wenn "Execute" den Wert WAHR bis nach Auftreten des Fehlers beibehält, dann bleibt "Error" ebenfalls WAHR und wechselt seinen Wert erst zusammen mit "Execute" in FALSCH.

⑤ Wenn "Execute" vor Auftreten des Fehlers auf FALSCH zurückgesetzt wurde, meldet "Error" den Wert WAHR nur einen Zyklus lang.

Hinweis

Unter den folgenden Bedingungen wird ein Fehler nicht am Ausgangsparameter "Error" angezeigt:

- Die parametrierte Geschwindigkeit wurde erreicht, Eingangsparameter "Execute" hat den Wert FALSCH und ein Achsenfehler tritt auf (z. B. Software-Endschalter wird angefahren).
- Wenn die parametrierte Geschwindigkeit erreicht ist und Eingangsparameter "Execute" den Wert FALSCH hat, ist die Aufgabe beendet. Nach Abschluss der Aufgabe wird der Achsenfehler nur in der Bewegungssteuerungsanweisung "MC_Power" gemeldet.

10.3.8.3 Anweisung MC_MoveJog überwachen

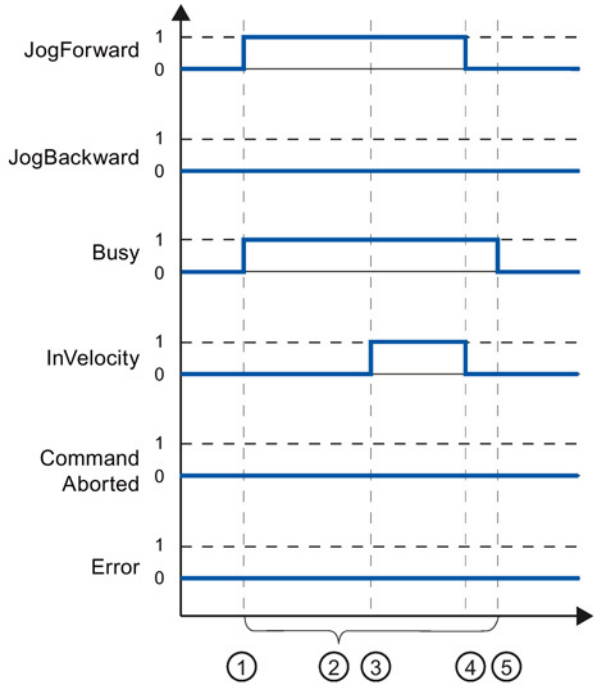
Die Aufgaben der Bewegungssteuerungsanweisung "MC_MoveJog" implementieren einen Tippbetrieb.

- Die Bewegungssteuerungsaufgaben "MC_MoveJog" haben kein definiertes Ende. Das Ziel der Aufgabe ist erfüllt, wenn die parametrierte Geschwindigkeit zum ersten Mal erreicht ist und die Achse mit konstanter Geschwindigkeit fährt. Wenn die parametrierte Geschwindigkeit erreicht ist, wird dies durch den Wert WAHR am Ausgangsparameter "InVelocity" gemeldet.
- Der Auftrag ist beendet, wenn der Eingangsparameter "JogForward" oder "JogBackward" auf den Wert FALSCH gesetzt wurde und die Achse zum Stillstand gekommen ist.
- Die Ausgangsparameter "Busy", "CommandAborted" und "Error" melden, dass die Aufgabe noch verarbeitet wird, abgebrochen wurde oder dass ein Fehler anliegt.
  - Während der Verarbeitung der Bewegungssteuerungsaufgabe meldet der Ausgangsparameter "Busy" den Wert WAHR. Wurde die Aufgabe beendet, abgebrochen oder durch einen Fehler gestoppt, ändert sich der Wert des Ausgangsparameters "Busy" in FALSCH.
  - Der Ausgangsparameter "InVelocity" meldet den Zustand WAHR, solange sich die Achse mit der parametrierten Geschwindigkeit bewegt. Die Ausgangsparameter "CommandAborted" und "Error" melden mindestens einen Zyklus lang den Status. Diese Statusmeldungen werden gespeichert gehalten, solange einer der Eingangsparameter "JogForward" oder "JogBackward" auf WAHR ist.

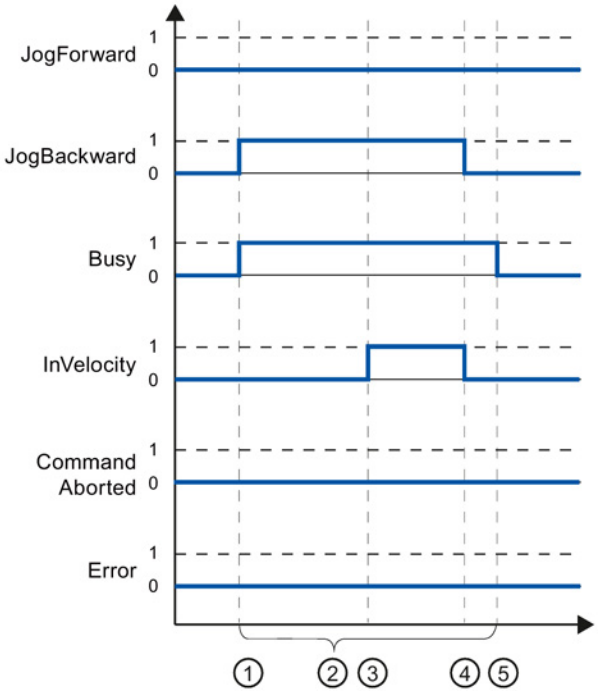
Das Verhalten der Statusbits wird im Folgenden für verschiedene Beispielsituationen aufgeführt.

- Das erste Beispiel zeigt das Verhalten der Achse, wenn die parametrierte Geschwindigkeit erreicht ist und gehalten wird. Wenn die Bewegungssteuerungsaufgabe zu dem Zeitpunkt, zu dem die parametrierte Geschwindigkeit erreicht wird, ausgeführt wurde, wird dies durch den Wert WAHR am Ausgangsparameter "InVelocity" gemeldet.
- Das zweite Beispiel zeigt das Verhalten der Achse, wenn die Aufgabe abgebrochen wird. Wenn die Bewegungssteuerungsaufgabe während der Ausführung abgebrochen wird, wird dies durch den Wert WAHR am Ausgangsparameter "CommandAborted" gemeldet. Das Verhalten ist unabhängig davon, ob die parametrierte Geschwindigkeit erreicht wurde oder nicht.
- Das dritte Beispiel zeigt das Verhalten der Achse im Fehlerfall. Tritt während der Ausführung der Bewegungssteuerungsaufgabe ein Fehler auf, wird dies durch den Wert WAHR am Ausgangsparameter "Error" gemeldet. Das Verhalten ist unabhängig davon, ob die parametrierte Geschwindigkeit erreicht wurde oder nicht.

Tabelle 10- 89 Beispiel 1 - Wenn die parametrierte Geschwindigkeit erreicht und gehalten wird



JogForward



JogBackward

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "JogForward" oder "JogBackward" gestartet.

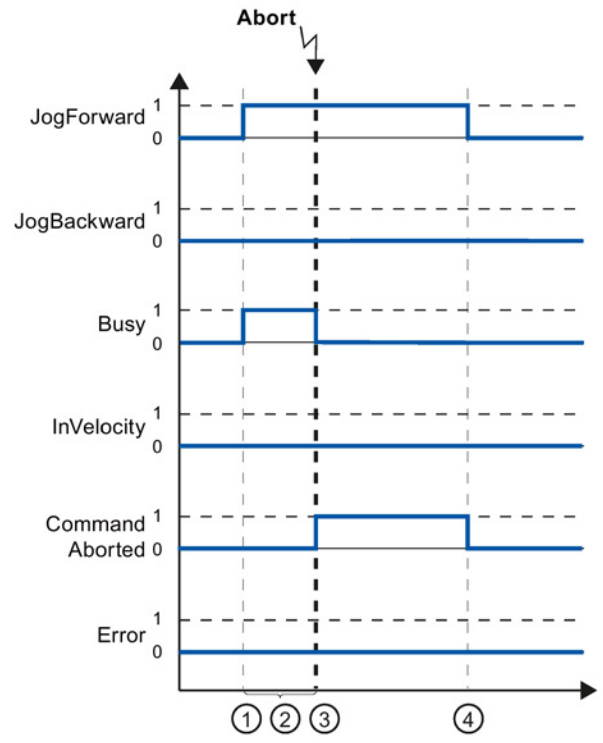
② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Wenn die parametrierte Geschwindigkeit erreicht ist, geht der Ausgangsparameter "InVelocity" nach WAHR.

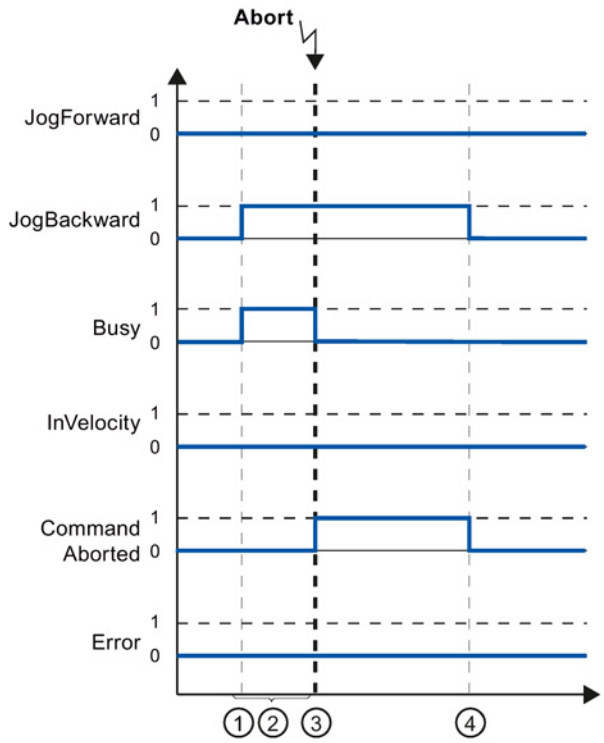
④ Wenn der Eingangsparameter "JogForward" oder "JogBackward" auf den Wert FALSCH zurückgesetzt wird, endet die Achsbewegung. Die Achse beginnt zu verzögern. Infolge dessen bewegt sich die Achse nicht mehr mit konstanter Geschwindigkeit und der Ausgangsparameter "InVelocity" geht nach FALSCH.

⑤ Wenn die Achse zum Stillstand gekommen ist, ist die Bewegungssteuerungsaufgabe beendet und der Ausgangsparameter "Busy" ändert seinen Wert in FALSCH.

Tabelle 10- 90 Beispiel 2 - Wenn die Aufgabe während der Ausführung abgebrochen wird



JogForward



JogBackward

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "JogForward" oder "JogBackward" gestartet.

② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Die Aufgabe wird während ihrer Ausführung von einer anderen Bewegungssteuerungsaufgabe abgebrochen. Wenn die Aufgabe abgebrochen wird, geht der Ausgangsparameter "Busy" auf FALSCH und "CommandAborted" auf WAHR.

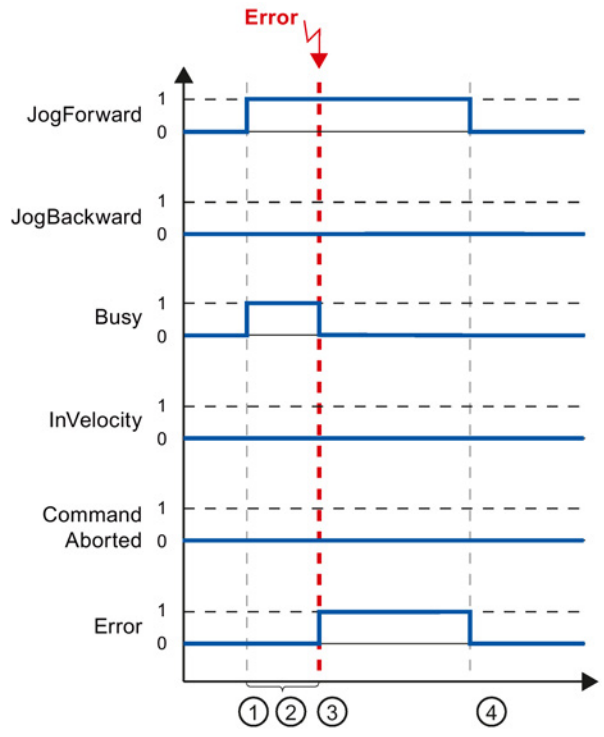
④ Wenn der Eingangsparameter "JogForward" oder "JogBackward" auf den Wert FALSCH zurückgesetzt wird, ändert der Ausgangsparameter "CommandAborted" seinen Wert in FALSCH.

Hinweis

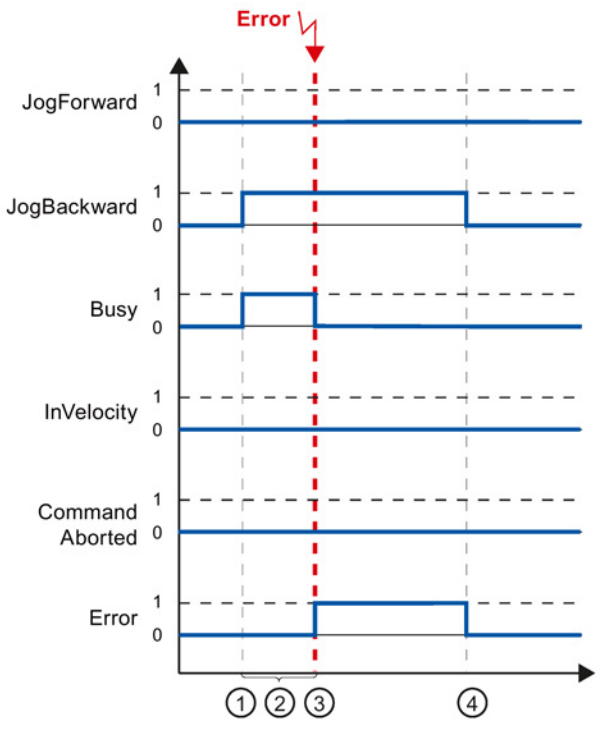
Der Abbruch der Aufgabe wird am Ausgangsparameter "CommandAborted" nur einen Ausführungszyklus lang gemeldet, wenn alle folgenden Bedingungen erfüllt sind:

Die Eingangsparameter "JogForward" und "JogBackward" haben den Wert FALSCH (doch die Achse verzögert noch) und eine neue Bewegungssteuerungsaufgabe wird gestartet.

Tabelle 10- 91 Beispiel 3 - Wenn während der Ausführung der Aufgabe ein Fehler aufgetreten ist



JogForward



JogBackward

① Die Aufgabe wird bei einer positiven Flanke am Eingangsparameter "JogForward" oder "JogBackward" gestartet.

② Während die Aufgabe aktiv ist, meldet der Ausgangsparameter "Busy" den Wert WAHR.

③ Während der Ausführung der Aufgabe tritt ein Fehler auf. Wenn der Fehler auftritt, geht der Ausgangsparameter "Busy" auf FALSCH und "Error" auf WAHR.

④ Wenn der Eingangsparameter "JogForward" oder "JogBackward" auf den Wert FALSCH zurückgesetzt wird, ändert der Ausgangsparameter "Error" seinen Wert in FALSCH.

Hinweis

Der aufgetretene Fehler wird am Ausgangsparameter "Error" nur einen Ausführungszyklus lang gemeldet, wenn alle folgenden Bedingungen erfüllt sind:

Die Eingangsparameter "JogForward" und "JogBackward" haben den Wert FALSCH (doch die Achse verzögert noch) und ein neuer Fehler tritt auf (z. B., Software-Endschalter wird angefahren).

Kommunikation

11

Die S7-1200 bietet mehrere Arten der Kommunikation zwischen CPUs und Programmiergeräten, HMI-Geräten und anderen CPUs.

⚠ WARNUNG

Wenn sich ein Angreifer physischen Zugriff auf Ihre Netzwerke verschaffen kann, kann er möglicherweise Daten lesen und schreiben.

Das TIA Portal, die CPU und HMI-Geräte (außer HMI-Geräten über GET/PUT) nutzen die sichere Kommunikation, die Schutz vor Replay- und Man-in-the-Middle-Angriffen bietet. Nach der Aktivierung der Kommunikation erfolgt der Austausch signierter Meldungen in Klartext, wodurch ein Angreifer Daten lesen kann, die Daten jedoch vor unberechtigtem Schreiben geschützt sind. Das TIA Portal, nicht der Kommunikationsprozess, verschlüsselt die Daten von Knowhow-geschützten Bausteinen.

Alle anderen Arten der Kommunikation (E/A-Austausch über PROFIBUS, PROFINET, AS-i oder einen anderen E/A-Bus, GET/PUT, T-Block und Kommunikationsmodule (CM)) haben keine Sicherheitsfunktionen. Sie müssen diese Arten der Kommunikation durch Begrenzung des physischen Zugriffs schützen. Wenn sich ein Angreifer über diese Arten der Kommunikation physischen Zugriff auf Ihre Netzwerke verschaffen kann, kann er möglicherweise Daten lesen und schreiben.

Informationen und Empfehlungen bezüglich der Sicherheit finden Sie in unseren Operational Guidelines für Industrial Security (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf) auf der Website "Service & Support" von Siemens.

PROFINET

PROFINET wird für den Austausch von Daten über das Anwenderprogramm mit anderen Kommunikationspartnern über Ethernet verwendet:

- Bei der S7-1200 unterstützt PROFINET 16 IO-Geräte mit maximal 256 Submodulen, und PROFIBUS gestattet 3 unabhängige PROFIBUS DP-Master und unterstützt 32 Slaves je DP-Master, mit maximal 512 Modulen je DP-Master.
- S7-Kommunikation
- User Datagram Protocol (UDP)
- ISO on TCP (RFC 1006)
- Transport Control Protocol (TCP)

PROFINET IO-Controller

Als IO-Steuerung über PROFINET IO kommuniziert die CPU mit bis zu 16 PN-Geräten im lokalen PN-Netzwerk oder über einen PN/PN-Koppler (Verbund). Ausführliche Informationen hierzu finden Sie unter PROFIBUS and PROFINET International, PI (www.profinet.com).

PROFIBUS

PROFIBUS wird für den Austausch von Daten über das Anwenderprogramm mit anderen Kommunikationspartnern über das PROFIBUS-Netzwerk verwendet:

- Beim CM 1242-5 fungiert die CPU als PROFIBUS DP-Slave.
- Beim CM 1243-5 fungiert die CPU als PROFIBUS DP-Master der Klasse 1.
- PROFIBUS DP-Slaves, PROFIBUS DP-Master und AS-i (die 3 Kommunikationsmodule auf der linken Seite) und PROFINET sind voneinander getrennte Kommunikationsnetzwerke, die sich nicht gegenseitig einschränken.

ASi

Der S7-1200 CM 1243-2 AS-i Master ermöglicht die Anbindung eines AS-i-Netzwerks an eine S7-1200 CPU.

CPU-zu-CPU-Kommunikation über S7

Sie können eine Kommunikationsverbindung mit einer Partnerstation aufbauen und mithilfe der Anweisungen GET und PUT mit S7-CPUs kommunizieren.

TeleService-Kommunikation

Bei TeleService über GPRS kommuniziert eine Engineering-Station, auf der STEP 7 installiert ist, über das GSM-Netz und das Internet mit einer SIMATIC S7-1200-Station mit einem CP 1242-7. Die Verbindung erfolgt über einen Telecontrol-Server, der als Vermittler dient und mit dem Internet verbunden ist.

IO-Link

Der S7-1200 SM 1278 4xIO-Link-Master ermöglicht die Verbindung von IO-Link-Geräten mit einer S7-1200 CPU.

11.1 Asynchrone Kommunikationsverbindungen in V4.1

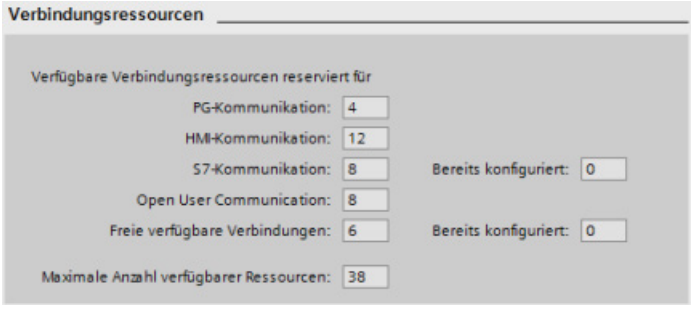
Überblick über die Kommunikationsdienste

Die CPU unterstützt die folgenden Kommunikationsdienste:

Kommunikationsdienst	Funktionalität	Bei PROFIBUS DP		Bei Ethernet
		CM 1243-5 DP-Mastermodul	CM 1242-5 DP-Slavemodul	
PG-Kommunikation	Inbetriebnahme, Test, Diagnose	Ja	Nein	Ja
HMI-Kommunikation	Bedienen und Beobachten	Ja	Nein	Ja
S7-Kommunikation	Datenaustausch über konfigurierte Verbindungen	Ja	Nein	Ja
Routing von PG-Funktionen	Beispielsweise Test und Diagnose außerhalb der Netzwerkgrenzen	Nein	Nein	Nein
PROFIBUS DP	Datenaustausch zwischen Master und Slave	Ja	Ja	Nein
PROFINET IO	Datenaustausch zwischen E/A-Steuerungen und E/A-Geräten	Nein	Nein	Ja
Webserver	Diagnose	Nein	Nein	Ja
SNMP (Simple Network Management Protocol)	Standardprotokoll für Netzwerkdiagnose und Parametrierung	Nein	Nein	Ja
Offene Kommunikation über TCP/IP	Datenaustausch über Industrial Ethernet mit TCP/IP-Protokoll (mit ladbaren FBs)	Nein	Nein	Ja
Offene Kommunikation über ISO-on-TCP	Datenaustausch über Industrial Ethernet mit ISO-on-TCP-Protokoll (mit ladbaren FBs)	Nein	Nein	Ja
Offene Kommunikation über UDP	Datenaustausch über Industrial Ethernet mit UDP-Protokoll (mit ladbaren FBs)	Nein	Nein	Ja

Verfügbare Verbindungen

Die CPU unterstützt maximal die folgende Anzahl von gleichzeitigen, asynchronen Kommunikationsverbindungen für PROFINET und PROFIBUS. Die maximale Anzahl der jeder Kategorie zugeordneten Verbindungsressourcen ist festgelegt. Sie können die Werte nicht ändern. Die 6 frei verfügbaren Verbindungen können jedoch genutzt werden, um die Anzahl Verbindungen in jeder Kategorie nach dem Bedarf Ihrer Anwendung zu erhöhen.



Basierend auf den zugeordneten Verbindungsressourcen steht die folgende Anzahl von Verbindungen je Gerät zur Verfügung:

	Programmier-gerät (PG)	Mensch-Maschine-Schnittstelle (HMI)	GET/PUT-Client/Server	Offene Benutzer-kommunikation	Webbrowser
Maximale Anzahl von Verbindungs-ressourcen	3 (garantierte Unterstützung eines PGs)	12 (garantierte Unterstützung von vier HMI-Geräten)	8	8	30 (garantierte Unterstützung von drei Webbrowsern)

Beispielsweise hat ein PG drei verfügbare Verbindungsressourcen. Je nach aktuell verwendeten PG-Funktionen verwendet das PG möglicherweise tatsächlich 1, 2 oder 3 der verfügbaren Verbindungsressourcen. Bei der S7-1200 besteht jederzeit die Garantie für mindestens 1 PG; es ist jedoch maximal 1 PG zulässig.

Ein weiteres Beispiel ist die Anzahl der HMI-Geräte, wie in der nachstehenden Abbildung gezeigt. HMI-Geräte haben 12 verfügbare Verbindungsressourcen. Je nach Typ oder Modell Ihres HMI-Geräts und den verwendeten HMI-Funktionen kann jedes HMI-Gerät tatsächlich 1, 2 oder 3 seiner verfügbaren Verbindungsressourcen nutzen. Abhängig von der Anzahl der verwendeten verfügbaren Verbindungsressourcen können möglicherweise mehr als vier HMI-Geräte gleichzeitig verwendet werden. Es besteht jedoch jederzeit die Garantie für mindestens vier HMI-Geräte. Ein HMI-Gerät kann seine verfügbaren Verbindungsressourcen (je 1 von insgesamt 3) für die folgenden Funktionen nutzen:

- Lesen
- Schreiben
- Alarme und Diagnosen

Beispiel	HMI 1	HMI 2	HMI 3	HMI 4	HMI 5	Gesamtzahl verfügbarer Verbin-dungsres-sourcen
Verwendete Verbin-dungsres-sourcen	2	2	2	3	3	12

Hinweis

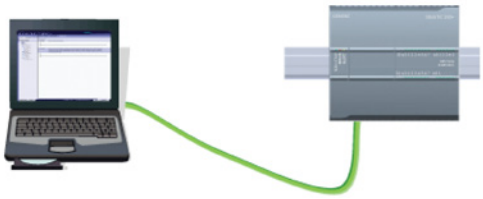
Webserver (HTTP)-Verbindungen: Die CPU bietet Verbindungen für mehrere Webbrowser. Die Anzahl von Browsern, die von der CPU gleichzeitig unterstützt werden können, richtet sich danach, wie viele Verbindungen ein Webbrowser anfordert/verwendet.

Hinweis

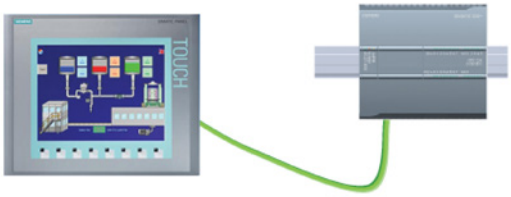
Die Kommunikationsverbindungen von offener Benutzerkommunikation, S7-Verbindung, HMI, Programmiergerät und Webserver (HTTP) können basierend auf den verwendeten Funktionen mehrere Verbindungsressourcen nutzen.

11.2 PROFINET

Die CPU kann mit anderen CPUs, mit Programmiergeräten, mit HMI-Geräten und Siemens-fremden Geräten über herkömmliche TCP-Kommunikationsprotokolle kommunizieren.



Mit der CPU verbundenes Programmiergerät

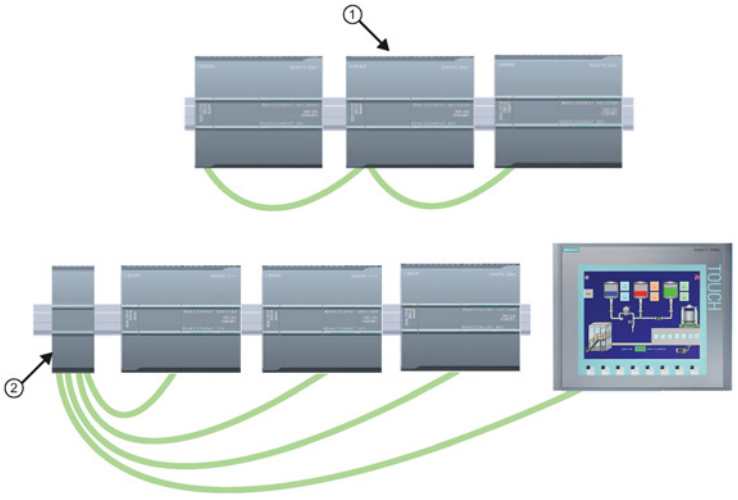


An die CPU angeschlossenes HMI-Gerät

Eine mit einer anderen CPU verbundene CPU

Ethernet-Switch

Der PROFINET-Anschluss an der CPU 1211C, 1212C und 1214C enthält keinen Ethernet-Switch. Eine direkte Verbindung zwischen einem Programmiergerät oder HMI-Gerät und einer CPU erfordert keinen Ethernet-Switch. Ein Netzwerk mit mehr als zwei CPUs oder HMI-Geräten jedoch benötigt einen Ethernet-Switch.



① CPU 1215C

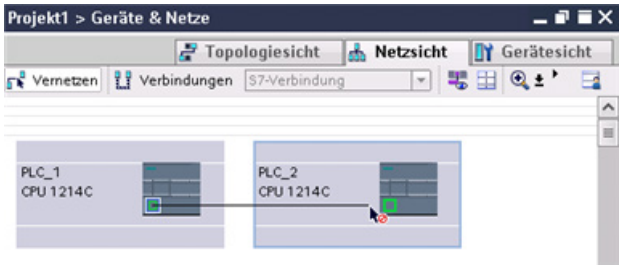
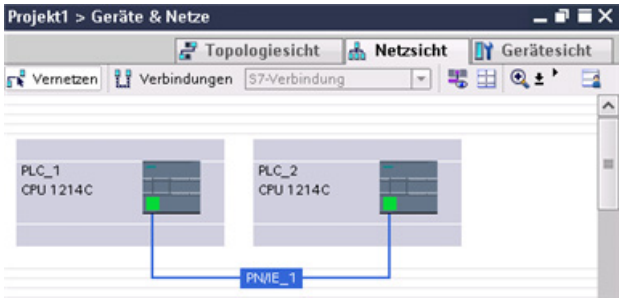
② Ethernet-Switch CSM1277

Die CPUs 1215C und 1217C besitzen einen integrierten Ethernet-Switch mit 2 Ports. Sie können ein Netzwerk mit einer CPU 1215C und zwei anderen S7-1200 CPUs aufbauen. Zum Anschließen mehrerer CPUs und HMI-Geräte können Sie auch einen auf dem Baugruppenträger montierten Ethernet-Switch CSM1277 mit 4 Ports verwenden.

11.2.1 Netzwerkverbindung erstellen

In der Netzsicht der Gerätekonfiguration können Sie die Netzwerkverbindungen zwischen den Geräten in Ihrem Projekt herstellen. Nach dem Herstellen der Netzwerkverbindung können Sie im Register "Eigenschaften" des Inspektorfensters die Netzwerkparameter konfigurieren.

Tabelle 11- 1 Netzwerkverbindung erstellen

Handlungsanweisung	Ergebnis
Wählen Sie die "Netzwerkansicht" für die Anzeige der Geräte, die verbunden werden sollen.	
Wählen Sie die Schnittstelle eines Geräts und ziehen Sie die Verbindung zur Schnittstelle des zweiten Geräts.	
Lassen Sie die Maustaste los, damit die Netzwerkverbindung erstellt wird.	

11.2.2 Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren

Eine Verbindung zwischen lokaler CPU und entfernter Partner-CPU definiert eine logische Zuweisung von zwei Kommunikationspartnern für die Herstellung von Kommunikationsdiensten. Eine Verbindung definiert folgende Elemente:

- Beteiligte Kommunikationspartner (einer aktiv, einer passiv)
- Art der Verbindung (z. B. eine PLC-, HMI- oder Geräteverbindung)
- Verbindungspfad

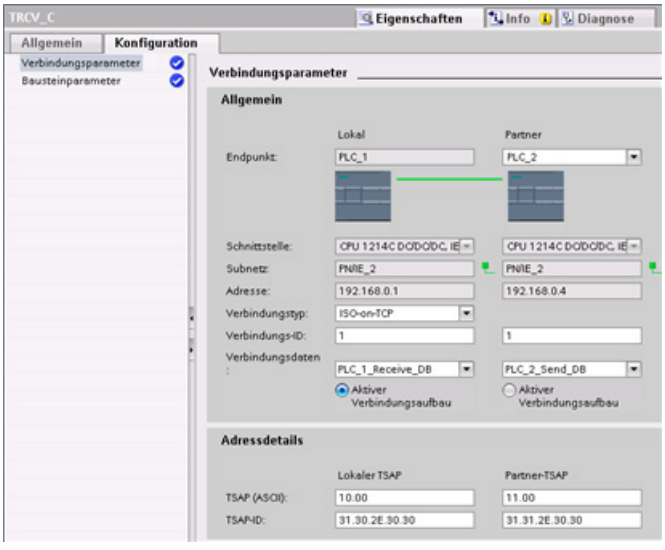
Die Kommunikationspartner führen die Anweisungen zum Einrichten und Herstellen der Kommunikationsverbindung aus. Mit Parametern geben Sie die aktiven und passiven Endpunkte der Kommunikation an. Nach dem Einrichten und Aufbauen der Verbindung wird diese automatisch von der CPU gehalten und überwacht.

Wird die Verbindung beendet (z. B. durch eine Leitungsstörung), so versucht der aktive Partner, sie wieder aufzubauen. Sie müssen die Kommunikationsanweisung nicht erneut ausführen.

Verbindungspfade

Nach dem Einfügen einer Anweisung TSEND_C, TRCV_C oder TCON in das Anwenderprogramm werden im Inspektorfenster die Eigenschaften der Verbindung anzeigt, wenn Sie einen Teil der Anweisung auswählen. Sie legen die Kommunikationsparameter über das Register "Konfiguration" im Dialog "Eigenschaften" der Kommunikationsanweisung fest.

Tabelle 11- 2 Verbindungspfad konfigurieren (über die Eigenschaften der Anweisung)

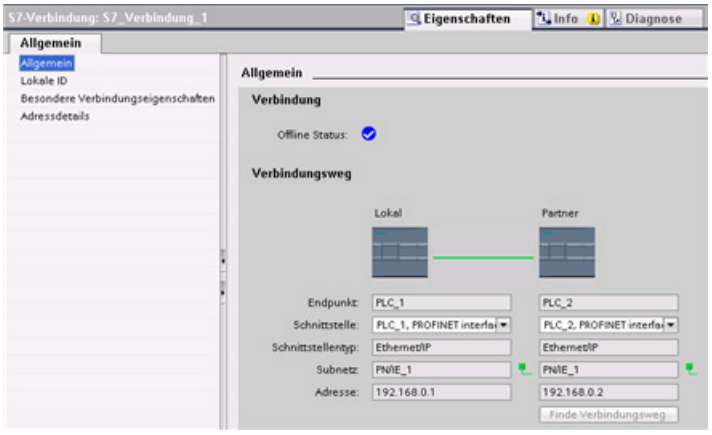
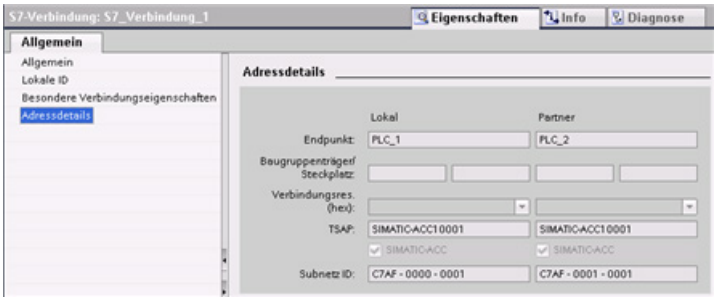
TCP, ISO-on-TCP und UDP	Verbindungseigenschaften
Bei den TCP-, ISO-on-TCP- und UDP-Ethernet-Protokollen konfigurieren Sie die Verbindungen der lokalen und der Partner-CPU in den "Eigenschaften" der Anweisung (TSEND_C, TRCV_C oder TCON). Die Abbildung zeigt die "Verbindungseigenschaften" im Register "Verbindung" einer ISO-on-TCP-Verbindung.	

Hinweis

Wenn Sie die Verbindungseigenschaften für eine CPU konfigurieren, können Sie in STEP 7 entweder einen bestimmten Verbindungs-DB in der Partner-CPU auswählen (sofern vorhanden) oder einen Verbindungs-DB für die Partner-CPU anlegen. Die Partner-CPU muss im Projekt bereits angelegt sein, es darf sich nicht um eine "nicht spezifizierte" CPU handeln.

Sie müssen trotzdem eine Anweisung TSEND_C, TRCV_C oder TCON in das Anwenderprogramm der Partner-CPU einfügen. Wenn Sie die Anweisung einfügen, wählen Sie den Verbindungs-DB aus, der von der Konfiguration angelegt wurde.

Tabelle 11- 3 Verbindungspfad für die S7-Kommunikation konfigurieren (Gerätekonfiguration)

S7-Kommunikation (GET und PUT)	Verbindungseigenschaften
Bei der S7-Kommunikation konfigurieren Sie die Verbindungen zwischen lokaler und Partner-CPU im Editor "Geräte & Netze" des Netzwerks. Sie können auf die Schaltfläche "Hervorgehoben: Verbindung" klicken, um die "Eigenschaften" aufzurufen. Im Register "Allgemein" werden mehrere Eigenschaften aufgeführt: • "Allgemein" (abgebildet) • "Lokale ID" • "Spezielle Verbindungseigenschaften" • "Adressdetails" (abgebildet)	 

Im Abschnitt "PROFINET" unter "Protokolle" (Seite 656) und im Abschnitt "S7-Kommunikation" unter "S7-Verbindung erstellen" (Seite 806) finden Sie weitere Informationen sowie eine Liste der verfügbaren Kommunikationsanweisungen.

Tabelle 11- 4 Parameter für die CPU-Verbindung

Parameter		Definition
Adresse		Zugewiesene IP-Adressen
Allgemein	Endpunkt	Name der Partner-CPU (Empfänger)
	Schnittstelle	Name der Schnittstellen
	Subnetz	Name der Subnetze
	Schnittstellentyp	*Nur S7-Kommunikation*: Typ der Schnittstelle
	Verbindungstyp	Typ des Ethernet-Protokolls
	Verbindungs-ID	ID-Nummer
	Verbindungsdaten	Datenspeicher für die lokale CPU und die Partner-CPU
	Aktive Verbindung herstellen	Optionsfeld zum Auswählen der lokalen CPU oder der Partner-CPU als aktive Verbindung
Adressdetails	Endpunkt	*Nur S7-Kommunikation*: Name der Partner-CPU (Empfänger)
	Baugruppenträger/Steckplatz	*Nur S7-Kommunikation*: Baugruppenträger- und Steckplatzadresse
	Verbindungsressource	*Nur S7-Kommunikation*: Komponente des TSAP für die Konfiguration einer S7-Verbindung mit einer S7-300 oder S7-400 CPU

Parameter		Definition
	Port (dezimal):	TCP und UPD: Port der Partner-CPU im Dezimalformat
	TSAP [1] und Subnetz-ID:	ISO on TCP (RFC 1006) und S7-Kommunikation: TSAPs der lokalen CPU und der Partner-CPU im ASCII- und Hexadezimalformat

[1] Verwenden Sie beim Konfigurieren einer Verbindung mit einer S7-1200 CPU über ISO-on-TCP in der TSAP-Erweiterung für die passiven Kommunikationsteilnehmer nur ASCII-Zeichen.

Transport Service Access Points (TSAPs)

Mit TSAPs gestatten das ISO-on-TCP-Protokoll und die S7-Kommunikation mehrere Verbindungen mit einer einzigen IP-Adesse (bis zu 64-K-Verbindungen). TSAPs ermitteln die eindeutige Zuordnung dieser Verbindungen der Kommunikationsendpunkte zu einer IP-Adresse.

Die zu verwendenden TSAPs definieren Sie im Dialog "Verbindungsparameter" unter "Adressdetails". Der TSAP einer Verbindung in der CPU wird im Feld "Lokaler TSAP" eingegeben. Der TSAP für die Verbindung in der Partner-CPU wird im Feld "Partner-TSAP" eingegeben.

Portnummern

Bei den TCP- und UDP-Protokollen müssen in der Konfiguration der Verbindungsparameter der lokalen (aktiven) CPU die dezentrale IP-Adresse und die Portnummer der (passiven) Partner-CPU angegeben werden.

Die zu verwendenden Ports definieren Sie im Dialog "Verbindungsparameter" unter "Adressdetails". Der Port einer Verbindung in der CPU wird im Feld "Lokaler Port" eingegeben. Der Port für die Verbindung in der Partner-CPU wird im Feld "Partner-Port" eingegeben.

11.2.3 IP-Adressen zuweisen

11.2.3.1 IP-Adressen zu Programmier- und Netzwerkgeräten zuweisen

Wenn Ihr Programmiergerät eine integrierte Adapterkarte nutzt, die an das LAN Ihrer Anlage (und möglicherweise an das Internet) angeschlossen ist, müssen die Netzwerk-ID der IP-Adresse und die Subnetzmaske Ihrer CPU mit der integrierten Adapterkarte des Programmiergeräts genau übereinstimmen. Die Netzwerk-ID ist der erste Teil der IP-Adresse (die ersten drei Oktette, z.B. **211.154.184**.16). Sie legt fest, welches IP-Netzwerk Sie nutzen. Die Subnetzmaske hat normalerweise den Wert **255.255.255.0**. Da sich Ihr Computer jedoch in einem Anlagen-LAN befindet, kann die Subnetzmaske andere Werte annehmen (z.B. **255.255.254.0**), um eindeutige Subnetze einzurichten. Die Subnetzmaske definiert in UND-Verknüpfung mit der IP-Adresse des Geräts die Grenzen eines IP-Subnetzes.

Hinweis

Im Internet, wo Ihre Programmiergeräte, Netzwerkgeräte und IP-Router mit der Welt kommunizieren, müssen eindeutige IP-Adressen zugeordnet werden, um Konflikte mit anderen Netzwerknutzern zu vermeiden. Wenden Sie sich an die IT-Abteilung Ihrer Firma, deren Mitarbeiter die Netzwerke in Ihrem Unternehmen kennen, um sich IP-Adressen zuweisen zu lassen.

⚠ WARNUNG

Unbefugter Zugriff auf die CPU über den Webserver

Ein unbefugter Zugriff auf die CPU oder das Einstellen von ungültigen Werten für PLC-Variablen kann den Prozessbetrieb stören und zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Weil durch die Aktivierung des Webservers berechtigte Benutzer in der Lage sind, den Betriebszustand zu ändern, PLC-Daten zu schreiben und Firmware-Updates durchzuführen, empfiehlt Siemens, die folgenden Sicherheitsvorkehrungen einzuhalten:

- Aktivieren Sie den Zugriff auf den Webserver nur über das HTTPS-Protokoll.
- Schützen Sie die Webserver-Benutzer-IDs (Seite 821) mit einem starken Passwort. Starke Passwörter sind mindestens zehn Zeichen lang, bestehen aus Buchstaben, Zahlen und Sonderzeichen, sind keine Wörter, die in einem Wörterbuch gefunden werden können, und sind keine Namen oder Kennungen, die sich aus persönlichen Daten ableiten lassen. Halten Sie das Passwort geheim und ändern Sie es häufig.
- Erweitern Sie nicht die voreingestellten Minimalrechte für den Benutzer "Jeder".
- Führen Sie eine Fehlerprüfung und eine Bereichsprüfung für die Variablen in Ihrer Programmlogik durch, weil die Benutzer von Webseiten für die PLC-Variablen ungültige Werte einrichten können.

Wenn Ihr Programmiergerät eine Ethernet/USB-Adapterkarte nutzt, die an ein einzelnes Netzwerk angeschlossen ist, müssen die Netzwerk-ID der IP-Adresse und die Subnetzmaske Ihrer CPU mit der Ethernet/USB-Adapterkarte des Programmiergeräts genau übereinstimmen. Die Netzwerk-ID ist der erste Teil der IP-Adresse (die ersten drei Oktette, z.B. **211.154.184**.16). Sie legt fest, welches IP-Netzwerk Sie nutzen. Die Subnetzmaske hat normalerweisen den Wert **255.255.255.0**. Die Subnetzmaske definiert in UND-Verknüpfung mit der IP-Adresse des Geräts die Grenzen eines IP-Subnetzes.

Hinweis

Eine Ethernet/USB-Adapterkarte ist nützlich, wenn Sie Ihr Programmiergerät nicht an das Firmen-LAN anschließen möchten. Dieser Aufbau ist besonders während anfänglicher Tests bzw. während der Inbetriebnahmeprüfungen nützlich.

Tabelle 11- 5 Ethernet-Adressen zuweisen

Adapterkarte des Programmiergeräts	Netzwerktyp	IP-Adresse	Subnetzmaske
Integrierte Adapterkarte	Angeschlossen an Ihr Firmen-LAN (und möglicherweise das Internet)	Die Netzwerk-ID Ihrer CPU und der integrierten Adapterkarte des Programmiergeräts müssen exakt übereinstimmen.[1]	Die Subnetzmaske Ihrer CPU und der integrierten Adapterkarte müssen exakt übereinstimmen. Die Subnetzmaske hat normalerweise den Wert **255.255.255.0**. Da sich Ihr Computer jedoch in einem Anlagen-LAN befindet, kann die Subnetzmaske andere Werte annehmen (z.B. **255.255.254.0**), um eindeutige Subnetze einzurichten.[2]
Ethernet/USB-Adapterkarte	Angeschlossen an ein einzelnes Netzwerk	Die Netzwerk-ID Ihrer CPU und der Ethernet/USB-Adapterkarte des Programmiergeräts müssen exakt übereinstimmen.[1]	Die Subnetzmaske Ihrer CPU und der Ethernet/USB-Adapterkarte müssen exakt übereinstimmen. Die Subnetzmaske hat normalerweise den Wert **255.255.255.0**.[2]

1 Die Netzwerk-ID ist der erste Teil der IP-Adresse (die ersten drei Oktette, z. B. **211.154.184**.16). Sie legt fest, welches IP-Netzwerk Sie nutzen.

2 Die Subnetzmaske definiert in UND-Verknüpfung mit der IP-Adresse des Geräts die Grenzen eines IP-Subnetzes.

IP-Adresse Ihres Programmiergeräts über "Netzwerkumgebung" (auf Ihrem Desktop) zuweisen oder prüfen

Die IP-Adresse Ihres Programmiergeräts können Sie mit den folgenden Menübefehlen zuweisen oder prüfen:

- (Klicken Sie mit der rechten Maustaste auf) "Netzwerkumgebung".
- "Eigenschaften"
- Klicken Sie mit der rechten Maustaste auf "LAN-Verbindung".
- "Eigenschaften"

Blättern Sie im Dialog "Eigenschaften von LAN-Verbindung" im Feld "Diese Verbindung verwendet folgende Elemente:" zum Eintrag "Internetprotokoll (TCP/IP)". Klicken Sie auf "Internetprotokoll (TCP/IP)" und dann auf die Schaltfläche "Eigenschaften". Wählen Sie "IP-Adresse automatisch beziehen (DHCP)" oder "Folgende IP-Adresse verwnden" (um eine statische IP-Adresse einzugeben) aus.

Hinweis

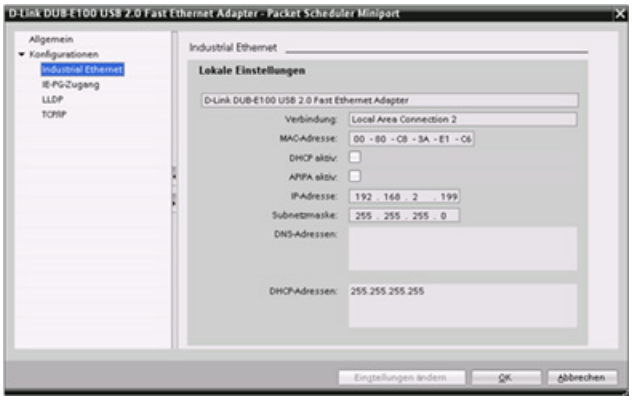
Das Dynamic Host Configuration Protocol (DHCP) weist Ihrem Programmiergerät beim Einschalten des DHCP-Servers automatisch eine IP-Adresse zu.

11.2.3.2 Ermitteln der IP-Adresse Ihres Programmiergeräts

Die MAC- und die IP-Adresse Ihres Programmiergeräts ermitteln Sie mit den folgenden Menübefehlen:

1. Erweitern Sie in der Projektnavigation den Knoten "Online-Zugänge".
2. Klicken Sie mit der rechten Maustaste auf das erforderliche Netzwerk und wählen Sie "Eigenschaften".
3. Erweitern Sie im Dialog "Netzwerk" die "Konfigurationen" und wählen Sie "Industrial Ethernet".

Dort werden die MAC- und die IP-Adresse Ihres Programmiergeräts angezeigt.

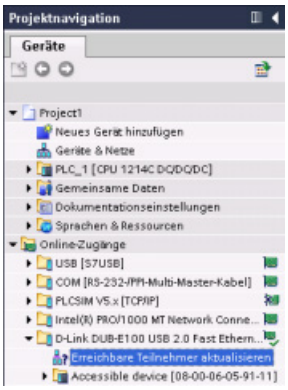


11.2.3.3 Online eine IP-Adresse zu einer CPU zuweisen

Sie können einem Netzwerkgerät online eine IP-Adresse zuweisen. Dies ist besonders bei der ersten Gerätekonfiguration nützlich.

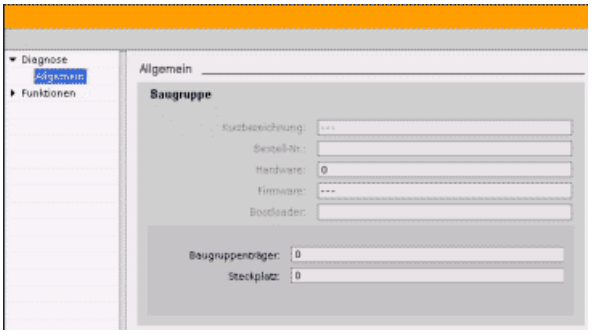
1. Stellen Sie in der "Projektnavigation" mit folgenden Befehlen sicher, dass der CPU keine IP-Adresse zugewiesen ist:
   - "Online-Zugänge"
   - <Adapterkarte für das Netzwerk, in dem sich das Gerät befindet>
   - "Erreichbare Teilnehmer aktualisieren"

   HINWEIS: Wenn STEP 7 statt einer IP-Adresse eine MAC-Adresse anzeigt, wurde keine IP-Adresse zugewiesen.

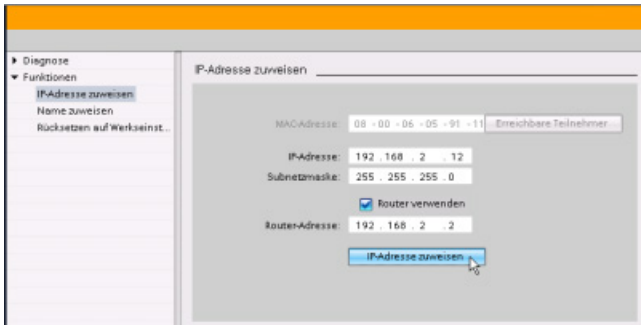


2. Doppelklicken Sie unter dem erforderlichen zugänglichen Gerät auf "Online & Diagnose".

3. Wählen Sie im Dialog "Online & Diagnose" die folgenden Menübefehle:
   - "Funktionen"
   - "IP-Adresse zuordnen"

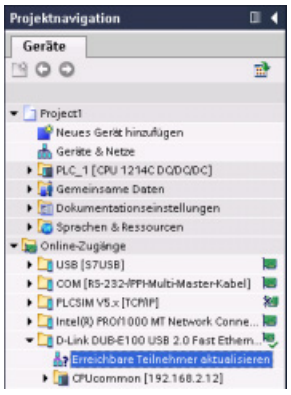


4. Geben Sie im Feld "IP-Adresse" Ihre neue IP-Adresse ein und klicken Sie auf die Schaltfläche "IP-Adresse zuweisen".



5. Prüfen Sie in der "Projektnavigation" mit den folgenden Befehlen, dass Ihre neue IP-Adresse der CPU zugewiesen wurde:

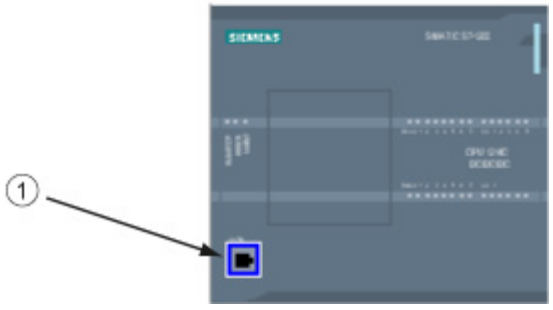
- "Online-Zugänge"
- <Adapterkarte für das Netzwerk, in dem sich das Gerät befindet>
- "Erreichbare Teilnehmer aktualisieren"



11.2.3.4 IP-Adresse für eine CPU in Ihrem Projekt konfigurieren

Konfigurieren der PROFINET-Schnittstelle

Um Parameter für die PROFINET-Schnittstelle zu konfigurieren, wählen Sie das grüne PROFINET-Feld auf der CPU. Im Inspektorfenster wird das Register "Eigenschaften" für den PROFINET-Port angezeigt.



① PROFINET-Port

Konfigurieren der IP-Adresse

Ethernet-Adresse (MAC-Adresse): In einem PROFINET-Netzwerk ist jedem Gerät für die Identifikation eine MAC-Adresse (Media-Access-Control-Adresse) vom Hersteller zugewiesen. Eine MAC-Adresse besteht aus sechs Gruppen zu je zwei Hexadezimalziffern mit Bindestrichen (-) oder Doppelpunkten (:) als Trennzeichen in der Reihenfolge der Übertragung (zum Beispiel 01-23-45-67-89-AB oder 01:23:45:67:89:AB).

IP-Adresse: Jedes Gerät benötigt außerdem eine IP-Adresse (Internet-Protocol-Adresse). Mit Hilfe dieser Adresse kann ein Gerät Daten über ein komplexeres Netzwerk liefern.

Jede IP-Adresse ist in vier 8-Bit-Segmente unterteilt und wird in einem Dezimalformat mit Punkttrennzeichen dargestellt (zum Beispiel 211.154.184.16). Der erste Teil der IP-Adresse ist die Netzwerk-ID (Kennung des aktiven Netzwerks), der zweite Teil der Adresse ist die Host-ID (für jedes Gerät im Netzwerk eindeutig). Eine IP-Adresse von 192.168.x.y ist eine Standardkennzeichnung, die als Teil eines privaten Netzwerks, das sich nicht im Internet befindet, erkannt wird.

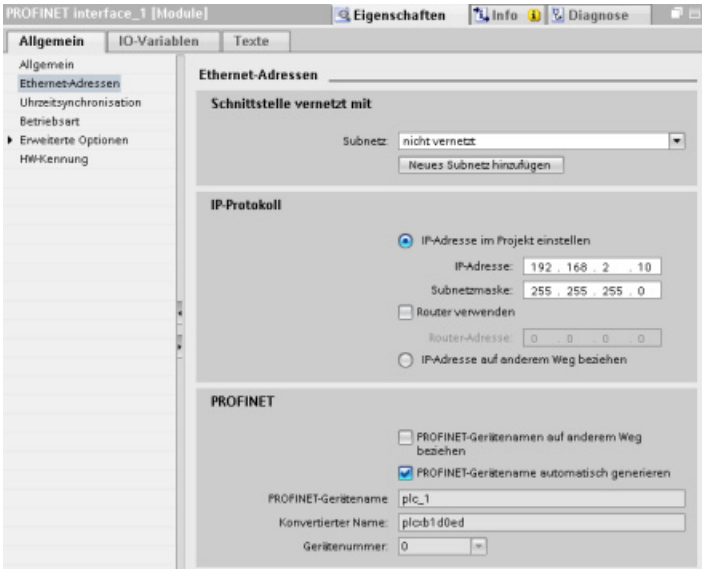
Subnetzmaske: Ein Subnetz ist eine logische Gruppierung miteinander verbundener Netzwerkgeräte. Die Teilnehmer eines Subnetzes befinden sich meist dicht nebeneinander auf einem LAN (Local Area Network). Eine Maske (die Subnetz- oder Netzmaske) legt die Grenzen eines IP-Subnetzes fest.

Eine Subnetzmaske von 255.255.255.0 eignet sich allgemein für ein kleines lokales Netzwerk. Das bedeutet, dass alle IP-Adressen in diesem Netzwerk dieselben ersten 3 Oktette haben müssen. Die einzelnen Geräte in diesem Netzwerk werden dann über das letzte Oktett (8-Bit-Feld) identifiziert. Ein Beispiel hierfür ist die Zuweisung einer Subnetzmaske von 255.255.255.0 und IP-Adressen von 192.168.2.0 bis 192.168.2.255 zu den einzelnen Geräten in einem kleinen lokalen Netzwerk.

Die einzige Verbindung zwischen verschiedenen Subnetzen läuft über einen Router. Werden Subnetze verwendet, so muss ein IP-Router eingesetzt werden.

IP-Router: Router sind die Verbindung zwischen LANs. Mit einem Router kann ein Rechner in einem LAN Meldungen an andere Netzwerke senden, die wiederum zu anderen LANs gehören. Liegt das Ziel der Daten nicht innerhalb des LANs, so leitet der Router die Daten an ein anderes Netzwerk oder eine Gruppen von Netzwerken weiter, wo die Daten ihrem Ziel zugestellt werden können.

Router benötigen IP-Adressen zum Senden und Empfangen von Datenpaketen.



Eigenschaften von IP-Adressen: Wählen Sie im Dialog "Eigenschaften" den Eintrag "Ethernet-Adressen". STEP 7 zeigt den Dialog für die Konfiguration der Ethernet-Adresse an, in dem Sie dem Softwareprojekt die IP-Adresse der CPU zuweisen, in die das Projekt geladen wird.

Tabelle 11- 6 Parameter für die IP-Adresse

Parameter		Beschreibung
Subnetz	Name des Subnetzes, mit dem das Gerät verbunden ist. Um ein neues Subnetz anzulegen, klicken Sie auf Schaltfläche "Neues Subnetz hinzufügen". Die Voreinstellung ist "nicht verbunden". Zwei Arten von Verbindungen sind möglich: • Die Voreinstellung "nicht verbunden" stellt eine lokale Verbindung her. • Wenn Ihr Netzwerk über zwei oder mehr Geräte verfügt, ist ein Subnetz erforderlich.	
IP-Protokoll	IP-Adresse	Zugewiesene IP-Adresse der CPU
	Subnetzmaske	Zugewiesene Subnetzmaske
	IP-Router verwenden	Aktivieren Sie das Optionskästchen, wenn ein IP-Router verwendet wird
	Router-Adresse	Zugewiesene IP-Adresse des Routers, sofern vorhanden

Hinweis

Alle IP-Adressen werden beim Laden des Projekts konfiguriert. Wenn die CPU keine vorkonfigurierte IP-Adresse besitzt, müssen Sie das Projekt mit der MAC-Adresse des Zielgeräts zuweisen. Ist Ihre CPU an einen Router oder ein Netzwerk angeschlossen, so muss auch die IP-Adresse des Routers eingegeben werden.

Über das Optionsfeld "IP-Adresse auf anderem Weg einstellen" können Sie die IP-Adresse online oder über die Anweisung "T_CONFIG (Seite 731)" ändern, nachdem Sie das Programm in die CPU geladen haben. Diese Zuweisungsart der IP-Adresse gilt nur für die CPU.

⚠WARNUNG

Laden einer Hardwarekonfiguration mit "IP-Adresse auf anderem Weg einstellen"

Nach dem Laden einer Hardwarekonfiguration mit aktivierter Option "IP-Adresse auf anderem Weg einstellen" ist es nicht möglich, den Betriebszustand der CPU von RUN in STOP oder von STOP in RUN zu versetzen.

Die Anwendergeräte laufen unter diesen Bedingungen weiter und es kann zu unvorhersehbarem Betrieb der Maschine bzw. des Prozesses kommen, was tödliche oder schwere Verletzungen oder Sachschaden verursachen kann, wenn keine geeigneten Sicherheitsvorkehrungen vorgenommen werden.

Stellen Sie sicher, dass die IP-Adresse(n) Ihrer CPU festgelegt sind, bevor Sie die CPU in einer echten Automatisierungsumgebung einsetzen. Für diese Überprüfung können Sie das Programmierpaket STEP 7, das S7-1200 Tool oder ein angeschlossenes HMI-Gerät in Verbindung mit der Anweisung T_CONFIG nutzen.

⚠WARNUNG

Bedingung, unter der das PROFINET-Netzwerk möglicherweise stoppt

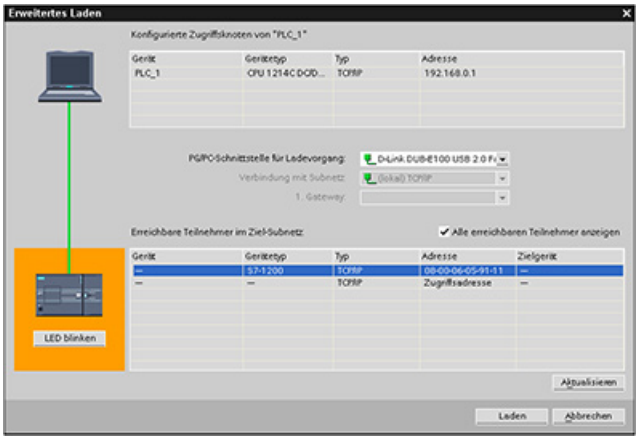
Wenn Sie die IP-Adresse einer CPU online oder über das Anwenderprogramm ändern, kann dies dazu führen, dass das PROFINET-Netzwerk stoppt.

Wird die IP-Adresse einer CPU durch eine IP-Adresse außerhalb des Subnetzes ersetzt, verliert das PROFINET-Netzwerk die Kommunikation und der gesamte Datenaustausch stoppt. Die Anwendergeräte können so konfiguriert werden, dass sie unter diesen Bedingungen weiterarbeiten. Der Verlust der PROFINET-Kommunikation kann zu unvorhersehbarem Betrieb der Maschine bzw. des Prozesses führen und tödliche oder schwere Verletzungen oder Sachschaden verursachen, wenn keine geeigneten Sicherheitsvorkehrungen vorgenommen werden.

Wenn eine IP-Adresse manuell geändert werden muss, stellen Sie sicher, dass sich die neue IP-Adresse innerhalb des Subnetzes befindet.

11.2.4 Testen des PROFINET-Netzwerks

Nachdem die Konfiguration beendet ist, laden Sie das Programm (Seite 211) in die CPU. Alle IP-Adressen werden beim Laden des Projekts konfiguriert.



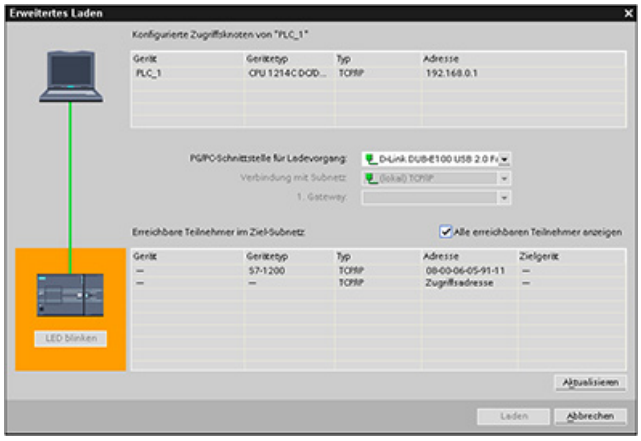
Online-Zuweisung einer IP-Adresse zu einem Gerät

Die S7-1200 CPU hat keine vorkonfigurierte IP-Adresse. Sie müssen der CPU daher manuell eine IP-Adresse zuweisen.

- Wie Sie einem Gerät online eine IP-Adresse zuweisen, erfahren Sie unter "Gerätekonfiguration: IP-Adresse einer CPU online zuweisen" (Seite 645).
- Für die Zuweisung einer IP-Adresse in Ihrem Projekt muss die IP-Adresse in der Gerätekonfiguration eingerichtet, die Konfiguration gespeichert und ins Zielsystem geladen werden. Weitere Informationen finden Sie unter "Gerätekonfiguration: IP-Adresse für eine CPU in Ihrem Projekt konfigurieren" (Seite 646).

Im Dialog "Erweitertes Laden" auf angeschlossene Netzwerkgeräte abfragen

Die Funktion "Laden in Gerät" der S7-1200 CPU und der zugehörigen Dialog "Erweitertes Laden" kann alle erreichbaren Netzwerkgeräte anzeigen und zusätzlich angeben, ob allen Geräten eindeutige IP-Adressen zugeordnet wurden. Um alle erreichbaren und verfügbaren Geräte mit ihren zugeordneten MAC- oder IP-Adressen anzuzeigen, aktivieren Sie das Optionskästchen "Alle ereichbaren Teilnehmer anzeigen".



Ist das erforderliche Netzwerkgerät in dieser Liste nicht enthalten, so wurde die Kommunikation mit diesem Gerät aus irgendeinem Grund unterbrochen. Das Gerät und das Netzwerk sind dann auf Hardware- und/oder Konfigurationsfehler zu überprüfen.

11.2.5 Ermitteln der Ethernet-Adresse (MAC-Adresse) der CPU

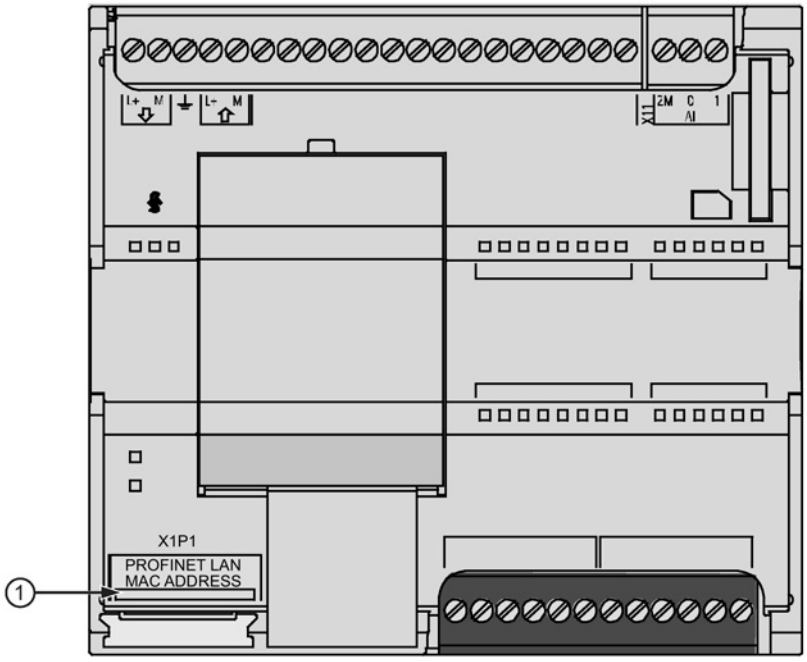
In PROFINET-Netzwerken ist die MAC-Adresse (Media-Access-Control-Adresse) eine Kennung, die den Netzwerkschnittstellen zur eindeutigen Kennzeichnung vom Hersteller zugewiesen wird. Die MAC-Adresse enthält üblicherweise die registrierte Identifikationsnummer des Herstellers.

Das Standardformat (IEEE 802.3) für den Druck von MAC-Adressen als Klartext besteht aus sechs Gruppen zu je zwei Hexadezimalziffern mit Bindestrichen (-) oder Doppelpunkten (:) als Trennzeichen in der Reihenfolge der Übertragung (zum Beispiel 01-23-45-67-89-ab oder 01:23:45:67:89:ab).

Hinweis

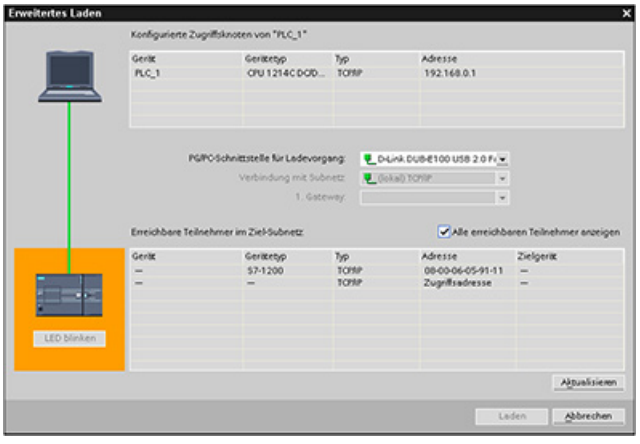
In jeder CPU wird im Werk eine dauerhafte, eindeutige MAC-Adresse eingestellt. Die MAC-Adresse einer CPU können Sie nicht ändern.

Die MAC-Adresse ist an der Vorderseite links unten auf der CPU aufgedruckt. Sie müssen die untere Abdeckklappe öffnen, um die MAC-Adresse lesen zu können.



① MAC-Adresse

Die CPU hat zunächst keine IP-Adresse, sondern nur eine im Werk eingestellte MAC-Adresse. In der PROFINET-Kommunikation ist es erforderlich, dass allen Geräten eine eindeutige IP-Adresse zugeordnet wird.



Über die Funktion "Laden in Gerät" der CPU und den Dialog "Erweitertes Laden" können Sie alle erreichbaren Teilnehmer im Netzwerk anzeigen und zusätzlich prüfen, ob allen Geräten eindeutige IP-Adressen zugeordnet wurden. Dieser Dialog zeigt alle erreichbaren und verfügbaren Geräte mit ihren zugeordnen MAC- oder IP-Adressen an. MAC-Adressen sind besonders wichtig, um Geräte zu identifizieren, die keine eindeutige IP-Adresse haben.

11.2.6 Konfigurieren der NTP-Synchronisierung

⚠ WARNUNG

Wenn ein Angreifer über die NTP-Synchronisierung (Network Time Protocol) auf Ihre Netzwerke zugreifen kann, kann er möglicherweise durch Veränderung der CPU-Systemzeit in gewissem Rahmen Ihren Prozess kontrollieren.

Die NTP-Client-Funktion der S7-1200 CPU ist standardmäßig deaktiviert und ermöglicht im aktivierten Zustand lediglich konfigurierten IP-Adressen, als NTP-Server zu fungieren. Die CPU deaktiviert diese Funktion standardmäßig, und Sie müssen die Funktion konfigurieren, um ferngesteuerte Korrekturen der CPU-Systemzeit zu gestatten.

Die S7-1200 CPU unterstützt Uhrzeitalarme und Uhrzeitoperationen, die von einer korrekten CPU-Systemzeit abhängig sind. Wenn Sie NTP konfigurieren und die Uhrzeitsynchronisierung von einem Server akzeptieren, müssen Sie sicherstellen, dass es sich bei dem Server um eine vertrauenswürdige Quelle handelt. Ist dies nicht der Fall, kann es zu einer Sicherheitsverletzung kommen, bei der ein unbekannter Benutzer die CPU-Systemzeit verändern und dadurch in gewissem Rahmen Ihren Prozess kontrollieren kann.

Informationen und Empfehlungen bezüglich der Sicherheit finden Sie in unseren Operational Guidelines für Industrial Security (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf) auf der Website "Service & Support" von Siemens.

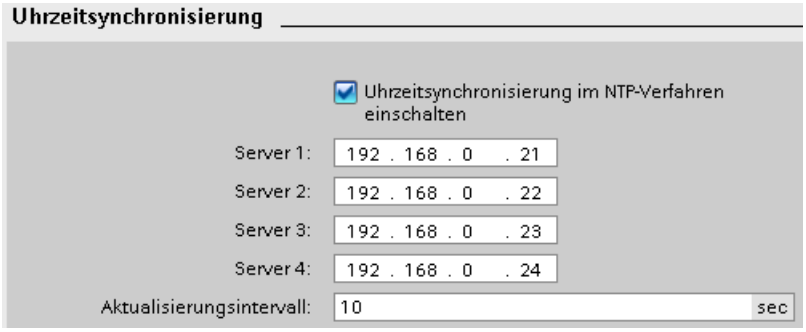
NTP (Network Time Protocol) ist weit verbreitet und dient der Synchronisierung der Uhren von Computersystemen mit Uhrzeitservern im Internet. Beim NTP-Verfahren sendet die CPU in regelmäßigen Zeitabständen Uhrzeitanfragen (im Client-Modus) an den NTP-Server im Subnetz (LAN). Anhand der Antworten der Server wird die zuverlässigste und genaueste Uhrzeit ermittelt und die Uhrzeit der Station synchronisiert.

Der Vorteil dieses Verfahrens ist, dass es Zeit für die subnetzübergreifende Synchronisierung gestattet.

Es müssen die IP-Adressen von bis zu vier NTP-Servern konfiguriert werden. Das Aktualisierungsintervall legt den Zeitraum zwischen den Uhrzeitabfragen fest (in Sekunden). Der Wert des Intervalls liegt zwischen 10 Sekunden und einem Tag.

Beim NTP-Verfahren wird im Allgemeinen UTC (Universal Time Coordinated) übertragen, dies entspricht GMT (Greenwich Mean Time).

Wählen Sie im Eigenschaftenfenster den Eintrag "Uhrzeitsynchronisation". In STEP 7 wird der Dialog "Uhrzeitsynchronisation" angezeigt:



Hinweis

Alle IP-Adressen werden beim Laden des Projekts konfiguriert.

Tabelle 11- 7 Parameter für die Uhrzeitsynchronisation

Parameter	Definition
Uhrzeitsynchronisation mit NTP-Servern freigeben	Aktivieren Sie das Optionskästchen, um die Uhrzeitsynchronisation mit NTP-Servern freizugeben.
Server 1	Zugewiesene IP-Adresse für Netzwerkzeit-Server 1
Server 2	Zugewiesene IP-Adresse für Netzwerkzeit-Server 2
Server 3	Zugewiesene IP-Adresse für Netzwerkzeit-Server 3
Server 4	Zugewiesene IP-Adresse für Netzwerkzeit-Server 4
Intervall Uhrzeitsynchronisation	Intervallwert (s)

11.2.7 Anlaufzeit, Benennung und Adresszuweisung von PROFINET-Geräten

PROFINET IO kann die Anlaufzeit für Ihr System verlängern (konfigurierbarer Timeout-Wert). Ein größere Anzahl von Geräten und langsame Geräte wirken sich auf die Zeitdauer aus, die die Umschaltung in RUN benötigt.

Bei V4.0 und späteren Versionen sind maximal 16 PROFINET IO-Geräte im S7-1200 PROFINET-Netzwerk möglich.

Jede Station (bzw. IO-Gerät) startet beim Anlauf unabhängig und dies wirkt sich auf die Gesamtanlaufzeit der CPU aus. Wenn Sie den konfigurierbaren Timeout-Wert zu niedrig setzen, ist möglicherweise nicht genügend Gesamtanlaufzeit der CPU vorhanden, damit alle Stationen den Anlauf beenden. Tritt diese Situation ein, kommt es zu falschen Stationsfehlern.

Die "Zuweisungszeit der Parameter für dezentrale E/A" (Timeout) finden Sie in den CPU-Eigenschaften unter "Anlauf". Der Standardwert für den konfigurierbaren Timeout ist 60.000 ms (1 Minute). Diese Zeit lässt sich vom Anwender konfigurieren.

Namen und Adressen von PROFINET-Geräten in STEP 7

Alle PROFINET-Geräte **müssen** einen Gerätenamen und eine IP-Adresse haben. Sie legen die Gerätenamen und die IP-Adressen in STEP 7 fest. Die Gerätenamen werden über PROFINET DCP (Discovery and Configuration Protocol, Erkennungs- und Konfigurationsprotokoll) in die IO-Geräte geladen.

PROFINET-Adresszuweisung beim Anlauf des Systems

Die Steuerung überträgt die Namen der Geräte zum Netzwerk, und die Geräte reagieren mit ihren MAC-Adressen. Die Steuerung weist dem Gerät dann über das PROFINET DCP-Protokoll eine IP-Adresse zu:

- Wenn die MAC-Adresse eine konfigurierte IP-Adresse hat, führt die Station den Anlauf durch.
- Wenn die MAC-Adresse keine konfigurierte IP-Adresse hat, weist STEP 7 die im Projekt konfigurierte Adresse zu und die Station führt dann den Anlauf durch.
- Gibt es bei diesem Vorgang ein Problem, tritt ein Stationsfehler auf und es findet kein Anlauf statt. Diese Situation verursacht, dass der konfigurierbare Timeout-Wert überschritten wird.

11.2.8 Offene Benutzerkommunikation

11.2.8.1 Protokolle

Der integrierte PROFINET-Port der CPU unterstützt mehrere Kommunikationsstandards über ein Ethernet-Netzwerk:

- Transport Control Protocol (TCP)
- ISO on TCP (RFC 1006)
- User Datagram Protocol (UDP)

Tabelle 11- 8 Protokolle und Kommunikationsanweisungen

Protokoll	Verwendungsbeispiele	Eintragen der Daten in den Empfangsbereich	Kommunikationsanweisungen	Adressierungsart
TCP	CPU-zu-CPU-Kommunikation Transport von Telegrammen	Ad-hoc-Modus	Nur TRCV_C und TRCV (V4.1 und ältere Anweisungen)	Weist den lokalen Geräten (aktiv) und Partnergeräten (passiv) Portnummern zu
		Datenempfang mit angegebener Länge	TSEND_C, TRCV_C, TCON, TDISCON, TSEND und TRCV(V4.1 und ältere Anweisungen)	
ISO on TCP	CPU-zu-CPU-Kommunikation Fragmentierung und Zusammensetzung von Meldungen	Ad-hoc-Modus	Nur TRCV_C und TRCV (V4.1 und ältere Anweisungen)	Weist den lokalen Geräten (aktiv) und Partnergeräten (passiv) TSAPs zu
		Protokollgesteuert	TSEND_C, TRCV_C, TCON, TDISCON, TSEND und TRCV (V4.1 und ältere Anweisungen)	
UDP	CPU-zu-CPU-Kommunikation Kommunikation im Anwenderprogramm	User Datagram Protocol	TUSEND und TURCV	Weist den lokalen Geräten (aktiv) und Partnergeräten (passiv) Portnummern zu, es handelt sich jedoch nicht um eine dedizierte Verbindung
S7-Kommunikation	CPU-zu-CPU-Kommunikation Daten aus einer CPU lesen/in eine CPU schreiben	Datenübertragung und -empfang mit angegebener Länge	GET und PUT	Weist den lokalen Geräten (aktiv) und Partnergeräten (passiv) TSAPs zu
PROFINET IO	Kommunikation zwischen CPU und PROFINET IO-Gerät	Datenübertragung und -empfang mit angegebener Länge	Integriert	Integriert

11.2.8.2 TCP und ISO on TCP

Transport Control Protocol (TCP) ist ein Standardprotokoll, beschrieben von RFC 793: Transmission Control Protocol. Wesentlicher Zweck des TCP ist die Bereitstellung einer zuverlässigen, sicheren Verbindung zwischen zwei Paaren von Prozessen. Dieses Protokoll hat die folgenden Merkmale:

- Effizientes Kommunikationsprotokoll, da es eng mit der Hardware verknüpft ist
- Geeignet für mittelgroße bis große Datenmengen (bis 8192 Byte)
- Bietet deutlich mehr Leistungsmerkmale für Anwendungen, insbesondere Wiederherstellung im Fehlerfall, Flusskontrolle und Zuverlässigkeit
- Ein verbindungsorientiertes Protokoll
- Lässt sich sehr flexibel mit Fremdsystemen einsetzen, die nur TCP unterstützen
- Routing-fähig
- Es gelten nur statische Datenlängen.
- Meldungen werden quittiert.
- Anwendungen werden über Portnummern adressiert.
- Die meisten Protokolle von Benutzeranwendungen wie TELNET und FTP nutzen TCP.
- Für die Datenverwaltung ist wegen der Programmierschnittstelle SEND/RECEIVE Programmieraufwand erforderlich.

International Standards Organization (ISO) on Transport Control Protocol (TCP) (RFC 1006) (ISO on TCP) ist ein Verfahren, das es ermöglicht, ISO-Anwendungen ins TCP/IP-Netzwerk zu übernehmen. Dieses Protokoll hat die folgenden Merkmale:

- Eng mit der Hardware verknüpftes effizientes Kommunikationsprotokoll
- Geeignet für mittelgroße bis große Datenmengen (bis 8192 Byte)
- Im Gegensatz zu TCP sind die Meldungsfunktion und die Datenendeerkennung meldungsorientiert.
- Routing-fähig, Einsatz in WAN möglich
- Dynamische Datenlängen sind möglich.
- Für die Datenverwaltung ist wegen der Programmierschnittstelle SEND/RECEIVE Programmieraufwand erforderlich.

Mit Transport Service Access Points (TSAPs) gestattet das TCP-Protokoll mehrere Verbindungen mit einer einzigen IP-Adesse (bis zu 64-K-Verbindungen). Bei RFC 1006 ermitteln TSAPs die eindeutige Zuordnung dieser Verbindungen der Kommunikationsendpunkte zu einer IP-Adresse.

11.2.8.3 Kommunikationsdienste und verwendete Portnummern

Die S7-1200 CPU unterstützt die in der nachstehenden Tabelle aufgeführten Protokolle. Bei jedem Protokoll weist die CPU die Adressparameter, die entsprechende Kommunikationsschicht sowie die Kommunikationsrolle und die Kommunikationsrichtung zu.

Diese Informationen ermöglichen die Anpassung der Sicherheitsmaßnahmen zum Schutz des Automatisierungssystem an die verwendeten Protokolle (z. B. Firewall). Nur die Ethernet- und PROFINET-Netzwerke haben Sicherheitsmaßnahmen. Da PROFIBUS keine Sicherheitsmaßnahmen bietet, enthält die Tabelle keine PROFIBUS-Protokolle.

Die nachstehende Tabelle zeigt die unterschiedlichen Schichten und Protokolle, die von der CPU verwendet werden:

Protokoll	Portnummer	(2) Sicherungsschicht (4) Transportschicht	Funktion	Beschreibung
PROFINET-Protokolle				
DCP (Discovery and Configuration Protocol, Erkennungs- und Konfigurationsprotokoll)	Nicht relevant	(2) Ethernet II und IEEE 802.1Q und Ethertype 0x8892 (PROFINET)	Erreichbare Teilnehmer PROFINET-Erkennung und -Konfiguration	PROFINET verwendet DCP, um Teilnehmer zu erkennen und Grundeinstellungen bereitzustellen. DCP verwendet die besondere Multicast-MAC-Adresse: xx-xx-xx-01-0E-CF, xx-xx-xx = Organizationally Unique Indentifier (Herstellerkennung)
LLDP (Link Layer Discovery Protocol)	Nicht relevant	(2) Ethernet II und IEEE 802.1Q und Ethertype 0x88CC (PROFINET)	PROFINET Link Layer Discovery Protocol	PROFINET verwendet LLDP, um Beziehungen zwischen benachbarten PROFINET-Geräten zu erkennen und zu verwalten. LLDP verwendet die besondere Multicast-MAC-Adresse: 01-80-C2-00-00-0E

11.2.8.4 Ad-hoc-Modus

Typischerweise empfangen die Protokolle TCP und ISO-on-TCP Datenpakete mit fest angegebener Länge von 1 bis 8192 Byte. Die Kommunikationsanweisungen TRCV_C und TRCV jedoch bieten auch einen Ad-hoc-Kommunikationsmodus, in dem Datenpakete variabler Länge von 1 bis 1472 Byte empfangen werden können.

Hinweis

Wenn Sie die Daten in einem "optimierten" DB (nur symbolisch adressierbar) speichern, können Sie Daten nur in Arrays der Datentypen Byte, Char, USInt und SInt empfangen.

Um die Anweisung TRCV_C oder TRCV für den Ad-hoc-Modus zu konfigurieren, setzen Sie den Eingangsparameter ADHOC der Anweisung.

Wenn Sie die Anweisung TRCV_C oder TRCV nicht sehr häufig im Ad-hoc-Modus aufrufen, können Sie in einem Aufruf mehrere Pakete empfangen. Beispiel: Wenn Sie fünf 100-Byte-Pakete in einem Aufruf empfangen möchten, liefert TCP diese fünf Pakete als ein 500-Byte-Paket, dagegen unterteilt ISO-on-TCP die Pakete in fünf 100-Byte-Pakete.

11.2.8.5 Verbindungs-IDs für Anweisungen für die offene Benutzerkommunikation

Wenn Sie die PROFINET-Anweisung TSEND_C, TRCV_C oder TCON in Ihr Anwenderprogramm einfügen, erstellt STEP 7 einen Instanz-DB für die Konfiguration des Kommunikationskanals (oder der Verbindung) zwischen den Geräten. Die Parameter der Verbindung konfigurieren Sie in den "Eigenschaften" (Seite 638) der Anweisung. Unter den Parametern ist auch die Verbindungs-ID der Verbindung.

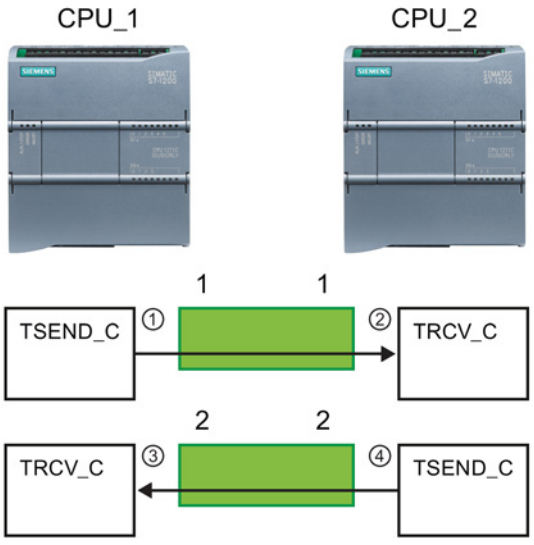
- Die Verbindungs-ID muss für die CPU eindeutig sein. Jede Verbindung, die Sie erstellen, benötigt einen anderen DB und eine andere Verbindungs-ID.
- Die lokale CPU und die Partner-CPU können dieselbe Verbindungs-ID für dieselbe Verbindung verwenden, doch die Verbindungs-IDs müssen nicht übereinstimmen. Die Verbindungs-ID ist nur für die PROFINET-Anweisungen innerhalb des Anwenderprogramms der jeweiligen CPU relevant.
- Für die Verbindungs-ID der CPU können Sie jede beliebige Nummer verwenden. Wenn Sie die Verbindungs-IDs jedoch sequenziell von "1" an vergeben, können Sie leichter die Anzahl der verwendeten Verbindungen für eine bestimmte CPU nachvollziehen.

Hinweis

Jede Anweisung TSEND_C, TRCV_C oder TCON in Ihrem Anwenderprogramm erstellt eine neue Verbindung. Es ist wichtig, dass Sie für jede Verbindung die korrekte Verbindungs-ID verwenden.

Das folgende Beispiel zeigt die Kommunikation zwischen zwei CPUs, die zwei getrennte Verbindungen zum Senden und Empfangen von Daten nutzen.

- Die Anweisung TSEND_C in CPU_1 bezieht sich auf die Anweisung TRCV_C in CPU_2 über die erste Verbindung ("Verbindungs-ID 1" bei beiden CPUs, CPU_1 und CPU_2).
- Die Anweisung TRCV_C in CPU_1 bezieht sich auf die Anweisung TSEND_C in CPU_2 über die zweite Verbindung ("Verbindungs-ID 2" bei beiden CPUs, CPU_1 und CPU_2).



① TSEND_C in CPU_1 erstellt eine Verbindung und weist dieser Verbindung eine Verbindungs-ID zu (Verbindungs-ID 1 bei CPU_1).

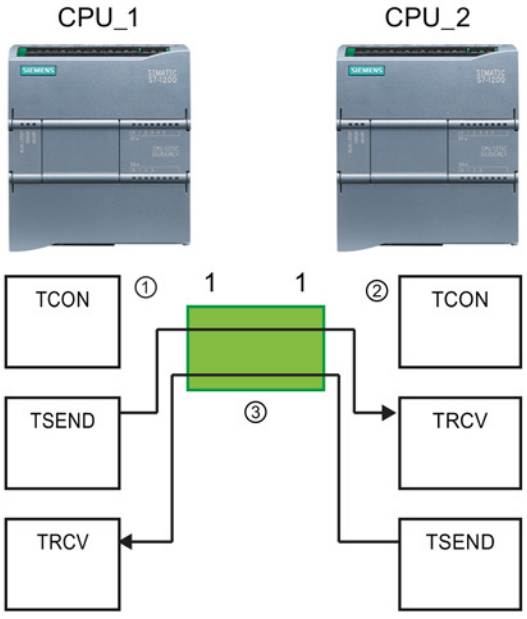
② TRCV_C in CPU_2 erstellt die Verbindung für CPU_2 und weist die Verbindungs-ID zu (Verbindungs-ID 1 bei CPU_2).

③ TRCV_C in CPU_1 erstellt eine zweite Verbindung für CPU_1 und weist dieser Verbindung eine andere Verbindungs-ID zu (Verbindungs-ID 2 bei CPU_1).

④ TSEND_C in CPU_2 erstellt eine zweite Verbindung und weist dieser Verbindung eine andere Verbindungs-ID zu (Verbindungs-ID 2 bei CPU_2).

Das folgende Beispiel zeigt die Kommunikation zwischen zwei CPUs, die nur eine Verbindung zum Senden und Empfangen von Daten nutzen.

- Jede CPU nutzt eine Anweisung TCON, um die Verbindung zwischen den beiden CPUs zu konfigurieren.
- Die Anweisung TSEND in CPU_1 bezieht sich auf die Anweisung TRCV in CPU_2 über die Verbindungs-ID ("Verbindungs-ID 1"), die von der Anweisung TCON in CPU_1 konfiguriert wurde. Die Anweisung TRCV in CPU_2 bezieht sich auf die Anweisung TSEND in CPU_1 über die Verbindungs-ID ("Verbindungs-ID 1"), die von der Anweisung TCON in CPU_2 konfiguriert wurde.
- Die Anweisung TSEND in CPU_2 bezieht sich auf die Anweisung TRCV in CPU_1 über die Verbindungs-ID ("Verbindungs-ID 1"), die von der Anweisung TCON in CPU_2 konfiguriert wurde. Die Anweisung TRCV in CPU_1 bezieht sich auf die Anweisung TSEND in CPU_2 über die Verbindungs-ID ("Verbindungs-ID 1"), die von der Anweisung TCON in CPU_1 konfiguriert wurde.



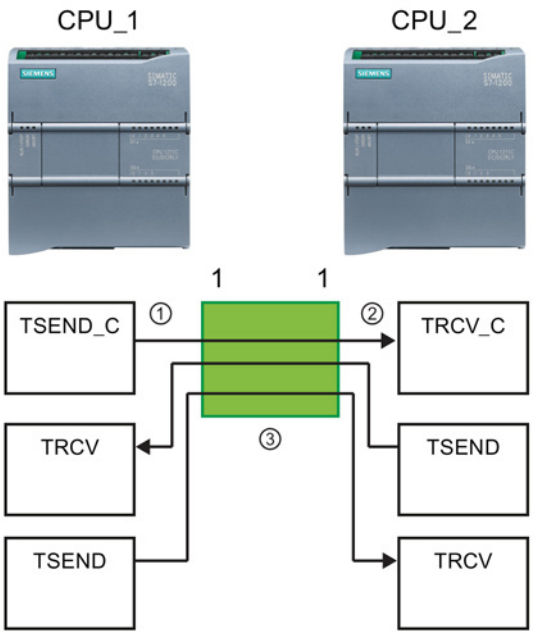
① TCON in CPU_1 erstellt eine Verbindung und weist dieser Verbindung eine Verbindungs-ID in CPU_1 zu (ID = 1).

② TCON in CPU_2 erstellt eine Verbindung und weist dieser Verbindung eine Verbindungs-ID in CPU_2 zu (ID = 1).

③ TSEND und TRCV in CPU_1 nutzen die von TCON in CPU_1 erstellte Verbindungs-ID (ID = 1).

TSEND und TRCV in CPU_2 nutzen die von TCON in CPU_2 erstellte Verbindungs-ID (ID = 1).

Wie das folgende Beispiel zeigt, können Sie auch mit einzelnen Anweisungen TSEND und TRCV über eine von einer Anweisung TSEND_C oder TRCV_C erstellte Verbindung kommunizieren. Die Anweisungen TSEND und TRCV erstellen selbst keine neue Verbindung, deshalb müssen sie den DB und die Verbindungs-ID nutzen, die von einer Anweisung TSEND_C, TRCV_C oder TCON erstellt wurden.



① TSEND_C in CPU_1 erstellt eine Verbindung und weist dieser Verbindung eine Verbindungs-ID zu (ID = 1).

② TRCV_C in CPU_2 erstellt eine Verbindung und weist dieser Verbindung eine Verbindungs-ID in CPU_2 zu (ID = 1).

③ TSEND und TRCV in CPU_1 nutzen die von TSEND_C in CPU_1 erstellte Verbindungs-ID (ID = 1).

TSEND und TRCV in CPU_2 nutzen die von TRCV_C in CPU_2 erstellte Verbindungs-ID (ID = 1).

11.2.8.6 Parameter für die PROFINET-Verbindung

Bei den Anweisungen TSEND_C, TRCV_C und TCON müssen verbindungsbezogene Parameter angegeben werden, um eine Verbindung zum Partnergerät aufbauen zu können. Diese Parameter werden von der Struktur TCON_Param für die Protokolle TCP, ISO-on-TCP und UDP zugewiesen. Üblicherweise geben Sie diese Parameter in den "Eigenschaften" der Anweisung im Register "Konfiguration" (Seite 638) an. Kann auf das Register "Konfiguration" nicht zugegriffen werden, müssen Sie die Struktur TCON_Param programmatisch angeben.

In V4.1 parametriert die Struktur TCON_IP_V4 das TCP-Protokoll und die Struktur TCON_IP_RFC parametriert das ISO-on-TCP-Protokoll.

TCON_Param

Tabelle 11- 9 Struktur der Verbindungsbeschreibung (TCON_Param)

Byte	Parameter und Datentyp		Beschreibung
0 … 1	block_length	UInt	Länge: 64 Bytes (fest)
2 … 3	id	CONN_OUC (Word)	Referenz auf diese Verbindung: Wertebereich: 1 (Standard) bis 4095. Geben Sie den Wert dieses Parameters für die Anweisung TSEND_C, TRCV_C oder TCON unter ID an.
4	connection_type	USInt	Verbindungstyp: • 17: TCP (Standard) • 18: ISO-on-TCP • 19: UDP
5	active_est	Bool	ID der Verbindungsart: • TCP und ISO-on-TCP: – FALSCH: Passive Verbindung – WAHR: Aktive Verbindung (Standard) • UDP: FALSCH
6	local_device_id	USInt	ID der lokalen PROFINET- oder Industrial Ethernet-Schnittstelle: 1 (Standard)
7	local_tsap_id_len	USInt	Länge des Parameters local_tsap_id in Bytes; mögliche Werte: • TCP: 0 (aktiv, Standard) oder 2 (passiv) • ISO-on-TCP: 2 bis 16 • UDP: 2
8	rem_subnet_id_len	USInt	Dieser Parameter wird nicht verwendet.
9	rem_staddr_len	USInt	Länge der Adresse des Partnerendpunkts in Bytes: • 0: nicht angegeben (Parameter rem_staddr ist irrelevant) • 4 (Standard): Gültige IP-Adresse im Parameter rem_staddr (nur bei TCP und ISO-on-TCP)

Byte	Parameter und Datentyp		Beschreibung
10	rem_tsap_id_len	USInt	Länge des Parameters rem_tsap_id in Bytes; mögliche Werte: • TCP: 0 (passiv) oder 2 (aktiv, Standard) • ISO-on-TCP: 2 bis 16 • UDP: 0
11	next_staddr_len	USInt	Dieser Parameter wird nicht verwendet.
12 … 27	local_tsap_id	Array [1..16] of Byte	Komponente der lokalen Adresse der Verbindung: • TCP und ISO-on-TCP: lokale Port-Nr. (mögliche Werte: 1 bis 49151; empfohlene Werte: 2000...5000): – local_tsap_id[1] = High Byte der Portnummer in Hexadezimalnotierung; – local_tsap_id[2] = Low Byte der Portnummer in Hexadezimalnotierung; – local_tsap_id[3-16] = irrelevant • ISO-on-TCP: lokale TSAP-ID: – local_tsap_id[1] = B#16#E0; – local_tsap_id[2] = Baugruppenträger und Steckplatz der lokalen Endpunkte (Bits 0 bis 4: Steckplatznummer, Bits 5 bis 7: Nummer des Baugruppenträgers); – local_tsap_id[3-16] = TSAP-Erweiterung, optional • UDP: Dieser Parameter wird nicht verwendet. Hinweis: Stellen Sie sicher, dass jeder Wert von local_tsap_id innerhalb der CPU eindeutig ist.
28 … 33	rem_subnet_id	Array [1..6] of USInt	Dieser Parameter wird nicht verwendet.
34 … 39	rem_staddr	Array [1..6] of USInt	Nur bei TCP und ISO-on-TCP: IP-Adresse des Partnerendpunkts. (Nicht relevant bei passiven Verbindungen.) Beispiel: Die IP-Adresse 192.168.002.003 wird in den folgenden Elementen des Arrays gespeichert: rem_staddr[1] = 192 rem_staddr[2] = 168 rem_staddr[3] = 002 rem_staddr[4] = 003 rem_staddr[5-6] = irrelevant

Byte	Parameter und Datentyp		Beschreibung
40 … 55	rem_tsap_id	Array [1..16] of Byte	Komponente der Partneradresse der Verbindung: • TCP: Portnummer der Partner-CPU. Bereich: 1 bis 49151; empfohlene Werte: 2000 bis 5000): – rem_tsap_id[1] = High Byte der Portnummer in Hexadezimalnotierung; – rem_tsap_id[2] = Low Byte der Portnummer in Hexadezimalnotierung; – rem_tsap_id[3-16] = irrelevant • ISO-on-TCP: Partner-TSAP-ID: – rem_tsap_id[1] = B#16#E0 – rem_tsap_id[2] = Baugruppenträger und Steckplatz des Partnerendpunkts (Bits 0 bis 4: Steckplatznummer, Bits 5 bis 7: Nummer des Baugruppenträgers) – rem_tsap_id[3-16] = TSAP-Erweiterung, optional • UDP: Dieser Parameter wird nicht verwendet.
56 … 61	next_staddr	Array [1..6] of Byte	Dieser Parameter wird nicht verwendet.
62 … 63	spare	Word	Reserviert: W#16#0000

TCON_IP_V4

Tabelle 11- 10 Struktur der Verbindungsbeschreibung (TCON_IP_V4): Zur Verwendung mit TCP

Byte	Parameter und Datentyp		Beschreibung
0 … 1	InterfaceId	HW_ANY	HW-Kennung des IE-Schnittstellen-Submoduls
2 … 3	ID	CONN_OUC (Word)	Verweis auf diese Verbindung: Wertebereich: 1 (Standard) bis 4095. Geben Sie den Wert dieses Parameters für die Anweisung TSEND_C, TRCV_C oder TCON unter ID an.
4	ConnectionType	Byte	Verbindungstyp: • 11: TCP/IP (Standard) • 17: TCP/IP (Dieser Verbindungstyp ist aus Gründen der Anbindung älterer Systeme vorhanden. Empfohlen wird die Verwendung von "11: TCP/IP (Standard)".) • 19: UDP
5	ActiveEstablished	Bool	Aktiver/passiver Verbindungsaufbau: • WAHR: Aktive Verbindung (Standard) • FALSCH: Passive Verbindung
	V4 IP-Adresse		
6	ADDR[1]	Byte	Oktett 1
7	ADDR[1]	Byte	Oktett 2
8	ADDR[1]	Byte	Oktett 3
9	ADDR[1]	Byte	Oktett 4

Byte	Parameter und Datentyp		Beschreibung
10 ... 11	RemotePort	UInt	Dezentrale UDP/TCP-Portnummer
12 ... 13	LocalPort	UInt	Lokale UDP/TCP-Portnummer

TCON_IP_RFC

Tabelle 11- 11 Struktur der Verbindungsbeschreibung (TCON_IP_RFC): Zur Verwendung mit ISO-on-TCP

Byte	Parameter und Datentyp		Beschreibung
0 … 1	InterfaceId	HW_ANY	HW-Kennung des IE-Schnittstellen-Submoduls
2 … 3	ID	CONN_OUC (Word)	Verweis auf diese Verbindung: Wertebereich: 1 (Standard) bis 4095. Geben Sie den Wert dieses Parameters für die Anweisung TSEND_C, TRCV_C oder TCON unter ID an.
4	ConnectionType	Byte	Verbindungstyp: • 12: ISO-on-TCP (Standard) • 17: ISO-on-TCP (Dieser Verbindungstyp ist aus Gründen der Anbindung älterer Systeme vorhanden. Empfohlen wird die Verwendung von "12: ISO-on-TCP (Standard)".)
5	ActiveEstablished	Bool	Aktiver/passiver Verbindungsaufbau: • WAHR: Aktive Verbindung (Standard) • FALSCH: Passive Verbindung
6 ... 7	Reserve		Nicht verwendet
	V4 IP-Adresse		
8	ADDR[1]	Byte	Oktett 1
9	ADDR[1]	Byte	Oktett 2
10	ADDR[1]	Byte	Oktett 3
11	ADDR[1]	Byte	Oktett 4
	Auswahl dezentrale Übertragung		
12 ... 13	TSelLength	UInt	Länge von TSelector
14 ... 45	TSel	array [1..32] of Byte	Zeichenfeld für TSAP-Name
	Auswahl lokale Übertragung		
46 ... 47	TSelLength	UInt	Länge von TSelector
48 ... 79	TSel	array [1..32] of Byte	Zeichenfeld für TSAP-Name

11.2.8.7 Anweisungen TSEND_C und TRCV_C

Version V4.1 der S7-1200 CPU zusammen mit STEP 7 V13 SP1 erweitert die Fähigkeit der Anweisungen TSEND_C und TRCV_C dahingehend, dass Verbindungsparameter mit Strukturen nach "TCON_IP_v4" und "TCON_IP_RFC" verwendet werden können.

Die S7-1200 unterstützt daher zwei Sätze TSEND_C- und TRCV_C-Anweisungen:

- Ältere TSEND_C- und TRCV_C-Anweisungen (Seite 678): Diese TSEND_C- und TRCV_C-Anweisungen wurden vor der Version V4.0 der S7-1200 verwendet und funktionieren nur mit Verbindungsparametern mit Strukturen nach "TCON_Param".
- TSEND_C- und TRCV_C-Anweisungen (Seite 668): Diese TSEND_C- und TRCV_C-Anweisungen stellen alle Funktionen der früheren Anweisungen bereit und zusätzlich die Möglichkeit zur Nutzung von Verbindungsparametern mit Strukturen nach "TCON_IP_v4" und "TCON_IP_RFC".

Version der Anweisungen TSEND_C und TRCV_C auswählen

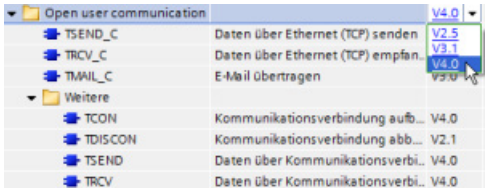
Es gibt zwei Versionen der Anweisungen TSEND_C und TRCV_C in STEP 7:

- Die Versionen 2.5 und 3.1 waren in STEP 7 Basic/Professional V13 und früher verfügbar.
- Version 4.0 ist in STEP 7 Basic/Professional V13 SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Verwenden Sie niemals Anweisungen unterschiedlicher Versionen zusammen in einem CPU-Programm.

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.



Um die Version einer TSEND_C- und TRCV_C-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine TSEND_C- oder TRCV_C-Anweisung über das Anweisungsverzeichnis in Ihr Programm platzieren, wird je nach der ausgewählten TSEND_C- oder TRCV_C-Anweisung eine neue FB- oder FC-Instanz in der Projektnavigation angelegt. Die neue FB- oder FC-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer TSEND_C- oder TRCV_C-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation eine FB- oder FC-Instanz einer TSEND_C- oder TRCV_C Anweisung aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der TSEND_C- oder TRCV_C-Anweisung anzuzeigen.

Anweisungen TSEND_C und TRCV_C (Daten über Ethernet senden und empfangen)

Die Anweisung TSEND_C verbindet die Funktionen der Anweisungen TCON, TDISCON und TSEND . Die Anweisung TRCV_C verbindet die Funktionen der Anweisungen TCON, TDISCON und TRCV. (Weitere Informationen zu diesen Anweisungen finden Sie unter "TCON, TDISCON, TSEND UND TRCV (Seite 688)".)

Die Mindestdatengröße, die gesendet (TSEND_C) oder empfangen (TRCV_C) werden kann, ist ein Byte; die maximale Datengröße beträgt 8192 Byte. TSEND_C unterstützt nicht die Datenübertragung von Booleschen Adressen und TRCV_C empfängt Daten nicht in Booleschen Adressen. Weitere Informationen zum Übertragen von Daten mit diesen Anweisungen finden Sie im Abschnitt zur Datenkonsistenz (Seite 191).

Hinweis

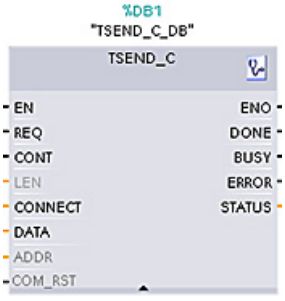
Initialisierung der Kommunikationsparameter

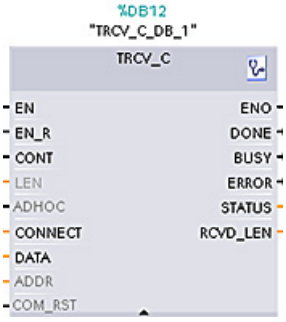
Nachdem Sie die Anweisung TSEND_C oder TRCV_C eingefügt haben, konfigurieren Sie in den "Eigenschaften" der Anweisung (Seite 638) die Kommunikationsparameter (Seite 663). Wenn Sie die Parameter für die Kommunikationspartner im Inspektorfenster eingeben, gibt STEP 7 die entsprechenden Daten in den DB der Anweisung ein.

Wenn Sie einen Multiinstanz-DB verwenden möchten, müssen Sie den DB manuell auf beiden CPUs konfigurieren.

Tabelle 11- 12 Anweisungen TSEND_C und TRCV_C

KOP/FUP	SCL	Beschreibung

	`"TSEND_C_DB"(`
	`req:=_bool_in_,`
	`cont:=_bool_in_,`
	`len:=_uint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`connect:=_struct_inout_,`
	`data:=_variant_inout_,`
	`com_rst:=_bool_inout_);`
		TSEND_C stellt eine TCP- oder ISO-on-TCP-Verbindung zu einem Partner her, sendet Daten und kann die Verbindung auch wieder beenden. Nach dem Einrichten und Aufbauen der Verbindung wird diese automatisch von der CPU gehalten und überwacht.

	`"TRCV_C_DB"(`
	`en_r:=_bool_in_,`
	`cont:=_bool_in_,`
	`len:=_uint_in_,`
	`adhoc:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`rcvd_len=>_uint_out_,`
	`connect:=_struct_inout_,`
	`data:=_variant_inout_,`
	`com_rst:=_bool_inout_);`
		TRCV_C stellt eine TCP- oder ISO on TCP-Verbindung zu einer CPU her, empfängt Daten und kann die Verbindung auch wieder beenden. Nach dem Einrichten und Aufbauen der Verbindung wird diese automatisch von der CPU gehalten und überwacht.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 11- 13 Datentypen für die Parameter von TSEND_C und TRCV_C

Parameter und Datentyp		Datentyp	Beschreibung
REQ (TSEND_C)	IN	Bool	Startet den Sendeauftrag bei einer steigenden Flanke
EN_R (TRCV_C)	IN	Bool	Empfang aktivieren
CONT	IN	Bool	Steuert die Kommunikationsverbindung: • 0: Verbindung automatisch herstellen. Kommunikationsverbindung nach dem Senden der Daten beenden. • 1: Kommunikationsverbindung nach dem Empfang der Daten beenden. Der Parameter CONT wird nur bei einer steigenden Flanke an REQ ausgewertet oder wenn COM_RST = "1".
LEN	IN	UDInt	Optionaler Parameter (versteckt) Maximale Bytezahl, die mit dem Auftrag gesendet (TSEND_C) oder empfangen (TRCV_C) werden soll. Wenn Sie am Parameter DATA rein symbolische Werte verwenden, muss der Parameter LEN den Wert 0 haben.

Parameter und Datentyp		Datentyp	Beschreibung
ADHOC (TRCV_C)	IN	Bool	Optionaler Parameter (versteckt) Ad-hoc-Modus-Anforderung für Verbindungstyp TCP.
CONNECT	IN_OUT	TCON_Param	Pointer auf die Verbindungsbeschreibung, die der zu beschreibenden Struktur der Verbindung entspricht. • Bei TCP oder UDP verwenden Sie die Struktur TCON_IP_v4. • Zur Beschreibung siehe: "Verbindungsparameter nach TCON_IP_v4". • Bei ISO-on-TCP verwenden Sie die Struktur TCON_IP_RFC. • Zur Beschreibung siehe: "Verbindungsparameter nach TCON_IP_RFC". Der Parameter CONNECT wird nur bei einer steigenden Flanke an REQ (TSEND_C) ausgewertet, wenn der Verbindungsaufbau beginnt (TRCV_C) oder wenn COM_RST = 1.
DATA	IN_OUT	Variant	Pointer auf den Sendebereich mit: • Adresse und Länge der Daten, die gesendet (TSEND_C) werden sollen • Adresse und maximale Länge der empfangenen Daten (TRCV_C)
ADDR	IN_OUT	Variant	Optionaler Parameter (versteckt) Pointer auf die Adresse des Empfängers mit dem Verbindungstyp UDP. Die Adressinformationen werden in der Struktur TADDR_Param ### zugeordnet.
COM_RST	IN_OUT	Bool	Optionaler Parameter (versteckt) Startet die Anweisung neu: • 0: Irrelevant • 1: Startet die Anweisung vollständig neu. Die vorhandene Verbindung wird entweder beendet oder zurückgesetzt und gemäß CONT erneut aufgebaut. Der Parameter COM_RST wird nach Auswertung durch die Anweisung TSEND_C oder TRCV_C zurückgesetzt und darf deshalb nicht statisch geschaltet werden.
DONE	OUT	Bool	Statusparameter mit den folgenden Werten: • 0: Sendeauftrag wurde noch nicht gestartet oder wird noch ausgeführt. • 1: Sendeauftrag fehlerfrei ausgeführt. Dieser Zustand wird nur einen Zyklus lang angezeigt.
BUSY	OUT	Bool	Statusparameter mit den folgenden Werten: • 0: Sendeauftrag wurde noch nicht gestartet oder ist bereits beendet. • 1: Sendeauftrag noch nicht beendet. Es kann kein neuer Sendeauftrag gestartet werden.

Parameter und Datentyp		Datentyp	Beschreibung
ERROR	OUT	Bool	Statusparameter mit den folgenden Werten: • 0: Kein Fehler • 1: Fehler bei Verbindungsaufbau, Datenübertragung oder Verbindungsbeendigung.
STATUS	OUT	Word	Status der Anweisung (siehe Beschreibung der Parameter ERROR und STATUS).
RCVD_LEN (TRCV_C)	OUT	Int	Tatsächliche empfangene Datenmenge (in Byte).

Hinweis

Die Anweisung TSEND_C erfordert einen Wechsel von 0 nach 1 am Eingangsparameter REQ, um einen Sendeauftrag zu starten. Der Parameter BUSY wird dann während der Verarbeitung auf 1 gesetzt. Die Fertigstellung des Sendeauftrags wird kenntlich gemacht, indem einer der Parameter DONE oder ERROR einen Zyklus lang auf 1 gesetzt wird. Während dieses Zeitraums wird ein Wechsel von 0 nach 1 am Eingangsparameter REQ ignoriert.

Hinweis

Die Standardeinstellung des Parameters LEN (LEN = 0) ermittelt anhand des Parameters DATA die Länge der zu übertragenden Daten. Stellen Sie sicher, dass die von der Anweisung TSEND_C gesendeten Daten (DATA) dieselbe Größe haben wie der Parameter DATA der Anweisung TRCV_C.

Funktionsweise von TSEND_C

Die Anweisung TSEND_C wird asynchron ausgeführt und implementiert die folgenden Funktionen der Reihe nach:

1. Einrichten und Herstellen einer Kommunikationsverbindung:

   TSEND_C richtet eine Kommunikationsverbindung ein und stellt diese Verbindung her, wenn am Parameter REQ eine steigende Flanke erkannt wird und noch keine Kommunikationsverbindung vorhanden ist. Nach dem Einrichten und Herstellen der Verbindung wird diese automatisch von der CPU gehalten und überwacht. Die am Parameter CONNECT angegebene Verbindungsbeschreibung dient zum Einrichten der Kommunikationsverbindung. Die folgenden Verbindungstypen sind verwendbar:

   – TCON_Param-Struktur für die Protokolle TCP, ISO-on-TCP und UDP

   – Bei V4.1, TCP/UDP: Verbindungsbeschreibung über die Struktur TCON_IP_v4 am Parameter CONNECT

   – Bei V4.1, ISO-on-TCP: Verbindungsbeschreibung über die Struktur TCON_IP_RFC am Parameter CONNECT

   Wenn die CPU in den Betriebszustand STOP wechselt, wird eine vorhandene Verbindung beendet und die eingerichtete Verbindung entfernt. Um die Verbindung wieder einzurichten und herzustellen, müssen Sie TSEND_C erneut ausführen.

Informationen zur Anzahl der möglichen Kommunikationsverbindungen finden Sie in den technischen Daten Ihrer CPU.

2. Senden von Daten über eine vorhandene Kommunikationsverbindung:

   Der Sendeauftrag wird bei einer steigenden Flanke am Parameter REQ ausgeführt. Wie oben beschrieben, wird zunächst die Kommunikationsverbindung hergestellt. Den Sendebereich geben Sie über den Parameter DATA an. Dies umfasst die Adresse und die Länge der zu sendenden Daten. Verwenden Sie im Parameter DATA keinen Datenbereich mit dem Datentyp BOOL oder Array of BOOL. Mit dem Parameter LEN geben Sie die maximale Anzahl mit einem Auftrag gesendeter Bytes an. Wenn Sie am Parameter DATA einen symbolischen Namen verwenden, muss der Parameter LEN den Wert 0 haben.

   Die zu sendenden Daten dürfen vor Fertigstellung des Sendeauftrags nicht bearbeitet werden.

3. Beenden der Kommunikationsverbindung:

   Die Kommunikationsverbindung wird nach dem Senden der Daten beendet, wenn der Parameter CONT zum Zeitpunkt der steigenden Flanke am Parameter REQ den Wert 0 hatte. Ansonsten wird die Kommunikationsverbindung aufrechterhalten.

Bei erfolgreicher Ausführung des Sendeauftrags wird der Parameter DONE auf 1 gesetzt. Die Kommunikationsverbindung kann vorab beendet werden (siehe hierzu die Beschreibung oben über die Abhängigkeit vom Parameter CONT). Der Signalzustand 1 am Parameter DONE ist keine Bestätigung dafür, dass die gesendeten Daten bereits vom Kommunikationspartner gelesen wurden.

TSEND_C wird zurückgesetzt, wenn der Parameter COM_RST auf 1 gesetzt wird. Werden zu diesem Zeitpunkt Daten übertragen, kann es zu Datenverlust kommen.

Abhängig vom Parameter CONT sind die folgenden Szenarien möglich:

- CONT = "0":

  Eine vorhandene Kommunikationsverbindung wird hergestellt.

- CONT = 1 und eine Kommunikationsverbindung wurde hergestellt:

  Eine vorhandene Kommunikationsverbindung wird zurückgesetzt und erneut hergestellt.

- CONT = 1 und es wurde keine Kommunikationsverbindung hergestellt.

  Es wird keine Kommunikationsverbindung hergestellt.

Der Parameter COM_RST wird nach Auswertung durch die Anweisung T_SEND zurückgesetzt. Um TSEND_C nach der Ausführung (DONE = 1) erneut zu aktivieren, rufen Sie die Anweisung einmal mit REQ = 0 auf.

Funktionsweise von TRCV_C

Die Anweisung TRCV_C wird asynchron ausgeführt und implementiert die folgenden Funktionen der Reihe nach:

1. Einrichten und Herstellen einer Kommunikationsverbindung:

   TRCV_C richtet eine Kommunikationsverbindung ein und stellt diese Verbindung her, wenn der Parameter EN_R = 1 ist und keine Kommunikationsverbindung vorhanden ist. Nach dem Einrichten und Herstellen der Verbindung wird diese automatisch von der CPU gehalten und überwacht.

   Die am Parameter CONNECT angegebene Verbindungsbeschreibung dient zum Einrichten der Kommunikationsverbindung. Die folgenden Verbindungstypen sind verwendbar:

   – TCON_Param-Struktur für die Protokolle TCP, ISO-on-TCP und UDP
   – Bei V4.1, TCP/UDP: Verbindungsbeschreibung über die Struktur TCON_IP_v4 am Parameter CONNECT
   – Bei V4.1, ISO-on-TCP: Verbindungsbeschreibung über die Struktur TCON_IP_RFC am Parameter CONNECT

   Wenn die CPU in den Betriebszustand STOP wechselt, wird eine vorhandene Verbindung beendet und die eingerichtete Verbindung entfernt. Um die Verbindung wieder einzurichten und herzustellen, müssen Sie TRCV_C erneut mit EN_R = 1 ausführen.

   Wenn EN_R vor dem Herstellen der Kommunikationsverbindung auf 0 gesetzt wird, wird die Verbindung hergestellt und bleibt bestehen, auch wenn CONT = 0. Es werden jedoch keine Daten empfangen (DONE bleibt 0).

   Informationen zur Anzahl der möglichen Kommunikationsverbindungen finden Sie in den technischen Daten Ihrer CPU.

2. Empfangen von Daten über eine vorhandene Kommunikationsverbindung:

   Der Datenempfang ist aktiviert, wenn der Parameter EN_R auf den Wert 1 gesetzt ist. Wie oben beschrieben, wird zunächst die Kommunikationsverbindung hergestellt. Die empfangenen Daten werden in einen Empfangsbereich geschrieben. Sie geben die Länge des Empfangsbereichs abhängig von der verwendeten Protokollvariante entweder mit dem Parameter LEN an (wenn LEN <> 0) oder mit der Längeninformation des Parameters DATA (wenn LEN = 0). Wenn Sie am Parameter DATA rein symbolische Werte verwenden, muss der Parameter LEN den Wert 0 haben.

   Wird EN_R auf "0" gesetzt, bevor erstmals Daten empfangen werden, bleibt die Kommunikationsverbindung auch dann erhalten, wenn CONT = 0. Es werden jedoch keine Daten empfangen (DONE bleibt auf "0").

3. Beenden der Kommunikationsverbindung:

   Die Kommunikationsverbindung wird nach dem Datenempfang beendet, wenn der Parameter CONT beim Herstellen der Verbindung den Wert 0 hatte. Ansonsten wird die Kommunikationsverbindung aufrechterhalten.

Bei erfolgreicher Ausführung des Empfangsauftrags wird der Parameter DONE auf 1 gesetzt. Die Kommunikationsverbindung kann vorab beendet werden (siehe hierzu die Beschreibung oben über die Abhängigkeit vom Parameter CONT).

TRCV_C wird zurückgesetzt, wenn der Parameter COM_RST gesetzt wird. Werden bei erneuter Ausführung Daten empfangen, kann dies zu Datenverlust führen. Abhängig vom Parameter CONT sind die folgenden Szenarien möglich:

- CONT = "0":

  Eine vorhandene Kommunikationsverbindung wird hergestellt.

- CONT = 1 und eine Kommunikationsverbindung wurde hergestellt:

  Eine vorhandene Kommunikationsverbindung wird zurückgesetzt und erneut hergestellt.

- CONT = 1 und es wurde keine Kommunikationsverbindung hergestellt:

  Es wird keine Kommunikationsverbindung hergestellt.

Der Parameter COM_RST wird nach Auswertung durch die Anweisung TRCV_ zurückgesetzt.

TRCV_C arbeitet mit den gleichen Empfangsmodi wie die Anweisung TRCV. Die folgende Tabelle zeigt, wie Daten in den Empfangsbereich geschrieben werden:

Protokollvariante	Verfügbarkeit der Daten im Empfangsbereich	Parameter Connection_type der Verbindungsbeschreibung	Parameter LEN	Parameter RCVD_LEN
TCP (Ad-hoc-Modus)	Die Daten sind sofort verfügbar.	B#16#11	Ausgewählt mit dem Eingang ADHOC der Anweisung TRCV_C	1 bis 1472
TCP (Datenempfang mit angegebener Länge)	Die Daten sind verfügbar, sobald die am Parameter LEN angegebene Datenlänge vollständig empfangen wurde.	B#16#11	1 bis 8192	Identisch mit dem Wert am Parameter LEN
ISO on TCP (protokollgesteuerte Datenübertragung)	Die Daten sind verfügbar, sobald die am Parameter LEN angegebene Datenlänge vollständig empfangen wurde.	B#16#12	1 bis 8192	Identisch mit dem Wert am Parameter LEN

Hinweis

Ad-hoc-Modus

Den "Ad-hoc-Modus" gibt es nur bei der Protokollvariante TCP. Um die Anweisung TRCV_C für den Ad-hoc-Modus zu konfigurieren, setzen Sie den Eingangsparameter ADHOC der Anweisung. Die Länge des Empfangsbereichs wird vom Pointer am Parameter DATA definiert. Die tatsächlich empfangene Datenlänge wird am Parameter RCVD_LEN ausgegeben. Es können maximal 1460 Bytes empfangen werden.

Hinweis

STEP 7-Projekte der S7-300/400 mit Ad-hoc-Modus in die S7-1200 importieren

In STEP 7-Projekten der S7-300/400 wählen Sie den Ad-hoc-Modus aus, indem Sie dem Parameter LEN den Wert 0 zuweisen. In der S7-1200 können Sie die Anweisung TRCV_C für den Ad-hoc-Modus konfigurieren, indem Sie den Eingangsparameter ADHOC der Anweisung setzen.

Wenn Sie ein STEP 7-Projekt der S7-300/400, das den Ad-hoc-Modus umfasst, in die S7-1200 importieren, müssen Sie den Parameter LEN in 65535 ändern.

Hinweis

TCP (Datenempfang mit angegebener Länge)

Mit dem Wert des Parameters LEN geben Sie die Länge für den Datenempfang an. Die am Parameter DATA angegebenen Daten sind im Empfangsbereich verfügbar, sobald die am Parameter LEN angegebene Länge vollständig empfangen wurde.

Hinweis

ISO on TCP (protokollgesteuerte Datenübertragung)

Bei der Protokollvariante ISO on TCP werden die Daten protokollgesteuert übertragen. Der Empfangsbereich wird von den Parametern LEN und DATA definiert.

Parameter BUSY, DONE und ERROR

Hinweis

Durch die asynchrone Bearbeitung von TSEND_C müssen die Daten im Sendebereich konsistent bleiben, bis einer der Parameter DONE oder ERROR den Wert WAHR annimmt.

Für TSEND_C bedeutet der Status WAHR des Parameters DONE, dass die Daten erfolgreich gesendet wurden. Er bedeutet nicht, dass die Partner-CPU den Empfangspuffer tatsächlich liest.

Durch die asynchrone Bearbeitung von TRCV_C sind die Daten im Empfangsbereich nur konsistent, wenn der Parameter DONE = 1 ist.

Tabelle 11- 14 Parameter BUSY, DONE und ERROR der Anweisungen TSEND_C und TRCV_C

BUSY	DONE	ERROR	Beschreibung
1	0	0	Der Sendeauftrag wird bearbeitet.
0	1	0	Der Sendeauftrag wurde erfolgreich ausgeführt.
0	0	1	Der Verbindungsaufbau oder der Sendeauftrag wurde mit Fehler beendet. Die Fehlerursache ist im Parameter STATUS hinterlegt.
0	0	0	Es wurde kein neuer Sendeauftrag zugewiesen.

Den Status der Ausführung können Sie mit den Parametern BUSY, DONE, ERROR und STATUS prüfen. Der Parameter BUSY zeigt den Bearbeitungszustand an. Mit dem Parameter DONE können Sie prüfen, ob ein Sendeauftrag erfolgreich ausgeführt wurde. Der Parameter ERROR wird gesetzt, wenn während der Ausführung von TSEND_C oder TRCV_C Fehler auftraten. Die Fehlerinformationen werden am Parameter STATUS ausgegeben.

Parameter Error und Status

Tabelle 11- 15 Bedingungscodes von TSEND_C und TRCV_C für ERROR und STATUS

ERROR	STATUS (W#16#...)	Beschreibung
0	0000	Sendeauftrag (TSEND_C) oder Empfangsauftrag (TRCV_C) fehlerfrei ausgeführt.
0	7000	Keine Ausführung eines Sendeauftrags aktiv; keine Kommunikationsverbindung hergestellt.
0	7001	• Ausführung des Sendeauftrags (TSEND_C) oder Empfangsauftrags (TRCV_C) starten. • Verbindung herstellen. • Auf Verbindungspartner warten.
0	7002	Daten werden gesendet (TSEND_C) oder empfangen (TRCV_C).
0	7003	Kommunikationsverbindung wird beendet.
0	7004	Kommunikationsverbindung hergestellt und überwacht; kein Sendeauftrag (TSEND_C) oder Empfangsauftrag (TRCV_C) aktiv.
0	7005	Kommunikationsverbindung wird zurückgesetzt.
1	80A0	Sammelfehler für Fehlercodes W#16#80A1 und W#16#80A2.
1	80A1	• Verbindung oder Port wird bereits benutzt. • Kommunikationsfehler: – Die angegebene Verbindung ist noch nicht hergestellt. – Die angegebene Verbindung wird beendet. Übertragung über diese Verbindung ist nicht möglich. – Die Schnittstelle wird neu initialisiert.
1	80A2	Lokaler oder entfernter Port wird vom System benutzt.
1	80A3	• Es wird versucht, eine bestehende Verbindung erneut herzustellen. • Es wird versucht, eine nicht vorhandene Verbindung zu beenden.
1	80A4	Die IP-Adresse des dezentralen Endpunkts der Verbindung ist ungültig, d.h. sie entspricht der IP-Adresse des lokalen Partners.
1	80A7	Kommunikationsfehler: Sie haben die Anweisung mit COM_RST = 1 aufgerufen, bevor der Sendeauftrag abgeschlossen war.
1	80B2	Der Parameter CONNECT zeigt auf einen Datenbaustein, der mit dem Attribut Nur im Ladespeicher ablegen generiert wurde.
1	80B3	Parametrierung nicht konsistent: Sammelfehler für Fehlercodes W#16#80A0 bis W#16#80A2, W#16#80A4, W#16#80B4 bis W#16#80B9.

ERROR	STATUS (W#16#...)	Beschreibung
1	80B4	Es wurden bei der passiven Herstellung einer Verbindung (active_est = FALSE) mit der Protokollvariante ISO on TCP (connection_type = B#16#12) eine oder mehrere der folgenden Bedingungen verletzt: • local_tsap_id_len >= B#16#02 • local_tsap_id[1] = B#16#E0
1	80B5	Für Verbindungstyp 13 = UDP ist nur die passive Herstellung einer Verbindung zulässig.
1	80B6	Parametrierungsfehler im Parameter connection_type des Datenbausteins für die Verbindungsbeschreibung.
1	80B7	Fehler in einem der folgenden Parameter des Datenbausteins für die Verbindungsbeschreibung: block_length, local_tsap_id_len, rem_subnet_id_len, rem_staddr_len, rem_tsap_id_len, next_staddr_len.
1	8085	• Der Parameter LEN ist größer als der größte zulässige Wert. • Der Wert am Parameter LEN oder DATA wurde nach dem ersten Aufruf geändert.
1	8086	Der Parameter ID im Parameter CONNECT liegt außerhalb des zulässigen Bereichs.
1	8087	Maximale Anzahl Verbindungen erreicht; keine weitere Verbindung möglich.
1	8088	Der Wert am Parameter LEN entspricht nicht dem am Parameter DATA festgelegten Empfangsbereich.
1	8089	Der Parameter CONNECT verweist nicht auf einen Datenbaustein.
1	8091	Maximale Schachtelungstiefe überschritten.
1	809A	Der Parameter CONNECT zeigt auf ein Feld, das nicht der Länge der Verbindungsbeschreibung entspricht.
1	809B	Die ID des lokalen Geräts in der Verbindungsbeschreibung entspricht nicht der CPU.
1	80C3	• Alle Verbindungsressourcen (Seite 659) sind belegt. • Ein Baustein mit dieser ID wird bereits in einer anderen Prioritätsklasse bearbeitet.
1	80C4	Temporärer Kommunikationsfehler: • Die Verbindung kann zur Zeit nicht hergestellt werden. • Die Schnittstelle empfängt neue Parameter oder die Verbindung wird gerade aufgebaut. • Die konfigurierte Verbindung wird gerade von einer "TDISCON (Seite 688)"-Anweisung gelöscht. • Die verwendete Verbindung wird gerade von einem Aufruf mit COM_RST = 1 beendet.
1	8722	Fehler im Parameter CONNECT: Ungültiger Quellbereich (Bereich nicht im Datenbaustein deklariert).
1	873A	Fehler im Parameter CONNECT: Der Zugriff auf die Verbindungsbeschreibung ist nicht möglich (kein Zugriff auf den Datenbaustein).
1	877F	Fehler im Parameter CONNECT: Interner Fehler
1	8822	TSEND_C: Parameter DATA: Ungültiger Quellbereich, der Bereich ist im DB nicht vorhanden.
1	8824	TSEND_C: Parameter DATA: Bereichsfehler im Pointer VARIANT.
1	8832	TSEND_C: Parameter DATA: Die DB-Nummer ist zu groß.
1	883A	TSEND_C: Parameter CONNECT: Zugriff auf angegebene Verbindungsdaten nicht möglich (z. B. weil der DB nicht vorhanden ist).
1	887F	TSEND_C: Parameter DATA: Interner Fehler (z. B. ungültiger Verweis auf VARIANT)

ERROR	STATUS (W#16#...)	Beschreibung
1	893A	TSEND_C: Parameter DATA: Zugriff auf Sendebereich nicht möglich (z. B. weil der DB nicht vorhanden ist).
1	8922	TRCV_C: Parameter DATA: Ungültiger Zielbereich, der Bereich ist im DB nicht vorhanden.
1	8924	TRCV_C: Parameter DATA: Bereichsfehler im Pointer VARIANT.
1	8932	TRCV_C: Parameter DATA: Die DB-Nummer ist zu groß.
1	893A	TRCV_C: Parameter CONNECT: Zugriff auf angegebene Verbindungsdaten nicht möglich (z. B. weil der DB nicht vorhanden ist).
1	897F	TRCV_C: Parameter DATA: Interner Fehler (z. B. ungültiger Verweis auf VARIANT).
1	8A3A	TRCV_C: Parameter DATA: Kein Zugriff auf den Datenbereich (z. B. weil der Datenbaustein nicht vorhanden ist).

Hinweis

Fehlermeldungen der Anweisungen TCON, TSEND, TRCV und TDISCON

Intern verwendet Anweisung TSEND_C die Anweisungen TCON, TSEND und TDISCON und die Anweisung TRCV_C verwendet die Anweisungen TCON, TRCV und TDISCON. Weitere Informationen zu den Fehlermeldungen dieser Anweisungen finden Sie unter TCON, TDISCON, TSEND UND TRCV (Seite 688).

Ethernet-Verbindungsprotokolle

Jede CPU hat einen integrierten PROFINET-Anschluss, der die standardmäßige PROFINET-Kommunikation unterstützt. Die Anweisungen TSEND_C und TRCV_C sowie TSEND und TRCV unterstützen alle die Ethernet-Protokolle TCP und ISO on TCP.

Weitere Informationen finden Sie unter "Gerätekonfiguration: Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren (Seite 638)".

11.2.8.8 Anweisungen TSEND_C und TRCV_C in älteren Systemen

Vor STEP 7 V13 SP1 und den S7-1200 V4.1-CPUs haben die Anweisungen TSEND_C und TRCV_C nur mit Verbindungsparametern mit Strukturen nach "TCON_Param" funktioniert. Für beide Anweisungsarten gelten die allgemeinen Konzepte. Informationen zur Programmierung finden sich in den Beschreibungen zu den einzelnen TSEND_C- und TRCV_C-Anweisungen.

Version der Anweisungen TSEND_C und TRCV_C auswählen

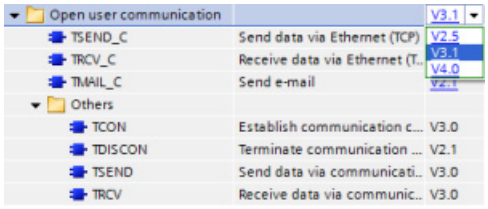
Es gibt zwei Versionen der Anweisungen TSEND_C und TRCV_C in STEP 7:

- Die Versionen 2.5 und 3.1 waren in STEP 7 Basic/Professional V13 und früher verfügbar.
- Version 4.0 ist in STEP 7 Basic/Professional V13 SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Verwenden Sie niemals Anweisungen unterschiedlicher Versionen zusammen in einem CPU-Programm.

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.



Um die Version einer TSEND_C- und TRCV_C-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine TSEND_C- oder TRCV_C-Anweisung über das Anweisungsverzeichnis in Ihr Programm platzieren, wird je nach der ausgewählten TSEND_C- oder TRCV_C-Anweisung eine neue FB- oder FC-Instanz in der Projektnavigation angelegt. Die neue FB- oder FC-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer TSEND_C- oder TRCV_C-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation eine FB- oder FC-Instanz einer TSEND_C- oder TRCV_C Anweisung aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der TSEND_C- oder TRCV_C-Anweisung anzuzeigen.

Alte Anweisungen TSEND_C und TRCV_C (Daten über Ethernet senden und empfangen)

Die alte Anweisung TSEND_C verbindet die Funktionen der alten Anweisungen TCON, TDISCON und TSEND . Die Anweisung TRCV_C verbindet die Funktionen der Anweisungen TCON, TDISCON und TRCV. (Weitere Informationen zu diesen Anweisungen finden Sie unter "Alte Anweisungen TCON, TDISCON, TSEND und TRCV (TCP-Kommunikation) (Seite 698)".)

Die Mindestdatengröße, die gesendet (TSEND_C) oder empfangen (TRCV_C) werden kann, ist ein Byte; die maximale Datengröße beträgt 8192 Byte. TSEND_C unterstützt nicht die Datenübertragung von Booleschen Adressen und TRCV_C empfängt Daten nicht in Booleschen Adressen. Weitere Informationen zum Übertragen von Daten mit diesen Anweisungen finden Sie im Abschnitt zur Datenkonsistenz (Seite 191).

Hinweis

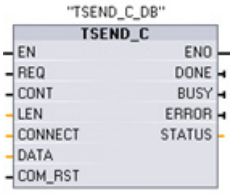
Initialisierung der Kommunikationsparameter

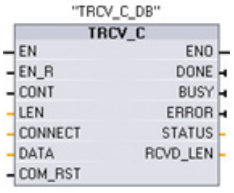
Nachdem Sie die Anweisung TSEND_C oder TRCV_C eingefügt haben, konfigurieren Sie in den "Eigenschaften" der Anweisung (Seite 638) die Kommunikationsparameter (Seite 663). Wenn Sie die Parameter für die Kommunikationspartner im Inspektorfenster eingeben, gibt STEP 7 die entsprechenden Daten in den DB der Anweisung ein.

Wenn Sie einen Multiinstanz-DB verwenden möchten, müssen Sie den DB manuell auf beiden CPUs konfigurieren.

Tabelle 11- 16 Anweisungen TSEND_C und TRCV_C

KOP/FUP	SCL	Beschreibung

	`"TSEND_C_DB"(`
	`req:=_bool_in_,`
	`cont:=_bool_in_,`
	`len:=_uint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`connect:=_struct_inout_,`
	`data:=_variant_inout_,`
	`com_rst:=_bool_inout_);`
		TSEND_C stellt eine TCP- oder ISO-on-TCP-Verbindung zu einem Partner her, sendet Daten und kann die Verbindung auch wieder beenden. Nach dem Einrichten und Aufbauen der Verbindung wird diese automatisch von der CPU gehalten und überwacht.

	`"TRCV_C_DB"(`
	`en_r:=_bool_in_,`
	`cont:=_bool_in_,`
	`len:=_uint_in_,`
	`adhoc:=_bool_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`rcvd_len=>_uint_out_,`
	`connect:=_struct_inout_,`
	`data:=_variant_inout_,`
	`com_rst:=_bool_inout_);`
		TRCV_C stellt eine TCP- oder ISO on TCP-Verbindung zu einer CPU her, empfängt Daten und kann die Verbindung auch wieder beenden. Nach dem Einrichten und Aufbauen der Verbindung wird diese automatisch von der CPU gehalten und überwacht.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 11- 17 Datentypen für die Parameter von TSEND_C und TRCV_C

Parameter und Datentyp		Datentyp	Beschreibung
REQ (TSEND_C)	IN	Bool	REQ = 1 startet den Sendeauftrag von TSEND_C bei einer steigenden Flanke über die im Parameter CONNECT beschriebene Verbindung. (CONT = 1 ist ebenfalls erforderlich, um die Kommunikationsverbindung herzustellen und aufrechtzuerhalten.
EN_R (TRCV_C)	IN	Bool	Ist EN_R = 1, ist TRCV_C empfangsbereit. Der Empfangsauftrag wird bearbeitet. (CONT = 1 ist ebenfalls erforderlich, um die Kommunikationsverbindung herzustellen und aufrechtzuerhalten.)

Parameter und Datentyp		Datentyp	Beschreibung
CONT	IN	Bool	Steuert die Kommunikationsverbindung: • 0: Trennen der Kommunikationsverbindung • 1: Aufbauen und Halten der Kommunikationsverbindung Beim Senden von Daten (TSEND_C) (steigende Flanke am Parameter REQ) muss der Parameter CONT den Wert TRUE haben, damit eine Verbindung aufgebaut oder gehalten werden kann. Beim Empfangen von Daten (TRCV_C) (steigende Flanke am Parameter EN_R) muss der Parameter CONT den Wert TRUE haben, damit eine Verbindung aufgebaut oder gehalten werden kann.
LEN	IN	UInt	Maximale Bytezahl, die gesendet (TSEND_C) oder empfangen (TRCV_C) werden soll: • Standard = 0: Der Parameter DATA legt die Länge der zu sendenden (TSEND_C) oder zu empfangenden (TRCV_C) Daten fest. • Ad-hoc-Modus = 65535: Für den Datenempfang ist eine variable Länge eingestellt (TRCV_C).
CONNECT	IN_OUT	TCON_Param	Pointer auf die Verbindungsbeschreibung (Seite 663)
DATA	IN_OUT	Variant	• Enthält Adresse und Länge der Daten, die gesendet (TSEND_C) werden sollen. • Enthält Anfangsadresse und maximale Länge der empfangenen Daten (TRCV_C).
COM_RST	IN_OUT	Bool	Ermöglicht den Neustart der Anweisung: • 0: Irrelevant • 1: Kompletter Neustart des Funktionsbausteins, bestehende Verbindung wird beendet.
DONE	OUT	Bool	• 0: Auftrag noch nicht gestartet oder läuft noch. • 1: Auftrag fehlerfrei ausgeführt.
BUSY	OUT	Bool	• 0: Auftrag beendet. • 1: Auftrag noch nicht beendet. Es kann kein neuer Auftrag gestartet werden.
ERROR	OUT	Bool	Statusparameter mit den folgenden Werten: • 0: Kein Fehler • 1: Fehler während der Verarbeitung aufgetreten. STATUS enthält Informationen zum Fehlertyp.
STATUS	OUT	Word	Statusinformationen umfassen Fehlerinformationen. (Siehe Tabelle "Fehler- und Statusparameter" unten.)
RCVD_LEN (TRCV_C)	OUT	Int	Tatsächliche empfangene Datenmenge in Byte

Hinweis

Die Anweisung TSEND_C erfordert einen Wechsel von 0 nach 1 am Eingangsparameter REQ, um einen Sendeauftrag zu starten. Der Parameter BUSY wird dann während der Verarbeitung auf 1 gesetzt. Die Fertigstellung des Sendeauftrags wird kenntlich gemacht, indem einer der Parameter DONE oder ERROR einen Zyklus lang auf 1 gesetzt wird. Während dieses Zeitraums wird ein Wechsel von 0 nach 1 am Eingangsparameter REQ ignoriert.

Hinweis

Die Standardeinstellung des Parameters LEN (LEN = 0) ermittelt anhand des Parameters DATA die Länge der zu übertragenden Daten. Stellen Sie sicher, dass die von der Anweisung TSEND_C gesendeten Daten (DATA) dieselbe Größe haben wie der Parameter DATA der Anweisung TRCV_C.

Funktionsweise von TSEND_C

Die folgenden Funktionen beschreiben die Funktionsweise der Anweisung TSEND_C:

- Um eine Verbindung herzustellen, ist TSEND_C mit CONT = 1 auszuführen.
- Nach der erfolgreichen Herstellung der Verbindung setzt TSEND_C den Parameter DONE für einen Zyklus.
- Um die Verbindung zu beenden, führen Sie TSEND_C mit CONT = 0 aus. Die Verbindung wird dann sofort abgebrochen. Dies betrifft auch den Empfänger. Die Verbindung wird beendet und es können Daten im Empfangspuffer verlorengehen.
- Um Daten über eine bestehende Verbindung zu senden, muss TSEND_C bei einer steigenden Flanke an REQ ausgeführt werden. Nach dem erfolgreichen Senden setzt TSEND_C den Parameter DONE für einen Zyklus auf 1.
- Um eine Verbindung herzustellen und Daten zu senden, muss TSEND_C mit CONT = 1 und REQ = 1 ausgeführt werden. Nach dem erfolgreichen Senden setzt TSEND_C den Parameter DONE für einen Zyklus auf 1.

Funktionsweise von TRCV_C

Die folgenden Funktionen beschreiben die Funktionsweise der Anweisung TRCV_C:

- Um eine Verbindung herzustellen, ist TRCV_C mit CONT = 1 auszuführen.
- Um Daten zu empfangen, führen Sie TRCV_C mit dem Parameter EN_R = 1 aus. TRCV_C empfängt kontinuierlich Daten, wenn die Parameter EN_R = 1 und CONT = 1 sind.
- Um die Verbindung zu beenden, führen Sie TRCV_C mit dem Parameter CONT = 0 aus. Die Verbindung wird sofort beendet, es können dabei Daten verloren gehen.

TRCV_C arbeitet mit den gleichen Empfangsmodi wie die Anweisung TRCV. Die folgende Tabelle zeigt, wie Daten in den Empfangsbereich geschrieben werden:

Tabelle 11- 18 Eintragen der Daten in den Empfangsbereich

Protokollvariante	Eintragen der Daten in den Empfangsbereich	Parameter "connection_type"	Wert des Parameters LEN	Wert des Parameters RCVD_LEN (Byte)
TCP	Ad-hoc-Modus	B#16#11	65535	1 bis 1472
TCP	Datenempfang mit angegebener Länge	B#16#11	0 (empfohlen) oder 1 bis 8192, außer 65535	1 bis 8192
ISO on TCP	Ad-hoc-Modus	B#16#12	65535	1 bis 1472
ISO on TCP	Protokollgesteuert	B#16#12	0 (empfohlen) oder 1 bis 8192, außer 65535	1 bis 8192

Hinweis

Ad-hoc-Modus

Den Ad-hoc-Modus gibt es nur bei den Protokollvarianten TCP und ISO on TCP. Den Ad-hoc-Modus stellen Sie ein, indem Sie dem Parameter LEN den Wert 65535 zuweisen. Der Empfangsbereich ist identisch mit dem von Parameter DATA angegebenen Bereich. Die Länge der empfangenen Daten wird am Parameter RCVD_LEN ausgegeben.

Wenn Sie die Daten in einem "optimierten" DB (nur symbolisch adressierbar) speichern, können Sie Daten nur in Arrays der Datentypen Byte, Char, USInt und SInt empfangen.

Hinweis

STEP 7-Projekte der S7-300/400 mit Ad-hoc-Modus in die S7-1200 importieren

In STEP 7-Projekten der S7-300/400 wählen Sie den Ad-hoc-Modus aus, indem Sie dem Parameter LEN den Wert 0 zuweisen. Bei der S7-1200 stellen Sie den Ad-hoc-Modus ein, indem Sie dem Parameter LEN den Wert 65535 zuweisen.

Wenn Sie ein STEP 7-Projekt der S7-300/400, das den Ad-hoc-Modus umfasst, in die S7-1200 importieren, müssen Sie den Parameter LEN in 65535 ändern.

Hinweis

Die Daten im Sendebereich müssen konsistent bleiben, bis einer der Parameter DONE oder ERROR den Wert WAHR annimmt.

Durch die asynchrone Bearbeitung von TSEND_C müssen die Daten im Sendebereich konsistent bleiben, bis einer der Parameter DONE oder ERROR den Wert WAHR annimmt.

Für TSEND_C bedeutet der Status WAHR des Parameters DONE, dass die Daten erfolgreich gesendet wurden. Er bedeutet nicht, dass die Partner-CPU den Empfangspuffer tatsächlich liest.

Durch die asynchrone Bearbeitung von TRCV_C sind die Daten im Empfangsbereich nur konsistent, wenn der Parameter DONE = 1 ist.

Tabelle 11- 19 Parameter BUSY, DONE und ERROR der Anweisungen TSEND_C und TRCV_C

BUSY	DONE	ERROR	Beschreibung
WAHR	Irrelevant	Irrelevant	Auftrag wird bearbeitet.
FALSCH	WAHR	FALSCH	Auftrag erfolgreich ausgeführt.
FALSCH	FALSCH	WAHR	Auftrag mit Fehler beendet. Die Fehlerursache ist im Parameter STATUS hinterlegt.
FALSCH	FALSCH	FALSCH	Kein neuer Auftrag zugewiesen.

Bedingungscodes von TSEND_C und TRCV_C Error und Status

ERROR	STATUS	Beschreibung
0	0000	Auftrag fehlerfrei ausgeführt
0	7000	Keine laufende Auftragsbearbeitung
0	7001	Auftragsbearbeitung starten, Verbindung herstellen, auf Verbindungspartner warten
0	7002	Daten werden gesendet oder empfangen
0	7003	Verbindung wird beendet
0	7004	Verbindung hergestellt, wird überwacht, keine laufende Auftragsbearbeitung
1	8085	Der Parameter LEN ist größer als der größte zulässige Wert.
1	8086	Der Parameter CONNECT liegt außerhalb des zulässigen Bereichs.
1	8087	Maximale Anzahl Verbindungen erreicht; keine weitere Verbindung möglich.
1	8088	LEN-Parameter gilt nicht für den in DATA angegebenen Speicherbereich.
1	8089	Der Parameter CONNECT verweist nicht auf einen Datenbaustein.
1	8091	Maximale Schachtelungstiefe überschritten.
1	809A	Der Parameter CONNECT zeigt auf ein Feld, das nicht mit der Länge der Verbindungsbeschreibung übereinstimmt.
1	809B	local_device_id in der Verbindungsbeschreibung stimmt nicht mit der CPU überein.

ERROR	STATUS	Beschreibung
1	80A1	Kommunikationsfehler: • Die angegebene Verbindung ist noch nicht hergestellt • Die angegebene Verbindung wird derzeit beendet; Datenübertragung über diese Verbindung nicht möglich • Schnittstelle wird neu initialisiert
1	80A3	Es wird versucht, eine nicht vorhandene Verbindung zu beenden
1	80A4	Die IP-Adresse des entfernten Teilnehmers ist ungültig. Beispiel: Die IP-Adresse des entfernten Teilnehmers ist dieselbe wie die IP-Adresse des lokalen Teilnehmers.
1	80A5	Verbindungs-ID (Seite 659) wird bereits verwendet.
1	80A7	Kommunikationsfehler: Sie haben TDISCON aufgerufen, bevor TSEND_C beendet war.
1	80B2	Der Parameter CONNECT zeigt auf einen Datenbaustein, der mit dem Schlüsselwort UNLINKED erzeugt wurde.
1	80B3	Inkonsistente Parameter: • Fehler in der Verbindungsbeschreibung • Die lokale Schnittstelle (Parameter local_tsap_id) ist bereits in einer anderen Verbindungsbeschreibung vorhanden. • Die ID in der Verbindungsbeschreibung stimmt nicht mit der als Parameter angegebenen ID überein.
1	80B4	Wenn Sie eine passive Verbindung über ISO on TCP (connection_type = B#16#12) aufbauen, werden Sie anhand des Fehlercodes 80B4 gewarnt, wenn der eingegebene TSAP keiner der folgenden Voraussetzungen für Adressen entspricht: • Hat der lokale TSAP eine Länge von 2 und einen ID-Wert von E0 oder E1 (hexadezimal) für das erste Byte, muss das zweite Byte entweder 00 oder 01 sein. • Hat der lokale TSAP eine Länge von 3 oder größer und einen ID-Wert von E0 oder E1 (hexadezimal) für das erste Byte, muss das zweite Byte entweder 00 oder 01 sein und alle Bytes müssen gültige ASCII-Zeichen sein. • Hat der lokale TSAP eine Länge von 3 oder größer und das erste Byte des ID-Werts des TSAP nicht den Wert E0 oder E1 (hexadezimal), müssen alle Bytes der TSAP-ID gültige ASCII-Zeichen sein. Gültige ASCII-Zeichen sind Bytewerte von 20 bis 7E (hexadezimal).
1	80B7	Der Datentyp und/oder die Länge der übertragenen Daten entsprechen nicht dem Bereich in der Partner-CPU, in den die Daten geschrieben werden sollen.
1	80C3	Alle Verbindungsressourcen sind belegt.
1	80C4	Temporärer Kommunikationsfehler: • Die Verbindung kann zur Zeit nicht hergestellt werden • Die Schnittstelle empfängt neue Parameter • Die konfigurierte Verbindung wird von TDISCON entfernt.
1	8722	Parameter CONNECT: Ungültiger Quellbereich: Der Bereich ist im DB nicht vorhanden.
1	873A	Parameter CONNECT: Der Zugriff auf die Verbindungsbeschreibung ist nicht möglich (z. B. DB nicht verfügbar)
1	877F	Parameter CONNECT: Interner Fehler, z. B. ungültiger Verweis auf ANY.
1	893A	Parameter enthält die Nummer eines DB, der nicht geladen ist.

Ethernet-Verbindungsprotokolle

Jede CPU hat einen integrierten PROFINET-Anschluss, der die standardmäßige PROFINET-Kommunikation unterstützt. Die Anweisungen TSEND_C und TRCV_C sowie TSEND und TRCV unterstützen alle die Ethernet-Protokolle TCP und ISO on TCP.

Weitere Informationen finden Sie unter "Gerätekonfiguration: Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren (Seite 638)".

11.2.8.9 Anweisungen TCON, TDISCON, TSEND und TRCV

Version V4.1 der S7-1200 CPU zusammen mit STEP 7 V13 SP1 erweitert die Fähigkeit der Anweisungen TCON, TDISCON, TSEND und TRCV dahingehend, dass Verbindungsparameter mit Strukturen nach "TCON_IP_v4" und "TCON_IP_RFC" verwendet werden können.

Die S7-1200 unterstützt daher zwei Sätze TCON-, TDISCON-, TSEND- und TRCV-Anweisungen:

- Ältere TCON-, TDISCON-, TSEND- und TRCV-Anweisungen (Seite 698): Diese TCON-, TDISCON-, TSEND- und TRCV-Anweisungen wurden vor der Version V4.0 der S7-1200 verwendet und funktionieren nur mit Verbindungsparametern mit Strukturen nach "TCON_Param".
- Anweisungen TCON, TDISCON, TSEND und TRCV (Seite 688): Diese TCON-, TDISCON-, TSEND- und TRCV-Anweisungen stellen alle Funktionen der alten Anweisungen bereit und zusätzlich die Möglichkeit zur Nutzung von Verbindungsparametern mit Strukturen nach "TCON_IP_v4" und "TCON_IP_RFC".

Version der Anweisungen TCON, TDISCON, TSEND und TRCV auswählen

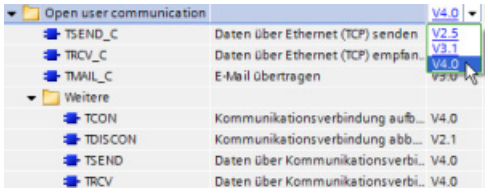
Es gibt zwei Versionen der Anweisungen TCON, TDISCON, TSEND oder TRCV in STEP 7:

- Die Versionen 2.5 und 3.1 waren in STEP 7 Basic/Professional V13 und früher verfügbar.
- Version 4.0 ist in STEP 7 Basic/Professional V13 SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Verwenden Sie niemals Anweisungen unterschiedlicher Versionen zusammen in einem CPU-Programm.

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.



Um die Version einer TCON-, TDISCON-, TSEND- oder TRCV-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine TCON-, TDISCON-, TSEND- oder TRCV-Anweisung über das Anweisungsverzeichnis in Ihr Programm platzieren, wird je nach der ausgewählten TCON-, TDISCON-, TSEND- oder TRCV-Anweisung eine neue FB- oder FC-Instanz in der Projektnavigation angelegt. Die neue FB- oder FC-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer TCON-, TDISCON-, TSEND- oder TRCV-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation eine FB- oder FC-Instanz einer TCON-, TDISCON-, TSEND- oder TRCV-Anweisung aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der TCON-, TDISCON-, TSEND- oder TRCV-Anweisung anzuzeigen.

Anweisungen TCON, TDISCON, TSEND und TRCV (TCP-Kommunikation)

Ethernet-Kommunikation mit den Protokollen TCP und ISO on TCP

Hinweis

Anweisungen TSEND_C und TRCV_C

Um die Programmierung der PROFINET/Ethernet-Kommunikation zu vereinfachen, verbinden die Anweisungen TSEND_C und TRCV_C die Funktionalität der Anweisungen TCON, TDISCON. TSEND und TRCV:

- TSEND_C verbindet die Anweisungen TCON, TDISCON und TSEND.
- TRCV_C verbindet die Anweisungen TCON, TDISCON und TRCV.

Die folgenden Anweisungen steuern den Kommunikationsprozess:

- TCON baut die TCP/IP-Verbindung zwischen dem Client- und dem Server-PC (CPU) auf.
- TSEND und TRCV senden und empfangen Daten.
- TDISCON trennt die Verbindung.

Die Mindestdatengröße, die gesendet (TSEND) oder empfangen (TRCV) werden kann, ist ein Byte. Die Maximalgröße beträgt 8192 Byte. TSEND unterstützt nicht die Übertragung von Daten von Booleschen Adressen und TRCV empfängt Daten nicht in Booleschen Adressen. Beachten Sie für weitere Informationen zum Übertragen von Daten mit diesen Anweisungen den Abschnitt zur Datenkonsistenz (Seite 191).

TCON, TDISCON, TSEND und TRCV laufen asynchron ab, so dass die Bearbeitung eines Auftrags mehrere Anweisungsausführungen umfasst. Beispielsweise starten Sie einen Auftrag zum Einrichten und Herstellen einer Verbindung durch die Ausführung von TCON mit Parameter REQ = 1. Dann überwachen Sie den Stand der Auftragsausführung, indem TCON wiederholt ausgeführt wird, und prüfen, ob Parameter DONE den Abschluss des Auftrags bestätigt.

Die folgende Tabelle zeigt die Beziehungen zwischen BUSY, DONE und ERROR. Den aktuellen Status eines Auftrags können Sie anhand der Tabelle feststellen:

Tabelle 11- 20 Interaktionen zwischen den Parametern BUSY, DONE und ERROR

BUSY	DONE	ERROR	Beschreibung
1	0	0	Auftrag wird bearbeitet.
0	1	0	Auftrag erfolgreich ausgeführt.
0	0	1	Auftrag mit Fehler beendet. Die Fehlerursache wird am Parameter STATUS ausgegeben.
0	0	0	Kein neuer Auftrag zugewiesen.

TCON und TDISCON

Hinweis

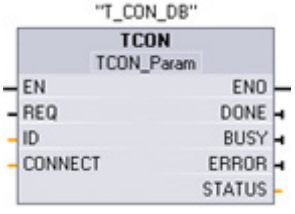
Initialisierung der Kommunikationsparameter

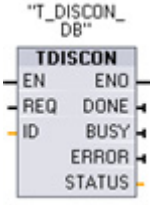
Nachdem Sie die Anweisung TCON eingefügt haben, konfigurieren Sie in den "Eigenschaften" der Anweisung (Seite 638) die Kommunikationsparameter (Seite 663). Wenn Sie die Parameter für die Kommunikationspartner im Inspektorfenster eingeben, gibt STEP 7 die entsprechenden Daten in den Instanz-DB der Anweisung ein.

Wenn Sie einen Multiinstanz-DB verwenden möchten, müssen Sie den DB manuell auf beiden CPUs konfigurieren.

Tabelle 11- 21 Anweisungen TCON und TDISCON

KOP/FUP		Beschreibung

	`"TCON_DB"(`
	`req:=_bool_in_,`
	`ID:=_undef_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`connect:=_struct_inout_);`
		TCP und ISO on TCP: TCON initiiert eine Kommunikationsverbindung von der CPU zu einem Kommunikationsteilnehmer.

	`"TDISCON_DB"(`
	`req:=_bool_in_,`
	`ID:=_word_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_);`
		TCP und ISO on TCP: TDISCON beendet eine Kommunikationsverbindung von der CPU zu einem Kommunikationsteilnehmer.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 11- 22 Datentypen für die Parameter von TCON und TDISCON

Parameter	Deklaration	Datentyp	Beschreibung
REQ	IN	Bool	Startet den Auftrag für den Aufbau der in der ID angegebenen Verbindung bei einer steigenden Flanke.
ID	IN	CONN_OUC (Word)	Verweis auf die zugewiesene Verbindung. Wertebereich: W#16#0001 bis W#16#0FFF
CONNECT (TCON)	IN_OUT	VARIANT	Pointer auf die Verbindungsbeschreibung • Für TCP oder UDP ist Struktur TCON_IP_v4 zu verwenden. Zur Beschreibung siehe: "Verbindungsparameter nach TCON_IP_v4" im TIA Portal. • Für ISO-on-TCP ist die Struktur TCON_IP_RFC zu verwenden. Zur Beschreibung siehe: "Verbindungsparameter nach TCON_IP_RFC" im TIA Portal.
DONE	OUT	Bool	Statusparameter mit den folgenden Werten: • 0: Auftrag noch nicht gestartet oder wird noch ausgeführt • 1: Auftrag fehlerfrei ausgeführt
BUSY	OUT	Bool	Statusparameter mit den folgenden Werten: • 0: Auftrag noch nicht gestartet oder bereits beendet • 1: Auftrag noch nicht beendet. Es kann kein neuer Auftrag gestartet werden
ERROR	OUT	Bool	Statusparameter ERROR: • 0: Kein Fehler • 1: Fehler aufgetreten
STATUS	OUT	Word	Zustand der Anweisung

Beide Kommunikationsteilnehmer führen die Anweisung TCON aus, um die Kommunikationsverbindung einzurichten und herzustellen. Mit Parametern geben Sie die aktiven und passiven Endpunkte der Kommunikation an. Nach dem Einrichten und Aufbauen der Verbindung wird sie automatisch von der CPU gehalten und überwacht.

Wird die Verbindung durch eine Leitungsstörung oder vom entfernten Teilnehmer abgebrochen, so versucht der aktive Teilnehmer, sie wieder aufzubauen. Sie müssen TCON nicht erneut ausführen.

Wird die Anweisung TDISCON ausgeführt oder ist die CPU in den Betriebszustand STOP gegangen, so wird die bestehende Verbindung beendet und die eingerichtete Verbindung gelöscht. Um die Verbindung einzurichten und wiederherzustellen, müssen Sie TCON erneut ausführen.

Tabelle 11- 23 Bedingungscodes von ERROR und STATUS für TCON und TDISCON

ERROR	STATUS (W#16#...)	Erklärung
0	0000	Verbindung erfolgreich aufgebaut.
0	7000	Keine laufende Auftragsbearbeitung
0	7001	Auftragsbearbeitung starten; Verbindung herstellen (TCON) oder Verbindung beenden (TDISCON).
0	7002	Verbindung wird hergestellt (REQ irrelevant); Verbindung herstellen (TCON) oder Verbindung beenden (TDISCON).
1	8085	TCON: Verbindungs-ID wird bereits verwendet.
1	8086	TCON: Der Parameter ID liegt außerhalb des zulässigen Bereichs.
1	8087	TCON: Maximale Anzahl Verbindungen erreicht; keine weitere Verbindung möglich
1	8089	TCON: Parameter CONNECT verweist nicht auf eine Verbindungsbeschreibung oder die Verbindungsbeschreibung wurde manuell erstellt.
1	809A	TCON: Die Struktur an Parameter CONNECT wird nicht unterstützt oder die Länge ist ungültig.
1	809B	TCON: Die ID des lokalen Geräts in der Verbindungsbeschreibung entspricht nicht der CPU oder dem CP oder lautet "0".
1	80A0	Sammelfehler für Fehlercodes W#16#80A1 und W#16#80A2.
1	80A1	TCON: Für TCP/UDP (TCON_IP_v4): Verbindung oder Port wird bereits benutzt.
1	80A2	TCON: Lokaler oder entfernter Port wird vom System verwendet.
1	80A3	TCON: Wert des Parameters ID wird bereits von einer Verbindung verwendet (TCON), die mit dem Anwenderprogramm erstellt wurde. Die Verbindung verwendet die gleiche ID, jedoch abweichende Verbindungseinstellungen an Parameter CONNECT.
1	80A4	TCON: IP-Adresse des entfernten Endpunkts der Verbindung ist ungültig oder entspricht der IP-Adresse des lokalen Partners.
1	80A5	TCON: Verbindungs-ID wird bereits verwendet.
1	80A7	TCON: Kommunikationsfehler: Sie haben "TDISCON" ausgeführt, bevor "TCON" beendet war.
1	80B2	TCON: Parameter CONNECT verweist auf einen Datenbaustein, der mit dem Attribut "Nur in Ladespeicher speichern" erzeugt wurde.
1	80B3	Parametrierung nicht konsistent: Sammelfehler für Fehlercodes W#16#80A0 bis W#16#80A2, W#16#80A4, W#16#80B4 bis W#16#80B9.
1	80B4	TCON: Nur mit TCON_IP_RFC Die lokale Transportauswahl wurde nicht angegeben oder das erste Byte enthält nicht den Wert 0x0E oder die lokale Transportauswahl beginnt mit "SIMATIC-".
1	80B5	TCON: Nur passiver Verbindungsaufbau zulässig für Verbindungstyp 13 = UDP (Parameter active_est der Struktur TCON_IP_v4 hat den Wert TRUE).
1	80B6	TCON: Parametrierungsfehler in Parameter connection_type des Datenbausteins für die Verbindungsbeschreibung. • Nur gültig für TCON_IP_v4: 0x11, 0x0B und 0x13. • Nur gültig für TCON_IP_RFC: 0x0C und 0x12

ERROR	STATUS (W#16#...)	Erklärung
1	80B7	TCON: Mit TCON_IP_v4: • TCP (aktiver Verbindungsaufbau): Entfernter Port ist "0". • TCP (passiver Verbindungsaufbau): Lokaler Port ist "0". • UDP: Lokaler Port ist "0". TCON: Mit TCON_IP_RFC: • Lokale (local_tselector) oder entfernte (remote_tselector) Tranportauswahl wurde mit einer Länge von mehr als 32 Bytes angegeben. • Für TSelLength der Transportauswahl (lokal oder entfernt) wurde eine Länge über 32 eingegeben. • Längenfehler der IP-Adresse des spezifischen Verbindungspartners.
1	80B8	TCON: Parameter ID in der lokalen Verbindungsbeschreibung (Struktur in Parameter CONNECT) und Parameter ID der Anweisung sind verschieden.
1	80C3	TCON: Alle Verbindungsressourcen (Seite 659) sind belegt.
1	80C4	Temporärer Kommunikationsfehler: • Die Verbindung kann zur Zeit nicht hergestellt werden (TCON). • Die Schnittstelle empfängt neue Parameter (TCON und TDISCON). • Die konfigurierte Verbindung wird durch eine Anweisung "TDISCON" entfernt (TCON).
1	80C5	TCON: Der entfernte Partner lehnt den Verbindungsaufbau ab, hat die Verbindung beendet oder aktiv abgebrochen.
1	80C6	TCON: Der entfernte Partner ist nicht erreichbar (Netzwerkfehler).
1	80C7	TCON: Zeitüberschreitung während der Ausführung.
1	80C8	TCON: ID wird von einer vom Anwenderprogramm erstellten Verbindung verwendet, die die gleiche Verbindungsbeschreibung in Parameter CONNECT verwendet.
1	80C9	TCON: Bestätigung des entfernten Partners fehlgeschlagen. Der entfernte Partner, der die Verbindung aufbauen will, stimmt nicht mit dem Partner überein, der in der Struktur von Parameter CONNECT festgelegt ist.
1	80CE	TCON: Die IP-Adresse der lokalen Schnittstelle lautet 0.0.0.0.

TSEND und TRCV

Hinweis

Wenn Sie in der offenen Benutzerkommunikation über PROFINET eine Anweisung TSEND ohne die entsprechende Anweisung TRCV auf dem Remote-Gerät ausführen, dann kann es passieren, dass die Anweisung TSEND auf unbestimmte Zeit im Zustand "Busy" bleibt und darauf wartet, dass die Anweisung TRCV die Daten empfängt. In diesem Zustand ist der Ausgang "Busy" der Anweisung TSEND gesetzt und der Ausgang "Status" hat den Wert "0x7002". Diese Bedingung kann auftreten, wenn Sie mehr als 4096 Byte an Daten übertragen. Das Problem wird bei der nächsten Ausführung der Anweisung TRCV behoben.

Tabelle 11- 24 Anweisungen TSEND und TRCV

KOP/FUP	SCL	Beschreibung
	"TSEND_DB"(req:=_bool_in_, ID:=_word_in_, len:=_udint_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, data:=_variant_inout_);	TCP und ISO on TCP: TSEND sendet über eine Kommunikationsverbindung Daten von der CPU zu einer Partnerstation.
	"TRCV_DB"(en_r:=_bool_in_, ID:=_word_in_, len:=_udint_in_, adhoc:=_bool_in_, ndr=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, rcvd_len=>_udint_out_, data:=_variant_inout_);	TCP und ISO on TCP: TRCV empfängt über eine Kommunikationsverbindung Daten von einer Partnerstation auf der CPU.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 11- 25 Datentypen für die Parameter von TSEND und TRCV

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	TSEND: Startet den Sendeauftrag bei einer steigenden Flanke. Die Daten werden aus dem mit DATA und LEN bezeichneten Bereich übertragen.
EN_R	IN	Bool	TRCV: Aktiviert die CPU für den Empfang, bei EN_R = 1 ist TRCV empfangsbereit. Der Empfangsauftrag wird bearbeitet.
ID	IN	CONN_OUC (Word)	Verweis auf die zugehörige Verbindung. Die ID muss mit der zugehörigen Parameter-ID in der lokalen Verbindungsbeschreibung übereinstimmen. Wertebereich: W#16#0001 bis W#16#0FFF
LEN	IN	UDInt	Maximale Bytezahl, die gesendet (TSEND) oder empfangen (TRCV) werden soll: • Standard = 0: Der Parameter DATA legt die Länge der zu sendenden (TSEND) oder zu empfangenden (TRCV) Daten fest. • Ad-hoc-Modus = 65535: Für den Datenempfang ist eine variable Länge eingestellt (TRCV).
ADHOC	IN	Bool	TRCV: Optionaler Parameter (versteckt) Ad-hoc-Modus-Anforderung für Verbindungstyp TCP.
DATA	IN_OUT	Variant	Pointer auf den Datenbereich für Senden (TSEND) oder Empfangen (TRCV). Der Datenbereich enthält die Adresse und Länge. Die Adresse bezieht sich auf einen der Speicherbereiche E, A oder M oder auf einen DB.
DONE	OUT	Bool	TSEND: • 0: Auftrag noch nicht gestartet oder läuft noch. • 1: Auftrag fehlerfrei ausgeführt.
NDR	OUT	Bool	TRCV: • NDR = 0: Auftrag noch nicht gestartet oder läuft noch. • NDR = 1: Auftrag erfolgreich beendet.
BUSY	OUT	Bool	• BUSY = 1: Der Auftrag ist noch nicht beendet. Es kann kein neuer Auftrag gestartet werden. • BUSY = 0: Auftrag beendet.
ERROR	OUT	Bool	ERROR = 1: Fehler während der Verarbeitung aufgetreten. STATUS enthält Informationen zum Fehlertyp.
STATUS	OUT	Word	Statusinformationen umfassen Fehlerinformationen. (Siehe Tabelle "Bedingungscodes für Error und Status" unten.)
RCVD_LEN	OUT	UDInt	TRCV: Tatsächliche empfangene Datenmenge in Byte

Hinweis

Die Anweisung TSEND erfordert einen Wechsel von 0 nach 1 am Eingangsparameter REQ, um einen Sendeauftrag zu starten. Der Parameter BUSY wird dann während der Verarbeitung auf 1 gesetzt. Die Fertigstellung des Sendeauftrags wird kenntlich gemacht, indem einer der Parameter DONE oder ERROR einen Zyklus lang auf 1 gesetzt wird. Während dieses Zeitraums wird ein Wechsel von 0 nach 1 am Eingangsparameter REQ ignoriert.

Funktionsweise von TRCV

Die Anweisung TRCV schreibt die empfangenen Daten in einen Empfangsbereich, der von den folgenden zwei Variablen angegeben wird:

- Zeigt auf den Anfang des Bereichs
- Länge des Bereichs oder der Wert, der am Eingang LEN bereitgestellt wird, sofern er nicht 0 ist

Hinweis

Die Standardeinstellung des Parameters LEN (LEN = 0) ermittelt anhand des Parameters DATA die Länge der zu übertragenden Daten. Stellen Sie sicher, dass die von der Anweisung TSEND gesendeten Daten (DATA) dieselbe Größe haben wie der Parameter DATA der Anweisung TRCV.

Sobald alle Auftragsdaten empfangen sind, werden sie von TRCV in den Empfangsbereich übertragen und NDR wird auf 1 gesetzt.

Tabelle 11- 26 Eintragen der Daten in den Empfangsbereich

Protokollvari-ante	Eintragen der Daten in den Empfangsbereich	Parameter "connection_type"	Wert des Parameters LEN	Wert des Parameters RCVD_LEN (Byte)
TCP	Ad-hoc-Modus	B#16#11	Ausgewählt mit dem Eingang ADHOC der Anweisung TRCV	1 bis 1472
TCP	Datenempfang mit angegebener Länge	B#16#11	0 (empfohlen) oder 1 bis 8192, außer 65535	1 bis 8192
ISO on TCP	Ad-hoc-Modus	B#16#12	65535	1 bis 1472
ISO on TCP	protokollgesteuert	B#16#12	0 (empfohlen) oder 1 bis 8192, außer 65535	1 bis 8192

Hinweis

Ad-hoc-Modus

Den Ad-hoc-Modus gibt es nur bei den Protokollvarianten TCP und ISO on TCP. Um die Anweisung TRCV für den Ad-hoc-Modus zu konfigurieren, setzen Sie den Eingangsparameter ADHOC der Anweisung. Der Empfangsbereich ist identisch mit dem von Parameter DATA angegebenen Bereich. Die Länge der empfangenen Daten wird am Parameter RCVD_LEN ausgegeben. Unmittelbar nach dem Empfang eines Datenblocks schreibt TRCV die Daten in den Empfangsbereich und setzt NDR auf 1.

Wenn Sie die Daten in einem "optimierten" DB (nur symbolisch adressierbar) speichern, können Sie Daten nur in Arrays der Datentypen Byte, Char, USInt und SInt empfangen.

Hinweis

STEP 7-Projekte der S7-300/400 mit Ad-hoc-Modus in die S7-1200 importieren

In STEP 7-Projekten der S7-300/400 wählen Sie den Ad-hoc-Modus aus, indem Sie dem Parameter LEN den Wert 0 zuweisen. In S7-1200 können Sie die Anweisung TRCV für den Ad-hoc-Modus konfigurieren, indem Sie den Eingangsparameter ADHOC der Anweisung setzen.

Wenn Sie ein STEP 7-Projekt der S7-300/400, das den Ad-hoc-Modus umfasst, in die S7-1200 importieren, müssen Sie den Parameter LEN in 65535 ändern.

Tabelle 11- 27 Bedingungscodes von ERROR und STATUS für TSEND und TRCV

ERROR	STATUS	Beschreibung
0	0000	• Sendeauftrag fehlerfrei ausgeführt (TSEND) • Neue Daten angenommen: Die aktuelle Länge der empfangenen Daten wird in RCVD_LEN (TRCV) gezeigt.
0	7000	• Keine laufende Auftragsbearbeitung (TSEND) • Baustein nicht empfangsbereit (TRCV)
0	7001	• Auftragsbearbeitung starten, Daten werden gesendet: Während dieser Bearbeitung greift das Betriebssystem auf die Daten im Sendebereich DATA zu (TSEND). • Baustein empfangsbereit, Empfangsauftrag wurde aktiviert (TRCV).
0	7002	• Folgeaufruf (REQ irrelevant), Auftrag wird bearbeitet: Während dieser Bearbeitung greift das Betriebssystem auf die Daten im Sendebereich DATA zu (TSEND). • Folgeaufruf, Empfangsauftrag wird bearbeitet: Während dieser Bearbeitung werden die Daten in den Empfangsbereich geschrieben. Deshalb kann ein Fehler zu inkonsistenten Daten im Empfangsbereich führen (TRCV).
1	8085	• Der Parameter LEN ist größer als der größte zulässige Wert (TSEND) und (TRCV). • Der Parameter LEN oder DATA hat sich seit der ersten Ausführung der Anweisung geändert (TRCV).
1	8086	Der Parameter ID ist nicht im zulässigen Adressbereich.
1	8088	Der Parameter LEN ist größer als der in DATA festgelegte Speicherbereich.

ERROR	STATUS	Beschreibung
1	80A1	Kommunikationsfehler: • Die angegebene Verbindung ist noch nicht hergestellt (TSEND und TRCV). • Die angegebene Verbindung wird beendet. Ein Sende- oder Empfangsauftrag ist über diese Verbindung nicht möglich (TSEND und TRCV). • Die Schnittstelle wird neu initialisiert (TSEND). • Die Schnittstelle empfängt neue Parameter (TRCV).
1	80C3	Interner Mangel an Verbindungsressourcen (Seite 659): Ein Baustein mit dieser ID wird bereits in einer anderen Prioritätsklasse bearbeitet.
1	80C4	Temporärer Kommunikationsfehler: • Die Verbindung zum Kommunikationspartner kann zur Zeit nicht hergestellt werden. • Die Schnittstelle empfängt neue Parametrierungen oder die Verbindung wird gerade aufgebaut.

Ethernet-Verbindungsprotokolle

Jede CPU hat einen integrierten PROFINET-Anschluss, der die standardmäßige PROFINET-Kommunikation unterstützt. Die Anweisungen TSEND_C, TRCV_C, TSEND und TRCV unterstützen alle die Ethernet-Protokolle TCP und ISO on TCP.

Weitere Informationen finden Sie unter "Gerätekonfiguration: Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren (Seite 638)".

11.2.8.10 Ältere Anweisungen TCON, TDISCON, TSEND und TRCV

Vor STEP 7 V13 SP1 und den S7-1200 V4.1 CPUs haben die Anweisungen TCON, TDISCON, TSEND und TRCV nur mit Verbindungsparametern mit Strukturen nach "TCON_Param" funktioniert. Für beide Anweisungsarten gelten die allgemeinen Konzepte. Informationen zur Programmierung finden sich in den Beschreibungen zu den einzelnen TCON-, TDISCON-, TSEND- und TRCV-Anweisungen.

Version der Anweisungen TCON, TDISCON, TSEND und TRCV auswählen

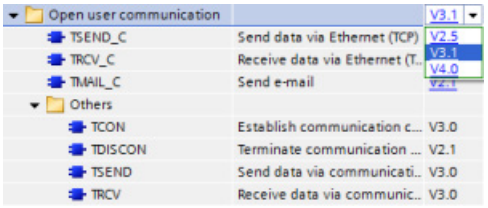
Es gibt zwei Versionen der Anweisungen TCON, TDISCON, TSEND oder TRCV in STEP 7:

- Die Versionen 2.5 und 3.1 waren in STEP 7 Basic/Professional V13 und früher verfügbar.
- Version 4.0 ist in STEP 7 Basic/Professional V13 SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Verwenden Sie niemals Anweisungen unterschiedlicher Versionen zusammen in einem CPU-Programm.

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.



Um die Version einer TCON-, TDISCON-, TSEND- oder TRCV-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine TCON-, TDISCON-, TSEND- oder TRCV-Anweisung über das Anweisungsverzeichnis in Ihr Programm platzieren, wird je nach der ausgewählten TCON-, TDISCON-, TSEND- oder TRCV-Anweisung eine neue FB- oder FC-Instanz in der Projektnavigation angelegt. Die neue FB- oder FC-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer TCON-, TDISCON-, TSEND- oder TRCV-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation eine FB- oder FC-Instanz einer TCON-, TDISCON-, TSEND- oder TRCV-Anweisung aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der TCON-, TDISCON-, TSEND- oder TRCV-Anweisung anzuzeigen.

Alte Anweisungen TCON, TDISCON, TSEND und TRCV (TCP-Kommunikation)

Ethernet-Kommunikation mit den Protokollen TCP und ISO on TCP

Hinweis

Anweisungen TSEND_C und TRCV_C

Um die Programmierung der PROFINET/Ethernet-Kommunikation zu vereinfachen, verbinden die Anweisungen TSEND_C und TRCV_C die Funktionalität der Anweisungen TCON, TDISCON. TSEND und TRCV:

- TSEND_C verbindet die Anweisungen TCON, TDISCON und TSEND.
- TRCV_C verbindet die Anweisungen TCON, TDISCON und TRCV.

Die folgenden Anweisungen steuern den Kommunikationsprozess:

- TCON baut die TCP/IP-Verbindung zwischen dem Client- und dem Server-PC (CPU) auf.
- TSEND und TRCV senden und empfangen Daten.
- TDISCON trennt die Verbindung.

Die Mindestdatengröße, die gesendet (TSEND) oder empfangen (TRCV) werden kann, ist ein Byte. Die Maximalgröße beträgt 8192 Byte. TSEND unterstützt nicht die Übertragung von Daten von Booleschen Adressen und TRCV empfängt Daten nicht in Booleschen Adressen. Beachten Sie für weitere Informationen zum Übertragen von Daten mit diesen Anweisungen den Abschnitt zur Datenkonsistenz (Seite 191).

TCON, TDISCON, TSEND und TRCV laufen asynchron ab, so dass die Bearbeitung eines Auftrags mehrere Anweisungsausführungen umfasst. Beispielsweise starten Sie einen Auftrag zum Einrichten und Herstellen einer Verbindung durch die Ausführung von TCON mit Parameter REQ = 1. Dann überwachen Sie den Stand der Auftragsausführung, indem TCON wiederholt ausgeführt wird, und prüfen, ob Parameter DONE den Abschluss des Auftrags bestätigt.

Die folgende Tabelle zeigt die Beziehungen zwischen BUSY, DONE und ERROR. Den aktuellen Status eines Auftrags können Sie anhand der Tabelle feststellen:

Tabelle 11- 28 Interaktionen zwischen den Parametern BUSY, DONE und ERROR

BUSY	DONE	ERROR	Beschreibung
WAHR	Irrelevant	Irrelevant	Auftrag wird bearbeitet.
FALSCH	WAHR	FALSCH	Auftrag erfolgreich ausgeführt.
FALSCH	FALSCH	WAHR	Auftrag mit Fehler beendet. Die Fehlerursache ist im Parameter STATUS hinterlegt.
FALSCH	FALSCH	FALSCH	Kein neuer Auftrag zugewiesen.

TCON und TDISCON

Hinweis

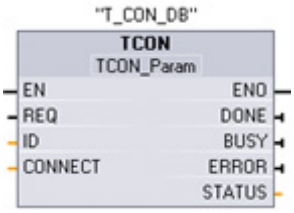
Initialisierung der Kommunikationsparameter

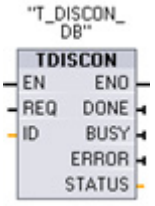
Nachdem Sie die Anweisung TCON eingefügt haben, konfigurieren Sie in den "Eigenschaften" der Anweisung (Seite 638) die Kommunikationsparameter (Seite 663). Wenn Sie die Parameter für die Kommunikationspartner im Inspektorfenster eingeben, gibt STEP 7 die entsprechenden Daten in den Instanz-DB der Anweisung ein.

Wenn Sie einen Multiinstanz-DB verwenden möchten, müssen Sie den DB manuell auf beiden CPUs konfigurieren.

Tabelle 11- 29 Anweisungen TCON und TDISCON

KOP/FUP		Beschreibung

	`"TCON_DB"(`
	`req:=_bool_in_,`
	`ID:=_undef_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`connect:=_struct_inout_);`
		TCP und ISO on TCP: TCON initiiert eine Kommunikationsverbindung von der CPU zu einem Kommunikationsteilnehmer.

	`"TDISCON_DB"(`
	`req:=_bool_in_,`
	`ID:=_word_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_);`
		TCP und ISO on TCP: TDISCON beendet eine Kommunikationsverbindung von der CPU zu einem Kommunikationsteilnehmer.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 11- 30 Datentypen für die Parameter von TCON und TDISCON

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Steuerparameter REQ startet den Auftrag durch Herstellung der von ID angegebenen Verbindung. Der Auftrag beginnt mit einer steigenden Flanke.
ID	IN	CONN_OUC (Word)	Verweis auf die Verbindung, die mit dem entfernten Teilnehmer oder zwischen dem Anwenderprogramm und der Kommunikationsschicht des Betriebssystems hergestellt (TCON) oder beendet (TDISCON) werden soll. Die ID muss mit der zugehörigen Parameter-ID in der lokalen Verbindungsbeschreibung übereinstimmen. Wertebereich: W#16#0001 bis W#16#0FFF
CONNECT (TCON)	IN_OUT	TCON_Param	Pointer auf die Verbindungsbeschreibung (Seite 663)
DONE	OUT	Bool	• 0: Auftrag noch nicht gestartet oder läuft noch. • 1: Auftrag fehlerfrei ausgeführt.
BUSY	OUT	Bool	• 0: Auftrag beendet. • 1: Auftrag noch nicht beendet. Es kann kein neuer Auftrag gestartet werden.
ERROR	OUT	Bool	Statusparameter mit den folgenden Werten: • 0: Kein Fehler • 1: Fehler während der Verarbeitung aufgetreten. STATUS enthält Informationen zum Fehlertyp.
STATUS	OUT	Word	Statusinformationen umfassen Fehlerinformationen. (Siehe Tabelle "Bedingungscodes für ERROR und STATUS" unten.)

Beide Kommunikationsteilnehmer führen die Anweisung TCON aus, um die Kommunikationsverbindung einzurichten und herzustellen. Mit Parametern geben Sie die aktiven und passiven Endpunkte der Kommunikation an. Nach dem Einrichten und Aufbauen der Verbindung wird sie automatisch von der CPU gehalten und überwacht.

Wird die Verbindung durch eine Leitungsstörung oder vom entfernten Teilnehmer abgebrochen, so versucht der aktive Teilnehmer, sie wieder aufzubauen. Sie müssen TCON nicht erneut ausführen.

Wird die Anweisung TDISCON ausgeführt oder ist die CPU in den Betriebszustand STOP gegangen, so wird die bestehende Verbindung beendet und die eingerichtete Verbindung gelöscht. Um die Verbindung einzurichten und wiederherzustellen, müssen Sie TCON erneut ausführen.

Tabelle 11- 31 Bedingungscodes von ERROR und STATUS für TCON und TDISCON

ERROR	STATUS	Beschreibung
0	0000	Verbindung wurde erfolgreich aufgebaut.
0	7000	Keine laufende Auftragsbearbeitung
0	7001	Auftragsbearbeitung starten, Verbindung herstellen (TCON) oder Verbindung beenden (TDISCON)
0	7002	Folgeaufruf (REQ irrelevant), Verbindung herstellen (TCON) oder Verbindung beenden (TDISCON)
1	8086	Parameter ID ist außerhalb des zulässigen Adressbereichs.
1	8087	TCON: Maximale Anzahl Verbindungen erreicht; keine weitere Verbindung möglich.
1	809B	TCON: local_device_id in der Verbindungsbeschreibung stimmt nicht mit der CPU überein.
1	80A1	TCON: Verbindung oder Port ist vom Anwender belegt.
1	80A2	TCON: Lokaler oder entfernter Port ist vom System belegt.
1	80A3	Es wird versucht, eine vorhandene Verbindung erneut aufzubauen (TCON) oder eine nicht vorhandene Verbindung zu beenden (TDISCON).
1	80A4	TCON: IP-Adresse des entfernten Verbindungspunkts ist ungültig, sie entspricht der lokalen IP-Adresse.
1	80A5	TCON: Verbindungs-ID (Seite 659) wird bereits verwendet.
1	80A7	TCON: Kommunikationsfehler: Sie haben TDISCON ausgeführt, bevor TCON beendet war. TDISCON muss zunächst die von der ID angegebene Verbindung vollständig beenden.
1	80B2	TCON: Der Parameter CONNECT zeigt auf einen Datenbaustein, der mit dem Attribut "Nur im Ladespeicher ablegen" generiert wurde.

ERROR	STATUS	Beschreibung
1	80B4	TCON: Wenn Sie eine passive Verbindung über ISO on TCP (connection_type = B#16#12) aufbauen, werden Sie anhand des Fehlercodes 80B4 gewarnt, wenn der eingegebene TSAP keiner der folgenden Voraussetzungen für Adressen entspricht: • Hat der lokale TSAP eine Länge von 2 und einen ID-Wert von E0 oder E1 (hexadezimal) für das erste Byte, muss das zweite Byte entweder 00 oder 01 sein. • Hat der lokale TSAP eine Länge von 3 oder größer und einen ID-Wert von E0 oder E1 (hexadezimal) für das erste Byte, muss das zweite Byte entweder 00 oder 01 sein und alle Bytes müssen gültige ASCII-Zeichen sein. • Hat der lokale TSAP eine Länge von 3 oder größer und das erste Byte des ID-Werts des TSAP nicht den Wert E0 oder E1 (hexadezimal), müssen alle Bytes der TSAP-ID gültige ASCII-Zeichen sein. Gültige ASCII-Zeichen sind Bytewerte von 20 bis 7E (hexadezimal).
1	80B5	TCON: Verbindungstyp "13 = UDP" gestattet nur passiven Verbindungsaufbau.
1	80B6	TCON: Parametrierungsfehler im Parameter CONNECTION_TYPE des SDT-TCON_Param.
1	80B7	TCON: Fehler in einem der folgenden Parameter des Datenbausteins für die Verbindungsbeschreibung: • block_length • local_tsap_id_len • rem_subnet_id_len • rem_staddr_len • rem_tsap_id_len • next_staddr_len Hinweis: Beim Betrieb von TCON im passiven TCP-Modus muss LOCAL_TSAP_ID_LEN "2" und REM_TSAP_ID_LEN muss "0" sein.
1	80B8	TCON: Parameter der lokalen Verbindungsbeschreibung und Parameter-ID sind verschieden.
1	80C3	TCON: Alle Verbindungsressourcen sind belegt.
1	80C4	Temporärer Kommunikationsfehler: • Die Verbindung kann zur Zeit nicht hergestellt werden (TCON). • Die konfigurierte Verbindung wird von TDISCON (TCON) entfernt. • Die angegebene Verbindung wird derzeit aufgebaut (TDISCON). • Die Schnittstelle empfängt neue Parameter (TCON und TDISCON).

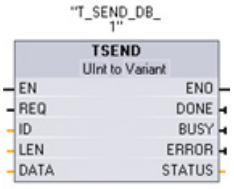
TSEND und TRCV

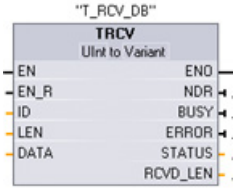
Hinweis

Wenn Sie in der offenen Benutzerkommunikation über PROFINET eine Anweisung TSEND ohne die entsprechende Anweisung TRCV auf dem Remote-Gerät ausführen, dann kann es passieren, dass die Anweisung TSEND auf unbestimmte Zeit im Zustand "Busy" bleibt und darauf wartet, dass die Anweisung TRCV die Daten empfängt. In diesem Zustand ist der Ausgang "Busy" der Anweisung TSEND gesetzt und der Ausgang "Status" hat den Wert "0x7002". Diese Bedingung kann auftreten, wenn Sie mehr als 4096 Byte an Daten übertragen. Das Problem wird bei der nächsten Ausführung der Anweisung TRCV behoben.

Tabelle 11- 32 Anweisungen TSEND und TRCV

KOP/FUP	SCL	Beschreibung

	`"TSEND_DB"(`
	`req:=_bool_in_,`
	`ID:=_word_in_,`
	`len:=_udint_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`data:=_variant_inout_);`
		TCP und ISO on TCP: TSEND sendet über eine Kommunikationsverbindung Daten von der CPU zu einer Partnerstation.

	`"TRCV_DB"(`
	`en_r:=_bool_in_,`
	`ID:=_word_in_,`
	`len:=_udint_in_,`
	`ndr=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`rcvd_len=>_udint_out_,`
	`data:=_variant_inout_);`
		TCP und ISO on TCP: TRCV empfängt über eine Kommunikationsverbindung Daten von einer Partnerstation auf der CPU.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 11- 33 Datentypen für die Parameter von TSEND und TRCV

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	TSEND: Startet den Sendeauftrag bei einer steigenden Flanke. Die Daten werden aus dem mit DATA und LEN bezeichneten Bereich übertragen.
EN_R	IN	Bool	TRCV: Aktiviert die CPU für den Empfang, bei EN_R = 1 ist TRCV empfangsbereit. Der Empfangsauftrag wird bearbeitet.
ID	IN	CONN_OUC (Word)	Verweis auf die zugehörige Verbindung. Die ID muss mit der zugehörigen Parameter-ID in der lokalen Verbindungsbeschreibung übereinstimmen. Wertebereich: W#16#0001 bis W#16#0FFF
LEN	IN	UInt	Maximale Bytezahl, die gesendet (TSEND) oder empfangen (TRCV) werden soll: • Standard = 0: Der Parameter DATA legt die Länge der zu sendenden (TSEND) oder zu empfangenden (TRCV) Daten fest. • Ad-hoc-Modus = 65535: Für den Datenempfang ist eine variable Länge eingestellt (TRCV).
DATA	IN_OUT	Variante	Pointer auf den Datenbereich für Senden (TSEND) oder Empfangen (TRCV). Der Datenbereich enthält die Adresse und Länge. Die Adresse bezieht sich auf einen der Speicherbereiche E, A oder M oder auf einen DB.
DONE	OUT	Bool	TSEND: • 0: Auftrag noch nicht gestartet oder läuft noch. • 1: Auftrag fehlerfrei ausgeführt.
NDR	OUT	Bool	TRCV: • NDR = 0: Auftrag noch nicht gestartet oder läuft noch. • NDR = 1: Auftrag erfolgreich beendet.
BUSY	OUT	Bool	• BUSY = 1: Der Auftrag ist noch nicht beendet. Es kann kein neuer Auftrag gestartet werden. • BUSY = 0: Auftrag beendet.
ERROR	OUT	Bool	ERROR = 1: Fehler während der Verarbeitung aufgetreten. STATUS enthält Informationen zum Fehlertyp.
STATUS	OUT	Word	Statusinformationen umfassen Fehlerinformationen. (Siehe Tabelle "Bedingungscodes für Error und Status" unten.)
RCVD_LEN	OUT	Int	TRCV: Tatsächliche empfangene Datenmenge in Byte

Hinweis

Die Anweisung TSEND erfordert einen Wechsel von 0 nach 1 am Eingangsparameter REQ, um einen Sendeauftrag zu starten. Der Parameter BUSY wird dann während der Verarbeitung auf 1 gesetzt. Die Fertigstellung des Sendeauftrags wird kenntlich gemacht, indem einer der Parameter DONE oder ERROR einen Zyklus lang auf 1 gesetzt wird. Während dieses Zeitraums wird ein Wechsel von 0 nach 1 am Eingangsparameter REQ ignoriert.

Funktionsweise von TRCV

Die Anweisung TRCV schreibt die empfangenen Daten in einen Empfangsbereich, der von den folgenden zwei Variablen angegeben wird:

- Zeigt auf den Anfang des Bereichs
- Länge des Bereichs oder der Wert, der am Eingang LEN bereitgestellt wird, sofern er nicht 0 ist

Hinweis

Die Standardeinstellung des Parameters LEN (LEN = 0) ermittelt anhand des Parameters DATA die Länge der zu übertragenden Daten. Stellen Sie sicher, dass die von der Anweisung TSEND gesendeten Daten (DATA) dieselbe Größe haben wie der Parameter DATA der Anweisung TRCV.

Sobald alle Auftragsdaten empfangen sind, werden sie von TRCV in den Empfangsbereich übertragen und NDR wird auf 1 gesetzt.

Tabelle 11- 34 Eintragen der Daten in den Empfangsbereich

Protokollvari-ante	Eintragen der Daten in den Empfangsbereich	Parameter "connection_type"	Wert des Parameters LEN	Wert des Parameters RCVD_LEN (Byte)
TCP	Ad-hoc-Modus	B#16#11	65535	1 bis 1472
TCP	Datenempfang mit angegebener Länge	B#16#11	0 (empfohlen) oder 1 bis 8192, außer 65535	1 bis 8192
ISO on TCP	Ad-hoc-Modus	B#16#12	65535	1 bis 1472
ISO on TCP	protokollgesteuert	B#16#12	0 (empfohlen) oder 1 bis 8192, außer 65535	1 bis 8192

Hinweis

Ad-hoc-Modus

Den Ad-hoc-Modus gibt es nur bei den Protokollvarianten TCP und ISO on TCP. Den Ad-hoc-Modus stellen Sie ein, indem Sie dem Parameter LEN den Wert 65535 zuweisen. Der Empfangsbereich ist identisch mit dem von Parameter DATA angegebenen Bereich. Die Länge der empfangenen Daten wird an den Parameter RCVD_LEN ausgegeben. Unmittelbar nach dem Empfang eines Datenblocks schreibt TRCV die Daten in den Empfangsbereich und setzt NDR auf 1.

Wenn Sie die Daten in einem "optimierten" DB (nur symbolisch adressierbar) speichern, können Sie Daten nur in Arrays der Datentypen Byte, Char, USInt und SInt empfangen.

Hinweis

STEP 7-Projekte der S7-300/400 mit Ad-hoc-Modus in die S7-1200 importieren

In STEP 7-Projekten der S7-300/400 wählen Sie den Ad-hoc-Modus aus, indem Sie dem Parameter LEN den Wert 0 zuweisen. Bei der S7-1200 stellen Sie den Ad-hoc-Modus ein, indem Sie dem Parameter LEN den Wert 65535 zuweisen.

Wenn Sie ein STEP 7-Projekt der S7-300/400, das den Ad-hoc-Modus umfasst, in die S7-1200 importieren, müssen Sie den Parameter LEN in 65535 ändern.

Bedingungscodes von TSEND und TRCV Error und Status

ERROR	STATUS	Beschreibung
0	0000	• Sendeauftrag fehlerfrei ausgeführt (TSEND) • Neue Daten angenommen: Die aktuelle Länge der empfangenen Daten wird in RCVD_LEN (TRCV) gezeigt.
0	7000	• Keine laufende Auftragsbearbeitung (TSEND) • Baustein nicht empfangsbereit (TRCV)
0	7001	• Auftragsbearbeitung starten, Daten werden gesendet: Während dieser Bearbeitung greift das Betriebssystem auf die Daten im Sendebereich DATA zu (TSEND). • Baustein empfangsbereit, Empfangsauftrag wurde aktiviert (TRCV).
0	7002	• Folgeaufruf (REQ irrelevant), Auftrag wird bearbeitet: Während dieser Bearbeitung greift das Betriebssystem auf die Daten im Sendebereich DATA zu (TSEND). • Folgeaufruf, Empfangsauftrag wird bearbeitet: Während dieser Bearbeitung werden die Daten in den Empfangsbereich geschrieben. Deshalb kann ein Fehler zu inkonsistenten Daten im Empfangsbereich führen (TRCV).
1	8085	• Der Parameter LEN ist größer als der größte zulässige Wert (TSEND) und (TRCV). • Der Parameter LEN oder DATA hat sich seit der ersten Ausführung der Anweisung geändert (TRCV).
1	8086	Der Parameter ID ist nicht im zulässigen Adressbereich.
1	8088	Der Parameter LEN ist größer als der in DATA festgelegte Speicherbereich.
1	80A1	Kommunikationsfehler: • Die angegebene Verbindung ist noch nicht hergestellt (TSEND und TRCV). • Die angegebene Verbindung wird beendet. Ein Sende- oder Empfangsauftrag ist über diese Verbindung nicht möglich (TSEND und TRCV). • Die Schnittstelle wird neu initialisiert (TSEND). • Die Schnittstelle empfängt neue Parameter (TRCV).
1	80C3	Interner Mangel an Ressourcen: Ein Baustein mit dieser ID wird bereits in einer anderen Prioritätsklasse bearbeitet.
1	80C4	Temporärer Kommunikationsfehler: • Die Verbindung zum Kommunikationspartner kann zur Zeit nicht hergestellt werden. • Die Schnittstelle empfängt neue Parametrierungen oder die Verbindung wird gerade aufgebaut.

Ethernet-Verbindungsprotokolle

Jede CPU hat einen integrierten PROFINET-Anschluss, der die standardmäßige PROFINET-Kommunikation unterstützt. Die Anweisungen TSEND_C, TRCV_C, TSEND und TRCV unterstützen alle die Ethernet-Protokolle TCP und ISO on TCP.

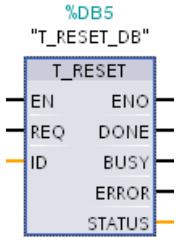
Weitere Informationen finden Sie unter "Gerätekonfiguration: Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren (Seite 638)".

11.2.8.11 Anweisung T_RESET (bestehende Verbindung beenden und neu aufbauen)

Mit der Anweisung "T_RESET" wird eine bestehende Verbindung beendet und anschließend neu aufgebaut:

Tabelle 11- 35 Anweisung T_RESET

KOP/FUP	SCL	Beschreibung

	`"T_RESET_DB"(`
	`req:=_bool_in_,`
	`id:=_word_in_,`
	`done=>_bool_out_,`
	`error=>_bool_out_,`
	`sta-`
	`tus=>_word_out_);`
		Mit der Anweisung "T_RESET" wird eine bestehende Verbindung beendet und anschließend neu aufgebaut.

Die lokalen Endpunkte der Verbindung bleiben erhalten. Sie werden automatisch erzeugt:

- wenn eine Verbindung konfiguriert und in die CPU geladen wurde
- wenn eine Verbindung vom Anwenderprogramm erzeugt wurde, zum Beispiel durch Aufruf der Anweisung "TCON (Seite 688)".

Die Anweisung "T_RESET" kann für alle Verbindungstypen ausgeführt werden, unabhängig davon, ob die lokale Schnittstelle der CPU oder die Schnittstelle eines CM/CP für die Verbindung verwendet wurde. Eine Ausnahme bilden Verbindungen für die Datenübertragung im Ad-hoc-Modus mit TCP, da auf diese Verbindungen nicht mit einer Verbindungs-ID verwiesen werden kann.

Wenn die Anweisung "T_RESET" mit dem Parameter REQ aufgerufen wurde, wird die mit dem Parameter ID angegebene Verbindung beendet und, falls erforderlich, der Sende- und Empfangspuffer für die Daten gelöscht. Durch den Abbau der Verbindung wird auch eine laufende Datenübertragung abgebrochen. Daher kann es zu Datenverlust kommen, wenn gerade eine Datenübertragung stattfindet. Die als aktiver Verbindungspartner eingerichtete CPU versucht dann automatisch, die unterbrochene Kommunikationsverbindung wieder aufzubauen. Es ist deshalb nicht nötig, die Anweisung "TCON (Seite 688)" aufzurufen, um die Kommunikationsverbindung wiederherzustellen.

Die Ausgangsparameter DONE, BUSY und STATUS geben den Auftragsstatus an.

Datentypen für die Parameter

Die folgende Tabelle zeigt die Parameter der Anweisung T_RESET:

Parameter	Deklaration	Datentyp	Speicherbereich	Beschreibung
REQ	Input	BOOL	E, A, M, D, L, T, C oder konstant	Der Steuerparameter REQUEST startet den Auftrag zur Beendigung der durch ID angegebenen Verbindung. Der Auftrag startet bei einer steigenden Flanke.
ID	Input	CONN_OUC (WORD)	L, D oder konstant	Verweis auf die Verbindung mit dem passiven Partner, die beendet werden soll. ID muss mit der entsprechenden Parameter-ID in der lokalen Verbindungsbeschreibung übereinstimmen. Wertebereich: W#16#0001 bis W#16#0FFF
DONE	Output	BOOL	E, A, M, D, L	Statusparameter DONE • 0: Auftrag noch nicht gestartet oder wird noch ausgeführt. • 1: Auftrag fehlerfrei ausgeführt.
BUSY	Output	BOOL	E, A, M, D, L	Statusparameter BUSY • 0: Auftrag beendet. • 1: Auftrag noch nicht beendet.
ERROR	Output	BOOL	E, A, M, D, L	Statusparameter ERROR • 0: Kein Fehler. • 1: Fehler bei der Bearbeitung aufgetreten. Parameter STATUS enthält ausführliche Informationen über die Fehlerart
STATUS	Output	WORD	E, A, M, D, L	Statusparameter STATUS Fehlerinformation (siehe Tabelle "STATUS-Parameter").

Parameter STATUS

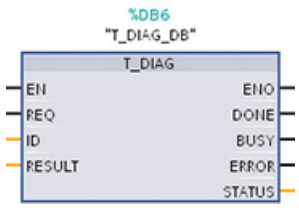
Fehlerbit	STATUS* (W#16#...)	Beschreibung
0	0000	Kein Fehler.
0	0001	Verbindung nicht aufgebaut.
0	7001	Verbindungsabbau gestartet.
0	7002	Verbindung wird beendet.
1	8081	Unbekannte Verbindung an Parameter ID angegeben.

11.2.8.12 Anweisung T_DIAG (Status einer Verbindung prüfen und Informationen lesen)

Die Anweisung T_DIAG prüft den Status einer Verbindung und liest weitere Informationen auf dem lokalen Endpunkt dieser Verbindung:

Tabelle 11- 36 Anweisung T_DIAG

KOP/FUP	SCL	Beschreibung

	`"T_DIAG_DB"(`
	`req:=_bool_in_,`
	`id:=_word_in_,`
	`done=>_bool_out_,`
	`error=>_bool_out_,`
	`sta-`
	`tus=>_dword_out_);`
		Mit der Anweisung T_DIAG prüfen Sie den Status einer Verbindung und lesen weitere Informationen auf dem lokalen Endpunkt dieser Verbindung.

Die Anweisung T_DIAG funktioniert wie folgt:

- Die Verbindung wird vom Parameter ID angegeben. Sie können beide im Verbindungseditor konfigurierte Verbindungsendpunkte und programmierte Verbindungsendpunkte (z. B. mit der Anweisung TCON) lesen.

  Temporäre Verbindungsendpunkte (z. B. bei der Verbindung mit einer Engineering Station erstellte Endpunkte) können nicht diagnostiziert werden, weil bei diesem Vorgang keine Verbindungs-ID erzeugt wird.
- Die gelesenen Verbindungsinformationen werden in einer vom Parameter RESULT angegebenen Struktur gespeichert.
- Der Ausgangsparameter STATUS gibt an, ob das Lesen der Verbindungsinformationen möglich war. Die Verbindungsinformationen in der Struktur am Parameter RESULT sind nur gültig, wenn die Anweisung T_DIAG mit STATUS = W#16#0000 und ERROR = FALSE abgeschlossen wurde.

  Tritt ein Fehler auf, können die Verbindungsinformationen nicht ausgewertet werden.

Mögliche Verbindungsinformationen

Die Struktur TDiag_Status kann zum Lesen der Verbindungsinformationen am Parameter RESULT verwendet werden. Die Struktur TDiag_Status enthält nur die wichtigsten Informationen über einen Verbindungsendpunkt (z. B. das verwendete Protokoll, den Verbindungszustand und die Anzahl der gesendeten oder empfangenen Datenbytes).

Die Struktur und Parameter der TDiag_Status-Struktur werden nachstehend beschrieben (siehe Tabelle zur Struktur TDIAG_Status).

Datentypen für die Parameter

Die folgende Tabelle zeigt die Parameter der Anweisung T_DIAG:

Parameter	Deklaration	Datentyp	Speicherbereich	Beschreibung
REQ	Input	BOOL	E, A, M, D, L, T, C oder Konstante	Startet die Anweisung zum Prüfen der im Parameter ID angegebenen Verbindung, wenn eine steigende Flanke vorliegt.
ID	Input	CONN_OUC (WORD)	L, D oder Konstante	Verweis auf die zugewiesene Verbindung. Wertebereich: W#16#0001 bis W#16#0FFF
RESULT	InOut	VARIANT	D	Pointer auf die Struktur, in der die Verbindungsinformationen gespeichert werden. Die Struktur TDiag_Status kann am Parameter RESULT verwendet werden (eine Beschreibung finden Sie in der Tabelle zur Struktur TDIAG_Status).
DONE	Output	BOOL	E, A, M, D, L	Statusparameter: • 0: Anweisung noch nicht gestartet oder noch in Bearbeitung. • 1: Anweisung fehlerfrei ausgeführt.
BUSY	Output	BOOL	E, A, M, D, L	Statusparameter: • 0: Anweisung noch nicht gestartet oder bereits beendet. • 1: Anweisung noch nicht abgeschlossen. Es kann kein neuer Auftrag gestartet werden.
ERROR	Output	BOOL	E, A, M, D, L	Statusparameter: • 0: Kein Fehler. • 1: Fehler aufgetreten.
STATUS	Output	WORD	E, A, M, D, L	Zustand der Anweisung

Parameter BUSY, DONE und ERROR

Den Status der Ausführung von T_DIAG können Sie mit den Parametern BUSY, DONE, ERROR und STATUS prüfen. Der Parameter BUSY zeigt den Bearbeitungszustand an. Mit dem Parameter DONE prüfen Sie, ob eine Anweisung erfolgreich ausgeführt wurde. Der Parameter ERROR wird gesetzt, wenn während der Ausführung von T_DIAG Fehler auftreten.

Die folgende Tabelle zeigt die Beziehungen zwischen den Parametern BUSY, DONE und ERROR:

BUSY	DONE	ERROR	Beschreibung
1	-	-	Die Anweisung wird bearbeitet.
0	1	0	Die Anweisung wurde erfolgreich ausgeführt. Die Daten in der von RESULT angegebenen Struktur sind nur gültig, wenn dies der Fall ist.
0	0	1	Anweisung mit Fehler beendet. Die Fehlerursache wird am Parameter STATUS ausgegeben.
0	0	0	Es wurde keine neue Anweisung zugewiesen.

Parameter STATUS

Die folgende Tabelle zeigt die Bedeutung der Werte am Parameter STATUS:

Fehlerbit	STATUS* (W#16#...)	Beschreibung
0	0000	Die Anweisung T_DIAG wurde erfolgreich ausgeführt. Die Daten in der am Parameter RESULT angegebenen Struktur können ausgewertet werden.
0	7000	Keine Anweisungsbearbeitung aktiv.
0	7001	Anweisungsbearbeitung gestartet.
0	7002	Verbindungsinformationen werden gelesen (Parameter REQ ist irrelevant).
1	8086	Der Wert am Parameter ID liegt außerhalb des gültigen Bereichs (W#16#0001 bis W#16#0FFF).
1	8089	Der Parameter RESULT zeigt auf einen ungültigen Datentyp (nur Strukturen TDIAG_Status und TDIAG_StatusExt).
1	80A3	Der Parameter ID verweist auf einen Verbindungsendpunkt, der nicht vorhanden ist. Bei programmierten Verbindungen kann dieser Fehler auch nach Aufruf der Anweisung TDISCON auftreten.
1	80C4	Interner Fehler. Der Zugriff auf den Verbindungsendpunkt ist kurzzeitig nicht verfügbar.

Struktur TDIAG_Status

Die folgende Tabelle enthält Einzelheiten zur Struktur TDIAG_Status. Der Wert jedes Elements ist nur gültig, wenn die Anweisung fehlerfrei ausgeführt wurde. Tritt ein Fehler auf, ändert sich der Inhalt der Parameter nicht:

Name	Datentyp	Beschreibung
Die folgenden Parameter befinden sich in der Struktur TDIAG_Status:		
InterfaceID	HW_ANY	Schnittstellen-ID (LADDR) der CPU oder des CM/CP.
ID	CONN_OUC	ID der diagnostizierten Verbindung. Nach einem erfolgreichen Aufruf ist der Wert dieses Elements mit der Parameter-ID der Anweisung T_DIAG identisch.
ConnectionType	BYTE	Für die Verbindung verwendeter Protokolltyp: • 0x01: Nicht verwendet. • ... • 0x0B: TCP-Protokoll (IP_v4) • 0x0C: ISO-on-TCP-Protokoll (RFC1006) • 0x0D: TCP-Protokoll (DNS) • 0x0E: Wählprotokoll • 0x0F: WDC-Protokoll • 0x10: SMTP-Protokoll • 0x11: TCP-Protokoll • 0x12: TCP- und ISO-on-TCP-Protokoll (RFC1006) • 0x13: UDP-Protokoll • 0x14: Reserviert • 0x15: PROFIBUS-Buszugriffsprotokoll (FDL) • 0x16: Transportprotokoll nach ISO 8073 (ISO nativ) • ... • 0x20: SMTP- oder SMTPS-Protokoll - basierend auf IPv4 • 0x21: SMTP- oder SMTPS-Protokoll - basierend auf IPv6 • 0x22: SMTP- oder SMTPS-Protokoll - basierend auf FQDN (**F**ully **Q**ualified **D**omain **N**ame) • ... • 0x70: S7-Verbindung • Andere: Reserviert
ActiveEstablished	BOOL	• FALSE: Lokal, der passive Verbindungsendpunkt • TRUE: Lokal, der aktive Verbindungsendpunkt

Name	Datentyp	Beschreibung
State	BYTE	Aktueller Status des Verbindungsendpunkts • 0x00: Nicht verwendet. • 0x01: Verbindung beendet. Temporärer Status, beispielsweise nach Aufruf der Anweisung T_RESET. Das System versucht dann automatisch, die Verbindung wiederherzustellen. • 0x02: Der aktive Verbindungsendpunkt versucht, eine Verbindung zum dezentralen Kommunikationspartner herzustellen. • 0x03: Der passive Verbindungsendpunkt wartet auf die Herstellung der Verbindung zum dezentralen Kommunikationspartner. • 0x04: Verbindung hergestellt. • 0x05: Die Verbindung wird beendet. Dies geschieht möglicherweise, weil die Anweisung "T_RESET" oder "T_DISCON" aufgerufen wurde. Andere mögliche Gründe sind Protokollfehler oder Leitungspausen. • 0x06..0xFF: Nicht verwendet.
Kind	BYTE	Modus des Verbindungsendpunkts: • 0x00: Nicht verwendet. • 0x01: Konfigurierte statische Verbindung, die konfiguriert und in die CPU geladen wurde. • 0x02: Konfigurierte dynamische Verbindung, die konfiguriert und in die CPU geladen wurde (wird derzeit nicht unterstützt). • 0x03: Programmierte Verbindung, die im Anwenderprogramm mit der Anweisung "TCON" erzeugt wurde. Ein Aufruf der Anweisung TDISCON oder ein Wechsel in den Betriebszustand STOP der CPU hat den Verbindungsendpunkt zerstört. • 0x04: Temporäre dynamische Verbindung, die beispielsweise von der Engineering Station (ES) oder der Operator Station (OS) aufgebaut wurde. (Dieser Verbindungstyp kann derzeit nicht diagnostiziert werden, weil er keine ID hat.) • 0x05..0xFF: Nicht verwendet.
SentBytes	UDINT	Anzahl der gesendeten Datenbytes.
ReceivedBytes	UDINT	Anzahl der empfangenen Datenbytes.

11.2.8.13 Anweisung TMAIL_C (Email über die Ethernet-Schnittstelle der CPU senden)

Überblick

Mit der Anweisung "TMAIL_C" können Sie eine E-Mail über die Ethernet-Schnittstelle der S7-1200 CPU senden.

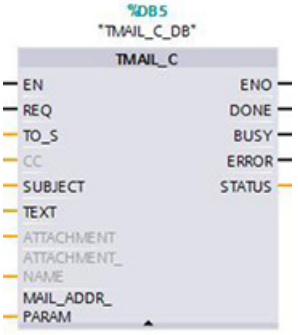
Die Anweisung TMAIL_C hat zwei Funktionalitäten:

- E-Mail über die CPU-Schnittstelle (nur SMTP ohne SSL)
- E-Mail über eine CP-Schnittstelle (entweder SMTP ohne SSL oder SMTP mit SSL). Wenn Sie die SSL-Funktionalität benötigen, müssen Sie den Eingangsparameter CERTINDEX von TMAIL_C = 1 setzen und die CP-Schnittstelle verwenden. Außerdem muss das richtige Zertifikat im Zertifikatspeicher des CP abgelegt sein.

Die Anweisung kann nur verwendet werden, wenn die Hardware konfiguriert ist und wenn die Netzwerkinfrastruktur eine Kommunikationsverbindung mit dem Mailserver ermöglicht.

Tabelle 11- 37 Anweisung TMAIL_C

KOP/FUP	SCL	Beschreibung

	`"TMAIL_C_DB"(`
	`req:=_bool_in_,`
	`to_s:=_string_in_,`
	`cc:=_string_in_,`
	`subject:=_string_in_,`
	`text:=_string_in_,`
	`attachment:=_variant_in_,`
	`attach-`
	`ment_name:=_string_in_,`
	
	`mail_addr_param:=_string_i`
	`n_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_);`
		Mit der Anweisung TMAIL_C können Sie eine E-Mail über die Ethernet-Schnittstelle der S7-1200 CPU senden.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Den Inhalt der E-Mail und die Verbindungsdaten legen Sie mit den folgenden Parametern fest:

- Mit den Parametern TO_S und CC legen Sie die Empfängeradresse fest.
- Den Inhalt der E-Mail legen Sie mit den Parametern SUBJECT und TEXT fest.
- Mit Pointern VARIANT an den Parametern ATTACHMENT und ATTACHMENT_NAME können Sie außerdem eine Anlage definieren.
- Mit dem Systemdatentyp Tmail_v4 oder Tmail_FQDN am Parameter MAIL_ADDR_PARAM werden die Verbindungsdaten definiert und die Adressierung und Authentifizierung für den Mailserver ausgeführt. Wenn Sie die Schnittstelle der S7-1200 CPU verwenden, muss der Systemdatentyp Tmail_v4 verwendet werden. In diesem Fall kann die E-Mail nur über SMTP gesendet werden.
- Das Senden der E-Mail wird gestartet, wenn die Flanke für den Parameter REQ von "0" auf "1" wechselt.
- Der Status des Sendeauftrags wird mit den Ausgangsparametern "BUSY", "DONE", "ERROR" und "STATUS" angegeben.

Sie können eine SMS nicht direkt mit der Anweisung "TMAIL_C" senden. Ob die E-Mail vom Mailserver als SMS weitergeleitet werden kann, hängt von Ihrem Telekommunikationsanbieter ab.

Funktionsweise der Anweisung

Die Anweisung "TMAIL_C" funktioniert asynchron, das heißt, ihre Ausführung erstreckt sich über mehrere Aufrufe. Beim Aufrufen der Anweisung "TMAIL_C" muss eine Instanz angegeben werden.

In den folgenden Fällen wird die Verbindung zum Mailserver abgebrochen:

- Wenn die CPU in den Betriebszustand STOP geht, während "TMAIL_C" aktiv ist.
- Wenn Kommunikationsprobleme auf dem Industrial Ethernet-Bus auftreten. In diesem Fall wird das Senden der E-Mail unterbrochen und die E-Mail erreicht den Empfänger nicht.

Die Verbindung wird auch beendet, wenn die Anweisung erfolgreich ausgeführt und die E-Mail gesendet wurde.

ACHTUNG
Anwenderprogramme ändern Sie können die Teile Ihres Anwenderprogramms, die direkt die Aufrufe von "TMAIL_C" betreffen, nur ändern, wenn: • sich die CPU in STOP befindet • gerade keine E-Mail gesendet wird (d. h., REQ = 0 und BUSY = 0). Dies betrifft insbesondere das Löschen und Ersetzen von Programmbausteinen, die Aufrufe von TMAIL_C oder der Instanz von TMAIL_C enthalten. Wird diese Einschränkung nicht beachtet, können Verbindungsressourcen blockiert werden. Das Automatisierungssystem kann gegenüber den TCP/IP-Kommunikationsfunktionen über Industrial Ethernet in einen undefinierten Zustand wechseln. Nach Übertragung der Änderungen ist ein Neustart der CPU mittels Warm- oder Kaltstart erforderlich.

Datenkonsistenz

Die Parameter TO_S, CC, SUBJECT, TEXT, ATTACHMENT und MAIL_ADDR_PARAM werden von der Anweisung "TMAIL_C" während ihrer Ausführung benutzt, so dass die Parameter erst geändert werden können, nachdem der Auftrag abgeschlossen ist (BUSY = 0).

SMTP-Authentifizierung

Authentifizierung bezeichnet hier ein Verfahren zur Identitätsprüfung, zum Beispiel mit einer Passwortabfrage.

Wenn Sie die Schnittstelle der S7-1200 CPU verwenden, unterstützt die Anweisung "TMAIL_C" das SMTP-Authentifizierungsverfahren AUTH-LOGIN, das für die meisten Mailserver erforderlich ist. Informationen zum Authentifizierungsverfahren Ihres Mailservers finden Sie im Handbuch zu Ihrem Mailserver oder auf der Webseite Ihres Internet-Dienstanbieters.

- Um das Authentifizierungsverfahren AUTH-LOGIN nutzen zu können, benötigt die Anweisung "TMAIL_C" den Benutzernamen, mit dem sie sich beim Mailserver anmelden soll. Dieser Benutzername ist der gleiche, mit dem Sie einen Mail-Account auf Ihrem Mailserver einrichten. Er wird mit dem Parameter UserName an die Struktur im Parameter MAIL_ADDR_PARAM übertragen.

  Wird im Parameter MAIL_ADDR_PARAM kein Benutzername eingegeben, wird das Authentifizierungsverfahren AUTH-LOGIN nicht verwendet. Die E-Mail wird dann ohne Authentifizierung gesendet.

- Für die Anmeldung benötigt die Anweisung "TMAIL_C" ferner das zugehörige Passwort. Dies ist das Passwort, das Sie beim Einrichten Ihres Mail-Accounts angegeben haben. Er wird mit dem Parameter PassWord an die Struktur im Parameter MAIL_ADDR_PARAM übertragen.

Datentypen für die Parameter

Die folgende Tabelle zeigt die Parameter der Anweisung TMAIL_C:

Parameter	Deklaration	Datentyp	Speicherbereich	Beschreibung
REQ	Input	BOOL	E, A, M, D, L, T, C oder konstant	Steuerparameter REQUEST: Aktiviert das Senden einer E-Mail bei einer steigenden Flanke.
TO_S (Seite 721)	Input	STRING	D	Empfängeradressen STRING mit einer maximalen Länge von 180 Zeichen (Byte). Das E-Mail-Adressformat entnehmen Sie bitte dem Beispiel in der Parameterbeschreibung.
CC (Seite 721)	Input	STRING	D	CC-Empfängeradressen (optional) STRING mit einer maximalen Länge von 180 Zeichen (Byte). Gleiches E-Mail-Adressformat wie für Parameter TO_S. Wird hier eine leere Zeichenkette zugewiesen, wird die E-Mail nicht an einen CC-Empfänger gesendet.
SUBJECT	Input	STRING	D	Betreff der E-Mail STRING mit einer maximalen Länge von 180 Zeichen (Byte).
TEXT	Input	STRING	D	Text der E-Mail (optional) STRING mit einer maximalen Länge von 180 Zeichen (Byte). Wird hier eine leere Zeichenkette zugewiesen, wird die Mail ohne Text gesendet.
ATTACHMENT	Input	VARIANT	D	E-Mail-Anlage (optional) Verweis auf ein Byte/Wort/Doppelwort-Feld (ArrayOfByte, ArrayOfWord oder ArrayOfDWord) mit einer maximalen Länge von 64 Byte. Wird kein Wert zugewiesen, wird die E-Mail ohne Anhang gesendet.

Parameter	Deklaration	Datentyp	Speicherbereich	Beschreibung
ATTACHMENT_NAME	Input	VARIANT	D	Name der E-Mail-Anlage (optional) Verweis auf eine Zeichenkette mit einer maximalen Länge von 50 Zeichen (Byte), die den Dateinamen der Anlage darstellt. Wird diesem Parameter eine leere Zeichenkette zugewiesen, wird die Mail-Anlage mit dem Dateinamen "attachment.bin" gesendet.
MAIL_ADDR_PARAM (Seite 719)	Input	VARIANT	D	Verbindungsparameter und Adresse des Mailservers Die Verbindungsparameter sind mit der Struktur Tmail_v4 oder Tmail_FQDN festzulegen (siehe Parameterbeschreibung).
DONE (Seite 721)	Output	BOOL	E, A, M, D, L	Statusparameter • DONE = 0: Auftrag noch nicht gestartet oder wird noch ausgeführt. • DONE = 1: Auftrag wurde fehlerfrei ausgeführt.
BUSY (Seite 721)	Output	BOOL	E, A, M, D, L	Statusparameter • BUSY = 0: Verarbeitung von "TMAIL_C" wurde angehalten. • BUSY = 1: Die Übertragung der E-Mail ist noch nicht beendet.
ERROR (Seite 721)	Output	BOOL	E, A, M, D, L	Statusparameter • ERROR = 0: Kein Fehler aufgetreten. • ERROR = 1: Fehler während der Verarbeitung aufgetreten. STATUS enthält Informationen zum Fehlertyp.
STATUS (Seite 722)	Output	WORD	E, A, M, D, L	Statusparameter Rückgabewert oder Fehlerinformationen der Anweisung "TMAIL_C" (siehe Parameterbeschreibung).

Ausführlichere Informationen zu den gültigen Datentypen enthält das Kapitel "Übersicht der gültigen Datentypen".

Hinweis

Optionale Parameter

Die optionalen Parameter CC, TEXT und ATTACHMENT werden nur mit der E-Mail gesendet, wenn die entsprechenden Parameter eine Zeichenkette mit einer Länge > 0 enthalten.

Parameter MAIL_ADDR_PARAM

Mit dem Parameter MAIL_ADDR_PARAM legen Sie die Verbindung für das Senden der E-Mail in der Struktur Tmail_v4 oder Tmail_FQDN fest und speichern die Mailserver-Adresse und -Anmeldedaten.

Welche Struktur für den Parameter MAIL_ADDR_PARAM benutzt wird, hängt von dem Format ab, in dem der Mailserver adressiert werden soll:

- Tmail_v4: Adressierung über die IP-Adresse (IPv4).
- Tmail_FQDN: Adressierung mit "Fully Qualified Domain Name" (FQDN).

Welche Struktur verwendet werden kann, hängt von der mit dem Parameter InterfaceId angesprochenen Schnittstelle ab. Soll die Anweisung "TMAIL_C" mit der internen Schnittstelle verwendet werden, muss für Parameter MAIL_ADDR_PARAM Struktur Tmail_v4 verwendet werden.

Tabelle 11- 38 **Tmail_v4: Adressierung des Mailservers über die IP-Adresse (IPv4)**

Parameter	Datentyp	Beschreibung
Tmail_v4	Struct	
InterfaceId	LADDR	Hardwarekennung der Schnittstelle
ID	CONN_OUC	Verbindungs-ID
ConnectionType	BYTE	Verbindungstyp. Wählen Sie 16#20 als Verbindungstyp für IPv4.
ActiveEstablished	BOOL	Status-Bit. Auf "1" zu setzen, wenn die Verbindung aufgebaut ist.
CertIndex	BYTE	=0: SMTP wird verwendet (**S**imple **M**ail **T**ransfer **P**rotocol). SMTP muss verwendet werden, wenn die E-Mail über die Schnittstelle einer S7-1200 CPU gesendet wird.
WatchDogTime	TIME	Ausführungszeitüberwachung. Mit diesem Parameter kann die maximale Ausführungszeit für das Senden festgelegt werden. Hinweis: Bei langsamen Verbindungen kann der Verbindungsaufbau länger dauern (ca. eine Minute). Berücksichtigen Sie beim Einstellen von Parameter WATCH_DOG_TIME die erforderliche Zeit für den Verbindungsaufbau. Die Verbindung wird beendet, sobald die eingegebene Zeit abgelaufen ist.
MailServerAddress	IP_v4	IP-Adresse des Mailservers. IPv4 in folgendem Format: XXX.XXX.XXX.XXX (dezimal). Beispiel: 192.142.131.237.
UserName	STRING[254]	Name für die Anmeldung am Mailserver
PassWord	STRING[254]	Passwort für den Mailserver
From	EMAIL_ADDR	E-Mail-Absenderadresse, die mit den folgenden beiden STRING-Parametern festgelegt wird. Beispiel: "myname@mymailserver.com".
LocalPartPlusAtSign	STRING[64]	Lokaler Teil der Absenderadresse einschließlich @-Zeichen. Beispiel: "myname@".

Parameter		Datentyp	Beschreibung
	FullQualifiedDomainName	STRING[254]	Fully Qualified Domain Name (kurz: FQDN) des Mailservers. Beispiel: "mymailserver.com".

Tabelle 11- 39 **Tmail_FQDN: Adressierung des Mailservers über FQDN**

Parameter			Datentyp	Beschreibung
Tmail_v6			Struct	
	Tmail_FQDN		LADDR	Hardwarekennung der Schnittstelle
	ID		CONN_OUC	Verbindungs-ID
	ConnectionType		BYTE	Verbindungstyp. Wählen Sie 16#22 als Verbindungstyp für FQDN.
	ActiveEstablished		BOOL	Status-Bit. Auf "1" zu setzen, wenn die Verbindung aufgebaut ist.
	CertIndex		BYTE	=0: SMTP wird verwendet (**S**imple **M**ail **T**ransfer **P**rotocol). SMTP muss verwendet werden, wenn die E-Mail über die Schnittstelle einer S7-1200 CPU gesendet wird.
	WatchDogTime		TIME	Ausführungszeitüberwachung. Mit diesem Parameter kann die maximale Ausführungszeit für das Senden festgelegt werden. Hinweis: Bei langsamen Verbindungen kann der Verbindungsaufbau länger dauern (ca. eine Minute). Berücksichtigen Sie beim Einstellen von Parameter WATCH_DOG_TIME die erforderliche Zeit für den Verbindungsaufbau. Die Verbindung wird beendet, sobald die eingegebene Zeit abgelaufen ist.
	MailServerAddress		STRING[254]	FQDN (**F**ully **Q**ualified **D**omain **N**ame) des Mailservers. Der Mailserver wird mit dem FQDN adressiert. Beispiel: "www.mymailserver.com.".
	UserName		STRING[254]	Name für die Anmeldung am Mailserver
	PassWord		STRING[254]	Passwort für den Mailserver
	From		Struct	E-Mail-Absenderadresse, die mit den folgenden beiden STRING-Parametern festgelegt wird. Beispiel: "myname@mymailserver.com".
		LocalPartPlusAtSign	STRING[64]	Lokaler Teil der Absenderadresse einschließlich @-Zeichen. Beispiel: "myname@".
		FullQualifiedDomainName	STRING[254]	Fully Qualified Domain Name (kurz: FQDN) des Mailservers. Beispiel: "mymailserver.com".

Parameter TO_S und CC

Die Parameter TO_S und CC sind Zeichenketten mit z. B. dem folgenden Inhalt:

- <wenna@mydomain.com>, <ruby@mydomain.com>
- <admin@mydomain.com>, <judy@mydomain.com>

Bei der Eingabe der Parameter sind die folgenden Regeln zu beachten:

- Ein Leerzeichen und eine öffnende spitze Klammer "<" müssen vor jeder Adresse eingegeben werden.
- Eine schließende spitze Klammer ">" muss nach jeder Adresse eingegeben werden.
- Zwischen den Adressen in TO und CC ist ein Komma einzugeben.

Um Laufzeit und Speicherplatz nicht unnötig zu belasten, führt die Anweisung "TMAIL_C" keine Syntaxprüfung für die Parameter TO_S oder CC durch.

Parameter DONE, BUSY und ERROR

Die Ausgangsparameter DONE, BUSY und ERROR werden jeweils nur für einen Zyklus angezeigt, wenn der Status des Ausgangsparameters BUSY von "1" zu "0" wechselt.

Die folgende Tabelle zeigt die Beziehungen zwischen DONE, BUSY und ERROR. Mithilfe dieser Tabelle können Sie den aktuellen Zustand der Anweisung "TMAIL_C feststellen bzw. den Zeitpunkt, zu dem der Sendevorgang beendet ist.

DONE	BUSY	ERROR	Beschreibung
0	1	0	Auftrag wird bearbeitet.
1	0	0	Auftrag erfolgreich ausgeführt.
0	0	1	Auftrag mit Fehler beendet. Die Fehlerursache ist im Parameter STATUS (Seite 722) hinterlegt.
0	0	0	Der Anweisung "TMAIL_C" wurde kein (neuer) Auftrag zugewiesen.

Parameter STATUS

Die folgende Tabelle zeigt die Rückgabewerte von TMAIL_C am Parameter STATUS:

Rückgabewert STATUS* (W#16#...):	Bedeutung	Hinweise
0000	Die Ausführung von TMAIL_C wurde fehlerfrei beendet.	Die fehlerfreie Beendigung von TMAIL_C bedeutet nicht, dass die gesendete Mail tatsächlich ankommt. Eine fehlerhafte Eingabe der Empfängeradressen erzeugt keinen Statusfehler der Anweisung TMAIL_C. In diesem Fall ist nicht gewährleistet, dass die Mail an andere Empfänger gesendet wird, auch wenn diese richtig eingegeben wurden.
7001	TMAIL_C ist aktiv (BUSY = 1).	Erster Aufruf: Auftrag gestartet.
7002	TMAIL_C ist aktiv (BUSY = 1).	Zwischenzeitlicher Aufruf: Auftrag bereits gestartet.
8xxx	Die Ausführung von TMAIL_C wurde mit einem Fehlercode der intern aufgerufenen Kommunikationsanweisungen beendet.	Ausführlichere Informationen enthalten die Beschreibungen des Parameters STATUS für die Kommunikationsanweisungen TCON, TDISCON, TSEND und TRCV (Seite 688).
8010	Fehler beim Verbindungsaufbau	Nähere Informationen zur Auswertung enthält der Parameter SFB_STATUS des Instanzdatenbausteins. Der am Parameter SFB_STATUS angezeigte Fehlercode wird in der Beschreibung des Parameters STATUS für die Anweisung TCON (Seite 688) erläutert.
8011	Fehler beim Senden der Daten	Nähere Informationen zur Auswertung enthält der Parameter SFB_STATUS des Instanzdatenbausteins. Der am Parameter SFB_STATUS angezeigte Fehlercode wird in der Beschreibung des Parameters STATUS für die Anweisung TSEND (Seite 688) erläutert.
8012	Fehler beim Empfangen der Daten	Nähere Informationen zur Auswertung enthält der Parameter SFB_STATUS des Instanzdatenbausteins. Der am Parameter SFB_STATUS angezeigte Fehlercode wird in der Beschreibung des Parameters STATUS für die Anweisung TRCV (Seite 688) erläutert.
8013	Fehler beim Verbindungsaufbau	Nähere Informationen zur Auswertung enthält der Parameter SFB_STATUS des Instanzdatenbausteins. Der am Parameter SFB_STATUS angezeigte Fehlercode wird in der Beschreibung des Parameters STATUS für die Anweisungen TCON (Seite 688) und TDISCON (Seite 688) erläutert.

Rückgabewert STATUS* (W#16#...):	Bedeutung	Hinweise
8014	Verbindungsaufbau nicht möglich.	Es wurde möglicherweise eine falsche Mailserver-IP-Adresse (MailServerAddress (Seite 719)) oder eine zu kurze Zeit (WatchDogTime (Seite 719)) für den Verbindungsaufbau eingegeben. Es ist auch möglich, dass die CPU keine Verbindung zum Netzwerk hat oder die CPU-Konfiguration fehlerhaft ist.
8015	Falscher Datentyp für MAIL_ADDR_PARAM	Gültige Datentypen sind nur die Systemdatentypen (Strukturen) Tmail_v4 und TMail_FQDN.
8016	Falscher Datentyp für Parameter ATTACHMENT	Gültig sind nur die Datentypen ArrayOfByte, ArrayOfWord und ArrayOfDWord.
8017	Falsche Datenlänge für Parameter ATTACHMENT	Die Datenlänge muss <= 65534 Byte betragen.
82xx, 84xx, oder 85xx	Die Fehlermeldung kommt vom Mailserver und entspricht, mit Ausnahme der "8", der Fehlernummer im SMTP-Protokoll. In den folgenden Zeilen sind mögliche Fehlercodes aufgeführt.	Ausführlichere Informationen über den SMTP-Fehlercode und andere Fehlercodes im SMTP-Protokoll finden Sie im Internet oder in der Fehlerdokumentation des Mailservers. Sie können außerdem die letzte Fehlermeldung des Mailservers in Ihrem Instanz-DB in Parameter BUFFER1 anzeigen. Die zuletzt mit der Anweisung TMAIL_C gesendeten Daten finden Sie unter DATEN im Instanz-DB.
8450	Vorgang nicht ausgeführt: Mailbox nicht verfügbar/nicht erreichbar	Wiederholen Sie den Vorgang später.
8451	Vorgang abgebrochen: Lokaler Verarbeitungsfehler	Wiederholen Sie den Vorgang später.
8500	Syntaxfehler: Fehler nicht erkannt. Dies betrifft auch Fehler durch eine zu lange Befehlszeichenkette. Der Fehler kann auch auftreten, wenn der Mailserver den LOGIN-Authentifizierungsvorgang nicht unterstützt.	Prüfen Sie die Parameter von TMAIL_C. Versuchen Sie, eine E-Mail ohne Authentifizierung zu senden. Ersetzen Sie hierfür den Inhalt von Parameter UserName durch eine leere Zeichenkette. Wird kein Benutzername angegeben, wird das Authentifizierungsverfahren LOGIN nicht verwendet.
8501	Syntaxfehler: Falsche Eingabe an einem Parameter	Mögliche Ursache: Falsche Adresse in Parameter TO_S oder CC (siehe auch: Parameter TO_S und CC (Seite 721)).
8502	Befehl ist unbekannt oder nicht implementiert	Prüfen Sie Ihre Eingaben, insbesondere den Parameter FROM. Möglicherweise ist er unvollständig und Sie haben "@" oder "." vergessen. (siehe auch: Parameter TO_S und CC (Seite 721)).
8535	SMTP-Authentifizierung unvollständig	Möglicherweise wurde ein falscher Benutzername oder ein falsches Passwort eingegeben.

Rückgabe-wert STATUS* (W#16#...):	Bedeutung	Hinweise
8550	Mailserver nicht erreichbar. Sie haben keine Zugriffsrechte.	Sie haben einen falschen Benutzernamen oder ein falsches Passwort eingegeben oder der Mailserver unterstützt Ihre Anmeldung nicht. Eine andere Fehlerursache kann ein Fehler im Domainnamen nach dem "@" in Parameter TO_S oder CC sein (siehe auch: Parameter TO_S und CC (Seite 721)).
8552	Vorgang abgebrochen: Zugewiesener Speicherplatz überschritten	Wiederholen Sie den Vorgang später.
8554	Übertragung fehlgeschlagen	Wiederholen Sie den Vorgang später.
* Fehlercodes können im Programmiereditor als Ganzzahlen oder Hexadezimalwerte angezeigt werden.		

11.2.8.14 UDP

UDP ist ein im RFC 768 beschriebenes Standardprotokoll und steht für User Datagram Protocol. UDP bietet ein Verfahren, damit eine Anwendung ein Datengramm zu einer anderen Anwendung senden kann. Die Zustellung der Daten ist jedoch nicht garantiert. Dieses Protokoll hat die folgenden Merkmale:

- Schnelles Kommunikationsprotokoll, weil eng mit der Hardware verbunden
- Geeignet für kleine bis mittlere Datenmengen (bis 1472 Byte)
- UDP ist ein einfacheres Transportprotokoll als TCP, mit einer dünnen Schicht und wenig Verwaltungsaufwand
- Lässt sich sehr flexibel in vielen Fremdsystemen einsetzen
- Routing-fähig
- Leitet die Datagramme über Portnummern
- Meldungen werden nicht quittiert: Die Anwendung hat die Verantwortung für Wiederherstellung im Fehlerfall und für Sicherheit zu übernehmen
- Für die Datenverwaltung ist aufgrund der Programmierschnittstelle SEND/RECEIVE Programmieraufwand erforderlich

UDP unterstützt die Broadcast-Kommunikation. Um die Broadcast-Funktion zu nutzen, konfigurieren Sie den entsprechenden Teil der IP-Adresse in der ADDR-Konfiguration. Beispiel: Eine CPU mit der IP-Adresse 192.168.2.10 und der Subnetzmaske 255.255.255.0 hätte die Broadcast-Adresse 192.168.2.255.

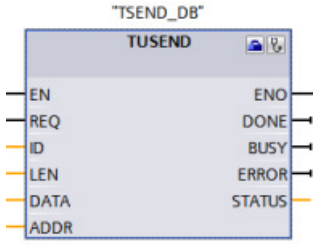
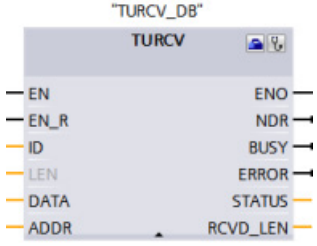
11.2.8.15 TUSEND und TURCV

Die folgenden Anweisungen steuern den UDP-Kommunikationsprozess:

- TCON baut die Kommunikation zwischen dem Client- und dem Server-PC (CPU) auf.
- TUSEND und TURCV senden und empfangen Daten.
- TDISCON trennt die Kommunikation zwischen dem Client und dem Server.

Weitere Informationen zu den Kommunikationsanweisungen TCON und TDISCON finden Sie im Abschnitt "TCP und ISO-on-TCP" unter TCON, TDISCON, TSEND und TRCV (Seite 688).

Tabelle 11- 40 Anweisungen TUSEND und TURCV

KOP/FUP	SCL	Beschreibung
	"TUSEND_DB"(req:=_bool_in_, ID:=_word_in_, len:=_udint_in_, done=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, data:=_variant_inout_);	Die Anweisung TUSEND sendet Daten über UDP an den entfernten Partner, der vom Parameter ADDR angegeben wird. Um den Auftrag zum Senden von Daten zu starten, rufen Sie die Anweisung TUSEND mit REQ = 1 auf.
	"TURCV_DB"(en_r:=_bool_in_, ID:=_word_in_, len:=_udint_in_, ndr=>_bool_out_, busy=>_bool_out_, error=>_bool_out_, status=>_word_out_, rcvd_len=>_udint_out_, data:=_variant_inout_);	Die Anweisung TURCV empfängt Daten über UDP. Der Parameter ADDR zeigt die Adresse des Senders an. Nach erfolgreicher Durchführung von TURCV enthält der Parameter ADDR die Adresse des entfernten Partners (des Senders). TURCV unterstützt den Ad-hoc-Modus nicht. Um den Auftrag zum Empfangen von Daten zu starten, rufen Sie die Anweisung TURCV mit EN_R = 1 auf.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

TCON, TDISCON, TUSEND und TURCV laufen asynchron ab, so dass die Bearbeitung eines Auftrags mehrere Anweisungsausführungen umfasst.

Tabelle 11- 41 Datentypen für die Parameter von TUSEND und TURCV

Parameter und Datentyp		Datentyp	Beschreibung
REQ (TUSEND)	IN	Bool	Startet den Sendeauftrag bei einer steigenden Flanke. Die Daten werden aus dem mit DATA und LEN bezeichneten Bereich übertragen.
EN_R (TURCV)	IN	Bool	• 0: CPU kann nicht empfangen. • 1: Aktiviert die CPU für den Empfang. Die Anweisung TURCV ist empfangsbereit und der Empfangsauftrag wird bearbeitet.
ID	IN	Word	Verweis auf die zugehörige Verbindung zwischen dem Anwenderprogramm und der Kommunikationsschicht des Betriebssystems. Die ID muss mit der zugehörigen Parameter-ID in der lokalen Verbindungsbeschreibung übereinstimmen. Wertebereich: W#16#0001 bis W#16#0FFF.
LEN	IN	UDInt	Anzahl der Bytes, die gesendet (TUSEND) oder empfangen (TURCV) werden sollen. • Standard = 0. Der Parameter DATA legt die Länge der zu sendenden oder zu empfangenden Daten fest. • Ansonsten, Wertebereich: 1 bis 1472
DONE (TUSEND)	IN	Bool	Statusparameter DONE (TUSEND): • 0: Auftrag noch nicht gestartet oder läuft noch. • 1: Auftrag fehlerfrei ausgeführt.
NDR (TURCV)	OUT	Bool	Statusparameter NDR (TURCV): • 0: Auftrag noch nicht gestartet oder läuft noch. • 1: Auftrag wurde erfolgreich ausgeführt.
BUSY	OUT	Bool	• 1: Auftrag noch nicht beendet. Es kann kein neuer Auftrag gestartet werden. • 0: Auftrag ist beendet.
ERROR	OUT	Bool	Statusparameter mit den folgenden Werten: • 0: Kein Fehler • 1: Ein Fehler ist während der Bearbeitung aufgetreten. STATUS enthält ausführliche Informationen über die Fehlerart.
STATUS	OUT	Word	Statusinformationen umfassen Fehlerinformationen. (Siehe Tabelle "Bedingungscodes für Error und Status" unten.)
RCVD_LEN	OUT	UDInt	Anzahl empfangener Bytes (TURCV)

Parameter und Datentyp		Datentyp	Beschreibung
DATA	IN_OUT	Variant	Adresse des Sendebereichs (TUSEND) oder Empfangsbereichs (TURCV): • das Prozessabbild der Eingänge • das Prozessabbild der Ausgänge • ein Merkerbit • einen Datenbaustein
ADDR	IN_OUT	Variant	Pointer auf die Adresse des Empfängers (bei TUSEND) oder des Senders (bei TURCV) (z. B. P#DB100.DBX0.0 Byte 8) Der Pointer kann auf einen beliebigen Speicherbereich verweisen. Erforderlich ist eine Struktur von 8 Byte, wie folgt: • Die ersten 4 Byte enthalten die dezentrale IP-Adresse. • Die nächsten 2 Byte geben die dezentrale Portnummer an. • Die letzten 2 Byte sind reserviert.

Der Auftragszustand wird an den Ausgangsparametern BUSY und STATUS angezeigt. STATUS entspricht dem Ausgangsparameter RET_VAL bei asynchron funktionierenden Anweisungen.

Die folgende Tabelle zeigt die Beziehungen zwischen BUSY, DONE (TUSEND), NDR (TURCV) und ERROR. Mithilfe dieser Tabelle können Sie den aktuellen Zustand der Anweisung (TUSEND oder TURCV) ermitteln bzw. den Zeitpunkt, zu dem der Sendevorgang (die Übertragung) bzw. der Empfangsvorgang beendet ist.

Tabelle 11- 42 Zustand der Parameter BUSY, DONE (TUSEND)/NDR (TURCV) und ERROR

BUSY	DONE / NDR	ERROR	Beschreibung
WAHR	Irrelevant	Irrelevant	Auftrag wird bearbeitet.
FALSCH	WAHR	FALSCH	Der Auftrag wurde erfolgreich ausgeführt.
FALSCH	FALSCH	WAHR	Auftrag mit Fehler beendet. Die Fehlerursache ist im Parameter STATUS hinterlegt.
FALSCH	FALSCH	FALSCH	Der Anweisung wurde kein (neuer) Auftrag zugewiesen.

[1] Wegen der asynchronen Funktion der Anweisungen: Bei TUSEND müssen die Daten im Sendebereich konsistent bleiben, bis einer der Parameter DONE oder ERROR den Wert WAHR annimmt. Bei TURCV sind die Daten im Empfangsbereich nur konsistent, wenn der Parameter NDR den Wert WAHR hat.

Tabelle 11- 43 Bedingungscodes von TUSEND und TURCV für ERROR und STATUS

ERROR	STATUS	Beschreibung
0	0000	• Sendeauftrag fehlerfrei ausgeführt (TUSEND). • Neue Daten wurden angenommen. Die aktuelle Länge der empfangenen Daten wird in RCVD_LEN (TURCV) gezeigt.
0	7000	• Keine laufende Auftragsbearbeitung (TUSEND) • Baustein nicht empfangsbereit (TURCV)
0	7001	• Auftragsbearbeitung starten, Daten werden gesendet (TUSEND): Während dieser Bearbeitung greift das Betriebssystem auf die Daten im Sendebereich DATA zu. • Baustein empfangsbereit, Empfangsauftrag wurde aktiviert (TURCV).
0	7002	• Folgeaufruf (REQ irrelevant), Auftrag wird bearbeitet (TUSEND): Während dieser Bearbeitung greift das Betriebssystem auf die Daten im Sendebereich DATA zu. • Folgeaufruf, Auftrag wird bearbeitet: Während der Bearbeitung schreibt die Anweisung TURCV Daten in den Empfangsbereich. Deshalb kann ein Fehler zu inkonsistenten Daten im Empfangsbereich führen.
1	8085	Der Parameter LEN ist größer als der größte zulässige Wert, hat den Wert 0 (TUSEND) oder Sie haben den Wert des Parameters LEN oder DATA seit der ersten Anweisungsausführung geändert (TURCV).
1	8086	Der Parameter ID ist nicht im zulässigen Adressbereich.
1	8088	• Der Parameter LEN ist größer als der in DATA festgelegte Speicherbereich (TUSEND) oder Empfangsbereich (TURCV). • Empfangsbereich ist zu klein (TURCV).
1	8089	Der Parameter ADDR verweist nicht auf einen Datenbaustein.
1	80A1	Kommunikationsfehler: • Die angegebene Verbindung zwischen dem Anwenderprogramm und der Kommunikationsschicht des Betriebssystems wurde noch nicht aufgebaut. • Die angegebene Verbindung zwischen dem Anwenderprogramm und der Kommunikationsschicht des Betriebssystems wird gerade beendet. Ein Sende- (TUSEND) oder Empfangsauftrag (TURCV) ist über diese Verbindung nicht möglich. • Schnittstelle wird neu initialisiert.
1	80A4	IP-Adresse des entfernten Verbindungspunkts ist ungültig, sie entspricht möglicherweise der lokalen IP-Adresse (TUSEND).
1	80B3	• Die festgelegte Protokollvariante (Parameter connection_type in der Verbindungsbeschreibung) ist kein UDP. Bitte verwenden Sie die Anweisung TSEND oder TRCV. • Parameter ADDR: Ungültige Einstellungen für die Portnummer (TUSEND)
1	80C3	• Ein Baustein mit dieser ID wird bereits in einer anderen Prioritätsklasse bearbeitet. • Interner Mangel an Ressourcen
1	80C4	Temporärer Kommunikationsfehler: • Die Verbindung zwischen dem Anwenderprogramm und der Kommunikationsschicht des Betriebssystems kann derzeit nicht hergestellt werden (TUSEND). • Die Schnittstelle empfängt neue Parameter (TUSEND). • Die angegebene Verbindung wird derzeit neu aufgebaut (TURCV).

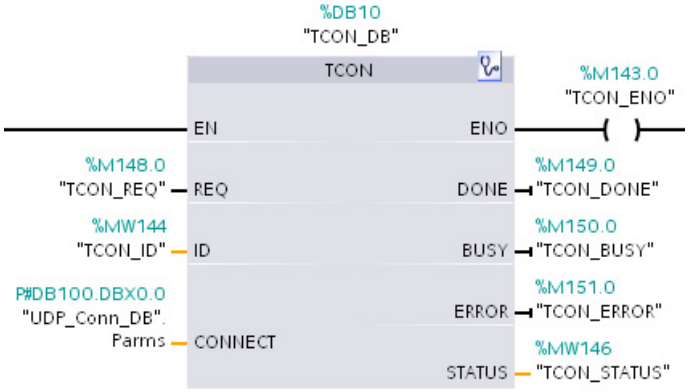
Ethernet-Verbindungsprotokolle

Jede CPU hat einen integrierten PROFINET-Anschluss, der die standardmäßige PROFINET-Kommunikation unterstützt. Die Anweisungen TUSEND und TURCV unterstützen das UDP-Ethernet-Protokoll.

Weitere Informationen finden Sie im Kapitel ""Gerätekonfiguration" unter "Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren" (Seite 638).

Betrieb

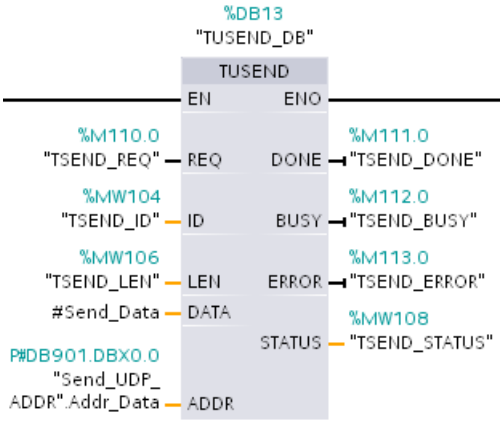
Beide Partner sind in der UDP-Kommunikation passiv. Typische Startwerte für Parameter des Datentyps "TCON_Param" werden in den folgenden Abbildungen gezeigt. Die Portnummern (LOCAL_TSAP_ID) sind im 2-Byte-Format geschrieben. Alle Ports mit Ausnahme von 161, 34962, 34963 und 34964 sind zulässig.



UDP_Conn_DB

	Name	Datentyp	Offset	Startwert	Kommentar
1	Static				
2	Parms	TCON_Param	0.0		
3	BLOCK_LENGTH	UInt	0.0	64	byte length of SDT
4	ID	CONN_OUC	2.0	1	reference to the connection
5	CONNECTION_TYPE	USInt	4.0	19	17: TCP/IP, 18: ISO on TCP
6	ACTIVE_EST	Bool	5.0	false	active/passive connection establishment
7	LOCAL_DEVICE_ID	USInt	6.0	1	1: local IE interface
8	LOCAL_TSAP_ID_LEN	USInt	7.0	2	byte length of local TSAP id/port number
9	REM_SUBNET_ID_LEN	USInt	8.0	0	byte length of remote subnet id
10	REM_STADDR_LEN	USInt	9.0	0	byte length of remote IP address
11	REM_TSAP_ID_LEN	USInt	10.0	0	byte length of remote port/TSAP id
12	NEXT_STADDR_LEN	USInt	11.0	0	byte length of next station address
13	LOCAL_TSAP_ID	Array[1..16] of Byte	12.0		TSAP id/local port number
14	LOCAL_TSAP_ID[1]	Byte		B#16#07	
15	LOCAL_TSAP_ID[2]	Byte		B#16#D0	

Die Anweisung TUSEND sendet Daten über UDP an den entfernten Partner, der vom Datentyp TADDR_Param angegeben wird. Die Anweisung TURCV empfängt Daten über UDP. Nach erfolgreicher Ausführung der Anweisung TURCV zeigt der Datentyp "TADDR_Param" Ihnen die Adresse des entfernten Partners (des Senders) an.



Send_UDP_ADDR

		Name	Datentyp	Offset	Startwert	Kommentar
1		Static				
2		Addr_Data	TADDR_Param	0.0		
3		REM_IP_ADDR	Array[1..4] of USint	0.0		remote station address
4		REM_IP_ADDR[1]	USint		0	
5		REM_IP_ADDR[2]	USint		0	
6		REM_IP_ADDR[3]	USint		0	
7		REM_IP_ADDR[4]	USint		0	
8		REM_PORT_NR	UInt	4.0	0	remote port number
9		RESERVED	Word	6.0	0	unused; has to be 0

11.2.8.16 T_CONFIG

Die Anweisung T_CONFIG ändert die IP-Konfigurationsparameter des PROFINET-Ports über das Anwenderprogramm. Dadurch ist die dauerhafte Änderung bzw. Einstellung der folgenden Funktionen möglich:

- Stationsname
- IP-Adresse
- Subnetzmaske
- Router-Adresse

Hinweis

Auf der Seite "Ethernet-Adresse" in den "Eigenschaften" der CPU können Sie über das Optionsfeld "IP-Adresse auf anderem Weg einstellen" (Seite 737) die IP-Adresse online oder über die Anweisung "T_CONFIG" ändern, nachdem Sie das Programm in die CPU geladen haben. Diese Zuweisungsart der IP-Adresse gilt nur für die CPU.

Auf der Seite "Ethernet-Adresse" in den "Eigenschaften" der CPU können Sie über das Optionsfeld "PROFINET-Gerätename auf anderem Weg einstellen" (Seite 738) den PROFINET-Gerätenamen online oder über die Anweisung "T_CONFIG" ändern, nachdem Sie das Programm in die CPU geladen haben. Diese Zuweisungsart des PROFINET-Gerätenamens gilt nur für die CPU.

⚠ WARNUNG

Das Ändern des IP-Konfigurationsparameters mit T_CONFIG verursacht einen CPU-Neustart.

Nachdem Sie mit T_CONFIG einen Parameter der IP-Konfiguration geändert haben, startet die CPU neu. Die CPU geht in den Betriebszustand STOP, führt einen Warmstart durch und kehrt in den Betriebszustand RUN zurück.

Verwenden Sie die Anweisung T_CONFIG nicht in einer Produktionsumgebung. Steuerungen können bei unsicheren Betriebszuständen ausfallen und dadurch unkontrolliertes Betriebsverhalten der gesteuerten Geräte verursachen. Derartiges unerwartetes Betriebsverhalten des Automatisierungssystems kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Stellen Sie sicher, dass Ihr Prozess in einen sicheren Zustand übergeht, wenn die CPU als Ergebnis der Ausführung der Anweisung T_CONFIG einen Warmstart durchführt. Bei einem Warmstart wird kein Urlöschen durchgeführt. Bei einem Warmstart werden alle nicht remanenten System- und Anwenderdaten initialisiert und die Werte aller remanenten Anwenderdaten gespeichert.

Hinweis

Führen Sie niemals mehrere Anweisungen T_CONFIG gleichzeitig aus.

Tabelle 11- 44 Anweisung T_CONFIG

KOP/FUP	SCL	Beschreibung

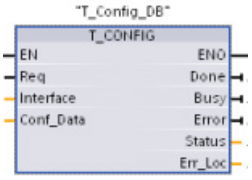
	`"T_CONFIG_DB"(`
	`req:=_bool_in_,`
	`interface:=_word_in_,`
	`conf_Data:=_variant_in_,`
	`done=>_bool_out_,`
	`busy=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_dword_out_,`
	`err_loc=>_word_out_);`
		Die Anweisung T_CONFIG ändert die Parameter der IP-Konfiguration über das Anwenderprogramm.
		T_CONFIG arbeitet asynchron. Die Ausführung erstreckt sich über mehrere Aufrufe.

Tabelle 11- 45 T_CONFIG-Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	Input	Bool	Startet die Anweisung an der steigenden Flanke.
INTERFACE	Input	HW_Interface	ID der Netzwerkschnittstelle
CONF_DATA	Input	Variant	Verweis auf die Struktur der Konfigurationsdaten; CONF_DATA ist über einen Systemdatentyp (SDT) festgelegt.
DONE	Output	Bool	• 0: Auftrag wurde noch nicht gestartet oder läuft noch. • 1: Auftrag wurde fehlerfrei ausgeführt.
BUSY	Output	Bool	• 0: Auftrag ist beendet. • 1: Der Auftrag ist noch nicht beendet. Es kann kein neuer Auftrag gestartet werden.
ERROR	Output	Bool	Statusparameter mit den folgenden Werten: • 0: Kein Fehler • 1: Ein Fehler ist während der Bearbeitung aufgetreten. STATUS enthält ausführliche Informationen über die Fehlerart.
STATUS	Output	DWord	Statusinformationen umfassen Fehlerinformationen. (Siehe Tabelle "Bedingungscodes für Error und Status" unten.)
ERR_LOC	Output	DWord	Fehlerstelle (Feld ID und Unterfeld ID des Fehlerparameters)

Die IP-Konfigurationsdaten werden im Datenbaustein CONF_DATA abgelegt, zusammen mit einem Variant-Pointer auf den oben angegebenen Parameter CONF_DATA. Die erfolgreiche Ausführung der Anweisung T_CONFIG endet mit der Übergabe der IP-Konfigurationsdaten an die Netzwerkschnittstelle. Fehler werden dem Ausgangsparameter STATUS zugewiesen.

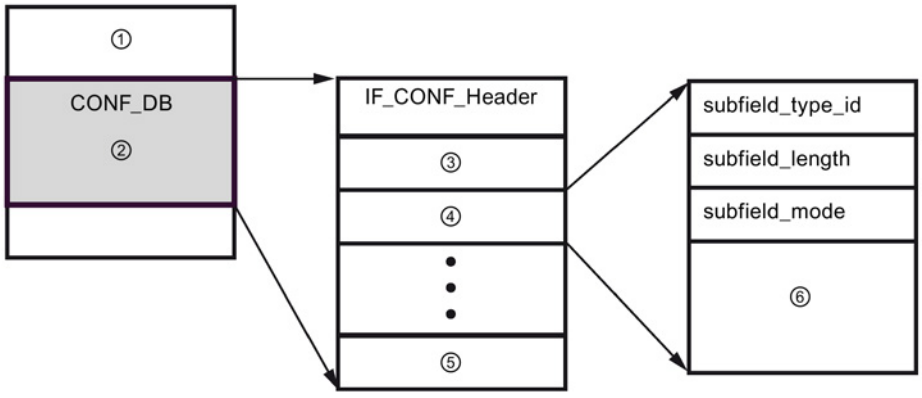
Tabelle 11- 46 Bedingungscodes für ERROR und STATUS

ERROR	STATUS (DW#16#...)	Beschreibung
0	00000000	Kein Fehler Hinweis: Bei erfolgreicher Ausführung der Anweisung wird der Status "Kein Fehler" möglicherweise nicht ausgegeben.
0	00700000	Der Auftrag ist noch nicht beendet (BUSY = 1).
0	00700100	Start der Auftragsausführung
0	00700200	Zwischenzeitlicher Aufruf (REQ irrelevant)
1	C08xyy00	Allgemeiner Fehler
1	C0808000	Die Parameter LADDR zur Identifikation der Schnittstelle sind ungültig.
1	C0808100	Den Parametern LADDR zur Identifikation der Schnittstelle wurde eine nicht unterstützte Hardware-Schnittstelle zugewiesen.
1	C0808200	Parameterfehler CONF_DATA: Datentyp des Variant-Pointers entspricht nicht dem Datentyp Byte.
1	C0808300	Parameterfehler CONF_DATA: Der Bereichs-Pointer ist nicht im DB des Variant-Pointers.
1	C0808400	Parameterfehler CONF_DATA: Der Variant-Pointer hat die falsche Länge.
1	C0808600	Reserviert
1	C0808700	Inkonsistenz in der Länge des CONF_DATA-Datenbausteins zur IP-Konfiguration
1	C0808800	Die Parameter der field_type_id des CONF_DATA-Datenbausteins sind ungültig. (Nur field_type_id = 0 ist zulässig.)
1	C0808900	Die Parameter der field_type_id des CONF_DATA-Datenbausteins sind ungültig oder wurden mehrmals verwendet.
1	C0808A00	Länge LEN der Parameter der IP-Konfiguration oder subfield_cnt fehlerhaft
1	C0808B00	Der Parameter ID der IP-Konfiguration ist ungültig oder wird nicht unterstützt.
1	C0808C00	Der Teilbaustein der IP-Konfiguration ist nicht korrekt abgelegt (Teilbaustein falsch, falsche Reihenfolge oder mehrmals verwendet).
1	C0808D00	Die Länge einer Anweisung LEN eines Teilbausteins ist ungültig.
1	C0808E00	Der Wert des Parameters im Teilbausteinmodus ist ungültig.
1	C0808F00	Teilbausteinkonflikt zwischen der IP-Konfiguration und einem vorherigen Teilbaustein.
1	C0809000	Die Parameter des Unterfelds sind schreibgeschützt (Beispiel: Parameter sind konfiguriert oder PNIO-Modus ist aktiv).
1	C0809100	Reserviert
1	C0809400	Ein Parameter in der IP-Konfiguration des Teilbausteins wurde nicht definiert oder kann nicht verwendet werden.
1	C0809500	Es besteht eine Inkonsistenz zwischen einem Parameter der IP-Konfiguration des Teilbausteins und anderen Parametern.
1	C080C200	Die Anweisung kann nicht ausgeführt werden. Dieser Fehler kann auftreten, wenn beispielsweise die Kommunikation mit der Schnittstelle verloren gegangen ist.

ERROR	STATUS (DW#16#...)	Beschreibung
1	C080C300	Es sind nicht genügend Ressourcen vorhanden. Dieser Fehler kann auftreten, wenn beispielsweise die Anweisung mehrmals mit verschiedenen Parametern aufgerufen wird.
1	C080C400	Kommunikation gestört. Der Fehler kann temporär auftreten und erfordert eine Wiederholung des Anwenderprogramms.
1	C080D200	Die Ausführung der Anweisung wird von der PROFINET-Schnittstelle nicht unterstützt.

Datenbaustein CONF_DATA

Die folgende Abbildung zeigt, wie die zu übertragenden Konfigurationsdaten im Konfigurations-DB gespeichert werden.



① Konfigurations-DB
② Konfigurationsdaten
③ Unterfeld 1
④ Unterfeld 2
⑤ Unterfeld *n*
⑥ Unterfeldspezifische Parameter

Die Konfigurationsdaten von CONF_DB bestehen aus einem Feld mit einem Header (IF_CONF_Header) und mehreren Unterfeldern. IF_CONF_Header verfügt über die folgenden Elemente:

- field_type_id (Datentyp UInt): Null
- field_id (Datentyp UInt): Nullabgleich
- subfield_cnt (Datentyp UInt): Anzahl der Unterfelder

Jedes Unterfeld besteht aus einem Header (subfield_type_id, subfield_length, subfield_mode) und den unterfeldspezifischen Parametern. Jedes Unterfeld muss eine gerade Anzahl von Bytes umfassen. Der Parameter subfield_mode unterstützt den Wert 1.

Hinweis

Derzeit ist nur ein Feld (IF_CONF_Header) zulässig. Dessen Parameter field_type_id und field_id müssen den Wert 0 haben. Andere Felder mit anderen Werten für field_type_id und field_id unterliegen künftigen Erweiterungen.

Im Feld IF_CONF_Header sind derzeit nur zwei Unterfelder "addr" (IP-Adresse) und "nos" (Name of station) zulässig.

Tabelle 11- 47 Unterstützte Unterfelder

subfield_type_id	Datentyp	Bedeutung
30	IF_CONF_V4	IP-Parameter: IP-Adresse, Subnetzmaske, Routeradresse
40	IF_CONF_NOS	PROFINET IO-Gerätename (Name of station)

Tabelle 11- 48 Elemente des Datentyps IF_CONF_V4

Name	Datentyp	Startwert	Beschreibung
Id	UInt	30	subfield_type_id
len	UInt	18	subfield_length
mode	UInt	1	subfield_mode (1: dauerhaft)
InterfaceAddress	IP_V4	-	Schnittstellenadresse
ADDR	Array [1..4] of Byte		
ADDR[1]	Byte	b#16#C8	High Byte der IP-Adresse: 200
ADDR[2]	Byte	b#16#0C	High Byte der IP-Adresse: 12
ADDR[3]	Byte	b#16#01	Low Byte der IP-Adresse: 1
ADDR[4]	Byte	b#16#90	Low Byte der IP-Adresse: 144
SubnetMask	IP_V4	-	Subnetzmaske
ADDR	Array [1..4] of Byte		
ADDR[1]	Byte	b#16#FF	High Byte der Subnetzmaske: 255
ADDR[2]	Byte	b#16#FF	High Byte der Subnetzmaske: 255
ADDR[3]	Byte	b#16#FF	Low Byte der Subnetzmaske: 255
ADDR[4]	Byte	b#16#00	Low Byte der Subnetzmaske: 0
DefaultRouter	IP_V4	-	Standard-Router
ADDR	Array [1..4] of Byte		
ADDR[1]	Byte	b#16#C8	High Byte des Routers: 200
ADDR[2]	Byte	b#16#0C	High Byte des Routers: 12
ADDR[3]	Byte	b#16#01	Low Byte des Routers: 1
ADDR[4]	Byte	b#16#01	Low Byte des Routers: 1

Tabelle 11- 49 Elemente des Datentyps IF_CONF_NOS

Name	Datentyp	Startwert	Beschreibung
id	UInt	40	subfield_type_id
len	UInt	246	subfield_length
mode	UInt	1	subfield_mode (1: dauerhaft)
Nos (Name of station)	Array[1..240] of Byte	0	Stationsname: Sie müssen das ARRAY ab dem ersten Byte belegen. Wenn das ARRAY länger als der zuzuweisende Stationsname ist, müssen Sie nach dem eigentlichen Stationsnamen ein Null-Byte eingeben (in Übereinstimmung mit IEC 61158-6-10). Andernfalls wird nos zurückgewiesen und die Anweisung "T_CONFIG und TC_CONFIG (Seite 731)" gibt den Fehlercode DW#16#C0809400 in STATUS ein. Wenn Sie das erste Byte mit 0 belegen, wird der Stationsname gelöscht.

Der Stationsname unterliegt den folgenden Einschränkungen:

- Eine Namenskomponente innerhalb des Stationsnamens, d.h. eine Zeichenfolge zwischen zwei Punkten, darf nicht länger sein als 63 Zeichen.
- Sonderzeichen wie Umlaute, Klammern, Unterstriche, Schrägstriche, Leerzeichen usw. sind nicht zulässig. Das einzig zulässig Sonderzeichen ist der Bindestrich.
- Der Stationsname darf nicht mit dem Zeichen "-" beginnen oder enden.
- Der Stationsname darf nicht mit einer Zahl beginnen.
- Das Format n.n.n.n (n = 0 ... 999) ist für den Stationsnamen nicht zulässig.
- Der Stationsname darf nicht mit der Zeichenfolge "Port-xyz" oder "Port-xyz-abcde" (a, b, c, d, e, x, y, z = 0 ... 9) beginnen.

Hinweis

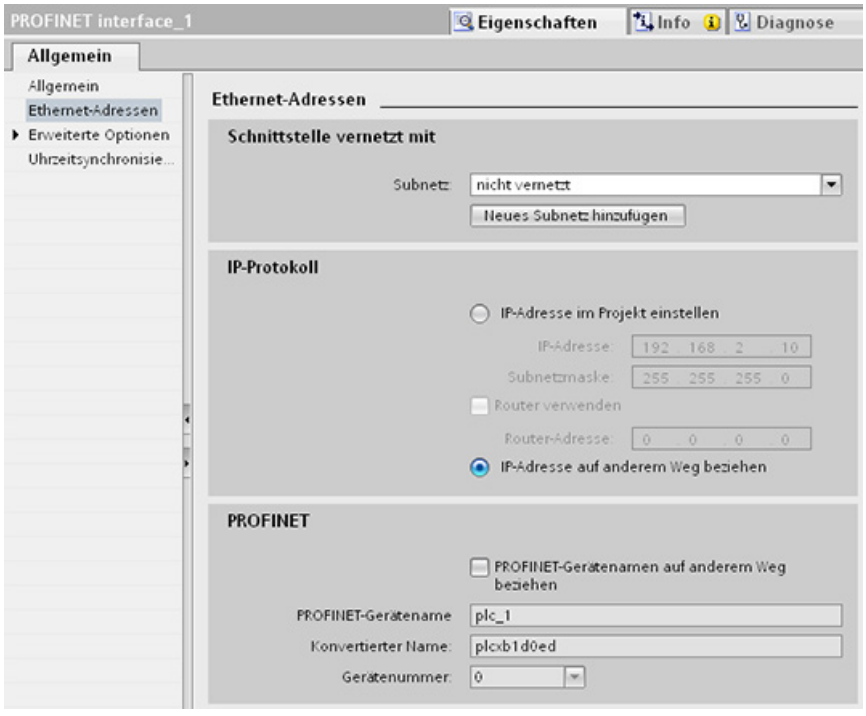
Sie können auch ein ARRAY "nos" erstellen, das kürzer ist als 240 Byte. Die Mindestlänge ist jedoch 2 Byte. Dann müssen Sie die Variable "len" (Länge des Unterfelds) entsprechend anpassen.

Beispiel: IP-Parameter mit Anweisung T_CONFIG ändern

Im folgenden Beispiel werden im Unterfeld "addr" die Parameter "InterfaceAddress" (IP-Adresse), "SubnetMask" und "DefaultRouter" (IP-Router) geändert. Auf der Seite "Ethernet-Adresse" in den "Eigenschaften" der CPU muss das Optionsfeld "IP-Adresse auf anderem Weg einstellen" aktiviert sein, um die IP-Adresse über die Anweisung "T_CONFIG" ändern zu können, nachdem Sie das Programm in die CPU geladen haben.

CONF_DATA_1

	Name	Datentyp	Startwert
1	Static		
2	Conf_data	Struct	
3	header	IF_CONF_Header	
4	FieldType	UInt	0
5	FieldId	UInt	0
6	SubfieldCount	UInt	1
7	addr	IF_CONF_v4	
8	Id	UInt	30
9	Length	UInt	18
10	Mode	UInt	1
11	InterfaceAddress	IP_V4	
12	ADDR	array [1..4] of Byte	
13	ADDR[1]	Byte	192
14	ADDR[2]	Byte	168
15	ADDR[3]	Byte	2
16	ADDR[4]	Byte	30
17	SubnetMask	IP_V4	
18	ADDR	array [1..4] of Byte	
19	ADDR[1]	Byte	255
20	ADDR[2]	Byte	255
21	ADDR[3]	Byte	255
22	ADDR[4]	Byte	0
23	DefaultRouter	IP_V4	
24	ADDR	array [1..4] of Byte	
25	ADDR[1]	Byte	192
26	ADDR[2]	Byte	168
27	ADDR[3]	Byte	2
28	ADDR[4]	Byte	1

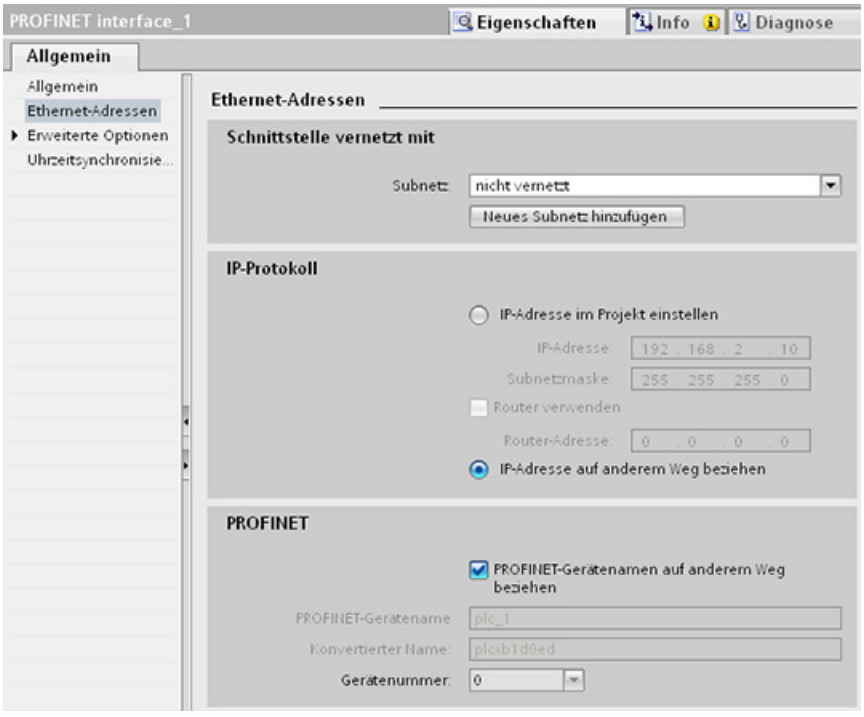


Beispiel: IP-Parameter und PROFINET IO-Gerätenamen mit Anweisung T_CONFIG ändern

Im folgenden Beispiel werden beide Unterfelder "addr" und "nos" (Name of station) geändert. Auf der Seite "Ethernet-Adresse" in den "Eigenschaften" der CPU muss das Optionsfeld "PROFINET-Gerätename auf anderem Weg einstellen" aktiviert sein, um den PROFINET-Gerätenamen über die Anweisung "T_CONFIG" ändern zu können, nachdem Sie das Programm in die CPU geladen haben.

CONF_DATA_2

	Name	Datentyp	Startwert
1	Static		
2	Conf_data	Struct	
3	header	IF_CONF_Header	
4	FieldType	UInt	0
5	FieldId	UInt	0
6	SubfieldCount	UInt	2
7	addr	IF_CONF_v4	
8	Id	UInt	30
9	Length	UInt	18
10	Mode	UInt	1
11	InterfaceAddress	IP_V4	
12	ADDR	array [1..4] of Byte	
13	SubnetMask	IP_V4	
14	ADDR	array [1..4] of Byte	
15	DefaultRouter	IP_V4	
16	ADDR	array [1..4] of Byte	
17	nos	IF_CONF_NOS	
18	Id	UInt	40
19	Length	UInt	246
20	Mode	UInt	1
21	NOS	array [1..240] of Byte	



11.2.8.17 Gemeinsame Parameter für Anweisungen

Eingangsparameter REQ

Viele der Anweisungen für die offene Benutzerkommunikation haben einen Eingang REQ, der die Anweisung bei einer steigenden Flanke (0 nach 1) initiiert. Der Eingang REQ muss während der Ausführung einer Anweisung 1 (WAHR) sein, doch der Eingang REQ kann so lange wie gewünscht WAHR bleiben. Die Anweisung stößt erst dann eine andere Anweisung an, wenn sie mit Eingang REQ = FALSCH ausgeführt wird, so dass die Anweisung den Zustand des Eingangs REQ zurücksetzen kann. Dies ist erforderlich, damit die Anweisung die steigende Flanke zum Starten der nächsten Anweisung erkennen kann.

Wenn Sie eine dieser Anweisungen in Ihr Programm einfügen, werden Sie von STEP 7 aufgefordert, den Instanz-DB anzugeben. Verwenden Sie für jeden Anweisungsaufruf einen eindeutigen DB. Dadurch wird sichergestellt, dass jede Anweisung die Eingänge wie REQ ordnungsgemäß verarbeitet.

Eingangsparameter ID

Dies ist ein Verweis auf die "Lokale ID (hex)" in der "Netzsicht" unter "Geräte & Netze" in STEP 7 und es ist die ID des Netzwerks, das Sie für diesen Kommunikationsbaustein verwenden möchten. Die ID muss mit der zugehörigen Parameter-ID in der lokalen Verbindungsbeschreibung übereinstimmen.

Ausgangsparameter DONE, NDR, ERROR und STATUS

Diese Anweisungen haben Ausgänge, die den Ausführungszustand anzeigen:

Tabelle 11- 50 Ausgabeparameter der Anweisungen für die offene Benutzerkommunikation

Parameter	Datentyp	Standard	Beschreibung
DONE	Bool	FALSCH	Wird eine Ausführung lang auf WAHR gesetzt, um anzuzeigen, dass die letzte Anforderung mit Fehler abgeschlossen wurde; andernfalls FALSCH.
NDR	Bool	FALSCH	Wird eine Ausführung lang auf WAHR gesetzt, um anzuzeigen, dass die angeforderte Aktion fehlerfrei abgeschlossen und neue Daten empfangen wurden; andernfalls FALSCH.
BUSY	Bool	FALSCH	Wird, wenn aktiv, auf WAHR gesetzt, um Folgendes anzuzeigen: • Der Auftrag ist noch nicht beendet. • Es kann kein neuer Auftrag gestartet werden. Wird auf FALSCH gesetzt, wenn der Auftrag beendet ist.

Parameter	Datentyp	Standard	Beschreibung
ERROR	Bool	FALSCH	Wird eine Ausführung lang auf WAHR gesetzt, um anzuzeigen, dass die letzte Anforderung mit Fehlern abgeschlossen wurde, der entsprechende Fehlercode befindet sich in STATUS; andernfalls FALSCH.
STATUS	Word	0	Ergebniszustand: • Wird das Bit DONE oder NDR gesetzt, wird STATUS auf 0 oder auf einen Informationscode gesetzt. • Wird das Bit ERROR gesetzt, wird STATUS auf einen Fehlercode gesetzt. • Wird keines der oben aufgeführten Bits gesetzt, kann die Anweisung Statusergebnisse zurückgeben, die den aktuellen Zustand der Funktion beschreiben. Der Wert in STATUS wird während der Ausführung der Funktion gehalten.

Hinweis

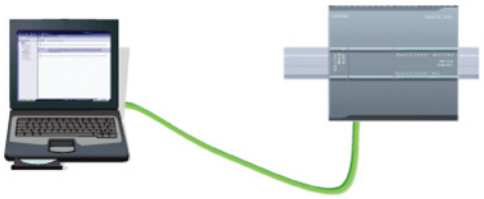
Beachten Sie, dass DONE, NDR und ERROR nur eine Ausführung lang gesetzt sind.

Eingeschränkte TSAPs und Portnummern für passive ISO- und TCP-Kommunikation

Wenn Sie mit der Operation TCON eine passive Kommunikationsverbindung einrichten und aufbauen, dürfen die folgenden Portadressen nicht verwendet werden:

- ISO TSAP (passiv):
  - 01.00, 01.01, 02.00, 02.01, 03.00, 03.01
  - 10.00, 10.01, 11.00, 11.01, ... BF0,00, BF0,01
- TCP-Port (passiv): 5001, 102, 123, 20, 21, 25, 34962, 34963, 34964, 80
- UDP-Port (passiv): 161, 34962, 34963, 34964

11.2.9 Kommunikation mit einem Programmiergerät



Eine CPU kann mit einem Programmiergerät STEP 7 in einem Netzwerk kommunizieren.

Berücksichtigen Sie beim Einrichten der Kommunikation zwischen einer CPU und einem Programmiergerät Folgendes:

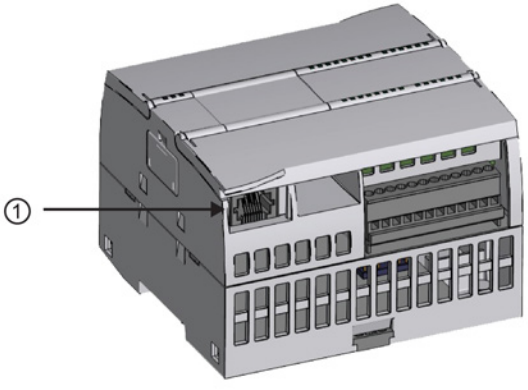
- Konfiguration/Setup: Eine Hardwarekonfiguration ist erforderlich.
- Für die Kommunikation zwischen zwei Teilnehmern ist kein Ethernet-Switch erforderlich; erst wenn mehr als zwei Geräte in einem Netzwerk vorhanden sind, wird ein Ethernet-Switch benötigt.

11.2.9.1 Hardware-Kommunikationsverbindung herstellen

Die PROFINET-Schnittstelle stellt die physische Verbindung zwischen einem Programmiergerät und einer CPU her. Da in der CPU die Auto-Cross-Over-Funktionalität implementiert ist, kann für die Schnittstelle ein Standard- oder Crossover-Ethernet-Kabel verwendet werden. Ein Ethernet-Switch wird für die direkte Verbindung des Programmiergeräts mit der CPU nicht benötigt.

Zum Herstellen der Hardwareverbindung zwischen einem Programmiergerät und einer CPU gehen Sie folgendermaßen vor:

1. Installieren Sie die CPU (Seite 60).
2. Schließen Sie das Ethernet-Kabel wie unten gezeigt am PROFINET-Anschluss an.
3. Schließen Sie das Ethernet-Kabel an das Programmiergerät an.



① PROFINET-Anschluss

Für die PROFINET-Verbindung ist eine optionale Zugentlastung verfügbar. Bestellinformationen finden Sie unter Ersatzteile und sonstige Hardware (Seite 60).

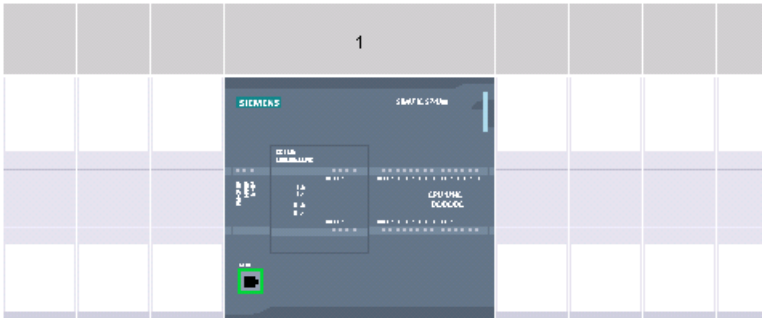
Siehe auch

Ersatzteile und sonstige Hardware (Seite 1345)

11.2.9.2 Konfigurieren der Geräte

Wenn Sie bereits ein Projekt mit einer CPU angelegt haben, öffnen Sie Ihr Projekt in STEP 7.

Sonst legen Sie ein Projekt an und fügen eine CPU (Seite 152) in den Baugruppenträger ein. Im folgenden Projekt sehen Sie eine CPU in der "Gerätesicht".



11.2.9.3 IP-Adressen zuweisen

IP-Adressen zuweisen

In einem PROFINET-Netzwerk benötigt jedes Gerät außerdem eine IP-Adresse (Internet-Protocol-Adresse). Mit Hilfe dieser Adresse kann ein Gerät Daten über ein komplexeres Netzwerk liefern:

- Verwenden Sie Programmiergeräte oder andere Netzwerkgeräte mit integrierter Adapterkarte, die an Ihr Werks-LAN angeschlossen ist, oder mit Ethernet/USB-Adapterkarte, die an ein getrenntes Netzwerk angeschlossen ist, so müssen Sie diesen Geräten IP-Adressen zuweisen. Weitere Informationen finden Sie unter "IP-Adressen zu Programmier- und Netzwerkgeräten zuweisen" (Seite 642).
- Sie können einer CPU oder einem Netzwerkgerät auch online eine IP-Adresse zuweisen. Dies ist besonders bei der ersten Gerätekonfiguration nützlich. Weitere Informationen finden Sie unter "IP-Adresse zu einer CPU online zuweisen (Seite 642)".
- Nachdem Sie Ihre CPU oder Ihr Netzwerkgerät in Ihrem Projekt konfiguriert haben, können Sie die Parameter für die PROFINET-Schnittstelle konfigurieren, um die IP-Adresse aufzunehmen. Weitere Informationen finden Sie unter "IP-Adresse für eine CPU in Ihrem Projekt konfigurieren" (Seite 645).

11.2.9.4 Testen Ihres PROFINET-Netzwerks

Nach Abschluss der Konfiguration müssen Sie Ihr Projekt in die CPU laden. Alle IP-Adressen werden beim Laden des Projekts konfiguriert.

Die Funktion "Laden in Gerät" der CPU und der zugehörigen Dialog "Erweitertes Laden" kann alle erreichbaren Netzwerkgeräte anzeigen und zusätzlich angeben, ob allen Geräten eindeutige IP-Adressen zugeordnet wurden. Ausführliche Informationen finden Sie unter "Testen des PROFINET-Netzwerks" (Seite 650).

11.2.10 Kommunikation HMI/PLC

Die CPU unterstützt PROFINET-Kommunikationsverbindungen mit HMIs (Seite 31). Zum Einrichten der Kommunikation zwischen CPUs und HMIs müssen die folgenden Bedingungen erfüllt sein:

Konfiguration/Setup:

- Der PROFINET-Port der CPU muss für die Verbindung mit der HMI konfiguriert sein.
- Die HMI muss eingerichtet und konfiguriert sein.
- Die HMI-Konfigurationsdaten sind Teil des CPU-Projekts und können im Projekt konfiguriert und geladen werden.
- Für die Kommunikation zwischen zwei Teilnehmern ist kein Ethernet-Switch erforderlich; erst wenn mehr als zwei Geräte in einem Netzwerk vorhanden sind, wird ein Ethernet-Switch benötigt.

Hinweis

Der auf dem Baugruppenträger montierte Ethernet-Switch CSM1277 mit 4 Ports dient zum Anschließen Ihrer CPUs und HMI-Geräte. Der PROFINET-Anschluss an der CPU enthält keinen Ethernet-Switch.

Unterstützte Funktionen:

- Die HMI kann Daten der CPU lesen/schreiben.
- Meldungen können anhand von Informationen der CPU ausgelöst werden.
- Systemdiagnose

Tabelle 11- 51 Erforderliche Schritte zum Konfigurieren der Kommunikation zwischen einer HMI und einer CPU

Schritt	Aufgabenstellung
1	Hardware-Kommunikationsverbindung herstellen Eine PROFINET-Schnittstelle stellt die physische Verbindung zwischen einer HMI und einer CPU her. Da in der CPU die Auto-Cross-Over-Funktionalität implementiert ist, kann für die Schnittstelle ein Standard- oder Crossover-Ethernet-Kabel verwendet werden. Ein Ethernet-Switch wird für die direkte Verbindung der HMI mit der CPU nicht benötigt. Weitere Informationen finden Sie unter "Kommunikation mit einem Programmiergerät: Hardware-Kommunikationsverbindung herstellen (Seite 741)".
2	Geräte konfigurieren Weitere Informationen finden Sie unter "Kommunikation mit einem Programmiergerät: Geräte konfigurieren (Seite 742)".
3	Konfigurieren der logischen Netzwerkverbindungen zwischen einer HMI und einer CPU Weitere Informationen finden Sie unter "Kommunikation zwischen HMI und PLC: Konfigurieren der logischen Netzwerkverbindungen zwischen zwei Geräten" (Seite 744).
4	IP-Adresse in Ihrem Projekt konfigurieren Gehen Sie auf dieselbe Weise vor. Sie müssen jedoch die IP-Adressen für die HMI und die CPU konfigurieren. Weitere Informationen finden Sie unter "Gerätekonfiguration: IP-Adresse für eine CPU in Ihrem Projekt konfigurieren" (Seite 646).
5	Testen des PROFINET-Netzwerks Sie müssen die Konfiguration für jede CPU und jedes HMI-Gerät laden. Ausführliche Informationen finden Sie unter "Gerätekonfiguration: Testen des PROFINET-Netzwerks" (Seite 650).

11.2.10.1 Konfigurieren der logischen Netzwerkverbindungen zwischen zwei Geräten

Nachdem Sie den Baugruppenträger mit der CPU konfiguriert haben, können Sie Ihre Netzwerkverbindungen einrichten.

Im Portal "Geräte & Netze" können Sie in der "Netzsicht" die Geräte in Ihrem Projekt vernetzen. Klicken Sie zunächst auf das Register "Verbindungen" und wählen Sie dann über die Klappliste rechts den Verbindungstyp aus (z. B. eine ISO-on-TCP-Verbindung).

Um eine PROFINET-Verbindung zu erstellen, klicken Sie auf das grüne Feld (PROFINET) auf dem ersten Gerät und ziehen eine Linie zum PROFINET-Feld auf dem zweiten Gerät. Lassen Sie die Maustaste los. Damit ist Ihre PROFINET-Verbindung hergestellt.

Ausführliche Informationen finden Sie unter "Gerätekonfiguration: Erstellen einer Netzwerkverbindung" (Seite 637).

11.2.11 Kommunikation PLC/PLC

Eine CPU kann mit den Anweisungen TSEND_C und TRCV_C mit einer anderen CPU im Netzwerk kommunizieren.

Berücksichtigen Sie beim Einrichten der Kommunikation zwischen zwei CPUs Folgendes:

- Konfiguration/Setup: Eine Hardwarekonfiguration ist erforderlich.
- Unterstützte Funktionen: Lesen/Schreiben von Daten in eine vernetzte CPU
- Für die Kommunikation zwischen zwei Teilnehmern ist kein Ethernet-Switch erforderlich; erst wenn mehr als zwei Geräte in einem Netzwerk vorhanden sind, wird ein Ethernet-Switch benötigt.

Tabelle 11- 52 Erforderliche Schritte zum Konfigurieren der Kommunikation zwischen zwei CPUs

Schritt	Aufgabenstellung
1	Hardware-Kommunikationsverbindung herstellen Eine PROFINET-Schnittstelle stellt die physische Verbindung zwischen zwei CPUs her. Da in der CPU die Auto-Cross-Over-Funktionalität implementiert ist, kann für die Schnittstelle ein Standard- oder Crossover-Ethernet-Kabel verwendet werden. Ein Ethernet-Switch wird für die Verbindung der zwei CPUs nicht benötigt. Weitere Informationen finden Sie unter "Kommunikation mit einem Programmiergerät: Hardware-Kommunikationsverbindung herstellen (Seite 741)".
2	Geräte konfigurieren Sie müssen zwei CPUs in Ihrem Projekt konfigurieren. Weitere Informationen finden Sie unter "Kommunikation mit einem Programmiergerät: Geräte konfigurieren (Seite 742)".
3	Konfigurieren der logischen Netzwerkverbindungen zwischen zwei CPUs Weitere Informationen finden Sie unter "Kommunikation zwischen PLC und PLC: Konfigurieren der logischen Netzwerkverbindungen zwischen zwei Geräten" (Seite 746).
4	IP-Adresse in Ihrem Projekt konfigurieren Gehen Sie auf dieselbe Weise vor. Sie müssen jedoch die IP-Adressen für zwei CPUs konfigurieren (z.B. PLC_1 und PLC_2). Weitere Informationen finden Sie unter "Gerätekonfiguration: IP-Adresse für eine CPU in Ihrem Projekt konfigurieren" (Seite 646).
5	Sende- und Empfangsparameter konfigurieren Sie müssen die Anweisungen TSEND_C und TRCV_C in beiden CPUs konfigurieren, um die Kommunikation zwischen ihnen zu ermöglichen. Weitere Informationen finden Sie unter "Einrichten der Kommunikation zwischen zwei CPUs: Sende- und Empfangsparameter konfigurieren (Seite 746)".
6	Testen des PROFINET-Netzwerks Sie müssen die Konfiguration für jede CPU laden. Ausführliche Informationen finden Sie unter "Gerätekonfiguration: Testen des PROFINET-Netzwerks" (Seite 650).

11.2.11.1 Konfigurieren der logischen Netzwerkverbindungen zwischen zwei Geräten

Nachdem Sie den Baugruppenträger mit der CPU konfiguriert haben, können Sie Ihre Netzwerkverbindungen einrichten.

Im Portal "Geräte & Netze" können Sie in der "Netzsicht" die Geräte in Ihrem Projekt vernetzen. Klicken Sie zunächst auf das Register "Verbindungen" und wählen Sie dann über die Klappliste rechts den Verbindungstyp aus (z. B. eine ISO-on-TCP-Verbindung).

Um eine PROFINET-Verbindung zu erstellen, klicken Sie auf das grüne Feld (PROFINET) auf dem ersten Gerät und ziehen eine Linie zum PROFINET-Feld auf dem zweiten Gerät. Lassen Sie die Maustaste los. Damit ist Ihre PROFINET-Verbindung hergestellt.

Ausführliche Informationen finden Sie unter "Gerätekonfiguration: Erstellen einer Netzwerkverbindung" (Seite 637).

11.2.11.2 Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren

Allgemeine Parameter konfigurieren

Sie legen die Kommunikationsparameter im Dialog "Eigenschaften" der Kommunikationsanweisung fest. Dieser Dialog wird am unteren Ende der Seite angezeigt, wenn Sie einen Teil der Anweisung selektiert haben.

Weitere Informationen finden Sie unter "Gerätekonfiguration: Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren (Seite 638)".

Die zu verwendenden TSAPs oder Ports definieren Sie im Dialog "Verbindungsparameter" unter "Adressdetails". Der TSAP oder Port einer Verbindung in der CPU wird im Feld "Lokaler TSAP" eingegeben. Der TSAP oder Port für die Verbindung in der Partner-CPU wird im Feld "Partner-TSAP" eingegeben.

11.2.11.3 Sende- und Empfangsparameter konfigurieren

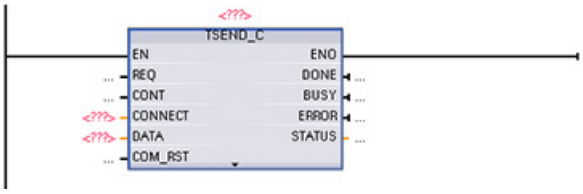
Über Kommunikationsbausteine (z. B. TSEND_C und TRCV_C) werden Verbindungen zwischen zwei CPUs hergestellt. Damit die PROFINET-Kommunikation von den CPUs aufgenommen werden kann, müssen die Parameter zum Senden und Empfangen von Meldungen eingerichtet werden. Diese Parameter legen fest, wie die Kommunikation abläuft, wenn Meldungen von einem Zielgerät gesendet oder von diesem empfangen werden.

Sendeparameter für TSEND_C konfigurieren

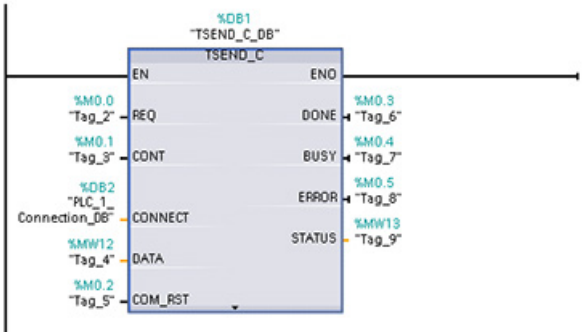
Anweisung TSEND_C

Die Anweisung TSEND_C (Seite 668) baut eine Kommunikationsverbindung zu einer Partnerstation auf. Die Verbindung wird eingerichtet, aufgebaut und automatisch überwacht, bis eine Anweisung die Verbindung trennt. Die Anweisung TSEND_C verbindet die Funktionen von TCON, TDISCON und TSEND.

In STEP 7 können Sie in der Gerätekonfiguration einstellen, wie Daten mit einer Anweisung TSEND_C gesendet werden sollen. Zuerst fügen Sie die Anweisung aus dem Ordner "Kommunikation" in der Taskcard "Anweisungen" in das Programm ein. Die Anweisung TSEND_C wird zusammen mit dem Dialog "Aufrufoptionen" angezeigt, in dem Sie einen DB zum Speichern der Parameter der Anweisung zuweisen können.



Sie können den Ein- und Ausgängen Speicherplätze im Variablenspeicher zuweisen. Dies wird in der folgenden Abbildung gezeigt:



Allgemeine Parameter konfigurieren

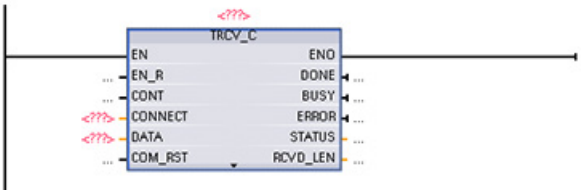
Sie können die Parameter im Dialog "Eigenschaften" der Anweisung TSEND_C festlegen. Dieser Dialog wird am unteren Ende der Seite angezeigt, wenn Sie einen Teil der Anweisung TSEND_C selektiert haben.

Empfangsparameter für TRCV_C konfigurieren

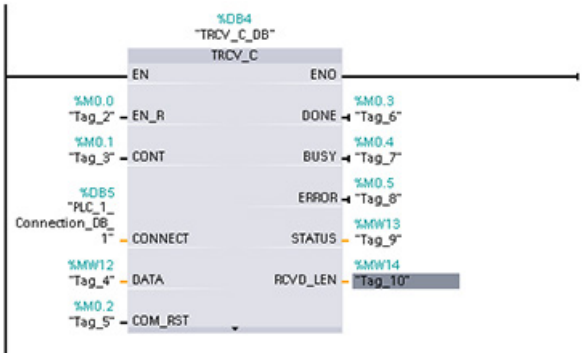
Anweisung TRCV_C

Die Anweisung TRCV_C (Seite 668) baut eine Kommunikationsverbindung zu einer Partnerstation auf. Die Verbindung wird eingerichtet, aufgebaut und automatisch überwacht, bis eine Anweisung die Verbindung trennt. Die Anweisung TRCV_C verbindet die Funktionen von TCON, TDISCON und TRCV.

In STEP 7 können Sie in der CPU-Konfiguration einstellen, wie eine Anweisung TRCV_C Daten empfangen soll. Zuerst fügen Sie die Anweisung aus dem Ordner "Kommunikation" in der Taskcard "Anweisungen" in das Programm ein. Die Anweisung TRCV_C wird zusammen mit dem Dialog "Aufrufoptionen" angezeigt, in dem Sie einen DB zum Speichern der Parameter der Anweisung zuweisen können.



Sie können den Ein- und Ausgängen Speicherplätze im Variablenspeicher zuweisen. Dies wird in der folgenden Abbildung gezeigt:



Allgemeine Parameter konfigurieren

Sie können die Parameter im Dialog "Eigenschaften" der Anweisung TRCV_C festlegen. Dieser Dialog wird am unteren Ende der Seite angezeigt, wenn Sie einen Teil der Anweisung TRCV_C selektiert haben.

11.2.12 CPU und PROFINET IO-Gerät konfigurieren

11.2.12.1 PROFINET IO-Gerät hinzufügen

PROFINET IO-Gerät hinzufügen

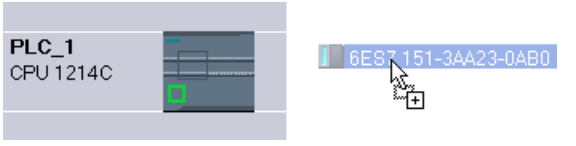
Verwenden Sie im Portal "Geräte & Netze" den Hardwarekatalog, um PROFINET IO-Geräte hinzuzufügen.

Hinweis

Um ein PROFINET IO-Gerät hinzuzufügen, können Sie STEP 7 Professional oder Basic ab V11 verwenden.

Zum Einfügen eines ET200S IO-Geräts erweitern Sie zum Beispiel die folgenden Behälter im Hardwarekatalog: Dezentrale Peripherie, ET200S, Schnittstellenmodule und PROFINET. Sie können dann das Schnittstellenmodul in der Liste der ET200S-Geräte (die nach der Bestellnummer sortiert sind) auswählen und das ET200S IO-Gerät hinzufügen.

Tabelle 11- 53 ET200S IO-Gerät zur Gerätekonfiguration hinzufügen

IO-Gerät einfügen	Ergebnis
	

Sie können nun das PROFINET IO-Gerät mit der CPU verbinden:

1. Klicken Sie auf dem Gerät mit der rechten Maustaste auf den Link "Nicht zugewiesen" und wählen Sie im Kontextmenü den Befehl "Neue IO-Steuerung zuweisen" aus, um den Dialog "IO-Steuerung auswählen" aufzurufen.
2. Wählen Sie in der Liste der IO-Steuerungen im Projekt Ihre S7-1200 CPU aus (in diesem Beispiel "PLC_1").
3. Klicken Sie auf "OK", um die Netzwerkverbindung anzulegen.

11.2.12.2 Konfigurieren der logischen Netzwerkverbindungen zwischen einer CPU und einem PROFINET IO-Gerät

Logische Netzwerkverbindungen konfigurieren

Nachdem Sie den Baugruppenträger mit der CPU konfiguriert haben, können Sie Ihre Netzwerkverbindungen einrichten.

Im Portal "Geräte & Netze" können Sie in der "Netzsicht" die Geräte in Ihrem Projekt vernetzen. Um eine PROFINET-Verbindung zu erstellen, klicken Sie auf das grüne Feld (PROFINET) auf dem ersten Gerät und ziehen eine Linie zum PROFINET-Feld auf dem zweiten Gerät. Lassen Sie die Maustaste los. Damit ist Ihre PROFINET-Verbindung hergestellt.

Ausführliche Informationen finden Sie unter "Gerätekonfiguration: Erstellen einer Netzwerkverbindung" (Seite 637).

11.2.12.3 CPUs und Gerätenamen zuweisen

CPUs und Gerätenamen zuweisen

Netzwerkverbindungen zwischen den Geräten weisen auch das PROFINET IO-Gerät der CPU zu, was erforderlich ist, damit die CPU das Gerät steuern kann. Um diese Zuweisung zu ändern, klicken Sie auf den PLC-Namen, der auf dem PROFINET IO-Gerät angezeigt wird. Daraufhin wird ein Dialog geöffnet, in dem Sie das PROFINET IO-Gerät von der aktuellen CPU trennen und neu zuweisen oder auch ohne Zuweisung lassen können.

Den Geräten in Ihrem PROFINET-Netzwerk muss ein Name zugewiesen sein, damit Sie eine Verbindung zur CPU herstellen können. Sie weisen Ihren PROFINET-Geräten in der "Netzsicht" Namen zu, wenn die Geräte noch keinen Namen haben oder wenn der Name eines Geräts geändert werden soll. Klicken Sie hierfür mit der rechten Maustaste auf das PROFINET IO-Gerät und wählen Sie "Gerätename zuweisen".

Den Namen eines PROFINET IO-Geräts müssen Sie sowohl im STEP 7-Projekt als auch über das Werkzeug "Online & Diagnose" im Konfigurationsspeicher des PROFINET IO-Geräts zuweisen (z. B. im Konfigurationsspeicher eines ET200 S-Schnittstellenmoduls). Fehlt ein Name oder entsprechen sich die Namen an den beiden Speicherorten nicht, funktioniert der Modus für den PROFINET IO-Datenaustausch nicht. Weitere Informationen finden Sie unter "Online- und Diagnose-Tools: einem PROFINET-Gerät online einen Namen zuweisen (Seite 1120)".

11.2.12.4 IP-Adressen zuweisen

IP-Adressen zuweisen

In einem PROFINET-Netzwerk benötigt jedes Gerät außerdem eine IP-Adresse (Internet-Protocol-Adresse). Mit Hilfe dieser Adresse kann ein Gerät Daten über ein komplexeres Netzwerk liefern:

- Verwenden Sie Programmiergeräte oder andere Netzwerkgeräte mit integrierter Adapterkarte, die an Ihr Werks-LAN angeschlossen ist, oder mit Ethernet/USB-Adapterkarte, die an ein getrenntes Netzwerk angeschlossen ist, so müssen Sie diesen Geräten IP-Adressen zuweisen. Weitere Informationen finden Sie unter "IP-Adressen zu Programmier- und Netzwerkgeräten zuweisen" (Seite 642).
- Sie können einer CPU oder einem Netzwerkgerät auch online eine IP-Adresse zuweisen. Dies ist besonders bei der ersten Gerätekonfiguration nützlich. Weitere Informationen finden Sie unter "IP-Adresse zu einer CPU online zuweisen (Seite 645)".
- Nachdem Sie Ihre CPU oder Ihr Netzwerkgerät in Ihrem Projekt konfiguriert haben, können Sie die Parameter für die PROFINET-Schnittstelle konfigurieren, um die IP-Adresse aufzunehmen. Weitere Informationen finden Sie unter "IP-Adresse für eine CPU in Ihrem Projekt konfigurieren" (Seite 646).

11.2.12.5 IO-Zykluszeit konfigurieren

IO-Zykluszeit konfigurieren

Ein PROFINET IO-Gerät wird innerhalb des Zeitraums "IO-Zyklus" von der CPU mit neuen Daten versorgt. Die Aktualisierungszeit kann für jedes Gerät separat konfiguriert werden und legt den Zeitraum fest, in dem Daten von der CPU zum und vom Gerät übertragen werden.

STEP 7 berechnet die Aktualisierungszeit für den "IO-Zyklus" automatisch in den Standardeinstellungen für jedes Gerät des PROFINET-Netzwerks und berücksichtigt dabei das Volumen der auszutauschenden Daten und die Anzahl der dieser Steuerung zugewiesenen Geräte. Wenn Sie nicht möchten, dass die Aktualisierungszeit automatisch berechnet wird, können Sie diese Einstellung ändern.

Sie legen die Parameter für den "IO-Zyklus" im Konfigurationsdialog "Eigenschaften" des PROFINET IO-Geräts fest. Dieser Dialog wird am unteren Ende der Seite angezeigt, wenn Sie einen Teil der Anweisung selektiert haben.

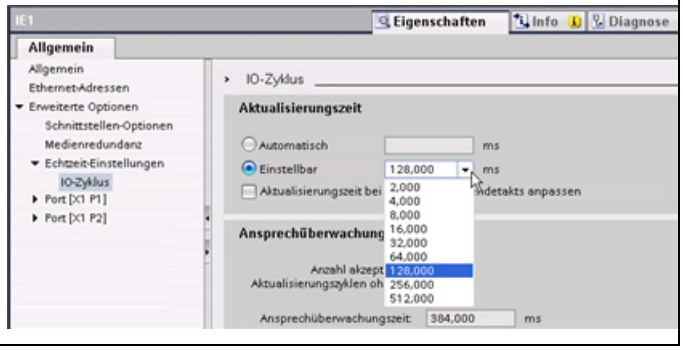
Klicken Sie in der "Gerätesicht" des PROFINET IO-Geräts auf den PROFINET-Port. Im Dialog "PROFINET-Schnittstelle" rufen Sie die Parameter "IO-Zyklus" mit der folgenden Menüauswahl auf:

- "Erweiterte Optionen"
- "Echtzeiteinstellungen"
- "IO-Zyklus"

Definieren Sie die "Aktualisierungszeit" für den IO-Zyklus wie folgt:

- Wenn eine geeignete Aktualisierungszeit automatisch berechnet werden soll, wählen Sie "Automatisch".
- Um die Aktualisierungszeit selbst einzustellen, wählen Sie "Kann eingestellt werden" und geben die gewünschte Aktualisierungszeit in ms ein.
- Um die Konsistenz zwischen dem Sendetakt und der Aktualisierungszeit sicherzustellen, aktivieren Sie die Option "Aktualisierungszeit anpassen, wenn sich der Sendetakt ändert". Diese Option gewährleistet, dass keine Aktualisierungszeit eingestellt wird, die kleiner als der Sendetakt ist.

Tabelle 11- 54 PROFINET IO-Zykluszeit für das ET200S konfigurieren

PROFINET IO-Gerät ET200 S	Dialog PROFINET IO-Zyklus für das ET200S
	

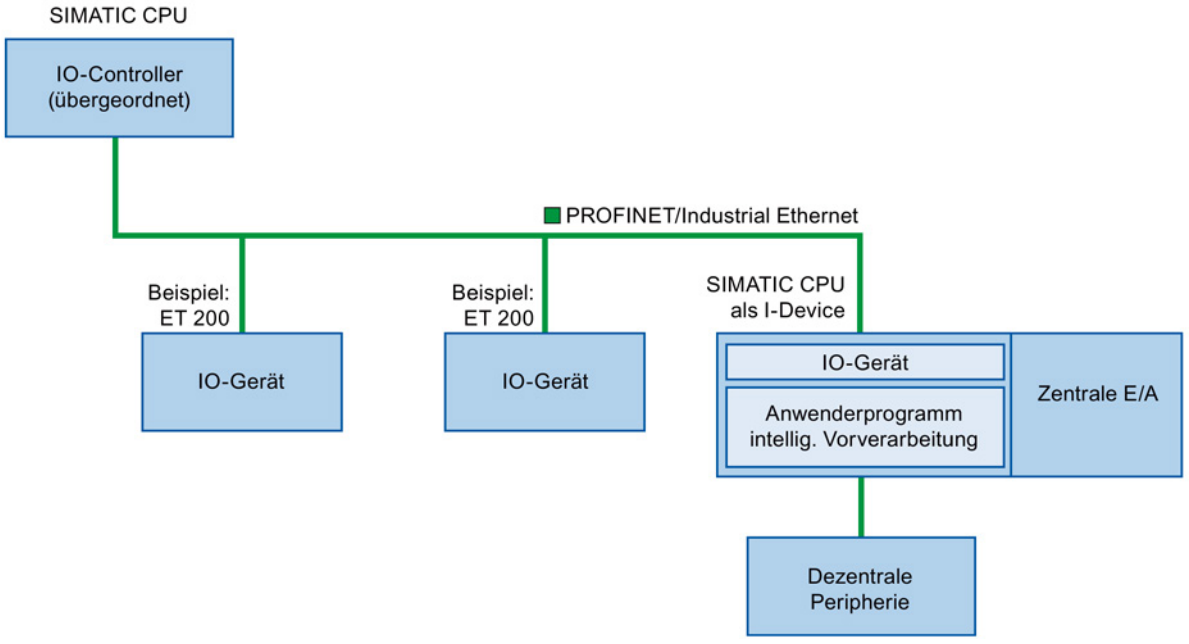
① PROFINET-Port

11.2.13 CPU und PROFINET I-Device konfigurieren

11.2.13.1 I-Device-Funktionalität

Die Funktion "I-Device" (intelligentes IO-Gerät) der CPU erleichtert den Datenaustausch mit einem IO-Controller und den Betrieb der CPU z. B. als intelligente Vorverarbeitungseinheit für Teilprozesse. Das I-Device ist als IO-Gerät mit einem übergeordneten IO-Controller verbunden.

Die Vorverarbeitung erfolgt durch das Anwenderprogramm der CPU. Die in den zentralen oder dezentralen E/A (PROFINET IO oder PROFIBUS DP) erfassten Prozesswerte werden vom Anwenderprogramm vorverarbeitet und der CPU einer übergeordneten Station über eine PROFINET IO-Schnittstelle zur Verfügung gestellt.



Benennungskonvention für "I-Device"

Eine CPU oder ein CP mit I-Device-Funktion wird in den folgenden Beschreibungen der Einfachheit halber als "I-Device" bezeichnet.

11.2.13.2 Eigenschaften und Vorteile des I-Device

Einsatzbereiche

Einsatzbereiche für ein I-Device:

- Dezentrale Verarbeitung:

  Komplexe Automatisierungsaufgaben können in kleinere Teilanlagen/Teilprozesse aufgeteilt werden. Die Prozesse werden somit handlicher und die Teilaufgaben vereinfacht.

- Abtrennen von Teilprozessen:

  Komplexe, weit verteilte und umfangreiche Prozesse können mit I-Devices in mehrere Teilprozesse mit schlankeren Schnittstellen aufgeteilt werden Die Teilprozesse können bei Bedarf in einzelnen STEP 7-Projekten gespeichert werden, die später in einem Hauptprojekt zusammengeführt werden können.

- Know-how-Schutz:

  Komponenten können mit lediglich einer GSD-Datei für die I-Device-Schnittstellenbeschreibung an Stelle eines STEP 7-Projekts geliefert werden. Der Anwender kann sein Programm schützen, weil es nicht mehr veröffentlicht werden muss.

Eigenschaften

Eigenschaften des I-Device:

- Getrenntes Arbeiten mit STEP 7-Projekten:

  Ersteller und Benutzer eines I-Device können mit vollständig getrennten STEP 7-Automatisierungsprojekten arbeiten. Die GSD-Datei bildet die Schnittstelle zwischen den STEP 7-Projekten. Auf diese Weise kann durch eine standardisierte Schnittstelle eine Verbindung zu Standard-IO-Controllern hergestellt werden.

- Real-Time-Kommunikation:

  Das I-Device ist über eine PROFINET IO-Schnittstelle mit einem deterministischen PROFINET IO-System ausgestattet und unterstützt daher RT (Real-Time) und IRT(Isochronous-Real-Time)-Kommunikation.

Vorteile

Ein I-Device hat die folgenden Vorteile:

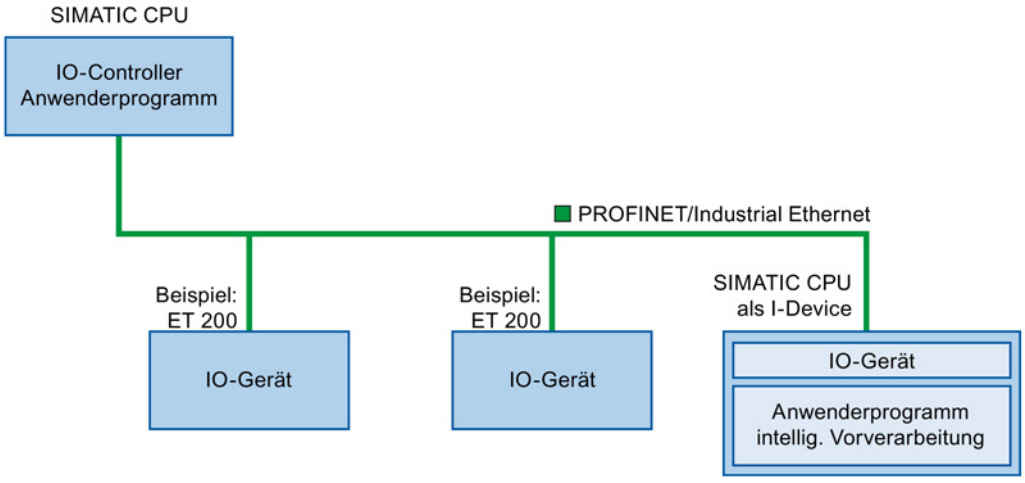
- Einfache Anbindung von IO-Controllern
- Echtzeit-Kommunikation zwischen IO-Controllern
- Entlastung des IO-Controllers durch Verteilung der Rechenleistung auf die I-Devices
- Geringere Kommunikationslast durch lokale Verarbeitung der Prozessdaten
- Handlichere Projekte durch die Verarbeitung von Teilaufgaben in separaten STEP 7-Projekten

11.2.13.3 Merkmale eines I-Device

Ein I-Device wird in einem E/A-System wie ein Standard-IO-Gerät eingefügt.

I-Device ohne untergeordnetes PROFINET IO-System

Ein I-Device hat keine eigene dezentrale Peripherie. Die Konfiguration und Parametrierung eines I-Device als IO-Gerät sind die gleichen wie bei einem dezentralen E/A-System (zum Beispiel ET 200).



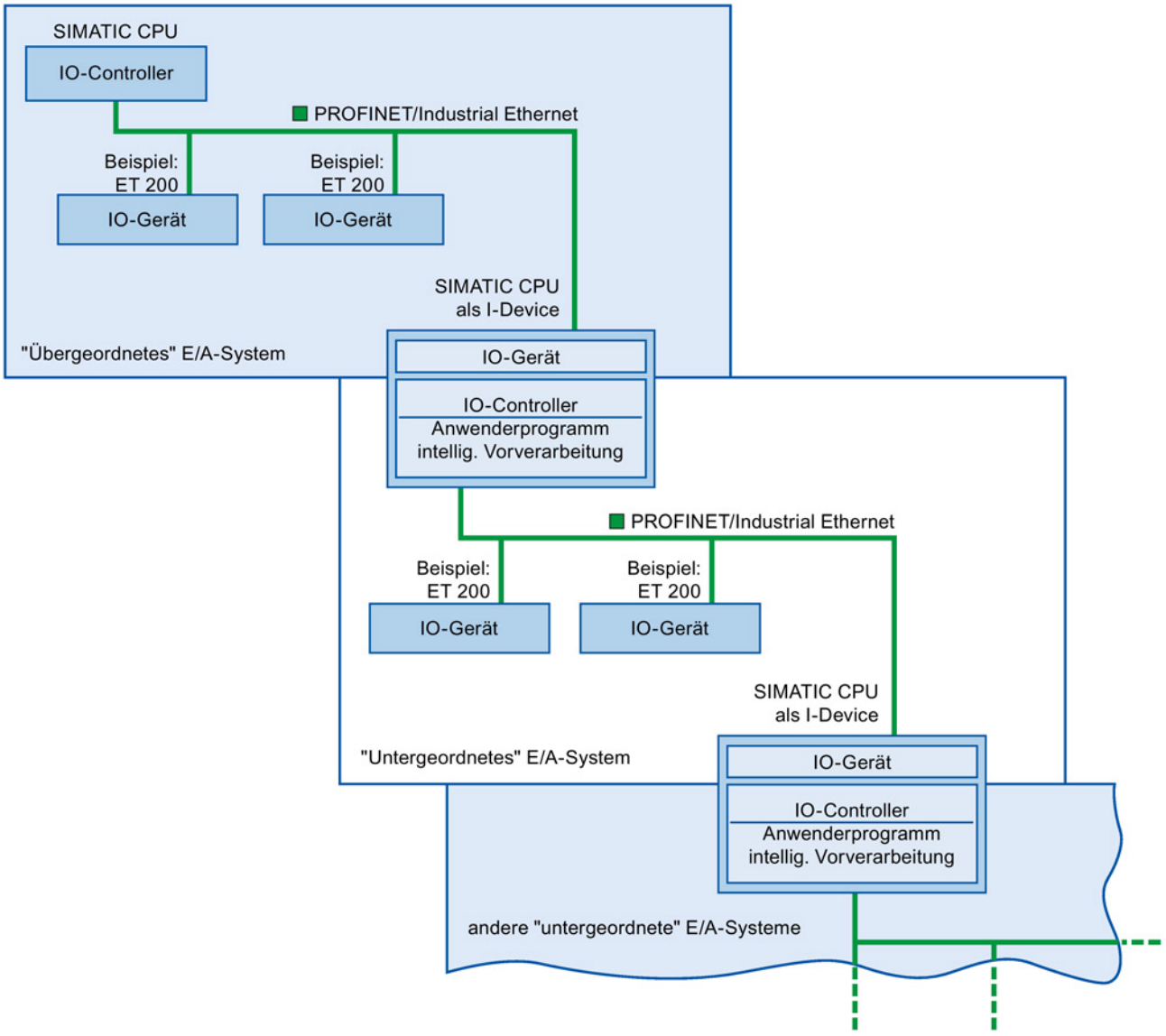
I-Device mit untergeordnetem PROFINET IO-System

Je nach Konfiguration kann ein I-Device zusätzlich zu seiner Funktion als IO-Gerät auch als IO-Controller an einer PROFINET-Schnittstelle genutzt werden.

Somit kann ein I-Device über seine PROFINET-Schnittstelle zu einem übergeordneten E/A-System gehören und als IO-Controller sein eigenes untergeordnetes E/A-System unterstützen.

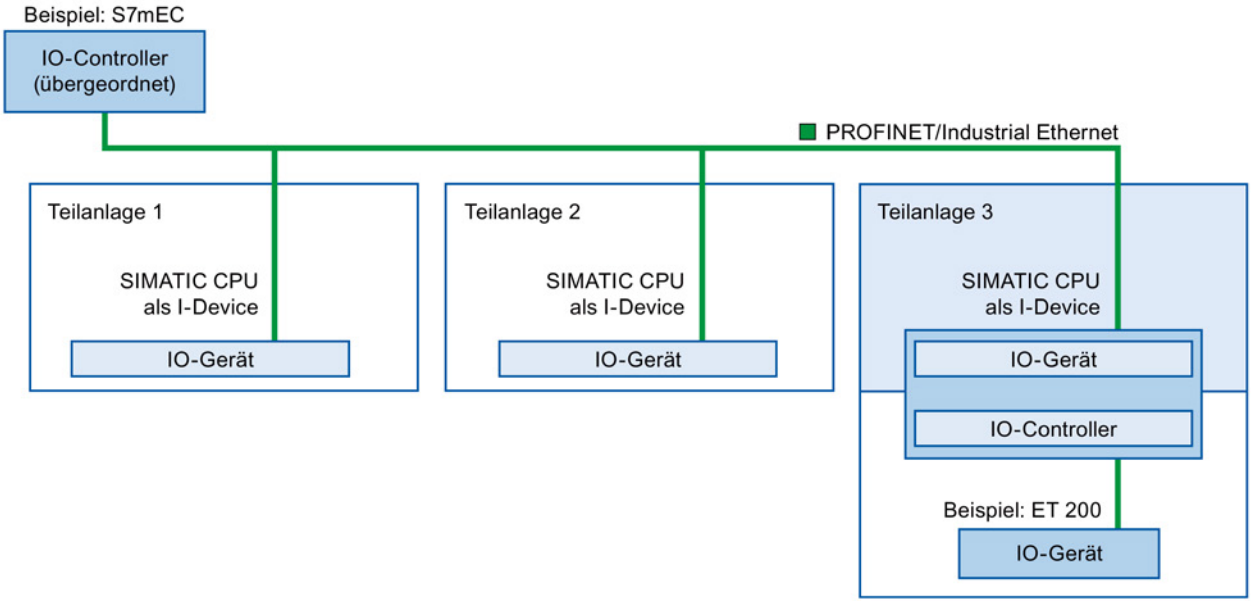
Das untergeordnete E/A-System kann wiederum eigene I-Devices enthalten (siehe Abbildung unten). Dies ermöglicht hierarchisch aufgebaute E/A-Systeme.

Zusätzlich zu seiner Funktion als IO-Controller kann ein I-Device auch über eine PROFIBUS-Schnittstelle als DP-Master für ein untergeordnetes PROFIBUS-System fungieren.



Beispiel: I-Device als IO-Gerät und IO-Controller

Der Funktionsablauf eines I-Device als IO-Gerät und IO-Controller wird hier am Beispiel eines Druckprozesses erklärt. Das I-Device steuert eine Teilanlage (einen Teilprozess). Eine von mehreren Teilanlagen wird beispielsweise verwendet, um zusätzliche Blätter wie Flyer oder Broschüren in ein Paket mit Druckerzeugnissen einzufügen.



Teilanlage 1 und Teilanlage 2 bestehen aus je einem I-Device mit zentralen E/A. Zusammen mit dem dezentralen E/A-System (zum Beispiel ET 200) bildet das I-Device die Teilanlage 3.

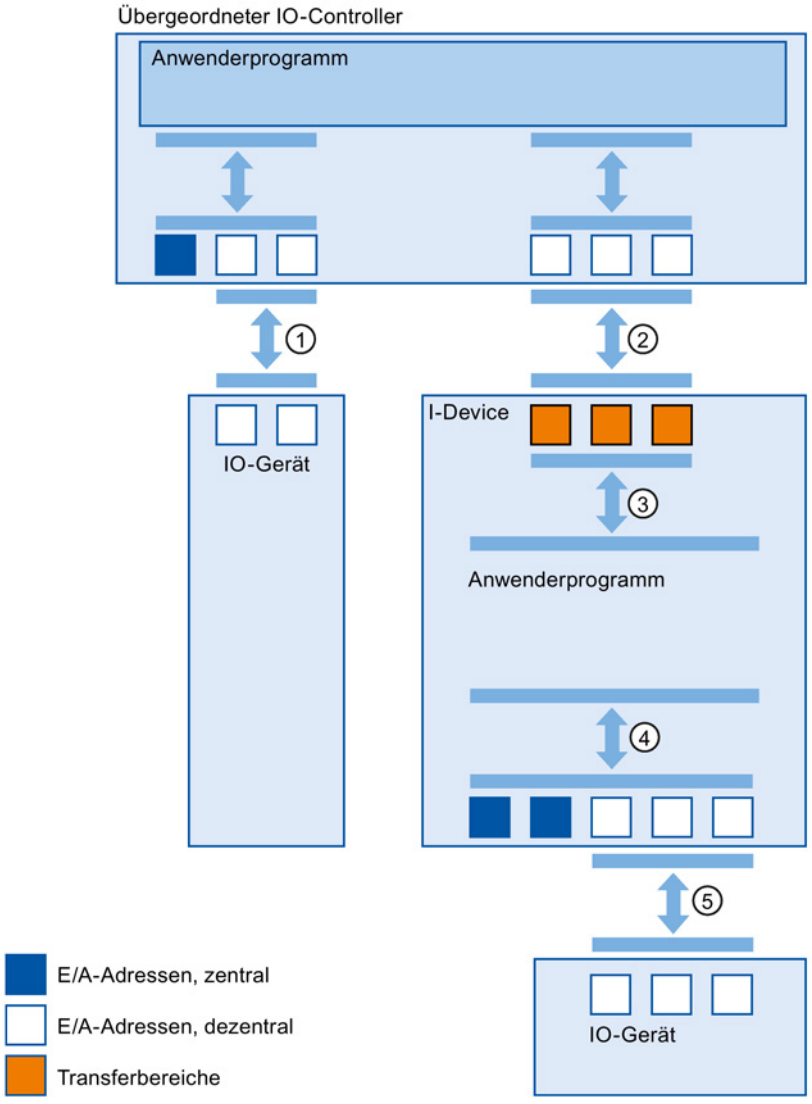
Das Anwenderprogramm im I-Device ist für die Vorverarbeitung der Prozessdaten verantwortlich. Für diese Aufgabe benötigt das Anwenderprogramm des I-Device Standardeinstellungen (zum Beispiel Steuerungsdaten) vom übergeordneten IO-Controller. Das I-Device liefert die Ergebnisse (zum Beispiel Status seiner Teilaufgabe) an den übergeordneten IO-Controller.

11.2.13.4 Datenaustausch zwischen über- und untergeordnetem E/A-System

Transferbereiche sind Schnittstellen zum Anwenderprogramm der I-Device-CPU. Die Eingänge werden im Anwenderprogramm verarbeitet, die Ausgänge sind das Ergebnis der Verarbeitung.

Die Daten für die Kommunikation zwischen IO-Controller und I-Device stehen in den Transferbereichen zur Verfügung. Ein Transferbereich enthält eine Informationseinheit, die konsistent zwischen IO-Controller und I-Device ausgetauscht wird. Weitere Informationen über die Konfiguration und Verwendung von Transferbereichen enthält das Kapitel "I-Device konfigurieren".

Die nächste Abbildung zeigt den Datenaustausch zwischen über- und untergeordnetem E/A-System. Die einzelnen Kommunikationsbeziehungen sind nachstehend anhand von Nummern erläutert:



① **Datenaustausch zwischen übergeordnetem IO-Controller und normalem IO-Gerät**

Der IO-Controller und die IO-Geräte tauschen Daten über PROFINET aus.

② **Datenaustausch zwischen übergeordnetem IO-Controller und I-Device**

Der IO-Controller und das I-Device tauschen Daten über PROFINET aus.

Der Datenaustausch zwischen einem übergeordneten IO-Controller und einem I-Device beruht auf der üblichen Beziehung zwischen IO-Controller und IO-Gerät.

Für den übergeordneten IO-Controller stellen die Transferbereiche der I-Devices Teilmodule einer vorkonfigurierten Station dar.

Die Ausgangsdaten des IO-Controllers sind die Eingangsdaten des I-Device. Umgekehrt sind die Eingangsdaten des IO-Controllers die Ausgangsdaten des I-Device.

③ **Transferbeziehung zwischen Anwenderprogramm und Transferbereich**

Es werden Ein- und Ausgangsdaten zwischen Anwenderprogramm und Transferbereich ausgetauscht.

④ **Datenaustausch zwischen Anwenderprogramm und E/A des I-Device**

Es werden Ein- und Ausgangsdaten zwischen Anwenderprogramm und der zentralen/dezentralen Peripherie ausgetauscht.

⑤ **Datenaustausch zwischen I-Device und einem untergeordneten IO-Gerät**
Es werden Daten zwischen I-Device und seinen IO-Geräten ausgetauscht. Die Datenübertragung erfolgt über PROFINET.

11.2.13.5 Konfigurieren des I-Device

Für die Konfiguration gibt es prinzipiell zwei Möglichkeiten:

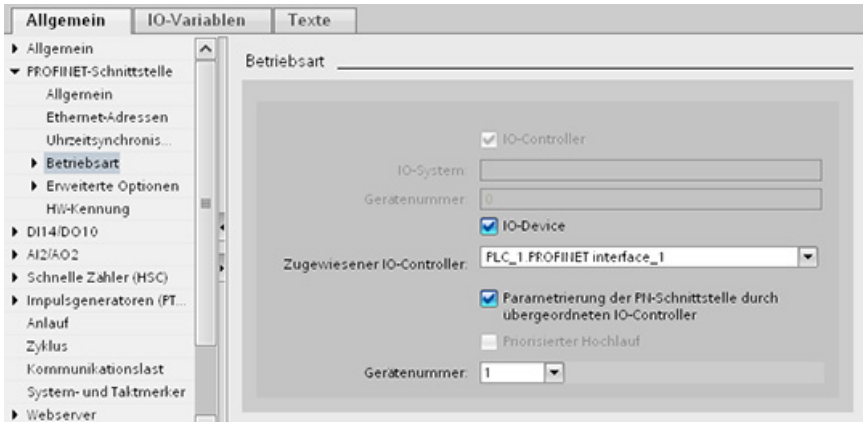
- Konfiguration eines I-Device in einem Projekt
- Konfiguration eines I-Device, das in einem anderen Projekt oder Engineering-System verwendet wird.

In STEP 7 können Sie ein I-Device für ein anderes Projekt oder ein anderes Engineering-System konfigurieren, indem Sie ein konfiguriertes I-Device in eine GSD-Datei exportieren. Dann importieren Sie die GSD-Datei wie andere GSD-Dateien in andere Projekte oder Engineering-Systeme. In dieser GSD-Datei sind neben anderen Daten auch die Transferbereiche für den Datenaustausch gespeichert.

Konfiguration eines I-Device in einem Projekt

1. Ziehen Sie mit der Maus eine PROFINET-CPU aus dem Hardwarekatalog in die Netzsicht.
2. Ziehen Sie mit der Maus eine PROFINET-CPU, die auch als IO-Device konfiguriert werden kann, aus dem Hardwarekatalog in die Netzsicht. Dieses Gerät wird als I-Device konfiguriert (z. B. eine CPU 1215C).
3. Wählen Sie die PROFINET-Schnittstelle für das I-Device.
4. Wählen Sie im Inspektorfenster im Navigationsbereich "Betriebsart" aus und aktivieren Sie das Kontrollkästchen "IO-Device".
5. Jetzt können Sie in der Klappliste "Zugewiesener IO-Controller" den IO-Controller auswählen.

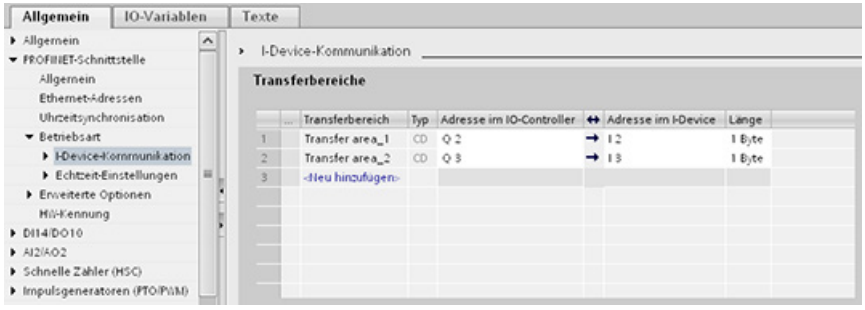
   Nachdem Sie den IO-Controller ausgewählt haben, werden in der Netzsicht die Vernetzung und das IO-System zwischen beiden Geräten angezeigt.



6. Über das Kontrollkästchen "Parametrierung der PN-Schnittstelle durch übergeordneten IO-Controller" geben Sie an, ob die Schnittstellenparameter vom I-Device selbst oder von einem übergeordneten IO-Controller zugewiesen werden.

   Wenn Sie das I-Device in einem untergeordneten IO-System betreiben, können die Parameter der PROFINET-Schnittstelle des I-Device (z. B. der Portparameter) nicht vom übergeordneten IO-Controller zugewiesen werden.

7. Konfigurieren Sie die Transferbereiche. Die Transferbereiche finden Sie im Navigationsbereich "I-Device-Kommunikation":
   - Klicken Sie in der Spalte "Transferbereich" in das erste Feld. STEP 7 weist einen Standardnamen zu, den Sie ändern können.
   - Wählen Sie die Art der Kommunikationsbeziehung aus: Derzeit können Sie lediglich CD oder F-CD auswählen.
   - Adressen werden automatisch voreingestellt. Sie können die Adressen bei Bedarf korrigieren und die Länge des Transferbereichs, der beständig übertragen wird, bestimmen.



8. Für jeden Transferbereich wird im Navigationsbereich ein eigener Eintrag angelegt. Wenn Sie einen dieser Einträge auswählen, können Sie die Details des Transferbereichs anpassen, sie korrigieren und Kommentare dazu eingeben.

Konfigurieren eines I-Device mit einer GSD-Datei

Wenn Sie ein I-Device in einem anderen Projekt nutzen oder wenn das I-Device in einem anderen Engineering-System verwendet wird, konfigurieren Sie den übergeordneten IO-Controller und das I-Device wie oben beschrieben.

Klicken Sie jedoch nach der Konfiguration der Transferbereiche auf die Schaltfläche "Exportieren", damit eine neue GSD-Datei für das I-Device erstellt wird. Diese GSD-Datei stellt das konfigurierte I-Device in anderen Projekten dar.

Die Schaltfläche "Exportieren" finden Sie im Inspektorfenster im Bereich "I-Device-Kommunikation".

Die Hardwarekonfiguration wird übersetzt und der Export-Dialog geöffnet.

Weisen Sie dem I-Device-Proxy einen Namen zu und geben Sie in die vorgesehenen Felder eine Beschreibung ein. Klicken Sie auf die Schaltfläche "Exportieren", um den Vorgang abzuschließen.

Dann können Sie die GSD-Datei beispielsweise in ein anderes Projekt importieren.

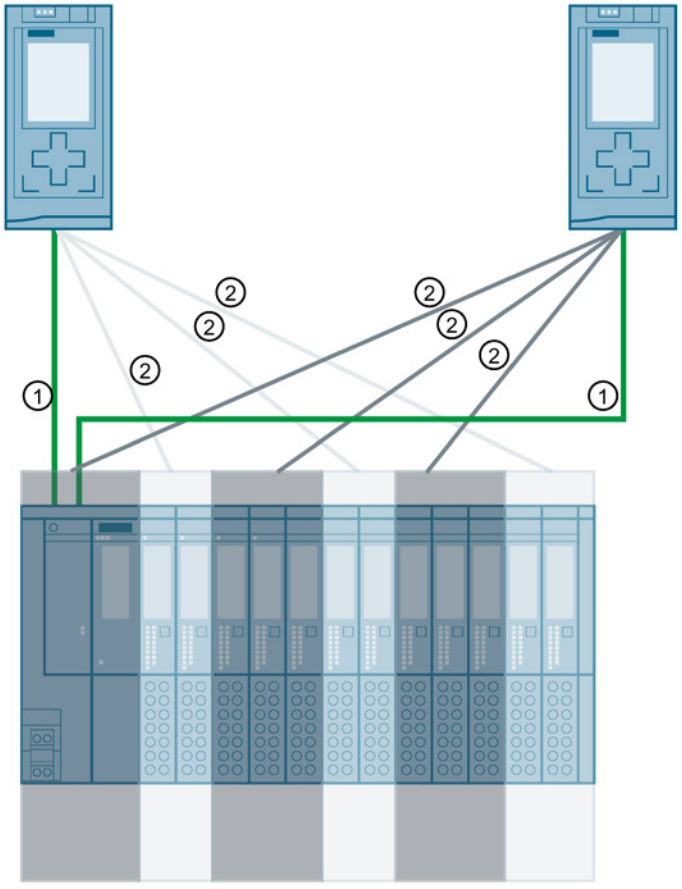
11.2.14 Shared Devices

11.2.14.1 Shared-Device-Funktion

In größeren und weiter verteilten Systemen werden häufig zahlreiche IO-Controller eingesetzt.

Ohne die "Shared-Device"-Funktion ist jedes E/A-Modul eines IO-Geräts dem gleichen IO-Controller zugewiesen. Wenn Sensoren, die sich physisch nahe beieinander befinden, verschiedenen IO-Controllern Daten liefern müssen, sind dafür mehrere IO-Geräte erforderlich.

Mit der "Shared-Device"-Funktion können die Module oder Submodule eines IO-Geräts auf verschiedene IO-Controller aufgeteilt werden. Somit sind flexible Automatisierungskonzepte möglich. Sie haben beispielsweise die Möglichkeit, E/A-Module, die sich nahe beieinander befinden, in einem IO-Gerät zu verbinden.



① PROFINET
② Logische Zuweisung

Prinzip

Der Zugriff auf die Submodule des Shared Device wird dann auf die einzelnen IO-Controller aufgeteilt. Jedes Submodul des Shared Device wird exklusiv einem IO-Controller zugewiesen.

Voraussetzung (GSD-Konfiguration)

- STEP 7 V12 Service Pack 1 oder höher
- CPU ab FW 1.1 als IO-Controller
- Unterstützung der Shared-Device-Funktion durch das IO-Gerät, z. B. Schnittstellenmodul IM 155-5 PN ST
- GSD-Datei für die Konfiguration des IO-Geräts ist installiert
- Eine als I-Device konfigurierte S7-1200 CPU unterstützt die Shared-Device-Funktion. Sie müssen die PROFINET GSD-Datei für das I-Device aus STEP 7 (ab V5.5) exportieren und dann in STEP 7 (TIA-Portal) importieren.

Zugriff konfigurieren

Das IO-Gerät muss in verschiedenen Projekten vorhanden sein, so dass die Module oder Submodule eines IO-Geräts verschiedenen IO-Controllern zugewiesen werden können. Für jeden IO-Controller ist ein eigenes Projekt erforderlich.

Mit dem Parameter "Shared Device" des Schnittstellenmoduls bestimmen Sie die Module und Submodule, auf die der IO-Controller Zugriff hat:

- Wenn der lokale IO-Controller Zugriff auf das konfigurierte Modul hat, wählen Sie den Namen des IO-Controllers in der Liste aus.
- Wenn der IO-Controller eines anderen Projekts und nicht der lokale IO-Controller auf das konfigurierte Modul Zugriff haben soll, wählen Sie den Eintrag "---".

Die Konfiguration ist bezüglich des Zugriffs konsistent, wenn jedes Modul oder Submodul in genau einem Projekt einem IO-Controller zugewiesen ist.

Modul oder Submodul ist einem anderen IO-Controller zugewiesen

Im folgenden Abschnitt werden die Konsequenzen der Einstellung "---" für den Parameter "Shared Device" aus Sicht des lokalen IO-Controllers beschrieben.

In diesem Fall hat der lokale IO-Controller keinen Zugriff auf das in dieser Weise konfigurierte Modul. Dies bedeutet im Einzelnen:

- Kein Datenaustausch mit dem Modul oder Submodul
- Kein Empfang von Alarm- und Diagnosemeldungen, also keine Anzeige des Diagnosestatus in der Onlinesicht
- Keine Parametrierung des Moduls oder Submoduls

Einstellung der Echtzeiteigenschaften

STEP 7 berechnet die Kommunikationslast und somit die resultierenden Aktualisierungszeiten. Sie müssen die Anzahl der projektexternen IO-Controller in dem Projekt, in dem die PROFINET-Schnittstelle des Shared Device dem IO-Controller zugewiesen ist, angeben, damit eine Berechnung mit Shared-Device-Konfigurationen möglich ist.

Die maximal mögliche Anzahl von IO-Controllern für das Shared Device ist vom Gerät abhängig. Diese Anzahl ist in der GSD-Datei des Shared Device gespeichert.

Sie können einen sehr kurzen Sendetakt mit einer CPU als IO-Controller festlegen. Der Sendetakt kann kürzer sein als der kürzeste vom Shared Device unterstützte Sendetakt. In diesem Fall wird das Shared Device vom IO-Controller mit einem Sendetakt betrieben, der vom Gerät unterstützt wird (Sendetaktanpassung).

Beispiel: Eine CPU unterstützt Sendetakte ab 0,25 ms. Ein konfiguriertes IO-Gerät unterstützt auch Sendetakte ab 0,25 ms, ein anderes IO-Gerät unterstützt Sendetakte ab 1 ms. In diesem Fall haben Sie die Möglichkeit, den kurzen Sendetakt von 0,25 ms für die CPU einzustellen. Die CPU betreibt beispielsweise das "langsame" IO-Gerät mit dem Sendetakt von 1 ms.

Regeln für die Konfiguration

- IO-Controller, die das Shared Device verwenden, werden in verschiedenen Projekten angelegt. In jedem Projekt muss darauf geachtet werden, dass das Shared Device in jeder Station identisch konfiguriert ist. Nur ein einziger IO-Controller darf jemals vollständigen Zugriff auf ein Submodul haben. Inkonsistenzen in der Konfiguration führen zu einem Fehler des Shared Device.
- E/A-Adressen eines Moduls oder Submoduls dürfen nur bearbeitet werden, wenn ein Modul oder Submodul dem IO-Controller im gleichen Projekt zugewiesen ist.
- Das Shared Device muss in jedem Projekt die gleichen IP-Parameter und den gleichen Gerätenamen haben.
- Der Sendetakt muss bei allen IO-Controllern, die Zugriff auf das Shared Device haben, identisch sein.
- Die S7-Subnetz-ID des Subnetzes, mit dem das Shared Device verbunden ist, muss in allen Projekten identisch sein.
- Die folgenden Funktionen sind nur verfügbar, wenn die PROFINET-Schnittstelle des Shared Device dem lokalen IO-Controller zugewiesen ist:
  - IRT-Betrieb
  - Priorisierter Anlauf
  - Parametrierung der Porteigenschaften

Grenzbedingungen

Die folgenden Grenzbedingungen entstehen, weil eine Shared-Device-Konfiguration über mehrere Projekte verteilt ist:

- Die Adressen von Modulen und Submodulen, die diesem IO-Controller nicht zugewiesen sind, fehlen in der Adressübersicht jedes IO-Controllers, der Zugriff auf ein Shared Device hat.
- Die nicht zugewiesenen Module und Submodule werden bei der Konsistenzprüfung nicht in der Grenzwertberechnung der Konfiguration für das Shared Device berücksichtigt. Deshalb müssen Sie selbst prüfen, ob die maximale Anzahl Submodule bzw. die maximale Menge zyklischer IO-Daten für das Shared Device nicht überschritten wird. Informationen zu den maximalen Werten entnehmen Sie der Dokumentation der Geräte, die Sie verwenden.
- Konfigurationsfehler wie die Zuweisung eines Moduls oder Submoduls zu mehreren IO-Controllern werden in STEP 7 nicht erkannt.
- CPUs, in die eine Shared-Device-Konfiguration geladen wurde, haben keine Informationen dazu, ob es sich bei dem IO-Gerät um ein Shared Device handelt. Module und Submodule, die anderen IO-Controllern und damit anderen CPUs zugewiesen sind, fehlen in der geladenen Konfiguration. Diese Module und Submodule werden deshalb weder auf dem CPU-Webserver noch in der CPU-Anzeige angezeigt.

11.2.14.2 Beispiel: Ein Shared Device konfigurieren (GSD-Konfiguration)

In diesem Beispiel wird beschrieben, wie Sie mit STEP 7 ab V13 SP1 ein dezentrales E/A-System als Shared Device konfigurieren.

Eine "dezentrale" Konfiguration mit unterschiedlichen Engineering-Tools für unterschiedliche IO-Controllerfamilien ist möglich. Die nachfolgend beschriebene Vorgehensweise basiert auf STEP 7 V13 SP1 und ist begrenzt auf die Konfiguration mit zwei IO-Controllern der S7-1200-Serie, die ein Shared Device gemeinsam nutzen.

Im Beispiel werden zwei Projekte mit jeweils einem IO-Controller angelegt:

- Controller1
- Controller2

Sie müssen das Shared Device in beiden Projekten erstellen, auch wenn es sich physisch um ein und dasselbe IO-Gerät handelt.

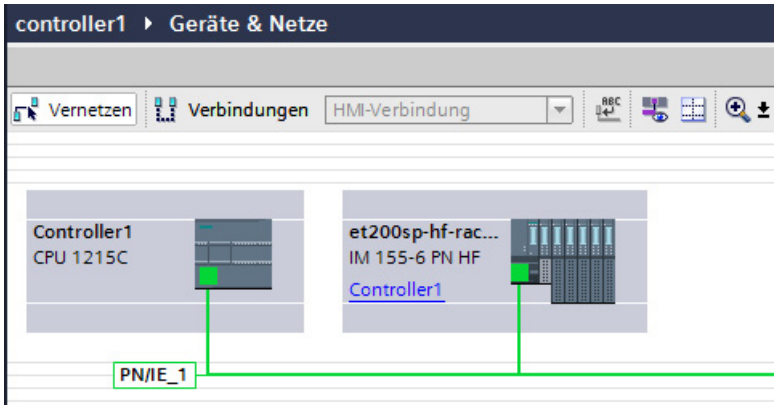
Voraussetzungen

- STEP 7 ab V13 SP1
- Unterstützung der Shared-Device-Funktion durch das IO-Gerät (z. B. ET 200SP IM 155-6 PN HF V3.1).
- GSD-Datei für die Konfiguration des IO-Geräts als Shared Device ist installiert.

Vorgehensweise: Projekt 1 anlegen

Um das erste Projekt mit einem Shared Device anzulegen, gehen Sie wie folgt vor:

1. Starten Sie STEP 7.
2. Legen Sie ein neues Projekt mit dem Namen "Controller1" an.
3. Fügen Sie eine CPU 1215C aus dem Hardwarekatalog in die Netzsicht ein. Benennen Sie es "Controller1".
4. Fügen Sie ein IO-Gerät mit der "Shared-Device"-Funktion (zum Beispiel ein ET 200SP) aus dem Hardwarekatalog ein (Hardwarekatalog: Weitere Feldgeräte > PROFINET IO > E/A).
5. Weisen Sie den IO-Controller "Controller1" dem IO-Gerät zu.



6. Doppelklicken Sie auf das IO-Gerät und fügen Sie alle erforderlichen Module und Submodule aus dem Hardwarekatalog in die Geräteübersichtstabelle ein.
7. Weisen Sie die Modulparameter zu.
8. Speichern Sie das Projekt.

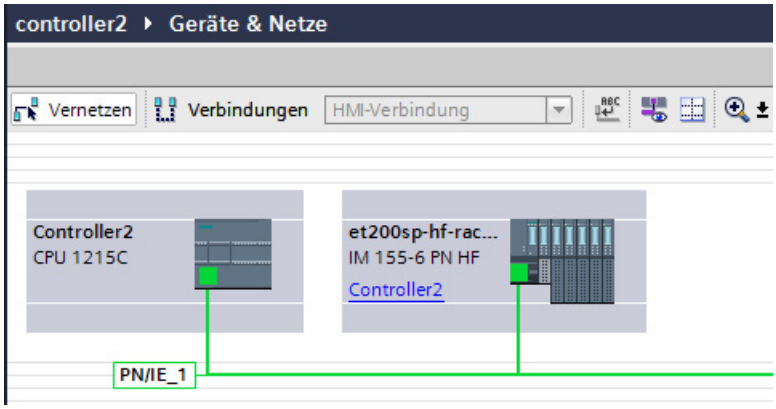
Vorgehensweise: Projekt 2 anlegen

Um das zweite Projekt mit einem Shared Device anzulegen, gehen Sie wie folgt vor:

1. Starten Sie STEP 7 erneut.

   Eine neue Instanz von STEP 7 wird geöffnet.

2. Legen Sie in der neuen Instanz ein neues Projekt mit dem Namen "Controller2" an.
3. Fügen Sie eine CPU 1215C in die Netzsicht ein. Benennen Sie sie "Controller2".
4. Kopieren Sie das IO-Gerät aus dem Projekt "Controller1" und fügen Sie es in die Netzsicht von Projekt "Controller2" ein.
5. Weisen Sie den IO-Controller "Controller2" dem IO-Gerät zu.



6. Speichern Sie das Projekt.

Beide Projekte haben jetzt ein identisch strukturiertes IO-Gerät, das im nächsten Schritt für die unterschiedlichen Arten des IO-Controllerzugriffs konfiguriert werden muss.

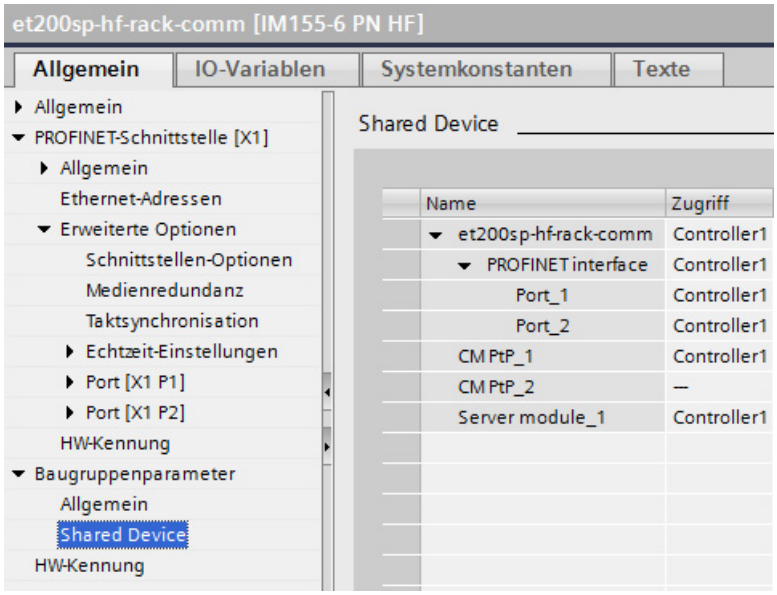
Vorgehensweise: Zugriff auf das Shared Device konfigurieren

Die Module und Submodule, die Sie in das Shared Device einfügen, werden automatisch der lokalen CPU zugewiesen. Um die Zuweisung zu ändern, gehen Sie wie folgt vor:

1. Wählen Sie in der Netz- oder Gerätesicht von Projekt "Controller1" das Schnittstellenmodul aus.
2. Wählen Sie den Bereich "Shared Device" im Inspektorfenster aus.

   Eine Tabelle zeigt für alle konfigurierten Module, welche CPU auf die Module und Submodule Zugriff hat. Die Standardeinstellung ist, dass die lokale CPU auf alle Module und Submodule Zugriff hat.
3. Behalten Sie die Einstellung "Controller1" für alle Module und Submodule bei, die im Adressbereich der lokalen CPU verbleiben sollen.

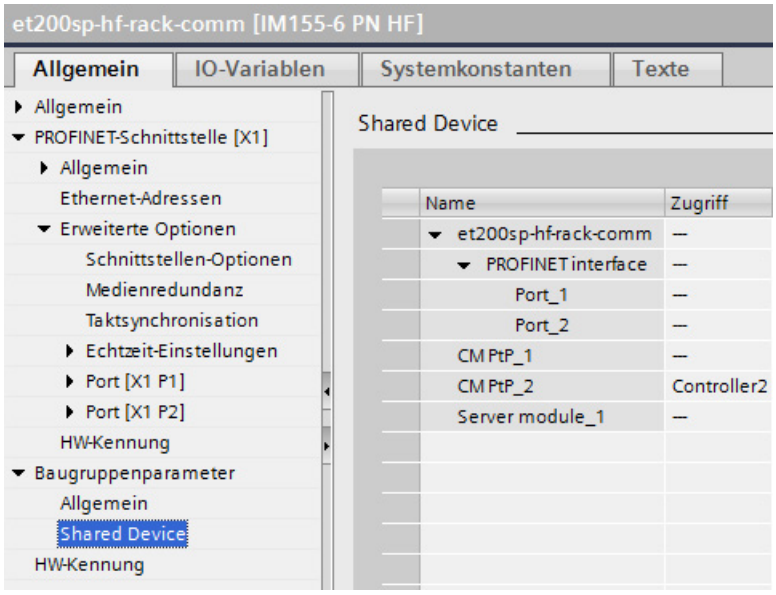
   Wählen Sie die Einstellung "---" für alle Module und Submodule aus, die sich im Adressbereich der CPU aus dem Projekt "Controller2" befinden sollen. Dies bedeutet, dass ein IO-Controller außerhalb des Projekts Zugriff auf das Modul bzw. Submodul erhält.



4. Wählen Sie in der Netz- oder Gerätesicht von Projekt "Controller2" das Schnittstellenmodul aus.
5. Wählen Sie den Bereich "Shared Device" im Inspektorfenster aus.

   Eine Tabelle zeigt für alle konfigurierten Module, welche CPU auf die Module und Submodule Zugriff hat.

6. Wählen Sie die Einstellung "---" für alle Module und Submodule, die sich im Adressbereich der CPU aus dem Projekt "Controller1" befinden sollen.



7. Prüfen Sie schließlich, ob sich die Einstellungen für den Zugriff auf die einzelnen Module und Submodule in beiden Projekten ergänzen. Damit ist gemeint, dass, wenn die lokale CPU in einem Projekt Zugriff hat, im anderen Projekt die Option "---" eingestellt sein muss und umgekehrt.

   Hinweis: Die Option "---" für die PROFINET-Schnittstelle und somit für die Ports bewirkt, dass die zugehörigen Parameter schreibgeschützt sind und sich nicht ändern lassen. Parameter der PROFINET-Schnittstelle und Portparameter können nur in dem Projekt bearbeitet werden, in dem die PROFINET-Schnittstelle der lokalen CPU zugewiesen ist. Die Ports können unabhängig hiervon in beiden Projekten miteinander verschaltet sein.

8. Prüfen Sie, ob in allen Projekten die gleichen IP-Adressparameter und der gleiche Gerätename für das Shared Device festgelegt sind.

   Prüfen Sie, ob in allen Projekten für das Subnetz, mit dem das Shared Device verbunden ist, die gleiche S7-Subnetz-ID eingestellt ist (Subnetzeigenschaften, Bereich "Allgemein" im Inspektorfenster).

Hinweis

Wenn Sie Änderungen am Shared Device vornehmen: Nehmen Sie die gleichen Änderungen in jedem Projekt auf dem Shared Device vor. Achten Sie darauf, dass nur genau ein IO-Controller Zugriff auf ein Modul oder Submodul hat.

Vorgehensweise: Echtzeiteinstellungen anpassen

Um zu gewährleisten, dass alle IO-Controller und Shared Devices mit dem passenden Sendetakt betrieben werden und dass die Aktualisierungszeiten anhand der Kommunikationslast richtig berechnet werden, müssen Sie die folgenden Einstellungen prüfen und anpassen:

1. Wählen Sie das Projekt aus, dessen IO-Controller Zugriff auf die PROFINET-Schnittstelle und die Ports des Shared Device hat.
2. Wählen Sie in der Netzsicht das Schnittstellenmodul des Shared Device aus.
3. Navigieren Sie im Inspektorfenster zum Bereich "PROFINET-Schnittstelle > Erweiterte Optionen > Echtzeit-Einstellungen > IO-Zyklus".
4. Legen Sie im Bereich "Shared Device" die Anzahl der projektexternen IO-Controller fest. Die maximale Anzahl ist vom IO-Gerät abhängig (in der GSD-Datei angegeben).
5. Sie müssen für jeden IO-Controller, der Zugriff auf die Module und Submodule des Shared Device hat, den gleichen Sendetakt einstellen:

- Wenn Sie den IO-Controller mit STEP 7 (TIA Portal) konfigurieren:
  - Öffnen Sie das entsprechende Projekt.
  - Wählen Sie die PROFINET-Schnittstelle des IO-Controllers aus.
  - Wählen Sie den Bereich "Erweiterte Optionen > Echtzeit-Einstellungen > IO-Kommunikation" im Inspektorfenster aus und legen Sie den Sendetakt fest.
- Wenn Sie den IO-Controller mit einem anderen Engineering-Tool konfigurieren:
  - Wählen Sie die PROFINET-Schnittstelle des Shared Device in STEP 7 (TIA Portal) aus und lesen Sie den Sendetakt des Shared Device aus (Bereich "Erweiterte Optionen > Echtzeit-Einstellungen").
  - Geben Sie den ausgelesenen Sendetakt in das Engineering-Tool ein.

Hinweis

Wenn Sie alle IO-Controller, die Zugriff auf das Shared Device haben, in STEP 7 (TIA Portal) konfigurieren, können Sie kürzere Sendetakte auf dem IO-Controller einstellen, als vom Shared Device unterstützt werden (Sendetaktanpassung).

Übersetzen und Laden

Sie müssen die Konfigurationen für die verschiedenen IO-Controller übersetzen und nacheinander in die CPUs laden.

Aufgrund der dezentralen Konfiguration bei getrennten Projekten gibt STEP 7 bei falscher Zugriffsparametrierung keine Konsistenzfehler aus. Hier einige Beispiele für falsche Zugriffsparametrierung:

- Mehrere IO-Controller haben Zugriff auf das gleiche Modul
- IP-Adressparameter oder Sendetakte sind nicht identisch

Diese Fehler werden erst im Controllerbetrieb gemeldet und als Konfigurationsfehler ausgegeben.

11.2.14.3 Beispiel: Ein I-Device als Shared Device konfigurieren

In diesem Beispiel wird beschrieben, wie Sie eine S7-1200 mit STEP 7 ab Version V13 SP1 als I-Device konfigurieren und sie dann in zwei Projekten als Shared Device einsetzen.

Eine "dezentrale" Konfiguration mit unterschiedlichen Engineering-Tools für unterschiedliche IO-Controllerfamilien ist möglich. Die nachfolgend beschriebene Vorgehensweise basiert auf STEP 7 V13 SP1 und ist begrenzt auf eine Konfiguration mit zwei IO-Controllern der S7-1200-Familie, die die Übertragungsbereiche eines I-Device als Shared Device gemeinsam nutzen. Das I-Device selbst ist eine CPU 1215C.

Im Beispiel werden drei Projekte mit jeweils einem IO-Controller angelegt:

- S7-1200-I-Device
- Controller1
- Controller2

Mit dem Projekt S7-1200-I-Device konfigurieren Sie das I-Device. Sie verwenden die PROFINET GSD-Variante von S7-1200-I-Device in den Projekten Controller1 und Controller2, um die Übertragungsbereiche in dem jeweiligen übergeordneten IO-Controller zuzuweisen.

Konzept des Shared I-Device

Für das Konzept des Shared I-Device werden mindestens drei getrennte Projekte benötigt:

- I-Device-Projekt: Sie konfigurieren und programmieren ein I-Device, um eine bestimmte Automatisierungsaufgabe durchzuführen. Sie definieren Übertragungsbereiche als die E/A-Schnittstelle für die übergeordneten Controller und weisen diese Übertragungsbereiche unterschiedlichen IO-Controllern zu. Für die Verbindung zu übergeordneten IO-Controllern stellen Sie eine PROFINET GSD-Datei bereit und verwenden die Übertragungsbereiche für den Zugriff auf das I-Device.
- Controller, die das I-Device gemeinsam nutzen (zwei Projekte): Sie verwenden das I-Device als eine PROFINET GSD-Variante bei der Konfiguration des PROFINET IO-Systems und geben dabei die E/A-Adressen an, unter denen die IO-Controller auf die Übertragungsbereiche zugreifen.

I-Device

Sie weisen die folgenden Parameter für eine S7-1200 CPU als I-Device zu:

- Zentrale und dezentrale Peripherie
- Gewünschte Übertragungsbereiche
- Anzahl der IO-Controller mit Zugriff auf dieses I-Device (bei einem Shared Device immer größer als 1)

Hinweis

Sie konfigurieren das I-Device ohne einen übergeordneten IO-Controller. Das Ergebnis ist, dass Sie nur die lokalen E/A-Adressen des Übertragungsbereichs (= "Adresse im I-Device") verwenden können, um das Anwenderprogramm zur Bearbeitung der Adressen aus dem Übertragungsbereich anzulegen. Sie laden das I-Device, vollständig konfiguriert mit Ausnahme der Verbindung zum übergeordneten IO-Controller, in die S7-1200 CPU.

Sie exportieren eine PROFINET GSD-Datei aus der I-Device-Konfiguration.

Controller, die das I-Device gemeinsam nutzen

Sie müssen die aus der I-Device-Konfiguration erstellte PROFINET GSD-Datei in alle Engineering-Systeme laden, die Sie bei der Konfiguration eines PROFINET IO-Systems mit diesem Shared I-Device verwenden. Wenn Sie alle Verwendungen dieses I-Device mit STEP 7 V13 SP1 konfigurieren, ist es ausreichend, die GSD-Datei in STEP 7 zu installieren.

Sie konfigurieren das I-Device als GSD-Variante auf dem PROFINET IO-System in den beteiligten Projekten. In STEP 7 V13 SP1 finden Sie dieses I-Device nach der Installation unter "Weitere Feldgeräte > PROFINET IO > PLCs & CPs".

In jedem der beteiligten Projekte weisen Sie ausschließlich den übergeordneten IO-Controllern Übertragungsbereiche zu (Standardeinstellung: alle). Für die anderen Übertragungsbereiche stellen Sie "---" ein (nicht zugewiesen). Hierdurch kann der lokale IO-Controller nicht auf diesen Übertragungsbereich zugreifen und Sie können den Übertragungsbereich einem anderen IO-Controller in einem anderen Projekt zuweisen.

Voraussetzungen

- STEP 7 ab V13 SP1
- Unterstützung der Shared-Device-Funktion durch das IO-Gerät (z. B. ET 200SP IM 155-6 PN HF V3.1).
- GSD-Datei für die Konfiguration des IO-Geräts als Shared Device ist installiert.

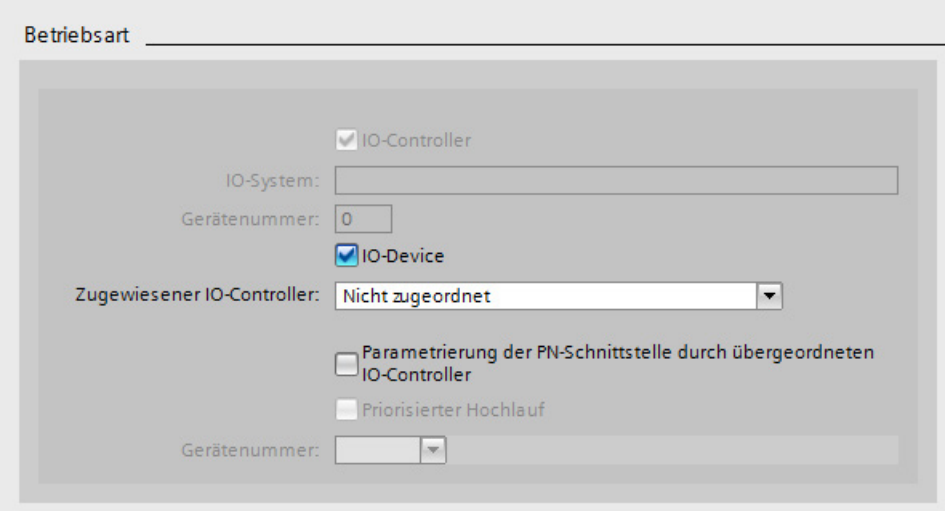
Vorgehensweise: Projekt S7-1200-I-Device anlegen

Um das Projekt mit einem Shared I-Device anzulegen, gehen Sie wie folgt vor:

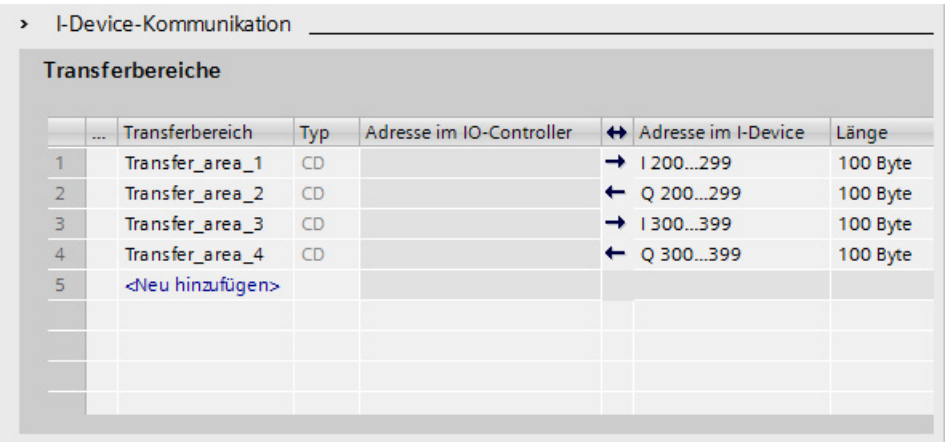
1. Starten Sie STEP 7.
2. Legen Sie ein neues Projekt mit dem Namen "S7-1200-I-Device" an.
3. Fügen Sie eine CPU 1215C aus dem Hardwarekatalog in die Netzsicht ein. Weisen Sie den Namen "S7-1200-I-Device" zu.

4. Doppelklicken Sie auf das IO-Gerät und konfigurieren Sie alle erforderlichen Module und Submodule.

5. Weisen Sie die Modulparameter zu. Insbesondere müssen Sie die folgenden Einstellungen für die CPU im Bereich der PROFINET-Schnittstelle [X1] konfigurieren:
   - Aktivieren Sie die Option "IO-Device" im Bereich "Betriebsart".

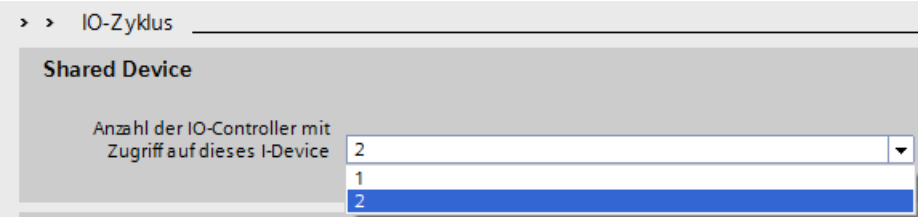


   - Konfigurieren Sie die Übertragungsbereiche im Bereich "Betriebsart > I-Device-Konfiguration". Die Spalte "Adresse im IO-Controller" bleibt leer, weil kein IO-Controller zugewiesen ist.

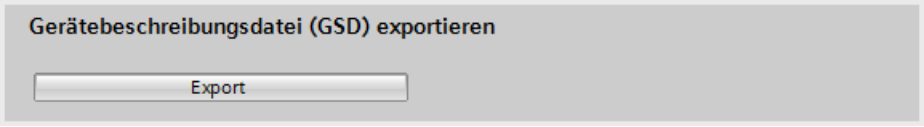


   Hinweis: Um einen Eingangsbereich in einen Ausgangsbereich zu ändern oder umgekehrt, müssen Sie in den Bereich des entsprechenden Übertragungsbereichs navigieren.

   - Wählen Sie die Anzahl der IO-Controller (mindestens zwei), die im Betrieb auf das Shared I-Device zugreifen (Bereich "Betriebsart > Echtzeit-Einstellungen", Bereich "Shared Device").



6. Speichern Sie das Projekt.
7. Klicken Sie auf die Schaltfläche "Exportieren" (Bereich "Betriebsart > I-Device-Konfiguration", Abschnitt "Allgemeine Gerätebeschreibungsdatei (GSD) exportieren"). Wenn Sie den Namen im Exportdialog nicht ändern, erhält die GSD-Datei einen Namen im zugewiesenen Format (zum Beispiel "GSDML-V2.31-#Siemens-PreConf_S7-1200-I-Device-20130925-123456").



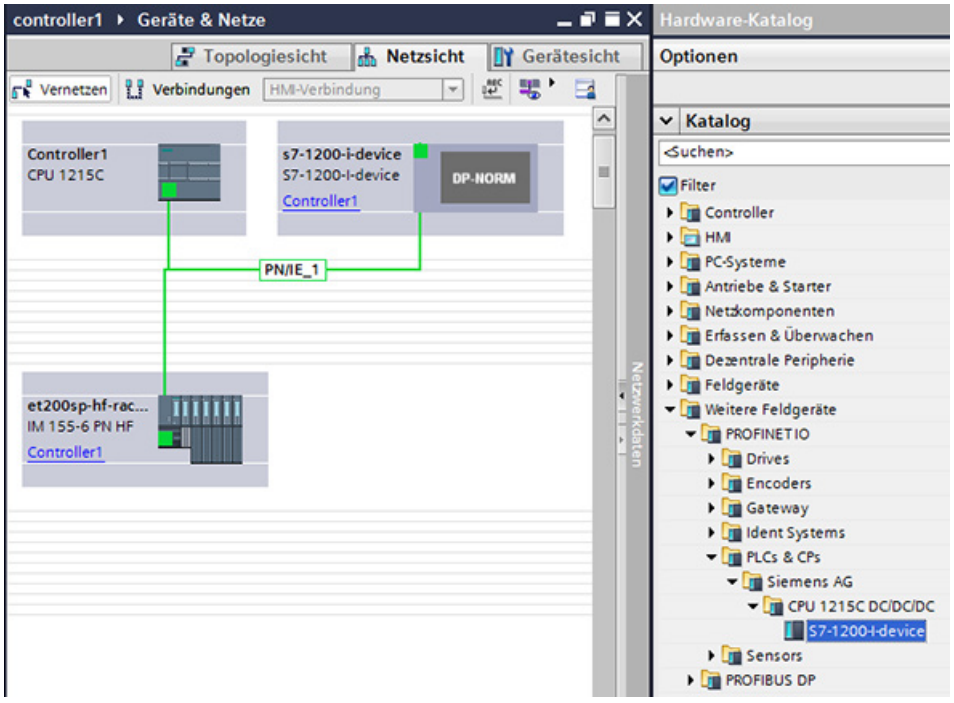
Vorgehensweise: Projekt Controller1 anlegen

Um das erste Projekt mit einem Shared I-Device anzulegen, gehen Sie wie folgt vor:

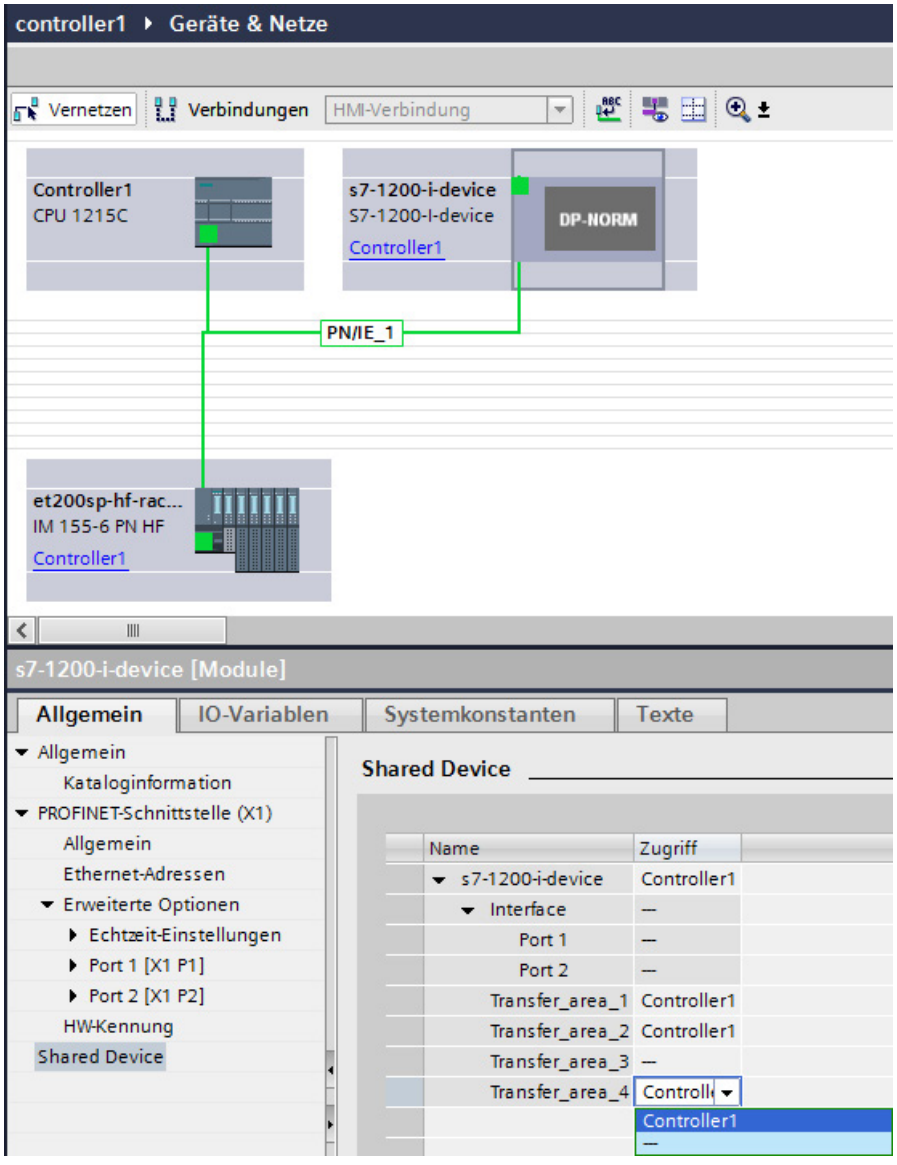
1. Starten Sie STEP 7.
2. Installieren Sie die aus der I-Device-CPU exportierte PROFINET GSD-Datei (S7-1200-I-Device).

3. Legen Sie ein neues Projekt mit dem Namen "Controller1" an.
4. Fügen Sie die CPU 1215C in die Netzsicht ein. Der Name der CPU müsste "Controller1" lauten.
5. Fügen Sie das I-Device aus dem Hardwarekatalog ein (Hardwarekatalog: Weitere Feldgeräte > PROFINET IO > PLCs & CPs).

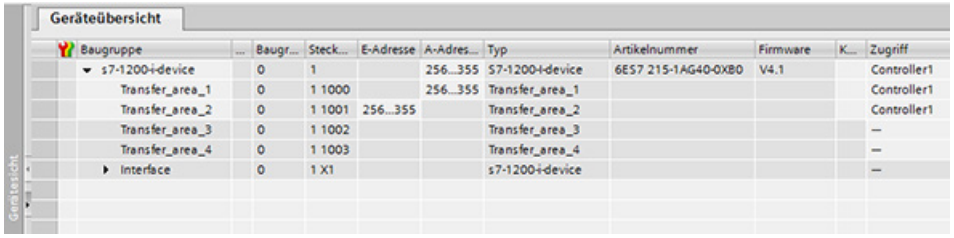
6. Weisen Sie den IO-Controller "Controller1" dem I-Device zu.



7. Wählen Sie den Bereich "Shared Device" in den Eigenschaften des I-Device aus:
   - In der Tabelle sind alle Übertragungsbereiche und die PROFINET-Schnittstelle dem lokalen IO-Controller zugewiesen (Controller1).
   - Definieren Sie die Übertragungsbereiche, auf die die Controller1-CPU **keinen** Zugriff haben soll. Wählen Sie für diese Bereiche den Eintrag "---" aus. Diese Übertragungsbereiche werden Controller2 zur Verfügung gestellt.



8. Sie können die Adressen aus der Gerätesicht des IO-Controllers in der Geräteübersicht anpassen. Um die Geräteübersicht aufzurufen, doppelklicken Sie auf das I-Device.



9. Speichern Sie das Projekt.

Vorgehensweise - Projekt Controller2 anlegen

Um das zweite Projekt mit einem Shared Device anzulegen, gehen Sie wie folgt vor:

1. Starten Sie STEP 7 erneut.

   Eine neue Instanz von STEP 7 wird geöffnet.

2. Legen Sie in der neuen Instanz ein neues Projekt mit dem Namen "Controller2" an.
3. Fügen Sie die CPU 1215C in die Netzsicht ein. Weisen Sie den Namen "Controller2" zu.
4. Fügen Sie das I-Device aus dem Hardwarekatalog ein (Hardwarekatalog: Weitere Feldgeräte > PROFINET IO > PLCs & CPs).
5. Weisen Sie den IO-Controller "Controller2" dem I-Device zu.
6. Passen Sie den Zugriff auf die Übertragungsbereiche im Projekt Controller1 an. Stellen Sie sicher, dass keine doppelten Zuweisungen zustande kommen.
7. Passen Sie die Parameter des Subnetzes und der PROFINET-Schnittstelle an. Weil das Shared I-Device dasselbe Gerät in verschiedenen Projekten verwendet, müssen diese Daten übereinstimmen.
8. Speichern Sie das Projekt.

Beide Projekte haben jetzt ein identisch konfiguriertes Shared I-Device. Der IO-Controllerzugriff und die Parameter der PROFINET-Schnittstelle müssen während des nächsten Schritts trotzdem in den verschiedenen Projekten geprüft werden.

Zusammenfassung - Parametrierung des Zugriffs auf das Shared Device

Die Übertragungsbereiche werden automatisch dem lokalen IO-Controller zugewiesen. Um die Zuweisung zu ändern, gehen Sie wie folgt vor:

1. Klicken Sie in der Netzsicht des Projekts "Controller1" auf das Gerät "S7-1200-I-Device" und wählen Sie den Bereich "Shared Device" aus.
2. Eine Tabelle zeigt an, welche CPU auf die einzelnen konfigurierten Übertragungsbereiche Zugriff hat. Die Standardeinstellung ist, dass die lokale CPU auf alle Module und Submodule Zugriff hat.
3. Behalten Sie die Einstellung "Controller1" für alle Übertragungsbereiche bei, die im Adressbereich der lokalen CPU verbleiben sollen.

   Wählen Sie die Einstellung "---" für alle Übertragungsbereiche aus, die sich im Adressbereich der CPU "Controller2" aus dem Projekt "Controller2" befinden sollen. Dies bedeutet, dass ein IO-Controller außerhalb des Projekts Zugriff auf den Übertragungsbereich erhält.
4. Gehen Sie für die übrigen Projekte genauso vor.
5. Prüfen Sie schließlich, ob sich die Einstellungen für den Zugriff auf die einzelnen Module und Submodule in beiden Projekten ergänzen. Damit ist gemeint, dass, wenn die lokale CPU in einem Projekt Zugriff hat, im anderen Projekt die Option "---" eingestellt sein muss und umgekehrt.

   Hinweis: Die Option "---" für die PROFINET-Schnittstelle und somit für die Ports bewirkt, dass die zugehörigen Parameter schreibgeschützt sind und sich nicht ändern lassen. Parameter der PROFINET-Schnittstelle und Portparameter können nur in dem Projekt bearbeitet werden, in dem die PROFINET-Schnittstelle der lokalen CPU zugewiesen ist. Die Ports können unabhängig hiervon in beiden Projekten miteinander verschaltet sein.
6. Prüfen Sie, ob in allen Projekten die gleichen IP-Adressparameter und der gleiche Gerätename für das Shared Device festgelegt sind.

   Prüfen Sie, ob in allen Projekten für das Subnetz, mit dem das Shared Device verbunden ist, die gleiche S7-Subnetz-ID eingestellt ist (Subnetzeigenschaften, Bereich "Allgemein" im Inspektorfenster).

Hinweis

Wenn Sie Änderungen am I-Device vornehmen (zum Beispiel die Anzahl oder Länge der Übertragungsbereiche ändern), exportieren Sie das I-Device erneut als GSD-Datei. Installieren Sie die GSD-Datei in jedem Projekt neu, in dem das I-Device als Shared Device verwendet wird. Achten Sie darauf, dass nur genau ein IO-Controller Zugriff auf einen Übertragungsbereich hat.

Vorgehensweise - Echtzeiteinstellungen anpassen

Um zu gewährleisten, dass alle IO-Controller und Shared Devices mit dem passenden Sendetakt betrieben werden und dass die Aktualisierungszeiten anhand der Kommunikationslast richtig berechnet werden, müssen Sie die folgenden Einstellungen prüfen und anpassen:

1. Sie müssen für jeden IO-Controller, der Zugriff auf die Module und Submodule des Shared Device hat, den gleichen Sendetakt einstellen:

- Wenn Sie den IO-Controller mit STEP 7 (TIA Portal) konfigurieren, gehen Sie wie folgt vor:
  - Öffnen Sie das entsprechende Projekt.
  - Wählen Sie die PROFINET-Schnittstelle des IO-Controllers aus.
  - Wählen Sie den Bereich "Erweiterte Optionen > Echtzeit-Einstellungen > IO-Kommunikation" im Inspektorfenster aus und legen Sie den Sendetakt fest.
- Wenn Sie den IO-Controller mit einem anderen Engineering-Tool konfigurieren, gehen Sie wie folgt vor:
  - Wählen Sie die PROFINET-Schnittstelle des Shared Device in STEP 7 (TIA Portal) aus und lesen Sie den Sendetakt des Shared Device aus (Bereich "Erweiterte Optionen > Echtzeit-Einstellungen").
  - Geben Sie den ausgelesenen Sendetakt in das Engineering-Tool ein.

Hinweis

Wenn Sie **alle** IO-Controller, die Zugriff auf das Shared I-Device haben, in STEP 7 (TIA Portal) konfigurieren, können Sie kürzere Sendetakte auf dem IO-Controller einstellen, als vom Shared Device unterstützt werden (Sendetaktanpassung).

Übersetzen und Laden

Sie müssen die Konfigurationen für die verschiedenen IO-Controller übersetzen und nacheinander in die CPUs laden.

Aufgrund der dezentralen Konfiguration bei getrennten Projekten gibt STEP 7 bei falscher Zugriffsparametrierung keine Konsistenzfehler aus. Hier einige Beispiele für falsche Zugriffsparametrierung:

- Mehrere IO-Controller haben Zugriff auf das gleiche Modul.
- IP-Adressparameter oder Sendetakte sind nicht identisch.

Diese Fehler werden erst im Controllerbetrieb gemeldet und als Konfigurationsfehler ausgegeben.

11.2.15 Diagnose

Informationen dazu, wie Sie Organisationsbausteine (OBs) für die Diagnose mit diesen Kommunikationsnetzwerken einsetzen, finden Sie unter "Organisationsbausteine (OBs)" (Seite 90).

11.2.16 Anweisungen für die dezentrale Peripherie

Weitere Informationen zur Verwendung der Anweisungen für die dezentrale Peripherie mit diesen Kommunikationsnetzwerken finden Sie unter "Dezentrale Peripherie (PROFINET, PROFIBUS oder AS-i)" (Seite 356).

11.2.17 Diagnoseanweisungen

Weitere Informationen zur Verwendung dieser Anweisungen mit diesen Kommunikationsnetzwerken finden Sie unter "Diagnose (PROFINET oder PROFIBUS): Diagnoseanweisungen" (Seite 390).

11.2.18 Diagnoseereignisse für die dezentrale Peripherie

Weitere Informationen zur Verwendung dieser Diagnoseinformationen mit diesen Kommunikationsnetzwerken finden Sie unter "Diagnose (PROFINET oder PROFIBUS): Diagnoseereignisse für die dezentrale Peripherie" (Seite 390).

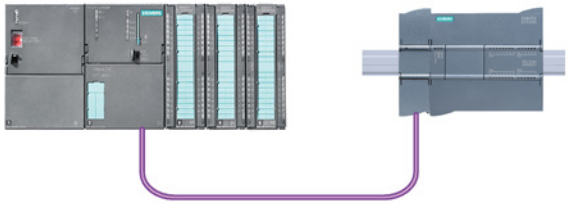
11.3 PROFIBUS

Ein PROFIBUS-System nutzt einen Bus-Master, um Slavegeräte abzufragen, die an mehreren Stellen auf einem seriellen RS485-Bus verteilt sind. Ein PROFIBUS-Slave ist ein beliebiges Peripheriegerät (E/A-Wandler, Ventil, Motorantrieb oder Messgerät), das Daten verarbeitet und die Ausgabe an den Master sendet. Der Slave stellt eine passive Station im Netzwerk dar, weil er keine Buszugriffsrechte besitzt. Er kann lediglich empfangene Meldungen quittieren oder auf Anforderung Antwortmeldungen an den Master senden. Alle PROFIBUS-Slaves haben die gleiche Priorität und die gesamte Netzwerkkommunikation stammt vom Master.

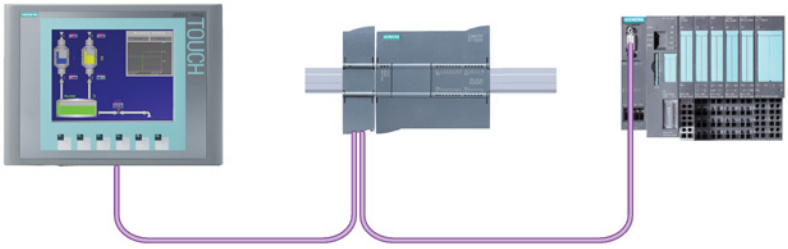
Ein PROFIBUS-Master stellt eine "aktive Station" im Netzwerk dar. PROFIBUS DP definiert zwei Klassen von Mastern. Ein Master der Klasse 1 (normalerweise eine zentrale programmierbare Steuerung (PLC) oder ein PC mit Spezialsoftware) bearbeitet die normale Kommunikation bzw. den Datenaustausch mit den ihm zugewiesenen Slaves. Ein Master der Klasse 2 (üblicherweise ein Konfigurationsgerät, z. B ein Laptop oder eine Programmierkonsole für Inbetriebnahme, Wartung und Diagnose) ist ein Sondergerät, das hauptsächlich für die Inbetriebnahme von Slaves und zu Diagnosezwecken eingesetzt wird.

Die S7-1200 ist mit dem Kommunikationsmodul CM 1242-5 als DP-Slave an ein PROFIBUS-Netzwerk angeschlossen. Das CM 1242-5 (DP-Slave) kann der Kommunikationspartner von DP-Mastern V0/V1 sein. Wenn Sie das Modul in einem Fremdsystem konfigurieren möchten, finden Sie auf der im Lieferumfang des Moduls enthaltenen CD sowie auf den Seiten des Siemens Automation-Kundensupports (http://support.automation.siemens.com/WW/llisapi.dll?func=cslib.csinfo&lang=en&objid=6GK72425DX300XE0&caller=view) im Internet eine GSD-Datei für das CM 1242-5 (DP-Slave).

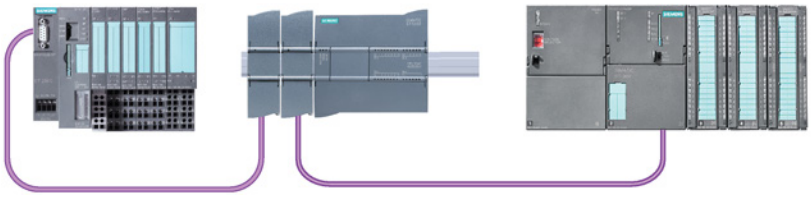
In der folgenden Abbildung fungiert die S7-1200 als DP-Slave einer S7-300 Steuerung:



Die S7-1200 ist mit dem Kommunikationsmodul CM 1243-5 als DP-Master an ein PROFIBUS-Netzwerk angeschlossen. Das CM 1243-5 (DP-Master) kann der Kommunikationspartner von DP-Slaves V0/V1 sein. In der folgenden Abbildung ist die S7-1200 ein Master und steuert einen ET200S DP-Slave:



Wenn ein CM 1242-5 und ein CM 1243-5 zusammen installiert sind, kann eine S7-1200 gleichzeitig sowohl als Slave eines übergeordneten DP-Mastersystems als auch als Master eines untergeordneten DP-Slavesystems fungieren.



In V4.0 können Sie maximal drei PROFIBUS-CMs je Station konfigurieren, wobei es sich um eine beliebige Kombination aus DP-Master- oder DP-Slave-CMs handeln kann. DP-Master in einer Implementierung mit einer CPU ab Firmware-Version 3.0 können jeweils maximal 32 Slaves steuern.

Die Konfigurationsdaten der PROFIBUS-CMs werden in der lokalen CPU gespeichert. Dadurch ist bei Bedarf der einfache Austausch dieser Kommunikationsmodule möglich.

Um PROFIBUS mit S7-1200 CPUs der Version 4.0 einzusetzen, müssen Sie die PROFIBUS Master CM-Firmware auf V1.3 hochrüsten.

Dieses Upgrade können Sie mithilfe einer Secure Digital (SD)-Karte durchführen.

Hinweis

Es ist empfehlenswert, die PROFIBUS CM-Firmware stets auf dem neuesten Stand zu halten (http://support.automation.siemens.com/WW/view/de/42131407). Die aktuelle Firmware finden Sie auf der Siemens-Website "Service & Support".

11.3.1 Kommunikationsdienste der PROFIBUS-CMs

Die PROFIBUS-CMs nutzen das PROFIBUS DP-V1-Protokoll.

Kommunikationstypen für DP-V1

Für DP-V1 stehen die folgenden Kommunikationstypen zur Verfügung:

- Zyklische Kommunikation (CM 1242-5 und CM 1243-5)

  Beide PROFIBUS-Module unterstützen die zyklische Kommunikation für die Prozessdatenübertragung zwischen DP-Slave und DP-Master.

  Die zyklische Kommunikation wird vom Betriebssystem der CPU bearbeitet. Hierfür sind keine Softwarebausteine erforderlich. Die E/A-Daten werden direkt aus dem/in das Prozessabbild der CPU gelesen bzw. geschrieben.

- Azyklische Kommunikation (nur CM 1243-5)

  Der DP-Master unterstützt auch die azyklische Kommunikation mit Softwarebausteinen:

  – Die Anweisung "RALRM" ist für die Alarmbearbeitung verfügbar.

  – Die Anweisungen "RDREC" und "WRREC" sind für die Übertragung von Konfigurations- und Diagnosedaten verfügbar.

Vom CM 1243-5 nicht unterstützte Funktionen: SYNC/FREEZE und Get_Master_Diag

Weitere Kommunikationsdienste des CM 1243-5

Der CM 1243-5 DP-Master unterstützt die folgenden zusätzlichen Kommunikationsdienste:

- S7-Kommunikation

  – PUT/GET-Dienste

    Der DP-Master fungiert als Client und Server für Abfragen anderer S7-Steuerungen oder PCs über PROFIBUS.

  – PG/OP-Kommunikation

    Mit den PG-Funktionen können Konfigurationsdaten und Anwenderprogramme von einem PG heruntergeladen und Diagnosedaten in ein PG geladen werden.

    Mögliche Kommunikationspartner für die OP-Kommunikation sind HMI-Panels, SIMATIC Panel-PCs mit WinCC flexible oder SCADA-Systeme, die die S7-Kommunikation unterstützen.

11.3.2 Verweis auf die Benutzerhandbücher für PROFIBUS-CMs

Weitere Informationen

Ausführliche Informationen über die PROFIBUS-CMs finden Sie in den Handbüchern zu den jeweiligen Geräten. Diese stehen im Internet auf den Seiten des Kundensupports von Siemens Industrial Automation unter den folgenden IDs zur Verfügung:

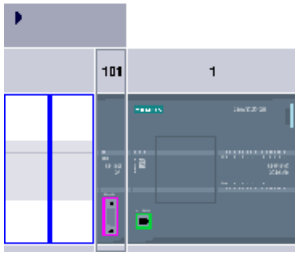
- CM 1242-5 (http://support.automation.siemens.com/WW/view/en/44632650)
- CM 1243-5 (http://support.automation.siemens.com/WW/view/en/44632657)

11.3.3 DP-Master und -Slave konfigurieren

11.3.3.1 CM 1243-5 (DP-Master) und DP-Slave hinzufügen

Verwenden Sie im Portal "Geräte & Netze" den Hardwarekatalog, um PROFIBUS-Module zur CPU hinzuzufügen. Diese Module werden an der linken Seite der CPU angeschlossen. Um ein Modul in die Hardwarekonfiguration einzufügen, selektieren Sie das Modul im Hardwarekatalog und doppelklicken, oder Sie ziehen es in den markierten Steckplatz.

Tabelle 11- 55 PROFIBUS CM 1243-5 (DP-Master) zur Gerätekonfiguration hinzufügen

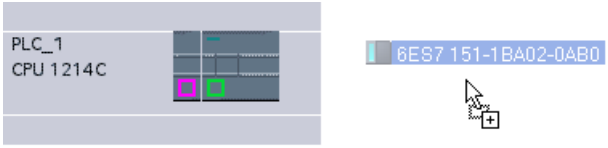
Modul	Modul auswählen	Modul einsetzen	Ergebnis
CM 1243-5 (DP-Master)			

Im Hardwarekatalog können Sie auch die DP-Slaves hinzufügen. Um beispielsweise einen ET200 S DP-Slave hinzuzufügen, erweitern Sie im Hardwarekatalog die folgenden Behälter:

- Dezentrale E/A
- ET200 S
- Schnittstellenmodule
- PROFIBUS

Wählen Sie dann "6ES7 151-1BA02-0AB0" (IM151-1 HF) in der Liste der Bestellnummern aus und fügen Sie den ET200 S DP-Slave wie in der folgenden Abbildung gezeigt ein.

Tabelle 11- 56 ET200 S DP-Slave zur Gerätekonfiguration hinzufügen

DP-Slave einfügen	Ergebnis
	

11.3.3.2 Konfigurieren der logischen Netzwerkverbindungen zwischen zwei PROFIBUS-Geräten

Nachdem Sie das Modul CM 1243-5 (DP-Master) konfiguriert haben, können Sie nun Ihre Netzwerkverbindungen konfigurieren.

Im Portal "Geräte & Netze" können Sie in der "Netzsicht" die Geräte in Ihrem Projekt vernetzen. Um eine PROFIBUS-Verbindung anzulegen, aktivieren Sie das violettfarbene Feld (PROFIBUS) am ersten Gerät. Ziehen Sie mit der Maus eine Linie zum PROFIBUS-Feld am zweiten Gerät. Lassen Sie die Maustaste los. Damit ist Ihre PROFIBUS-Verbindung hergestellt.

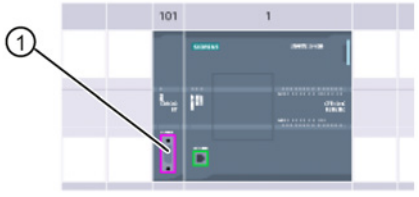
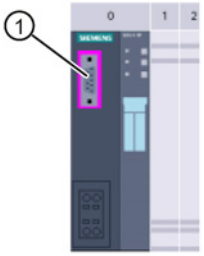
Ausführliche Informationen finden Sie unter "Gerätekonfiguration: Erstellen einer Netzwerkverbindung" (Seite 637).

11.3.3.3 PROFIBUS-Adressen zum CM 1243-5 und DP-Slave zuweisen

Konfigurieren der PROFIBUS-Schnittstelle

Nachdem Sie die logischen Netzwerkverbindungen zwischen zwei PROFIBUS-Geräten konfiguriert haben, können Sie die Parameter für die PROFIBUS-Schnittstellen konfigurieren. Hierfür klicken Sie auf das violettfarbene PROFIBUS-Feld am Modul CM 1243-5. Daraufhin zeigt das Register "Eigenschaften" im Inspektorfenster die PROFIBUS-Schnittstelle an. Die PROFIBUS-Schnittstelle des DP-Slaves wird auf dieselbe Weise konfiguriert.

Tabelle 11- 57 PROFIBUS-Schnittstellen von CM 1243-5 (DP-Master) und ET200 S DP-Slave konfigurieren

CM 1243-5 (DP-Master)	ET200 S DP-Slave
	

① PROFIBUS-Port

PROFIBUS-Adresse zuweisen

In einem PROFIBUS-Netzwerk wird jedem Gerät eine PROFIBUS-Adresse zugewiesen. Diese Adresse kann im Bereich von 0 bis 127 liegen, mit folgenden Ausnahmen:

- Adresse 0: Reserviert für die Netzwerkkonfiguration und/oder an den Bus angeschlossene Programmierwerkzeuge
- Adresse 1: Reserviert von Siemens für den ersten Master
- Adresse 126: Reserviert für Geräte im Werk, die keine Schaltereinstellung haben und über das Netzwerk neu adressiert werden müssen
- Adresse 127: Reserviert für Broadcast-Meldungen an alle Geräte im Netzwerk; darf keinen betriebsfähigen Geräten zugewiesen werden

Deshalb liegen die Adressen, die für betriebsfähige PROFIBUS-Geräte verwendet werden können, im Bereich von 2 bis 125.

Wählen Sie im Eigenschaftsfenster den Eintrag "PROFIBUS-Adresse". STEP 7 zeigt den Konfigurationsdialog für die PROFIBUS-Adresse an, in dem Sie die PROFIBUS-Adresse des Geräts zuweisen.

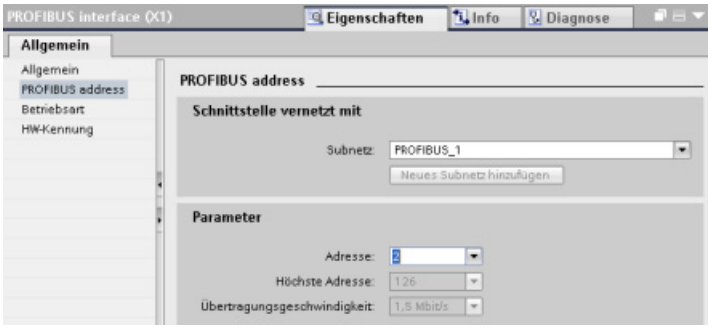


Tabelle 11- 58 Parameter für die PROFIBUS-Adresse

Parameter		Beschreibung
Subnetz	Name des Subnetzes, mit dem das Gerät verbunden ist. Um ein neues Subnetz anzulegen, klicken Sie auf Schaltfläche "Neues Subnetz hinzufügen". Die Voreinstellung ist "nicht verbunden". Zwei Arten von Verbindungen sind möglich: • Die Voreinstellung "nicht verbunden" stellt eine lokale Verbindung her. • Wenn Ihr Netzwerk über zwei oder mehr Geräte verfügt, ist ein Subnetz erforderlich.	
Parameter	Adresse	Dem Gerät zugewiesene PROFIBUS-Adresse
	Höchste Adresse	Die höchste PROFIBUS-Adresse basiert auf den aktiven Stationen auf dem PROFIBUS (z. B. DP-Master). Passive DP-Slaves haben unabhängige PROFIBUS-Adressen von 1 bis 125, auch wenn beispielsweise für die höchste PROFIBUS-Adresse 15 eingestellt ist. Die höchste PROFIBUS-Adresse ist für die Token-Weiterleitung (Weiterleitung der Senderechte) von Bedeutung, der Token wird nur an aktive Stationen weitergeleitet. Durch die Angabe der höchsten PROFIBUS-Adresse wird der Bus optimiert.
	Übertragungsgeschwindigkeit	Übertragungsgeschwindigkeit im konfigurierten PROFIBUS-Netzwerk: Die PROFIBUS-Übertragungsgeschwindigkeiten liegen zwischen 9,6 kBit/s und 12 MBit/s. Die Übertragungsgeschwindigkeit ist von den Eigenschaften der verwendeten PROFIBUS-Teilnehmer abhängig. Die Übertragungsgeschwindigkeit darf nicht größer sein als die vom langsamsten Teilnehmer unterstützte Geschwindigkeit. Die Übertragungsgeschwindigkeit wird normalerweise für den Master im PROFIBUS-Netzwerk festgelegt, wobei alle DP-Slaves automatisch die gleiche Übertragungsgeschwindigkeit nutzen (Auto-Baud).

11.3.4 Anweisungen für die dezentrale Peripherie

Weitere Informationen zur Verwendung der Anweisungen für die dezentrale Peripherie mit diesen Kommunikationsnetzwerken finden Sie unter "Dezentrale Peripherie (PROFINET, PROFIBUS oder AS-i)" (Seite 356).

11.3.5 Diagnoseanweisungen

Weitere Informationen zur Verwendung dieser Anweisungen mit diesen Kommunikationsnetzwerken finden Sie unter "Diagnose (PROFINET oder PROFIBUS): Diagnoseanweisungen" (Seite 390).

11.3.6 Diagnoseereignisse für die dezentrale Peripherie

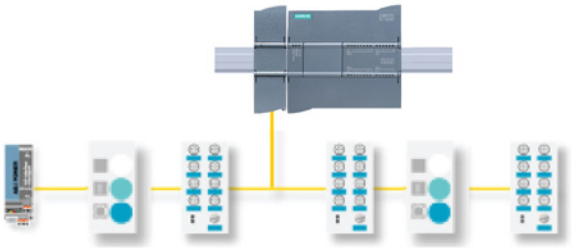
Weitere Informationen zur Verwendung dieser Diagnoseinformationen mit diesen Kommunikationsnetzwerken finden Sie unter "Diagnose (PROFINET oder PROFIBUS): Diagnoseereignisse für die dezentrale Peripherie" (Seite 390).

11.4 ASi

Das S7-1200 AS-i Master CM 1243-2 ermöglicht die Anbindung eines AS-i-Netzwerks an eine S7-1200 CPU.

Die Aktor-/Sensorschnittstelle bzw. AS-i ist ein Netzwerkverbindungssystem für einen Master auf der niedrigsten Stufe in einem Automatisierungssystem. Das CM 1243-2 dient als AS-i-Master im Netzwerk. Mithilfe eines einzigen AS-i-Kabels können Sensoren und Aktoren (AS-i-Slavegeräte) über das CM 1243-2 an die CPU angeschlossen werden. Das CM 1243-2 wickelt die gesamte AS-i-Netzwerkkoordination ab und überträgt mit Hilfe der dem CM 1243-2 zugewiesenen E/A-Adressen Daten und Statusinformationen von den Aktoren und Sensoren an die CPU. Je nach Slavetyp können Sie auf binäre oder analoge Werte zugreifen. Die AS-i-Slaves sind die Eingangs- und Ausgangskanäle des AS-i-Systems und sie sind nur aktiv, wenn sie vom CM 1243-2 aufgerufen werden.

In der folgenden Abbildung ist die S7-1200 ein AS-i-Master, der digitale/analoge Slavegeräte von AS-i-E/A-Modulen steuert.



Um AS-i mit S7-1200 CPUs der Version 4.0 einzusetzen, müssen Sie die AS-i Master CM-Firmware auf V1.1 hochrüsten.

Dies kann mit dem Webserver oder einer SIMATIC Memory Card erfolgen.

Hinweis

Bei S7-1200 CPUs der Version 4.0 müssen Sie bei Verwendung des Webservers oder einer SIMATIC Memory Card für das Upgrade der AS-i-Firmware von V1.0 auf V1.1 die AS-i-Firmware im AS-i Master CM 1243-2 wie folgt aktualisieren:

1. Laden Sie das Firmware-Upgrade ins AS-i Master CM 1243-2.
2. Schalten Sie nach beendetem Ladevorgang die S7-1200 CPU aus und wieder ein, um das Firmware-Upgrade im AS-i Master CM 1243-2 vollständig abzuschließen.
3. Wiederholen Sie die Schritte 1 und 2 für jedes weitere AS-i Master CM 1243-2. Der S7-1200 PLC gestattet maximal drei AS-i Master CM 1243-2.

Hinweis

Es ist empfehlenswert, die AS-i CM-Firmware stets auf dem neuesten Stand zu halten (http://support.automation.siemens.com/WW/view/de/43416171). Die aktuelle Firmware finden Sie auf der Siemens-Website "Service & Support".

11.4.1 AS-i-Master und -Slavegeräte konfigurieren

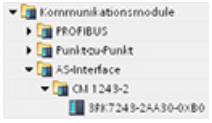
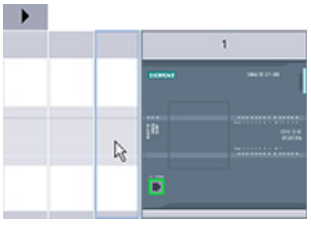
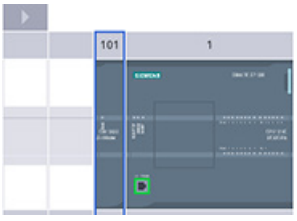
Der AS-i-Master CM 1243-2 ist als Kommunikationsmodul in das S7-1200 Automatisierungssystem integriert.

Ausführliche Informationen zum AS-i-Master CM 1243-2 finden Sie im Handbuch "AS-I Master CM 1243-2 und AS-i-Datenentkopplungsmodul DCM 1271 für SIMATIC S7-1200" (http://support.automation.siemens.com/WW/view/en/50414115/133300).

11.4.1.1 AS-i-Mastermodul CM 1243-2 und AS-i-Slave hinzufügen

Im Hardwarekatalog können Sie AS-i-Mastermodule CM1243-2 zur CPU hinzufügen. Diese Module werden an der linken Seite der CPU angeschlossen. Es können maximal drei AS-i-Mastermodule CM1243-2 verwendet werden. Um ein Modul in die Hardwarekonfiguration einzufügen, selektieren Sie das Modul im Hardwarekatalog und doppelklicken, oder Sie ziehen es in den markierten Steckplatz.

Tabelle 11- 59 AS-i-Mastermodul CM1243-2 zur Gerätekonfiguration hinzufügen

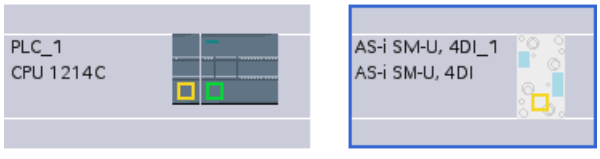
Modul	Modul auswählen	Modul einsetzen	Ergebnis
CM 1243-2 AS-i-Master			

Im Hardwarekatalog können Sie auch die AS-i-Slaves hinzufügen. Um beispielsweise einen Slave mit den Merkmalen "E/A-Modul, kompakt, digital, Eingang" hinzuzufügen, erweitern Sie im Hardwarekatalog die folgenden Behälter:

- Feldgeräte
- AS-Schnittstellen-Slaves

Wählen Sie anschließend in der Liste der Bestellnummern "3RG9 001-0AA00" (AS-i SM-U, 4DI) aus und fügen Sie den Slave mit den Merkmalen "E/A-Modul, kompakt, digital, Eingang" wie in der folgenden Abbildung gezeigt ein.

Tabelle 11- 60 AS-i-Slave zur Gerätekonfiguration hinzufügen

AS-i-Slave stecken	Ergebnis
	

11.4.1.2 Logische Netzwerkverbindungen zwischen zwei AS-i-Geräten konfigurieren

Nachdem Sie den AS-i-Master CM1243-2 konfiguriert haben, können Sie nun Ihre Netzwerkverbindungen konfigurieren.

Im Portal "Geräte & Netze" können Sie in der "Netzsicht" die Geräte in Ihrem Projekt vernetzen. Um eine AS-i-Verbindung anzulegen, aktivieren Sie das gelbe Feld (AS-i) am ersten Gerät. Ziehen Sie mit der Maus eine Linie zum AS-i-Feld am zweiten Gerät. Lassen Sie die Maustaste los. Damit ist Ihre AS-i-Verbindung hergestellt.

Weitere Informationen finden Sie unter "Gerätekonfiguration: Erstellen einer Netzwerkverbindung" (Seite 637).

11.4.1.3 Eigenschaften des AS-i-Masters CM1243-2 konfigurieren

Um Parameter für die AS-i-Schnittstelle zu konfigurieren, klicken Sie auf das gelbe AS-i-Feld am AS-i-Mastermodul CM1243-2. Daraufhin zeigt das Register "Eigenschaften" im Inspektorfenster die AS-i-Schnittstelle an.

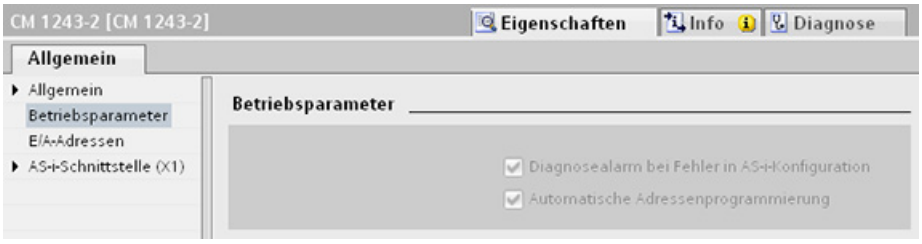
Im STEP 7-Inspektorfenster können Sie allgemeine Informationen, Adressen und Betriebsparameter anzeigen, konfigurieren und ändern:

Tabelle 11- 61 Eigenschaften des AS-i-Mastermoduls CM1243-2

Eigenschaft	Beschreibung
Allgemein	Name des AS-i-Masters CM 1243-2
Betriebsparameter	Parameter für die Antwort des AS-i-Masters
E/A-Adressen	Adressbereich für die Slave-E/A-Adressen
AS-i-Schnittstelle (X1)	Zugewiesenes AS-i-Netzwerk

Hinweis

"Diagnosealarm für Fehler in der AS-i-Konfiguration" und "Automatische Adressenprogrammierung" sind stets aktiv und deshalb grau dargestellt.



11.4.1.4 Einem AS-i-Slave eine AS-i-Adresse zuweisen

AS-i-Slaveschnittstelle konfigurieren

Um Parameter für die AS-i-Schnittstelle zu konfigurieren, klicken Sie auf das gelbe AS-i-Feld am AS-i-Slave. Daraufhin zeigt das Register "Eigenschaften" im Inspektorfenster die AS-i-Schnittstelle an.

① AS-i-Port

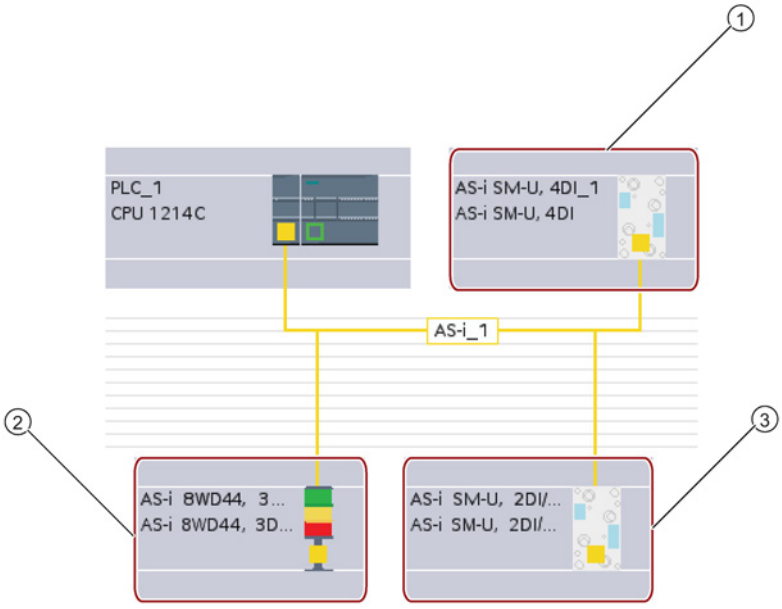
AS-i-Slaveadresse zuweisen

In einem AS-i-Netzwerk wird jedem Gerät eine AS-i-Slaveadresse zugewiesen. Diese Adresse kann im Bereich von 0 bis 31 liegen. Die Adresse 0 ist jedoch ausschließlich für neue Slavegeräte reserviert. Die Slaveadressen sind 1(A oder B) bis 31(A oder B) für insgesamt maximal 62 Slavegeräte.

"Standard"-AS-i-Geräte verwenden die gesamte Adresse und haben eine Zahlenadresse ohne den Bezeichner A oder B. AS-i-Geräte vom Typ "A/B-Knoten" verwenden jeweils den A- oder B-Anteil der Adresse, so dass jede der 31 Adressen zweimal verwendet werden kann. Der Adressbereich umfasst 1A bis 31A plus 1B bis 31B.

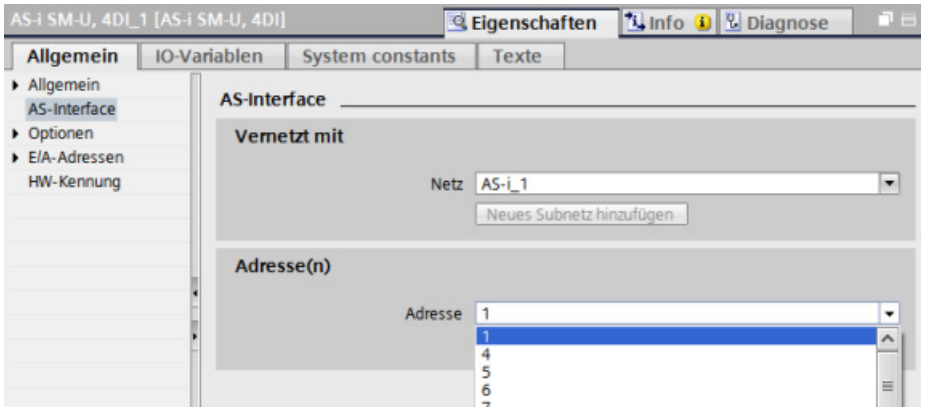
Jede Adresse im Bereich von 1 bis 31 kann einem AS-i-Slavegerät zugewiesen werden. D. h. mit anderen Worten: Es ist nicht wichtig, ob die Slaves mit Adresse 21 beginnen oder ob der erste Slave tatsächlich die Adresse 1 erhält.

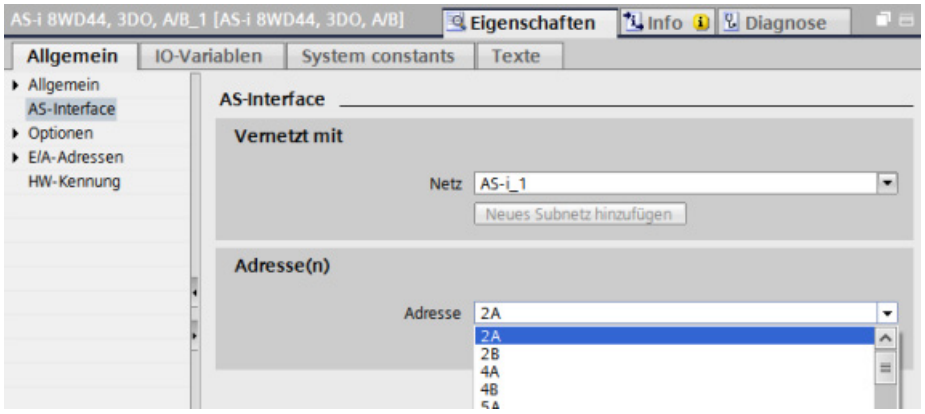
Im folgenden Beispiel wurden drei AS-i-Geräten die Adressen "1" (Standardgerät), "2A" (Gerät vom Typ A/B-Knoten) und "3" (Standardgerät) zugewiesen:



① AS-i-Slaveadresse 1; Gerät: AS-i SM-U, 4DI; Bestellnummer: 3RG9 001-0AA00
② AS-i-Slaveadresse 2A; Gerät: AS-i 8WD44, 3DO, A/B; Artikelnummer: 8WD4 428-0BD
③ AS-i-Slaveadresse 3; Gerät: AS-i SM-U, 2DI/2DO; Artikelnummer: 3RG9 001-0AC00

Geben Sie hier die AS-i-Slaveadresse ein:





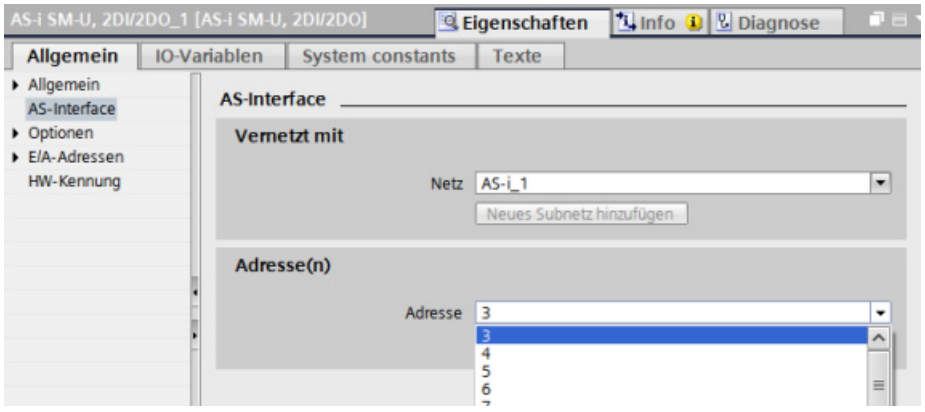


Tabelle 11- 62 Parameter für die AS-i-Schnittstelle

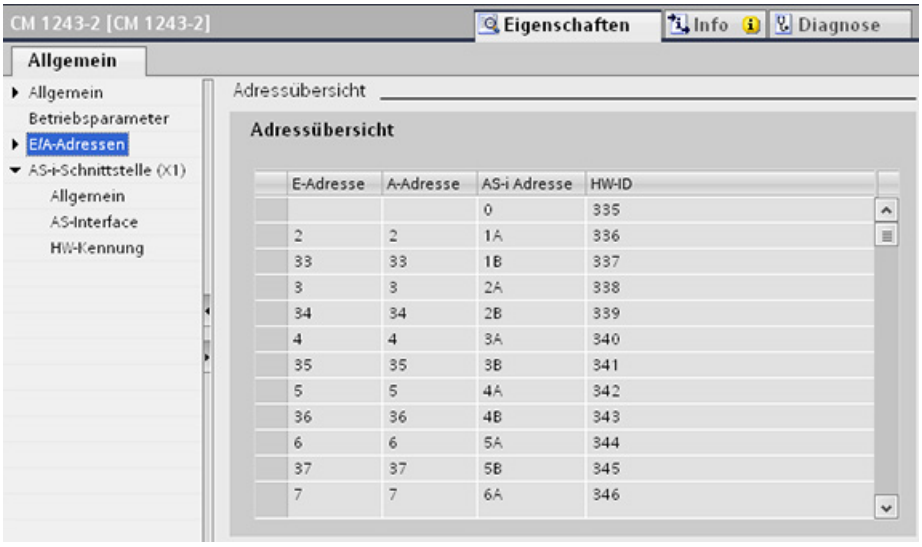
Parameter	Beschreibung
Netzwerk	Name des Netzwerks, mit dem das Gerät verbunden ist.
Adresse(n)	Zugewiesene AS-i-Adresse für das Slavegerät im Bereich von 1(A oder B) bis 31(A oder B) für insgesamt bis zu 62 Slavegeräte

11.4.2 Datenaustausch zwischen dem Anwenderprogramm und AS-i-Slaves

11.4.2.1 STEP 7 Basic konfigurieren

Der AS-i-Master reserviert einen 62 Byte großen Datenbereich im E/A-Bereich der CPU. Der Zugriff auf die Digitaldaten erfolgt hier in Byte. Für jeden Slave gibt es ein Byte Eingangs- und ein Byte Ausgangsdaten.

Die Zuweisung der AS-i-Verbindungen der digitalen AS-i-Slaves zu den Datenbits des zugewiesenen Bytes wird im Inspektorfenster des AS-i-Mastermoduls CM 1243-2 angezeigt.



Sie können auf die Daten der AS-i-Slaves im Anwenderprogramm zugreifen, indem Sie die angezeigten E/A-Adressen mit den entsprechenden Bitverknüpfungsoperationen (z. B. "UND") oder Bitzuweisungen verwenden.

Hinweis

Wenn Sie die AS-i-Slaves nicht mit STEP 7 konfigurieren, wird die "Systemzuweisung" automatisch aktiviert.

Wenn Sie keine Slaves konfigurieren, müssen Sie das AS-i-Master CM1243-2 über die tatsächliche Buskonfiguration informieren. Hierfür verwenden Sie die Online-Funktion "IST > SOLL".

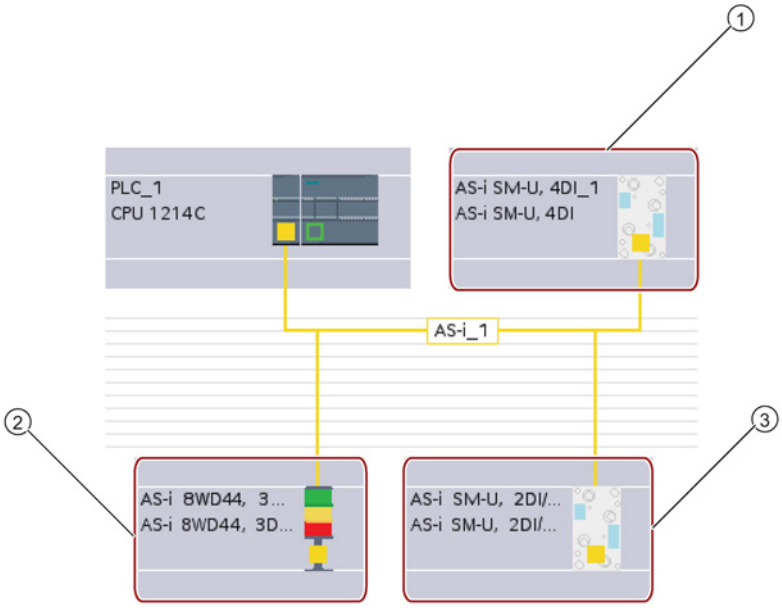
Weitere Informationen

Ausführliche Informationen zum AS-i-Master CM 1243-2 finden Sie im Handbuch "AS-I Master CM 1243-2 und AS-i-Datenentkopplungsmodul DCM 1271 für SIMATIC S7-1200" (http://support.automation.siemens.com/WW/view/en/50414115/133300).

11.4.2.2 Slaves mit STEP 7 konfigurieren

Digitale AS-i-Werte übertragen

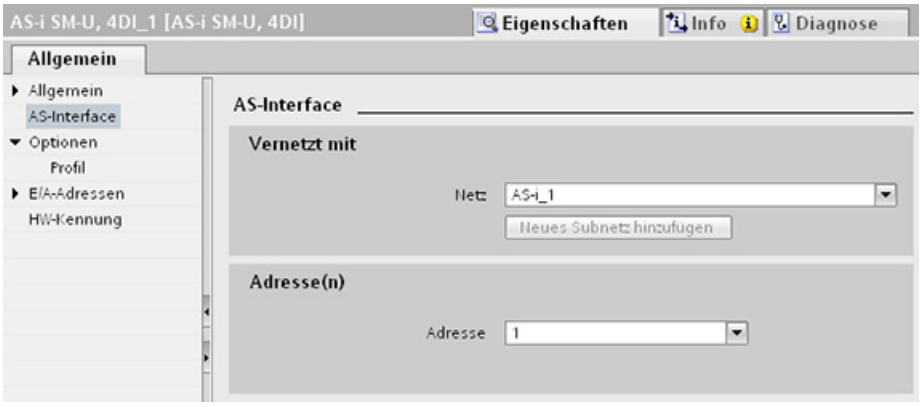
Die CPU greift im zyklischen Betrieb über den AS-i-Master CM1243-2 auf die digitalen Eingänge und Ausgänge der AS-i-Slaves zu. Der Zugriff auf die Daten erfolgt über E/A-Adressen oder mittels Datensatzübertragung.



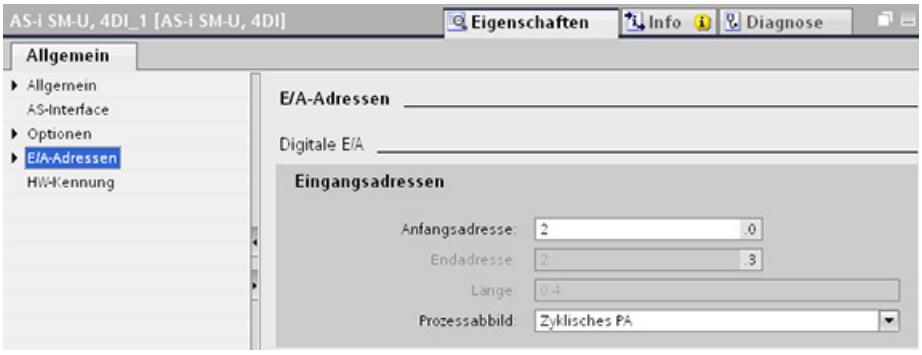
① AS-i-Slaveadresse 1
② AS-i-Slaveadresse 2A
③ AS-i-Slaveadresse 3

Der Zugriff auf die Digitaldaten erfolgt hier in Bytes (mit anderen Worten: jedem digitalen AS-i-Slave ist ein Byte zugeordnet). Wenn Sie die AS-i-Slaves in STEP 7 konfigurieren, wird für den jeweiligen AS-i-Slave die E/A-Adresse für den Zugriff auf die Daten aus dem Anwenderprogramm im Inspektorfenster angezeigt.

Dem Digitaleingabemodul (AS-i SM-U, 4DI) im AS-i-Netzwerk oben wurde die Slaveadresse 1 zugewiesen. Wenn Sie auf das Digitaleingabemodul klicken, wird in den "Eigenschaften" des Geräts im Register "AS-Schnittstelle" die Slaveadresse wie im Folgenden angezeigt:

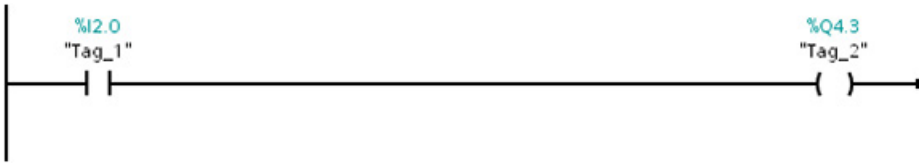


Dem Digitaleingabemodul (AS-i SM-U, 4DI) im AS-i-Netzwerk oben wurde die E/A-Adresse 2 zugewiesen. Wenn Sie auf das Digitaleingabemodul klicken, wird in den "Eigenschaften" des Geräts im Register "E/A-Adressen" die E/A-Adresse wie im Folgenden angezeigt:



Sie können auf die Daten der AS-i-Slaves im Anwenderprogramm zugreifen, indem Sie die deren E/A-Adressen mit den entsprechenden Bitverknüpfungsoperationen (z. B. "UND") oder Bitzuweisungen verwenden. Das folgende einfache Programm zeigt die Funktionsweise der Zuweisung:

In diesem Programm wird Eingang 2.0 abgefragt. Im AS-i-System gehört dieser Eingang zu Slave 1 (Eingangsbyte 2, Bit 0). Ausgang 4.3, der dann gesetzt wird, entspricht dem AS-i-Slave 3 (Ausgangsbyte 4, Bit 3).



Analoge AS-i-Werte übertragen

Wenn Sie einen AS-i-Slave in STEP 7 als analogen Slave konfiguriert haben, können Sie über das Prozessabbild der CPU auf die analogen Daten dieses AS-i-Slaves zugreifen.

Wenn Sie den analogen Slave nicht in STEP 7 konfiguriert haben, können Sie nur über die azyklischen Funktionen (Datensatzschnittstelle) auf die Daten des AS-i-Slaves zugreifen. Im Anwenderprogramm der CPU werden AS-i-Aufrufe über die Anweisungen für die dezentrale Peripherie, nämlich RDREC (Datensatz lesen) und WRREC (Datensatz schreiben), gelesen und geschrieben.

Hinweis

Eine über STEP 7 angegebene und in die S7-Station geladene Konfiguration der AS-i-Slaves wird beim Anlauf der S7-Station von der CPU in den AS-i-Master CM1243-2 geladen. Eine ggf. bereits vorhandene Konfiguration, die über die Online-Funktion der "Systemzuweisung" (Seite 796) ("IST -> SOLL") bestimmt wurde, wird überschrieben.

Weitere Informationen

Ausführliche Informationen zum AS-i-Master CM 1243-2 finden Sie im Handbuch "AS-I Master CM 1243-2 und AS-i-Datenentkopplungsmodul DCM 1271 für SIMATIC S7-1200" (http://support.automation.siemens.com/WW/view/en/50414115/133300).

11.4.3 Anweisungen für die dezentrale Peripherie

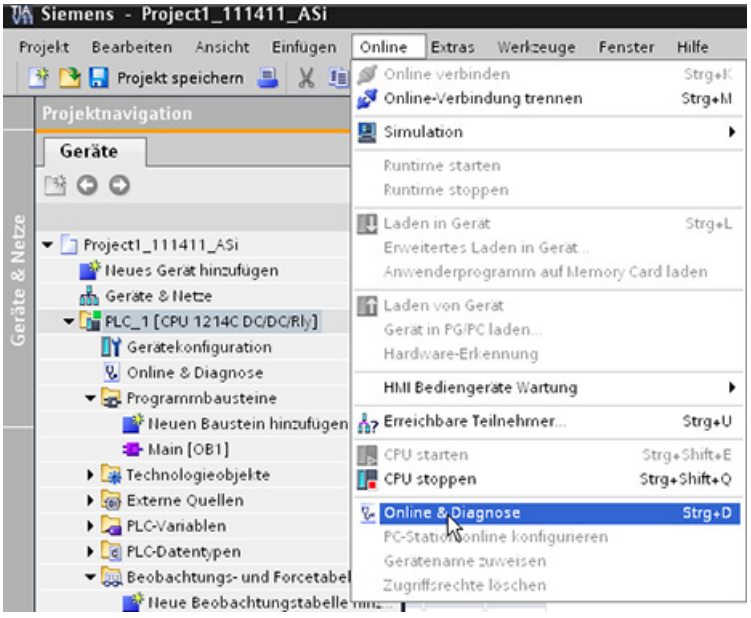
Weitere Informationen zur Verwendung der Anweisungen für die dezentrale Peripherie mit diesen Kommunikationsnetzwerken finden Sie unter "Dezentrale Peripherie (PROFINET, PROFIBUS oder AS-i)" (Seite 356).

11.4.4 Mit AS-i-Online-Werkzeugen arbeiten

Betriebsarten der AS-i online ändern

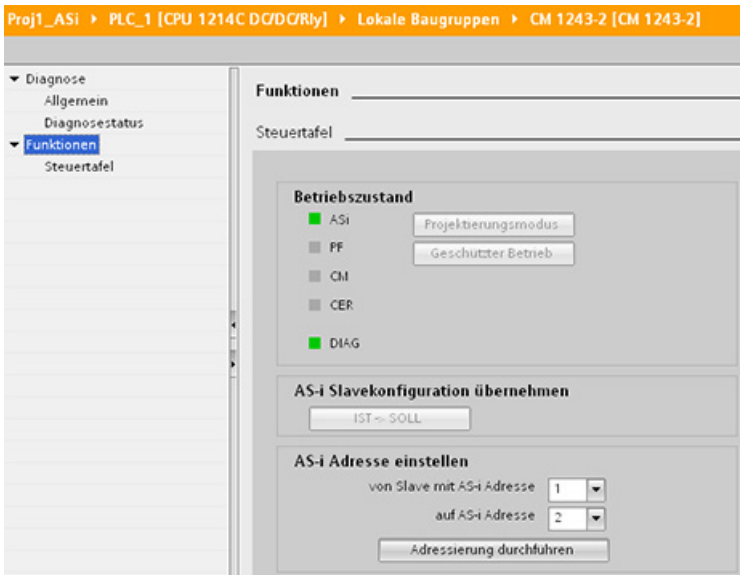
Sie müssen online gehen, um die Betriebsarten der AS-i anzuzeigen und zu ändern.

Um online zu gehen, müssen Sie zunächst das AS-i-Mastermodul CM1243-2 auswählen und die "Gerätekonfiguration" aufrufen. Klicken Sie dann in der Symbolleiste auf die Schaltfläche "Online gehen". Wählen Sie dann im Menü "Online" den Befehl "Online und Diagnose".



Es gibt zwei AS-i-Betriebsarten:

- Schutzmodus:
  - Das AS-i-Slavegerät und die E/A-Adressen der CPU können nicht geändert werden.
  - Die grüne LED "CM" ist ausgeschaltet.
- Konfigurationsmodus:
  - Sie können in Ihrem AS-i-Slavegerät und an den E/A-Adressen der CPU erforderliche Änderungen vornehmen.
  - Die grüne LED "CM" ist eingeschaltet.



Die AS-i-Slaveadresse können Sie im Feld "AS-i-Adresse festlegen" ändern. Ein neuer Slave, dem noch keine Adresse zugewiesen wurde, hat immer die Adresse 0. Er wird vom Master als neuer Slave ohne Adresszuweisung erkannt und wird erst nach Zuweisung einer Adresse in die normale Kommunikation einbezogen.

Konfigurationsfehler

Wenn die gelbe LED "CER" leuchtet, liegt ein Fehler in der Gerätekonfiguration des AS-i-Slaves vor. Wählen Sie die Schaltfläche "IST > SOLL", um die Slave-Gerätekonfiguration des AS-i-Mastermoduls CM1243-2 durch die Slave-Gerätekonfiguration des AS-i-Feldnetzwerks zu überschreiben.

11.5 S7-Kommunikation

11.5.1 Anweisungen GET und PUT (Daten aus entfernter CPU auslesen/in entfernte CPU schreiben)

Mittels der Anweisungen GET und PUT können Sie mit S7-CPUs über PROFINET- und PROFIBUS-Verbindungen kommunizieren. Dies ist nur möglich, wenn die Funktion "Zugriff über PUT/GET-Kommunikation erlauben" für die Partner-CPU in der Eigenschaft "Schutz" der lokalen CPU-Eigenschaften aktiviert ist:

- Zugriff auf Daten in einer dezentralen CPU: Eine S7-1200 CPU kann im Eingabefeld ADDR_x nur absolute Adressen verwenden, um Variablen dezentraler CPUs (S7-200/300/400/1200) anzusprechen.
- Zugriff auf Daten in einem Standard-DB: Eine S7-1200 CPU kann im Eingabefeld ADDR_x nur absolute Adressen verwenden, um DB-Variablen in einem Standard-DB einer dezentralen S7-CPU anzusprechen.
- Zugriff auf Daten in einem optimierten DB: Eine S7-1200 CPU kann nicht auf DB-Variablen in einem optimierten DB einer dezentralen S7-1200 CPU zugreifen.
- Zugriff auf Daten in einer lokalen CPU: Eine S7-1200 CPU kann entweder absolute Adressen oder symbolische Adressen als Eingabe im Eingabefeld RD_x oder SD_x der Anweisung GET oder PUT verwenden.

Hinweis

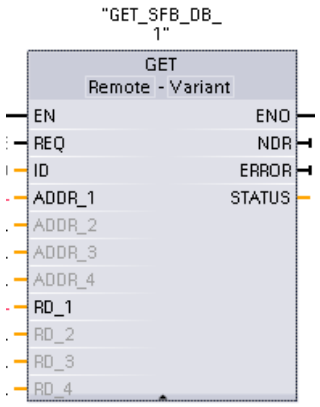
GET/PUT-Funktion im Programm einer V4.0-CPU nicht automatisch aktiviert

Die GET/PUT-Funktion des Programms einer V3.0 CPU ist in einer V4.0-CPU automatisch aktiviert.

Die GET/PUT-Funktion des Programms einer V4.0 CPU jedoch ist in einer V4.0-CPU nicht automatisch aktiviert. Sie müssen den GET/PUT-Zugriff (Seite 205) in der Gerätekonfiguration der CPU im Register "Eigenschaften" des Inspektorfensters unter der Eigenschaft "Schutz" aktivieren.

Tabelle 11- 63 Anweisungen GET und PUT

KOP/FUP	SCL	Beschreibung

	`"GET_DB"(`
	`req:=_bool_in_,`
	`ID:=_word_in_,`
	`ndr=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`addr_1:=_remote_inout_,`
	`[...addr_4:=_remote_inout_,]`
	`rd_1:=_variant_inout_`
	`[,...rd_4:=_variant_inout_]);`
		Mit der Anweisung GET lesen Sie Daten aus einer entfernten S7-CPU aus. Die entfernte CPU kann sich dabei im Betriebszustand RUN oder STOP befinden.
		STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

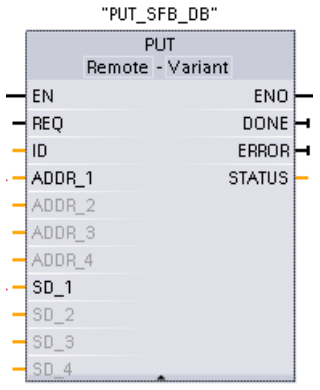
	`"PUT_DB"(`
	`req:=_bool_in_,`
	`ID:=_word_in_,`
	`done=>_bool_out_,`
	`error=>_bool_out_,`
	`status=>_word_out_,`
	`addr_1:=_remote_inout_,`
	`[...addr_4:=_remote_inout_,]`
	`sd_1:=_variant_inout_,`
	`[....sd_4:=_variant_inout_]);`
		Mit der Anweisung PUT schreiben Sie Daten in eine entfernte S7-CPU. Die entfernte CPU kann sich dabei im Betriebszustand RUN oder STOP befinden.
		STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 11- 64 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	Input	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet.
ID	Input	CONN_PRG (Word)	S7 Verbindungs-ID (Hex)
NDR (GET)	Output	Bool	Neue Daten bereit: • 0: Die Anforderung wurde noch nicht gestartet oder läuft noch. • 1: Der Auftrag wurde erfolgreich beendet.
DONE (PUT)	Output	Bool	DONE: • 0: Die Anforderung wurde noch nicht gestartet oder läuft noch. • 1: Der Auftrag wurde erfolgreich beendet.

Parameter und Datentyp		Datentyp	Beschreibung
ERROR STATUS	Output Output	Bool Word	• ERROR=0 STATUS-Wert: – 0000H: Weder Warnung noch Fehler. – <> 0000H: Warnung, STATUS liefert ausführliche Informationen. • ERROR=1 Fehler. STATUS liefert ausführliche Informationen über die Natur des Fehlers.
ADDR_1	InOut	Entfernt	Pointer auf die Speicherbereiche in der entfernten CPU, die die zu lesenden (GET) oder zu sendenden (PUT) Daten speichert.
ADDR_2	InOut	Entfernt	
ADDR_3	InOut	Entfernt	
ADDR_4	InOut	Entfernt	
RD_1 (GET) SD_1 (PUT)	InOut	Variante	Pointer auf die Speicherbereiche in der lokalen CPU, die die zu lesenden (GET) oder zu sendenden (PUT) Daten speichert. Zulässige Datentypen: Bool (nur ein Bit zulässig), Byte, Char, Word, Int, DWord, DInt oder Real. Hinweis: Wenn der Pointer auf einen DB zugreift, müssen Sie die absolute Adresse angeben, z. B.: P# DB10.DBX5.0 Byte 10 In diesem Fall stellt der Wert 10 die Anzahl von Bytes für die Anweisung GET oder PUT dar.
RD_2 (GET) SD_2 (PUT)	InOut	Variante	
RD_3 (GET) SD_3 (PUT)	InOut	Variante	
RD_4 (GET) SD_4 (PUT)	InOut	Variante	

Sie müssen sicherstellen, dass Länge (Anzahl der Bytes) und Datentypen der Parameter ADDR_x (entfernte CPU) und RD_x oder SD_x (lokale CPU) übereinstimmen. Die Zahl nach der Kennung "Byte" ist die Anzahl von Bytes, die vom Parameter ADDR_x, RD_x oder SD_x angegeben wird.

Hinweis

Die Gesamtzahl der von einer Anweisung GET empfangenen Bytes bzw. die Gesamtzahl der von einer Anweisung PUT gesendeten Bytes ist begrenzt. Die Begrenzung richtet sich danach, wie viele der vier möglichen Adress- und Speicherbereiche Sie verwenden:

- Wenn Sie nur ADDR_1 und RD_1/SD_1 verwenden, kann eine Anweisung GET 222 Bytes abrufen und eine Anweisung PUT 212 Bytes senden.
- Wenn Sie ADDR_1, RD_1/SD_1, ADDR_2 und RD_2/SD_2 verwenden, kann eine Anweisung GET insgesamt 218 Bytes abrufen und eine Anweisung PUT insgesamt 196 Bytes senden.
- Wenn Sie ADDR_1, RD_1/SD_1, ADDR_2, RD_2/SD_2, ADDR_3 und RD_3/SD_3 verwenden, kann eine Anweisung GET insgesamt 214 Bytes abrufen und eine Anweisung PUT insgesamt 180 Bytes senden.
- Wenn Sie ADDR_1, RD_1/SD_1, ADDR_2, RD_2/SD_2, ADDR_3, RD_3/SD_3, ADDR_4, RD_4/SD_4 verwenden, kann eine Anweisung GET insgesamt 210 Bytes abrufen und eine Anweisung PUT insgesamt 164 Bytes senden.

Die Summe der Byteanzahlen von jeder Ihrer Adress- und Speicherbereichsparameter muss kleiner oder gleich den festgelegten Grenzwerten sein. Wenn Sie diese Grenzwerte überschreiten, gibt die Anweisung GET oder PUT einen Fehler aus.

Bei der steigenden Flanke am Parameter REQ lädt die Leseanweisung (GET) oder die Schreibanweisung (PUT) die Parameter ID, ADDR_1 und RD_1 (GET) oder SD_1 (PUT).

- Bei GET: Die entfernte CPU gibt die angeforderten Daten an die Empfangsbereiche aus (RD_x). Sie beginnt damit im nächsten Zyklus. Wenn der Lesevorgang fehlerfrei beendet wurde, wird der Parameter NDR auf 1 gesetzt. Ein neuer Vorgang kann erst gestartet werden, wenn der vorhergehende Vorgang beendet ist.
- Bei PUT: Die lokale CPU beginnt, die Daten (SD_x) an die Adresse (ADDR_x) im Speicher der entfernten CPU zu senden. Wenn die Schreibanweisung fehlerfrei beendet wurde, gibt die entfernte CPU eine Ausführungsquittierung aus. Der Parameter DONE der Anweisung PUT wird dann auf 1 gesetzt. Ein neuer Schreibvorgang kann erst gestartet werden, wenn der vorhergehende Vorgang beendet ist.

Hinweis

Um Datenkonsistenz sicherzustellen, prüfen Sie stets, ob die Anweisung beendet wurde (NDR = 1 bei GET bzw. DONE = 1 bei PUT), bevor Sie auf die Daten zugreifen oder eine weitere Lese- oder Schreibanweisung ausführen.

Die Parameter ERROR und STATUS liefern Informationen zum Zustand der Lese- (GET) oder Schreibanweisung (PUT).

Tabelle 11- 65 Fehlerinformation

ERROR	STATUS (dezimal)	Beschreibung
0	11	• Der neue Auftrag kann nicht wirksam werden, weil der vorherige Auftrag noch nicht beendet ist. • Der Auftrag wird nun in einer Prioritätsklasse mit geringerer Priorität verarbeitet.
0	25	Kommunikation gestartet. Auftrag wird bearbeitet.
1	1	Kommunikationsprobleme, zum Beispiel: • Verbindungsbeschreibung nicht geladen (lokal oder entfernt) • Verbindung unterbrochen (Beispiel: Kabel, CPU ausgeschaltet oder CM/CB/CP ist in STOP) • Verbindung zum Partner noch nicht aufgebaut
1	2	Negative Quittierung vom Partnergerät. Die Aufgabe kann nicht ausgeführt werden.
1	4	Fehler in den Sendebereichs-Pointern (RD_x bei GET, SD_x bei PUT) bezüglich Datenlänge oder Datentyp.
1	8	Zugriffsfehler auf der Partner-CPU
1	10	Zugriff auf den lokalen Anwenderspeicher nicht möglich (Beispiel: Zugriffsversuch auf gelöschten DB)
1	12	Als der SFB aufgerufen wurde: • Wurde ein Instanz-DB angegeben, der nicht zu GET oder PUT gehört. • Wurde kein Instanz-DB angegeben, sondern ein gemeinsam genutzter DB • Wurde kein Instanz-DB gefunden (Laden eines neuen Instanz-DB).

ERROR	STATUS (dezimal)	Beschreibung
1	20	• Maximale Anzahl paralleler Aufträge/Instanzen überschritten • Die Instanzen wurden bei CPU-RUN überladen Dieser Zustand ist während der ersten Ausführung der Anweisung GET oder PUT möglich.
1	27	Es gibt keine entsprechende Anweisung GET oder PUT in der CPU.

11.5.2 S7-Verbindung erstellen

Verbindungsmechanismen

Um mit Hilfe der Anweisungen PUT/GET mit entfernten Teilnehmern zu kommunizieren, benötigt der Benutzer entsprechende Rechte.

Die Option "Zugriff über PUT/GET-Kommunikation erlauben" ist standardmäßig nicht aktiviert. Der Lese- und Schreibzugriff auf CPU-Daten ist dann nur mit Kommunikationsverbindungen möglich, die für die lokale CPU und für den entfernten Teilnehmer konfiguriert bzw. programmiert werden müssen. Beispielsweise ist der Zugriff mit Hilfe der Anweisungen BSEND/BRCV möglich.

Verbindungen, für die die lokale CPU nur als Server dient (d. h., die Kommunikation mit dem Teilnehmer ist in der lokalen CPU nicht konfiguriert/programmiert), sind während des Betriebs der CPU dann nicht möglich. Dies gilt z. B. für:

- PUT/GET-, FETCH/WRITE- oder FTP-Zugriff über Kommunikationsmodule
- PUT/GET-Zugriff von anderen S7 CPUs
- HMI-Zugriff über PUT/GET-Kommunikation

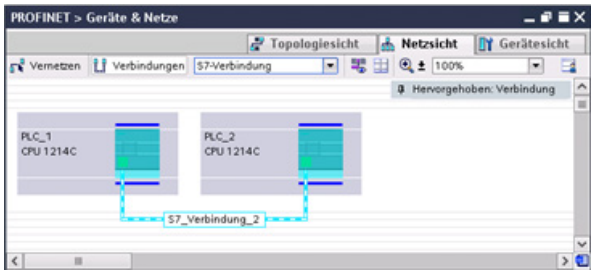
Soll auch der clientseitige Zugriff auf CPU-Daten möglich sein, d. h., sollen die Kommunikationsdienste der CPU nicht beschränkt werden, können Sie den Zugriffsschutz für die S7-1200 CPU (Seite 205) für diese Sicherheitsstufe konfigurieren.

Verbindungsarten

Die ausgewählte Verbindungsart baut eine Kommunikationsverbindung zu einer Partnerstation auf. Die Verbindung wird eingerichtet, aufgebaut und automatisch überwacht.

Im Portal "Geräte & Netze" können Sie in der "Netzsicht" die Geräte in Ihrem Projekt vernetzen. Klicken Sie zunächst auf das Register "Verbindungen" und wählen Sie dann über die Klappliste rechts den Verbindungstyp aus (z. B. eine S7-Verbindung). Klicken Sie auf das grüne Feld (PROFINET) auf dem ersten Gerät und ziehen Sie eine Linie zum PROFINET-Feld auf dem zweiten Gerät. Lassen Sie die Maustaste los. Damit ist Ihre PROFINET-Verbindung hergestellt.

Ausführliche Informationen finden Sie unter "Erstellen einer Netzwerkverbindung" (Seite 637).



Klicken Sie auf die Schaltfläche "Hervorgehoben: Verbindung", um den Konfigurationsdialog "Eigenschaften" der Kommunikationsanweisung aufzurufen.

11.5.3 Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren

Allgemeine Parameter konfigurieren

Sie legen die Kommunikationsparameter im Dialog "Eigenschaften" der Kommunikationsanweisung fest. Dieser Dialog wird am unteren Ende der Seite angezeigt, wenn Sie einen Teil der Anweisung selektiert haben.

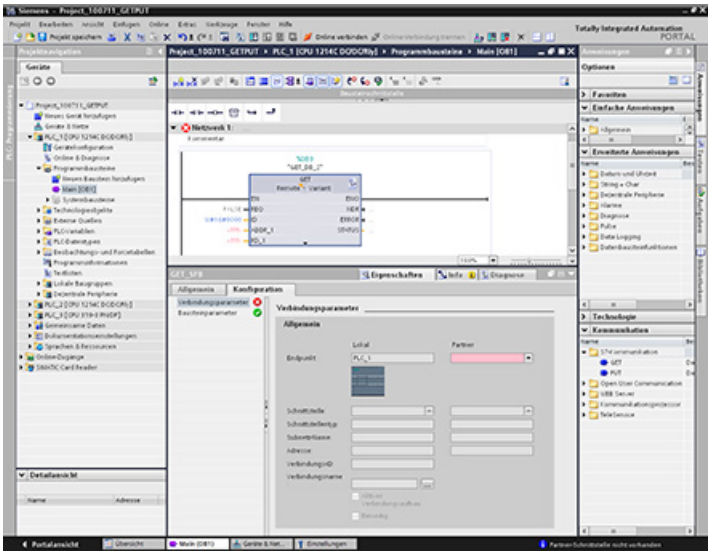
Weitere Informationen finden Sie unter "Gerätekonfiguration: Verbindungspfad zwischen lokaler und Partner-CPU konfigurieren (Seite 638)".

Die zu verwendenden TSAPs oder Ports definieren Sie im Dialog "Verbindungsparameter" unter "Adressdetails". Der TSAP oder Port einer Verbindung in der CPU wird im Feld "Lokaler TSAP" eingegeben. Der TSAP oder Port für die Verbindung in der Partner-CPU wird im Feld "Partner-TSAP" eingegeben.

11.5.4 Zuweisung von Verbindungsparametern für GET/PUT

Die Zuweisung der Verbindungsparameter für die GET/PUT-Anweisungen ist eine Hilfe für den Anwender bei der Konfiguration der S7-Verbindungen für die S7-Kommunikation von CPU zu CPU.

Nach dem Einfügen eines GET- oder PUT-Bausteins zeigt STEP 7 den Zuweisungsdialog für die Verbindungsparameter der GET/PUT-Anweisungen an:



Im Inspektorfenster werden die Eigenschaften der Verbindung angezeigt, wenn Sie einen Teil der Anweisung auswählen. Sie können die Kommunikationsparameter über das Register "Konfiguration" im Dialog "Eigenschaften" der Kommunikationsanweisung konfigurieren.

Hinweis

Die GET/PUT-Funktion ist im Programm einer CPU ab V4.1 nicht automatisch aktiviert

Die GET/PUT-Funktion im Programm einer CPU V3.0 ist in einer CPU ab V4.1 automatisch aktiviert.

Die GET/PUT-Funktion im Programm einer CPU ab V4.1 jedoch ist in einer CPU ab V4.1 nicht automatisch aktiviert. Sie müssen den GET/PUT-Zugriff (Seite 205) in der Gerätekonfiguration der CPU im Register "Eigenschaften" des Inspektorfensters unter der Eigenschaft "Schutz" aktivieren.

11.5.4.1 Verbindungsparameter

Auf der Seite "Verbindungsparameter" können Sie die erforderliche S7-Verbindung und den Parameter "Verbindungs-ID" konfigurieren, auf den vom Parameter "ID" des GET/PUT-Bausteins verwiesen wird. Die Seite enthält Informationen zum lokalen Endpunkt und Sie können hier auch die lokale Schnittstelle festlegen. Ferner können Sie den Partner-Endpunkt definieren.

Auf der Seite "Bausteinparameter" können Sie weitere Bausteinparameter konfigurieren.

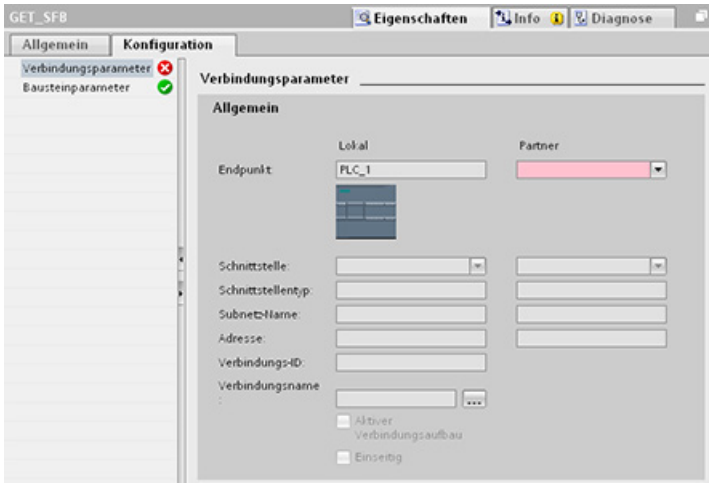


Tabelle 11- 66 Verbindungsparameter: Allgemeine Definitionen

Parameter		Definition
Verbindungs-parameter: Allgemein	Endpunkt	"Lokaler Endpunkt": Name der lokalen CPU "Partner-Endpunkt": Name der Partner-CPU (dezentral) Hinweis: In der Klappliste "Partner-Endpunkt" zeigt das System alle potenziellen S7-Verbindungspartner des aktuellen Projekts sowie die Option "Nicht spezifiziert" an. Ein nicht spezifizierter Partner ist ein Kommunikationspartner, der sich gegenwärtig nicht im STEP 7-Projekt befindet (z. B. ein Kommunikationspartner eines Fremdherstellers).
	Schnittstelle	Name der Schnittstellen Hinweis: Sie können die Verbindung ändern, indem Sie die lokale und die Partnerschnittstelle ändern.
	Schnittstellentyp	Typ der Schnittstelle
	Subnetzname	Name der Subnetze
	Adresse	Zugewiesene IP-Adressen Hinweis: Für einen "nicht spezifizierten" Kommunikationspartner können Sie die entfernte Adresse eines Fremdgeräts angeben.
	Verbindungs-ID	ID-Nummer: Wird von der Parameterzuweisung der GET/PUT-Verbindung automatisch generiert
	Verbindungsname	Datenspeicheradresse für die lokale CPU und die Partner-CPU: Wird von der Parameterzuweisung der GET/PUT-Verbindung automatisch generiert
	Aktiver Verbindungsaufbau	Kontrollkästchen zur Auswahl der lokalen CPU als aktive Verbindung

Parameter		Definition
	Unidirektional	Kontrollkästchen, um eine unidirektionale oder eine bidirektionale Verbindung anzugeben; schreibgeschützt Hinweis: In einer GET/PUT-Verbindung mit PROFINET können sowohl das lokale Gerät als auch das Partnergerät als Server oder Client fungieren. Dies ermöglicht eine bidirektionale Verbindung, das Kontrollkästchen "Unidirektional" ist dabei nicht aktiviert. Bei einer GET/PUT-Verbindung mit PROFIBUS kann das Partnergerät in einigen Fällen nur als Server fungieren (z. B. bei einer S7-300) und das Kontrollkästchen "Unidirektional" ist aktiviert.

Parameter Verbindungs-ID

Es gibt drei Möglichkeiten, die systemdefinierten Verbindungs-IDs zu ändern:

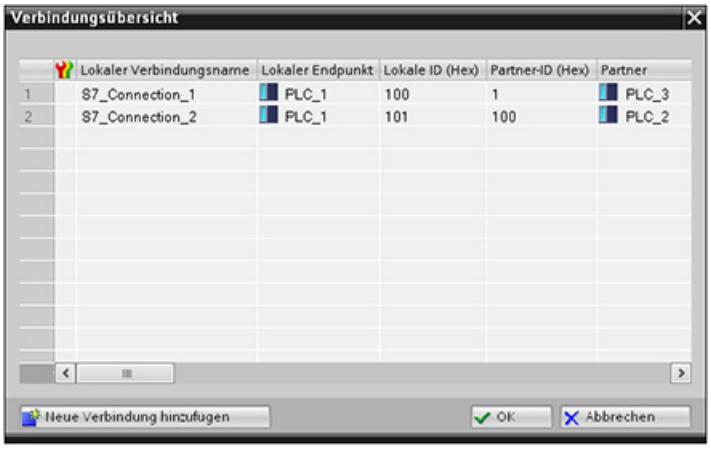
1. Sie können die aktuelle ID direkt im GET/PUT-Baustein ändern. Wenn die neue ID zu einer bereits vorhandenen Verbindung gehört, wird die Verbindung geändert.
2. Sie können die aktuelle ID direkt im GET/PUT-Baustein ändern, doch die neue ID ist noch nicht vorhanden. Das System erstellt eine neue S7-Verbindung.
3. Sie können die aktuelle ID über den Dialog "Verbindungsübersicht" ändern: Ihre Eingabe wird mit dem ID-Parameter des entsprechenden GET/PUT-Bausteins synchronisiert.

Hinweis

Der Parameter "ID" des GET/PUT-Bausteins ist kein Verbindungsname, sondern ein numerischer Ausdruck, der wie im folgenden Beispiel geschrieben wird: W#16#1

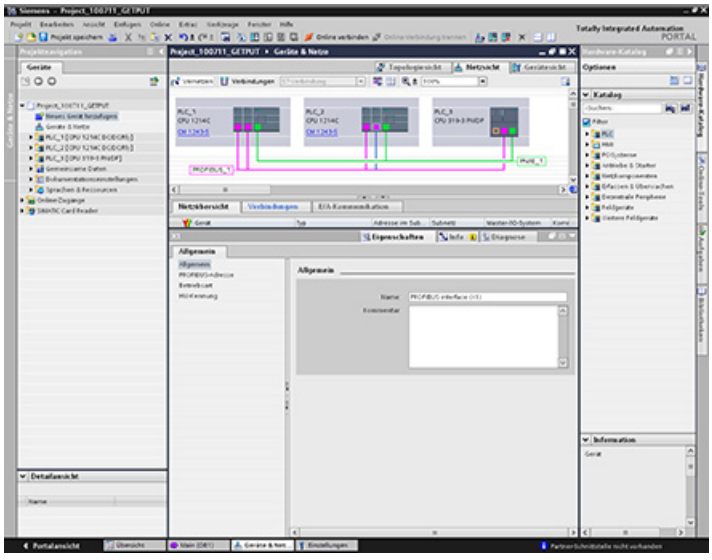
Parameter Verbindungsname

Der Verbindungsname lässt sich über ein besonderes Bedienelement, den Dialog "Verbindungsübersicht", bearbeiten. In diesem Dialog werden alle verfügbaren S7-Verbindungen angezeigt, die als Alternative für die aktuelle GET/PUT-Kommunikation ausgewählt werden können. Sie können in dieser Tabelle eine vollständig neue Verbindung anlegen. Klicken Sie auf die Schaltfläche rechts vom Feld "Verbindungsname", um den Dialog "Verbindungsübersicht" aufzurufen.



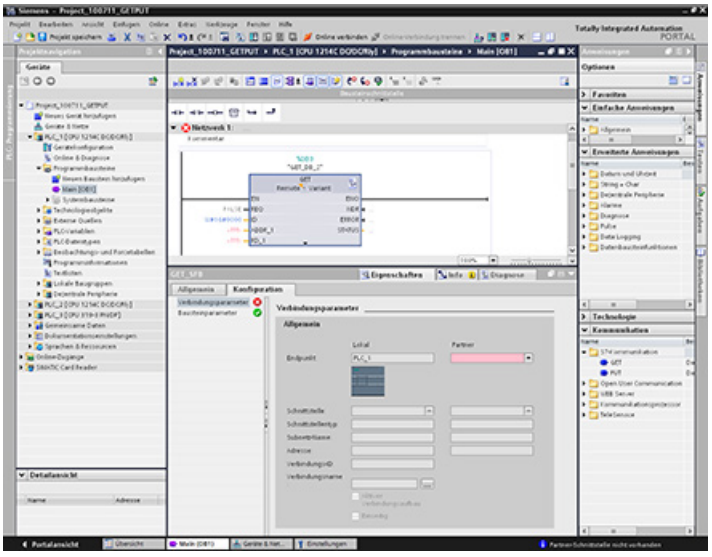
11.5.4.2 S7-Verbindung von CPU zu CPU konfigurieren

Bei einer Konfiguration von PLC_1, PLC_2 und PLC_3, wie in der folgenden Abbildung gezeigt, fügen Sie GET- oder PUT-Bausteine für "PLC_1" ein.



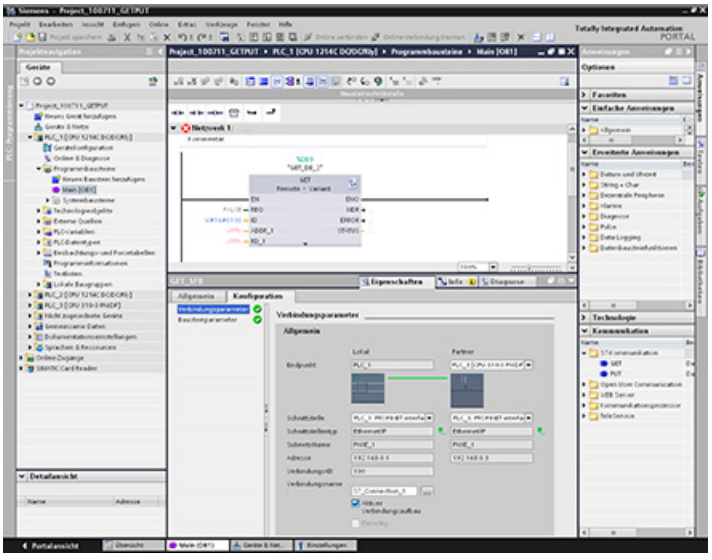
Für die Anweisung GET oder PUT wird im Inspektorfenster automatisch das Register "Eigenschaften" mit den folgenden Menüoptionen angezeigt:

- "Konfiguration"
- "Verbindungsparameter"



PROFINET S7-Verbindung konfigurieren

Wählen Sie für den "Partner-Endpunkt" die Option "PLC_3" aus.



Das System reagiert mit den folgenden Änderungen:

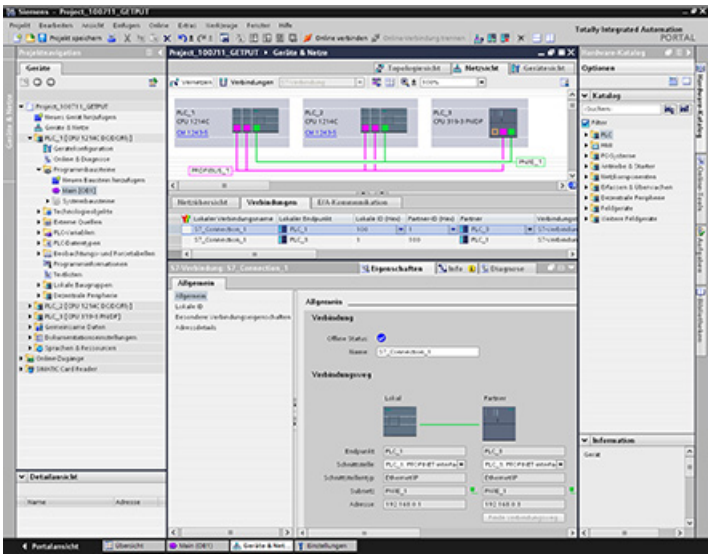
Tabelle 11- 67 Verbindungsparameter: Allgemeine Werte

Parameter		Definition
Verbindungs-parameter: Allgemein	Endpunkt	"Lokaler Endpunkt" enthält "PLC_1" (schreibgeschützt). Das Feld "Partner-Endpunkt" enthält "PLC_3[CPU319-3PN/DP]": • Die Farbe wechselt von Rot nach Weiß. • Das "Partner"-Gerätebild wird angezeigt. • Eine Verbindungslinie erscheint zwischen den PLC_1- und PLC_3-Gerätebildern (grüne Ethernet-Linie).
	Schnittstelle	"Lokale Schnittstelle" enthält "CPU1214C DC/DC/DC, PROFINET-Schnittstelle (R0/S1)". "Partner-Schnittstelle" enthält: "CPU319-3PN/DP, PROFINET-Schnittstelle (R0/S2)".
	Schnittstellentyp	"Lokaler Schnittstellentyp" enthält "Ethernet/IP"; die Bedienung ist schreibgeschützt. "Partner-Schnittstellentyp" enthält "Ethernet/IP"; die Bedienung ist schreibgeschützt. Die Schnittstellentypbilder werden rechts neben dem lokalen und dem Partner-Schnittstellentyp angezeigt (grünes Ethernet-Symbol).
	Subnetzname	"Lokaler Subnetzname" enthält "PN/IE_1"; die Bedienung ist schreibgeschützt. "Partner-Subnetzname" enthält "PN/IE_1"; die Bedienung ist schreibgeschützt.
	Adresse	"Lokale Adresse" enthält die lokale IP-Adresse; die Bedienung ist schreibgeschützt. "Partneradresse" enthält die Partner-IP-Adresse; die Bedienung ist schreibgeschützt.
	Verbindungs-ID	"Verbindungs-ID" enthält "100". Im Hauptprogramm [OB1] des Programmiereditors enthält die "Verbindungs-ID" des GET/PUT-Bausteins ebenfalls den Wert "100".
	Verbindungsname	"Verbindungsname" enthält den Standardverbindungsnamen (Beispiel: "S7_Verbindung_1"); die Bedienung ist aktiviert.
	Aktiver Verbindungsaufbau	Aktiviert, um die lokale CPU als aktive Verbindung auszuwählen.
	Unidirektional	Schreibgeschützt und nicht aktiviert. Hinweis: "PLC_1" (eine S7-1200 CPU 1214CDC/DC/Rls) und "PLC_3" (eine S7-300 CPU 319-3PN/DP) können in einer PROFINET GET/PUT-Verbindung beide als Server und als Client fungieren, so dass eine bidirektionale Verbindung möglich ist.

Das GET/PUT-Symbol in der Eigenschaftsansicht wechselt von Rot nach Grün.

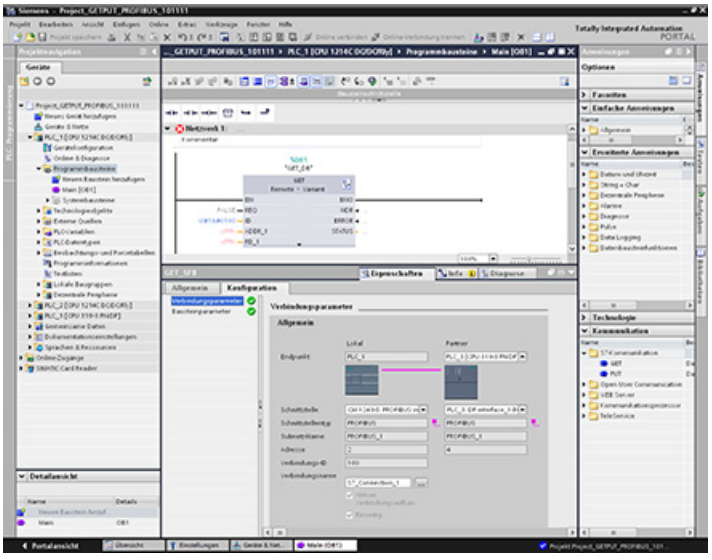
Vollständige PROFINET S7-Verbindung

In der "Netzsicht" wird in der Tabelle "Verbindungen" zwischen "PLC_1" und "PLC_3" eine bidirektionale S7-Verbindung angezeigt.



PROFIBUS S7-Verbindung konfigurieren

Wählen Sie für den "Partner-Endpunkt" die Option "PLC_3" aus.



Das System reagiert mit den folgenden Änderungen:

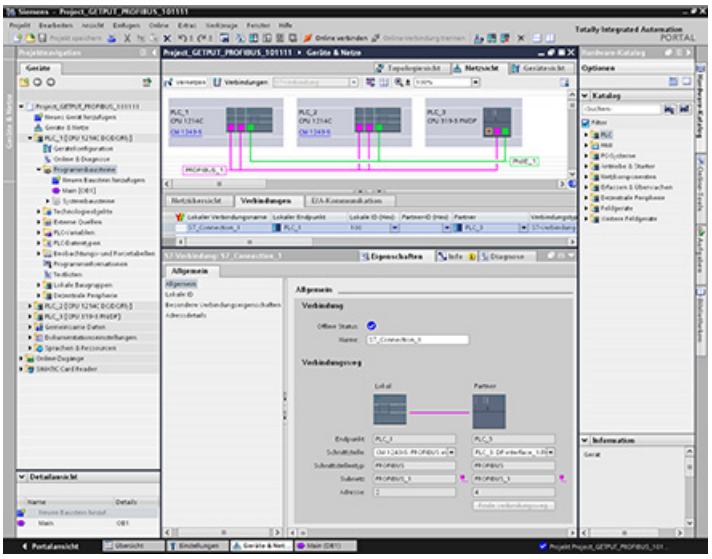
Tabelle 11- 68 Verbindungsparameter: Allgemeine Werte

Parameter		Definition
Verbindungs-parameter: Allgemein	Endpunkt	"Lokaler Endpunkt" enthält "PLC_1" (schreibgeschützt). Das Feld "Partner-Endpunkt" enthält "PLC_3[CPU319-3PN/DP]": • Die Farbe wechselt von Rot nach Weiß. • Das "Partner"-Gerätebild wird angezeigt. • Eine Verbindungslinie erscheint zwischen den PLC_1- und PLC_3-Gerätebildern (violettfarbene PROFIBUS-Linie).
	Schnittstelle	"Lokale Schnittstelle" enthält "CPU1214C DC/DC/DC, PROFIBUS-Schnittstelle (R0/S1)". "Partner-Schnittstelle" enthält: "CPU319-3PN/DP, PROFIBUS-Schnittstelle (R0/S2)".
	Schnittstellentyp	"Lokaler Schnittstellentyp" enthält "PROFIBUS"; die Bedienung ist schreibgeschützt. "Partner-Schnittstellentyp" enthält "PROFIBUS"; die Bedienung ist schreibgeschützt. Die Schnittstellentypbilder werden rechts neben dem lokalen und dem Partner-Schnittstellentyp angezeigt (violettfarbenes PROFIBUS-Symbol).
	Subnetzname	"Lokaler Subnetzname" enthält "PROFIBUS_1"; die Bedienung ist schreibgeschützt. "Partner-Subnetzname" enthält "PROFIBUS_1"; die Bedienung ist schreibgeschützt.
	Adresse	"Lokale Adresse" enthält die lokale IP-Adresse; die Bedienung ist schreibgeschützt. "Partneradresse" enthält die Partner-IP-Adresse; die Bedienung ist schreibgeschützt.
	Verbindungs-ID	"Verbindungs-ID" enthält "100". Im Hauptprogramm [OB1] des Programmiereditors enthält die "Verbindungs-ID" des GET/PUT-Bausteins ebenfalls den Wert "100".
	Verbindungsname	"Verbindungsname" enthält den Standardverbindungsnamen (Beispiel: "S7_Verbindung_1"); die Bedienung ist aktiviert.
	Aktiver Verbindungsaufbau	Schreibgeschützt und aktiviert, um die lokale CPU als aktive Verbindung auszuwählen.
	Unidirektional	Schreibgeschützt und aktiviert. Hinweis: "PLC_3" (eine S7-300 CPU319-3PN/DP) kann in einer GET/PUT-Verbindung mit PROFIBUS nur als Server fungieren (und nicht auch als Client), so dass nur eine unidirektionale Verbindung möglich ist.

Das GET/PUT-Symbol in der Eigenschaftsansicht wechselt von Rot nach Grün.

Vollständige PROFIBUS S7-Verbindung

In der "Netzsicht" wird in der Tabelle "Verbindungen" zwischen "PLC_1" und "PLC_3" eine unidirektionale S7-Verbindung angezeigt.



Webserver

12

Der Webserver der S7-1200 bietet Webseitenzugriff auf Daten über Ihre CPU und auf Prozessdaten.

Sie können die S7-1200 Webseiten über einen PC oder ein Mobilgerät aufrufen. Der Webserver zeigt die Seiten in einem Format und einer Größe an, die für das zum Anzeigen der Webseiten verwendete Gerät geeignet sind. Der Webserver unterstützt eine Mindestauflösung von 240 x 240 Pixel.

Verwenden Sie einen Webbrowser für den Zugriff auf die IP-Adresse der S7-1200 CPU oder der IP-Adresse eines über Webserver aktivierten CP-(Kommunikationsprozessor-)Moduls (Seite 826) im lokalen Baugruppenträger mit der CPU, um die Verbindung herzustellen. Die S7-1200 unterstützt mehrere gleichzeitige Verbindungen.

Standard-Webseiten

Die S7-1200 umfasst Standard-Webseiten, auf die Sie über einen Webbrowser auf Ihrem PC (Seite 823) oder über ein Mobilgerät (Seite 824) zugreifen können:

- Einführung (Seite 833) - Einstiegspunkt in die Standard-Webseiten
- Startseite (Seite 834) - allgemeine Informationen über die CPU
- Identifikation (Seite 835) - ausführliche Informationen über die CPU einschließlich Serien-, Bestell- und Versionsnummer
- Modulinformationen (Seite 837) - Informationen zu den Modulen im lokalen Baugruppenträger und Möglichkeit zur Aktualisierung der Firmware
- Kommunikation (Seite 840) - Informationen über die Netzwerkadressen, physikalischen Eigenschaften der Kommunikationsschnittstellen und Kommunikationsstatistik
- Diagnosepuffer (Seite 836) - der Diagnosepuffer
- Variablenzustand (Seite 841) - CPU-Variablen und E/A, zugänglich über Adressen oder PLC-Variablennamen
- Dateibrowser (Seite 844) - Browser für in der CPU oder auf einer Memory Card gespeicherte Dateien, z. B. Datenprotokolle oder Rezepte
- Anmeldung (Seite 829) - Anmeldung als anderer Nutzer oder Abmeldung.

Diese Seiten sind in die S7-1200 CPU integriert und stehen in Englisch, Deutsch, Französisch, Spanisch, Italienisch und vereinfachtem Chinesisch zur Verfügung. Zum Aufrufen einiger dieser Seiten sind zusätzliche Benutzerrechte (Seite 821) erforderlich, die Sie in STEP 7 konfigurieren. Einzelheiten über die Standard-Webseiten und den Zugriff auf diese Seiten finden Sie im Abschnitt Standard-Webseiten (Seite 827).

Hinweis

S7-1200 CPUs haben keine eigene Standard-Webseite für die Firmware-Aktualisierung. Die Firmware-Aktualisierungsfunktion ist in der Modulinformationsseite enthalten.

Benutzerdefinierte Webseiten

Bei der S7-1200 können Sie auch benutzerdefinierte Webseiten erstellen, die auf CPU-Daten zugreifen können. Sie können diese Seiten mit einer HTML-Authoring-Software Ihrer Wahl anlegen und vordefinierte "AWP"-Befehl (Automation Web Programming) in Ihren HTML-Code einfügen, um auf CPU-Daten zuzugreifen. Im Kapitel Benutzerdefinierte Webseiten (Seite 847) finden Sie spezielle Informationen zum Entwickeln von benutzerdefinierten Webseiten und zur zugehörigen Konfiguration und Programmierung in STEP 7.

Die benutzerdefinierten Webseiten können Sie über die Standard-Webseiten auf einem PC oder Mobilgerät aufrufen.

Voraussetzungen für den Webbrowser

Der Webserver unterstützt die folgenden PC-Webbrowser:

- Internet Explorer 8.0
- Internet Explorer 9.0
- Mozilla Firefox 17.0.1
- Google Chrome 23.0
- Apple Safari 5.1.7 (Windows)
- Apple Safari 6.0.2 (Mac)

Der Webserver unterstützt die folgenden Mobilgeräte-Webbrowser:

- Internet Explorer bis V6.0 für HMI-Panels
- Mobile Safari 7534.48.3 (iOS 5.0.1)
- Mobile Android Browser 2.3.4
- Mobile Google Chrome 23.0

Informationen zu browserspezifischen Einschränkungen, die sich auf die Anzeige von Standard- oder Benutzer-Webseiten auswirken können, finden Sie im Abschnitt Einschränkungen (Seite 892).

12.1 Webserver aktivieren

Sie aktivieren den Webserver in STEP 7 über die Gerätekonfiguration der CPU, zu der Sie eine Verbindung herstellen möchten.

Um den Webserver zu aktivieren, gehen Sie folgendermaßen vor:

1. Wählen Sie die CPU in der Gerätekonfiguration aus.
2. Wählen Sie im Inspektorfenster unter den CPU-Eigenschaften "Webserver" aus.
3. Aktivieren Sie das Kontrollkästchen für "Webserver auf allen Modulen dieses Geräts aktivieren".
4. Wählen Sie zur verbesserten Sicherheit "Erlaubt nur HTTPS-Zugriff" aus, sodass der sichere Zugriff auf den Webserver erforderlich ist.

⚠ WARNUNG

Nicht berechtigter Zugriff auf die CPU über den Webserver

Nicht berechtigter Zugriff auf die CPU oder das Einstellen von ungültigen Werten für PLC-Variablen kann den Prozessbetrieb stören und zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Weil durch die Aktivierung des Webservers berechtigte Benutzer Betriebszustandsänderungen vornehmen, PLC-Daten schreiben und Firmware-Updates durchführen können, empfiehlt Siemens, die folgenden Sicherheitsvorkehrungen einzuhalten:

- Aktivieren Sie den Zugriff auf den Webserver nur über das HTTPS-Protokoll.
- Schützen Sie die Benutzerkennungen für den Webserver (Seite 821) durch ein starkes Passwort. Starke Passwörter sind mindestens zehn Zeichen lang, bestehen aus Buchstaben, Zahlen und Sonderzeichen, sind keine Wörter, die in einem Wörterbuch gefunden werden können, und sind keine Namen oder Kennungen, die sich aus persönlichen Daten ableiten lassen. Halten Sie das Passwort geheim und ändern Sie es häufig.
- Erweitern Sie die standardmäßigen Mindestrechte des Benutzers "Jeder".
- Führen Sie eine Fehlerprüfung und eine Bereichsprüfung für die Variablen in Ihrer Programmlogik durch, weil Nutzer von Webseiten für die PLC-Variablen ungültige Werte einrichten können.
- Verwenden Sie für die Verbindung mit dem S7-1200 PLC-Webserver von einem Standort außerhalb Ihres geschützten Netzwerks ein sicheres Virtual Private Network (VPN).

Nachdem Sie die Gerätekonfiguration geladen haben, können Sie mit den Standard-Webseiten auf die CPU zugreifen. Wenn Sie für "Automatische Aktualisierung" die Option "Aktivieren" wählen, werden die Standard-Webseiten alle zehn Sekunden aktualisiert.

Wenn Sie benutzerdefinierte Webseiten (Seite 847) erstellt und aktiviert haben, können Sie über das Menü der Standard-Webseiten darauf zugreifen.

Hinweis

Geräteaustausch: Ersetzen einer V3.0-CPU durch eine V4.1-CPU

Wenn Sie eine vorhandene V3.0-CPU durch eine V4.1-CPU ersetzen (Seite 1349) und Ihr V3.0-Projekt in ein V4.1-Projekt konvertieren, müssen Sie beachten, dass STEP 7 und die V4.1-CPU die folgenden Webservereinstellungen beibehalten:

- "Webserver auf diesem Modul aktivieren"
- "Erlaubt nur HTTPS-Zugriff"

Hinweis

Wenn ein "Laden in RUN" (Seite 1138) durchgeführt wird, werden Datenwerte auf Standard- und benutzerdefinierten Webseiten nicht aktualisiert und Sie können auch erst dann wieder Datenwerte schreiben, wenn der Ladevorgang abgeschlossen ist. Der Webserver bricht jeden Versuch, während des Ladevorgangs Datenwerte zu schreiben, ab.

Aktivieren von anderen Sprachen für den Webserver

Für die Anzeige der Standard-Webseiten können Sie optional andere Sprachen auswählen. Wählen Sie im Register "Eigenschaften" Ihrer Gerätekonfiguration "Oberflächensprachen", um einer der sechs vom Webserver unterstützten Sprachen eine STEP 7-Projektsprache zuzuweisen. Nachdem Sie die Gerätekonfiguration geladen haben, bieten die Standard-Webseiten die Auswahl der Oberflächensprache an. Wenn Sie keine Sprache auswählen, ist die Standardsprache Englisch.

12.2 Konfigurieren von Webserver-Benutzern

Sie können Benutzer mit verschiedenen Berechtigungsstufen konfigurieren, die über den Webserver Zugriff auf die CPU erhalten.

Um Webserver-Benutzer und die zugehörigen Rechte zu konfigurieren, gehen Sie wie folgt vor:

1. Wählen Sie die CPU in der Gerätekonfiguration aus.
2. Wählen Sie im Inspektorfenster unter den CPU-Eigenschaften "Webserver" aus und aktivieren Sie den Webserver (Seite 819).
3. Wählen Sie in den Eigenschaften des Webservers "Benutzerverwaltung" aus.
4. Geben Sie Benutzernamen, Zugriffsstufen und Passwörter für die bereitzustellenden Benutzer-Logins ein.

Nachdem Sie die Konfiguration in die CPU geladen haben, können nur berechtigte Benutzer auf die Webserverfunktionen, für die sie die entsprechenden Rechte haben, zugreifen.

Zugriffsstufen für den Webserver

STEP 7 bietet den Standardbenutzer "Jeder" ohne Passwort. Dieser Benutzer hat standardmäßig keine zusätzlichen Rechte und kann nur die Standard-Webseiten Start (Seite 834) und Einführung (Seite 833) öffnen. Sie können dem Benutzer "Jeder" jedoch wie allen anderen von Ihnen konfigurierten Benutzern zusätzliche Rechte erteilen:

- Diagnoseabruf
- Variablen lesen
- Variablen schreiben
- Variablenzustand lesen
- Variablenzustand schreiben
- Benutzerdefinierte Webseiten öffnen
- In benutzerdefinierte Webseiten schreiben
- Dateien lesen
- Dateien schreiben/löschen
- Betriebszustand wechseln
- LEDs blinken lassen
- Firmware-Update durchführen
- Systemparameter ändern
- Anwendungsparameter ändern

⚠WARNUNG

Zugriff auf den Webserver

Werden dem Benutzer "Jeder" Rechte erteilt, ist es möglich, sich ohne Passwort am Webserver anzumelden. Nicht berechtigter Zugriff auf die CPU oder das Einstellen von ungültigen Werten für PLC-Variablen kann den Prozessbetrieb stören und zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Weil der Benutzer "Jeder" mit ausreichenden Rechten Betriebszustandsänderungen vornehmen, PLC-Daten schreiben und Firmware-Updates durchführen kann, empfiehlt Siemens, die folgenden Sicherheitsvorkehrungen einzuhalten:

- Aktivieren Sie den Zugriff auf den Webserver nur über das HTTPS-Protokoll.
- Schützen Sie die Benutzerkennungen für den Webserver durch ein starkes Passwort. Starke Passwörter sind mindestens zehn Zeichen lang, bestehen aus Buchstaben, Zahlen und Sonderzeichen, sind keine Wörter, die in einem Wörterbuch gefunden werden können, und sind keine Namen oder Kennungen, die sich aus persönlichen Daten ableiten lassen. Halten Sie das Passwort geheim und ändern Sie es häufig.
- Erweitern Sie die standardmäßigen Mindestrechte des Benutzers "Jeder".
- Führen Sie eine Fehlerprüfung und eine Bereichsprüfung für die Variablen in Ihrer Programmlogik durch, weil Nutzer von Webseiten für die PLC-Variablen ungültige Werte einrichten können.
- Verwenden Sie für die Verbindung mit dem S7-1200 PLC-Webserver von einem Standort außerhalb Ihres geschützten Netzwerks ein sicheres Virtual Private Network (VPN).

12.3 Über den PC auf die Webseiten zugreifen

Auf die S7-1200 Standard-Webseiten können Sie mit einem PC oder einem mobilen Gerät über die IP-Adresse der S7-1200 CPU oder die IP-Adresse eines über Webserver aktivierten CP (Seite 826) im lokalen Baugruppenträger zugreifen.

Um über einen PC auf die S7-1200 Standard-Webseiten zuzugreifen, gehen Sie folgendermaßen vor:

1. Vergewissern Sie sich, dass sich die S7-1200 und der PC in einem gemeinsamen Ethernet-Netzwerk befinden oder über ein herkömmliches Ethernet-Kabel direkt miteinander verbunden sind.
2. Öffnen Sie einen Webbrowser und geben Sie die URL "https://ww.xx.yy.zz" ein. Dabei entspricht "ww.xx.yy.zz" der IP-Adresse der S7-1200 CPU bzw. der IP-Adresse eines CP im lokalen Baugruppenträger.

Der Webbrowser öffnet die Einführungsseite.

Hinweis

Verwenden Sie für die Verbindung mit dem S7-1200 PLC-Webserver von einem Standort außerhalb Ihres geschützten Netzwerks ein sicheres Virtual Private Network (VPN). Beachten Sie außerdem alle Einschränkungen (Seite 892), die möglicherweise durch Ihre Webumgebung oder Ihr Betriebssystem gegeben sind.

Alternativ können Sie in Ihrem Webbrowser die Adresse einer spezifischen Standard-Webseite eingeben. Hierfür geben Sie die URL im Format "https://ww.xx.yy.zz/<Seite>.html" ein. Dabei entspricht <Seite> einer der folgenden Standard-Webseiten:

- start (Seite 834) - allgemeine Informationen über die CPU
- Identifikation (Seite 835) - ausführliche Informationen über die CPU einschließlich Serien-, Bestell- und Versionsnummer
- Modul (Seite 837) - Informationen zu den Modulen im lokalen Baugruppenträger und über die Möglichkeit von Firmware-Updates
- Kommunikation (Seite 840) - Informationen über die Netzwerkadressen, physikalischen Eigenschaften der Kommunikationsschnittstellen und Kommunikationsstatistik
- diagnostic (Seite 836) - der Diagnosepuffer
- variable (Seite 841) - CPU-Variablen und E/A, zugänglich über Adressen oder PLC-Variablennamen
- Dateibrowser (Seite 844) - Browser für den Zugriff auf Datenprotokoll- oder Rezeptdateien, die in der CPU oder auf einer Memory Card gespeichert sind
- index (Seite 833) - Einführungsseite für den Einstieg in die Standard-Webseiten
- Anmelden (Seite 829) - Anmelden als anderer Benutzer oder Abmelden (Auf dem PC wird auf jeder Standard-Webseite ein Anmelde-Fenster angezeigt, auf einem mobilen Gerät ist dagegen eine eigene Anmeldeseite vorhanden.)

Wenn Sie beispielsweise "https://ww.xx.yy.zz/communication.html" eingeben, zeigt der Browser die Kommunikationsseite an.

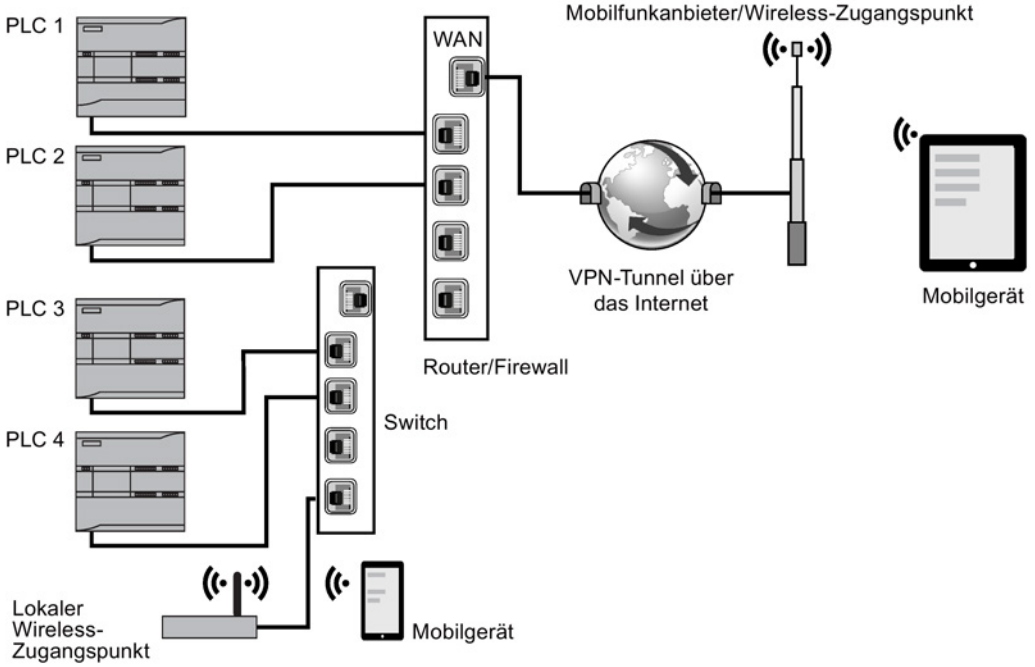
Sicherer Zugriff

Verwenden Sie für die Verbindung mit dem S7-1200 PLC-Webserver von einem Standort außerhalb Ihres geschützten Netzwerks ein sicheres Virtual Private Network (VPN). Verwenden Sie statt http:// für den sicheren Zugriff (Seite 819) auf die Standard-Webseiten https:// und legen Sie diesen Zugriff als erforderlich fest. Wenn Sie über https:// eine Verbindung zur S7-1200 herstellen, verschlüsselt die Website die Sitzung mit einem digitalen Zertifikat. Der Webserver überträgt die Daten über eine sichere Verbindung, die für niemanden sichtbar ist. Üblicherweise wird eine Sicherheitswarnung angezeigt, die Sie mit "Ja" bestätigen müssen, um die Standard-Webseiten aufzurufen. Um die Anzeige der Sicherheitswarnung bei jedem sicheren Zugriff zu vermeiden, können Sie das Siemens-Softwarezertifikat in Ihren Webbrowser importieren (Seite 894).

12.4 Über ein mobiles Gerät auf die Webseiten zugreifen

Für den Zugriff auf eine S7-1200 von einem mobilen Gerät aus müssen Sie Ihre PLC an ein Netzwerk mit Internetverbindung oder an einen lokalen Wireless Access Point anschließen. Verwenden Sie für die Verbindung eines Mobilgeräts mit dem S7-1200 PLC-Webserver ein sicheres Virtual Private Network (VPN). Mit Hilfe der Portweiterleitung im Wireless-Router können Sie die IP-Adresse der PLC in eine Adresse abbilden, durch die ein mobiles Gerät über das Internet auf die PLC zugreifen kann. Zum Einrichten der Portweiterleitung beachten Sie bitte die Anweisungen für die Softwarekonfiguration Ihres Routers. Sie können so viele PLCs und Schaltgeräte anschließen, wie von Ihrem Router unterstützt werden.

Ohne Portweiterleitung ist die Verbindung mit einer PLC nur lokal innerhalb der Reichweite des Funksignals möglich.



In diesem Beispiel kann ein mobiles Gerät, das sich in Reichweite des lokalen Wireless Access Points befindet, die Verbindung zu PLC 3 und PLC 4 über deren IP-Adressen herstellen. Außerhalb der Reichweite des lokalen Funksignals kann ein mobiles Gerät über das Internet die Verbindung zu PLC 1 und PLC 2 mit Hilfe der weitergeleiteten Adresse der jeweiligen PLC herstellen.

Für den Zugriff auf die Standard-Webseiten benötigen Sie Zugang zu einem Mobilfunkdienst oder einem Wireless Access Point. Um auf eine PLC über das Internet zuzugreifen, geben Sie in den Webbrowser Ihres mobilen Geräts die weitergeleitete Adresse für den Zugang zur PLC ein, also zum Beispiel http://ww.xx.yy.zz:pppp oder https::/ww.xx.yy.zz:pppp, wobei ww.xx.yy.zz die Adresse des Routers und pppp die Portzuordnung für eine bestimmte PLC ist.

Für den lokalen Zugriff über einen lokalen Wireless-Zugangspunkt geben Sie die IP-Adresse der S7-1200 CPU oder eines über Webserver aktivierten CP (Seite 826) im lokalen Baugruppenträger ein: http://ww.xx.yy.zz oder https::/ww.xx.yy.zz. Sie können außerdem auf eine Webseite auch über deren Namen zugreifen, siehe die Beschreibung in Über den PC auf die Webseiten zugreifen (Seite 823).

Um die Sicherheit zu erhöhen, konfigurieren Sie den Webserver so, dass nur sicherer Zugriff (HTTPS) (Seite 819) möglich ist.

12.5 Verwenden eines CP-Moduls für den Zugriff auf die Webseiten

Unabhängig davon, ob auf den Webserver über einen PC oder über ein mobiles Gerät zugegriffen wird, können die Standard-Webseiten über eines der folgenden CP-Module geöffnet werden, wenn dieses in STEP 7 konfiguriert und im lokalen Baugruppenträger mit der S7-1200 CPU installiert ist:

- CP 1242-7 GPRS V2
- CP 1243-7 LTE-EU

Für den Zugriff auf die Webseiten über diese CP-Module wird Standard-Webseite "Start" (Seite 834) verwendet. Die Start-Seite zeigt alle konfigurierten und installierten CP-Module im lokalen Baugruppenträger, der Zugriff auf Webseiten ist jedoch nur über die oben angegebenen Module möglich.

Hinweis

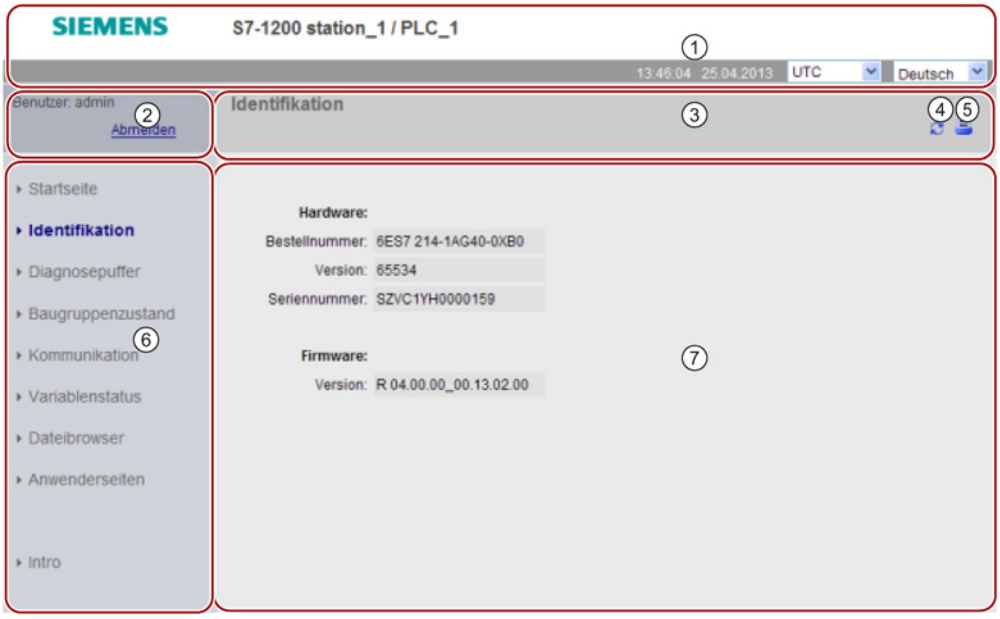
Zugriff auf Standard-Webseiten, wenn über Webserver aktivierte CPs im lokalen Baugruppenträger vorhanden sind

Bei der Verbindung mit den S7-1200 Standard-Webseiten sind bis zu ein oder zwei Minuten lang Verzögerungen möglich, wenn über Webserver aktivierte CPs im lokalen Baugruppenträger installiert sind. Scheinen die Seiten zunächst nicht verfügbar oder werden Fehlermeldungen angezeigt, warten Sie ein oder zwei Minuten und versuchen Sie den Zugriff erneut.

12.6 Standard-Webseiten

12.6.1 Aufbau der Standard-Webseiten

Alle S7-1200 Standard-Webseiten haben einen gemeinsamen Aufbau und verfügen über Links für die Navigation und Bedienelemente für die Seite. Unabhängig davon, ob Sie die Seite auf einem PC oder einem mobilen Gerät aufrufen, hat jede Seite den gleichen Inhalt. Aufbau und Navigationselemente sind jedoch je nach Bildschirmgröße und Auflösung des Geräts unterschiedlich. Auf einem Standard-PC und großen mobilen Geräten ist eine Standard-Webseite wie folgt aufgebaut:



① Webserver-Kopfzeile mit Auswahlmöglichkeiten für PLC-Ortszeit oder UTC-Zeit und die Anzeigesprache (Seite 168)

② Anmelden oder Abmelden

③ Kopfzeile der Standard-Webseite mit dem Namen der angezeigten Seite. In diesem Beispiel handelt es sich um die Seite der CPU-Identifikation. Auf einigen der Standard-Webseiten, z. B. der Modulinformationsseite, wird hier auch ein Navigationspfad angezeigt, wenn mehrere Seiten dieses Typs aufrufbar sind.

④ Symbol zum Aktualisieren: bei Seiten mit automatischer Aktualisierung wird die automatische Aktualisierungsfunktion aktiviert bzw. deaktiviert; bei Seiten ohne automatische Aktualisierung wird die Seite mit den aktuellen Daten aktualisiert

⑤ Symbol zum Drucken: erstellt eine druckbare Version der Informationen auf der angezeigten Seite und zeigt sie an

⑥ Navigationsbereich zum Wechseln zu einer anderen Seite

⑦ Inhaltsbereich der spezifischen Standard-Webseite, die Sie gerade anzeigen. In diesem Beispiel handelt es sich um die Seite der CPU-Identifikation.

Seitenaufbau auf mobilen Geräten

Auf Geräten mit einer Breite unter 768 Pixel zeigt der Webserver eine mobile Version jeder Seite an. Die Seite wird ohne Navigationsleiste, Anmeldebereich und Kopfzeile angezeigt und enthält Schaltflächen für die Navigation zwischen Webseiten sowie eine Schaltfläche zum Umschalten auf eine Startseite mit der Navigationsleiste. Sie können jedoch auch die Navigationselemente des mobilen Geräte verwenden. Beispiel: Auf einem mobilen Gerät mit einer Bildschirmbreite unter 768 Pixel wird die Identifikationsseite folgendermaßen im Hochformat angezeigt:

Beachten Sie, dass die Abbildungen der Standard-Webseiten in diesem Kapitel der Darstellung auf einem Standard-PC entsprechen. Für die Darstellung auf mobilen Geräten ist eine Standard-Webseite entsprechend anders aufgebaut.

Hinweis

Standard-Webseiten für CP-Module

Einige CP-Module (Seite 826) bieten Standard-Webseiten an, deren Aussehen und Funktionalität den S7-1200 CPU Standard-Webseiten sehr ähnlich sind. Beschreibungen der CP Standard-Webseiten sind der CP-Dokumentation zu entnehmen.

12.6.2 Anmeldung und Benutzerrechte

Auf dem PC bietet jede der Standard-Webseiten ein Anmeldefenster oberhalb des Navigationsbereichs. Aufgrund von Platzbeschränkungen bieten die Webseiten auf Mobilgeräten eine eigene Anmeldeseite. Die S7-1200 unterstützt mehrere Benutzer-Logins mit verschiedenen Zugriffsstufen (Rechten):

- Diagnoseabruf
- Variablen lesen
- Variablen schreiben
- Variablenzustand lesen
- Variablenzustand schreiben
- Benutzerdefinierte Webseiten öffnen
- In benutzerdefinierte Webseiten schreiben
- Dateien lesen
- Dateien schreiben/löschen
- Betriebszustand wechseln
- LEDs blinken lassen
- Firmware-Update durchführen
- Systemparameter ändern
- Anwendungsparameter ändern

In der Gerätekonfiguration der CPU in STEP 7 konfigurieren Sie in den Eigenschaften der Benutzerverwaltung für den Webserver die Benutzerrollen, weisen Zugriffsstufen (Rechte) und Passwörter zu (Seite 821).

Anmelden

STEP 7 bietet den Standardbenutzer "Jeder" ohne Passwort. Dieser Benutzer hat standardmäßig keine zusätzlichen Rechte und kann nur die Standard-Webseiten Start (Seite 834) und Einführung (Seite 833) öffnen. Sie können dem Benutzer "Jeder" jedoch wie allen anderen von Ihnen konfigurierten Benutzern zusätzliche Rechte erteilen:

⚠ WARNUNG

Zugriff auf den Webserver

Werden dem Benutzer "Jeder" Rechte erteilt, ist es möglich, sich ohne Passwort am Webserver anzumelden. Nicht berechtigter Zugriff auf die CPU oder das Einstellen von ungültigen Werten für PLC-Variablen kann den Prozessbetrieb stören und zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Weil der Benutzer "Jeder" mit ausreichenden Rechten Betriebszustandsänderungen vornehmen, PLC-Daten schreiben und Firmware-Updates durchführen kann, empfiehlt Siemens, die folgenden Sicherheitsvorkehrungen einzuhalten:

- Aktivieren Sie den Zugriff auf den Webserver nur über das HTTPS-Protokoll.
- Schützen Sie die Webserver-Benutzer-IDs (Seite 821) mit einem starken Passwort. Starke Passwörter sind mindestens zehn Zeichen lang, bestehen aus Buchstaben, Zahlen und Sonderzeichen, sind keine Wörter, die in einem Wörterbuch gefunden werden können, und sind keine Namen oder Kennungen, die sich aus persönlichen Daten ableiten lassen. Halten Sie das Passwort geheim und ändern Sie es häufig.
- Erweitern Sie nicht die voreingestellten Minimalrechte für den Benutzer "Jeder".
- Führen Sie eine Fehlerprüfung und eine Bereichsprüfung für die Variablen in Ihrer Programmlogik durch, weil Nutzer von Webseiten für die PLC-Variablen ungültige Werte einrichten können.
- Verwenden Sie für die Verbindung mit dem S7-1200 PLC-Webserver von einem Standort außerhalb Ihres geschützten Netzwerks ein sicheres Virtual Private Network (VPN).

Wenn Sie bestimmte Aktionen durchführen möchten, z. B. die Steuerung in einen anderen Betriebszustand versetzen, Werte in den Speicher schreiben oder die Firmware der CPU aktualisieren, brauchen Sie die dafür erforderlichen Rechte. Wenn Sie für die Schutzstufe der CPU (Seite 205) die Option "Kein Zugriff (kompletter Schutz)" eingerichtet haben, kann der Benutzer "Jeder" nicht auf den Webserver zugreifen.

Das Feld für die Anmeldung befindet sich bei der Anzeige auf einem PC oder einem Mobilgerät mit großem Display oben links auf jeder Standard-Webseite.

Auf Mobilgeräten mit kleinem Display befinden sich die Anmeldefelder auf einer eigenen Seite, die über die Startseite aufgerufen werden kann.

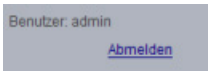
Zum Anmelden gehen Sie wie folgt vor:

1. Geben Sie den Benutzernamen in das Feld "Benutzername" ein.
2. Geben Sie das Passwort in das Feld "Passwort" ein.

Nach dreißig Minuten Inaktivität werden Sie automatisch abgemeldet. Wenn die angezeigte Seite ständig aktualisiert wird, läuft der Anmeldezeitraum nicht ab.

Wenn bei der Anmeldung Fehler auftreten, kehren Sie zur Einführungsseite (Seite 833) zurück und laden das Siemens-Sicherheitszertifikat herunter (Seite 894). Danach können Sie sich fehlerfrei anmelden.

Abmelden



Zum Abmelden klicken Sie auf einem PC oder einem Mobilgerät mit großem Display einfach auf einer beliebigen Seite auf den Link "Abmelden".

Auf einem Mobilgerät mit kleinem Display navigieren Sie von der Startseite zur Seite für die An- und Abmeldung und klicken auf die Schaltfläche "Abmelden".

Nachdem Sie sich abgemeldet haben, können Sie nur die Standard-Webseiten gemäß den Rechten des Benutzers "Jeder" aufrufen und anzeigen. In den Beschreibungen der einzelnen Standard-Webseiten werden die erforderlichen Rechte für die Seite definiert.

Hinweis

Vor dem Schließen des Webservers abmelden

Wenn Sie sich am Webserver angemeldet haben, sollten Sie sich vor dem Schließen des Webbrowsers unbedingt abmelden. Der Webserver unterstützt maximal sieben gleichzeitig angemeldete Benutzer.

12.6.3 Einleitung

Die Einführungsseite ist die Begrüßungsseite für den Einstieg in die S7-1200 Standard-Webseiten.

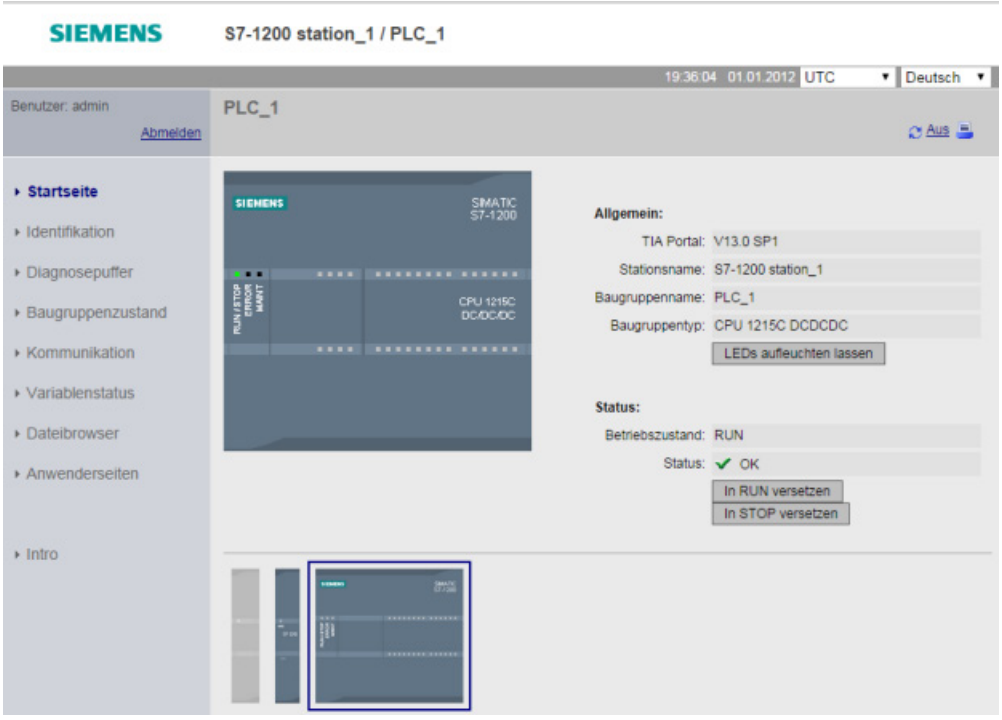
Auf dieser Seite klicken Sie auf "Enter", um die S7-1200 Standard-Webseiten aufzurufen. Oben auf der Seite finden Sie Links zu nützlichen Websites von Siemens sowie einen Link, um das Siemens-Sicherheitszertifikat herunterzuladen (Seite 894). Sie können auch angeben, dass die Einführungsseite bei künftigen Zugriffen auf den Webserver übersprungen werden soll.

12.6.4 Start

Die Startseite zeigt eine Darstellung der CPU oder des CP an, mit der Sie verbunden sind, und führt allgemeine Informationen über das Gerät auf. Für die CPU können Sie über die Schaltflächen den Betriebszustand ändern und die LEDs steuern, sofern Sie sich mit den entsprechenden Rechten angemeldet haben (Seite 829).

Im unteren Teil des Bildschirms wird angezeigt, ob über Webserver aktivierte CP-Module (Seite 826) im lokalen Baugruppenträger mit der S7-1200 CPU konfiguriert und installiert sind. Für den Zugriff auf die Standard-Webseiten fahren Sie mit dem Mauszeiger über die über Webserver aktivierten CP-Module und klicken Sie auf ein Modul. Informationen über die Webseiten der CP-Module finden Sie in der Dokumentation Ihres CP-Moduls. Der Name des CP-Moduls wird angezeigt, wenn Sie die Maus darüber bewegen.

Der Webserver zeigt ferner weitere CM- und CP-Module im lokalen Baugruppenträger, diese können jedoch nicht angeklickt werden, da sie keine Webseiten enthalten. Diese CM- und CP-Module werden hellgrau dargestellt, um kenntlich zu machen, dass sie lediglich angezeigt werden, aber nicht angeklickt werden können.



Beachten Sie, dass die fehlersicheren S7-1200 CPUs auf dieser Seite zusätzliche Daten in Bezug auf die funktionale Sicherheit anzeigen.

12.6.5 Identifikation

Die Identifikationsseite zeigt identifizierende Merkmale der CPU an:

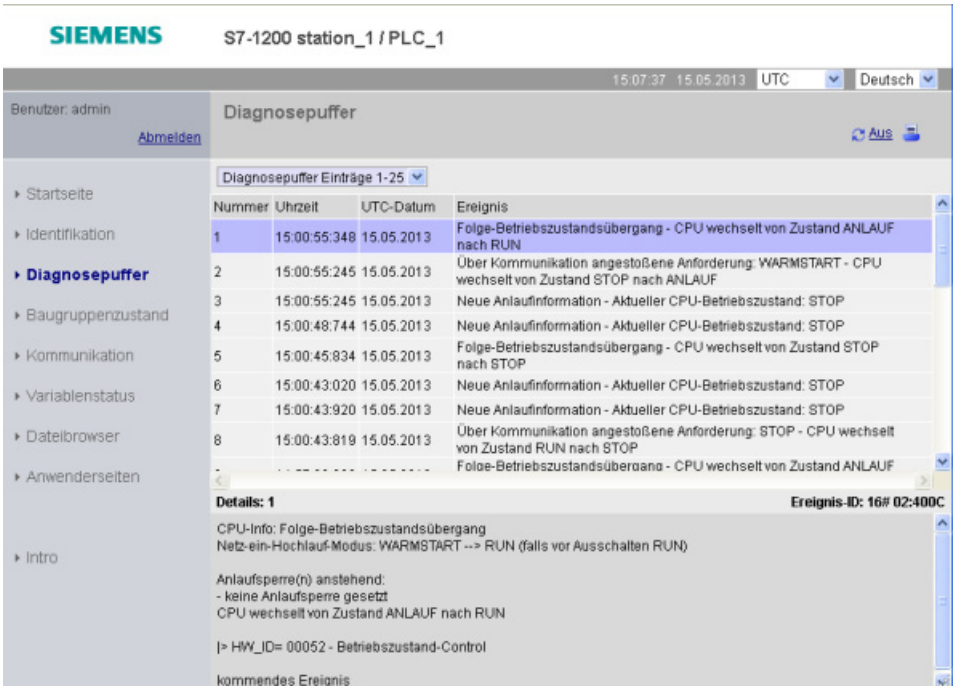
- Seriennummer
- Artikelnummer
- Versionsinformationen

Für die Anzeige der Identifikationsseite benötigen Sie das Recht (Seite 821) zur Abfrage von Diagnosen.

12.6.6 Diagnosepuffer

Die Diagnosepufferseite zeigt Diagnoseereignisse an. In der Auswahlleiste links können Sie wählen, welcher Bereich der Einträge des Diagnosepuffers (Einträge von 1-25 oder von 26-50) angezeigt werden soll. Rechts können Sie einstellen, ob die Zeit als UTC- oder PLC-Ortszeit angezeigt werden soll. Im oberen Bereich der Seite werden diese Diagnoseeinträge mit Uhrzeit und Datum des Auftretens angezeigt.

Im oberen Bereich der Seite können Sie einen einzelnen Eintrag auswählen, um dazu weitere Informationen im unteren Bereich der Seite anzuzeigen.



Zum Anzeigen der Diagnosepufferseite ist das Recht (Seite 821) für den Diagnoseabruf erforderlich.

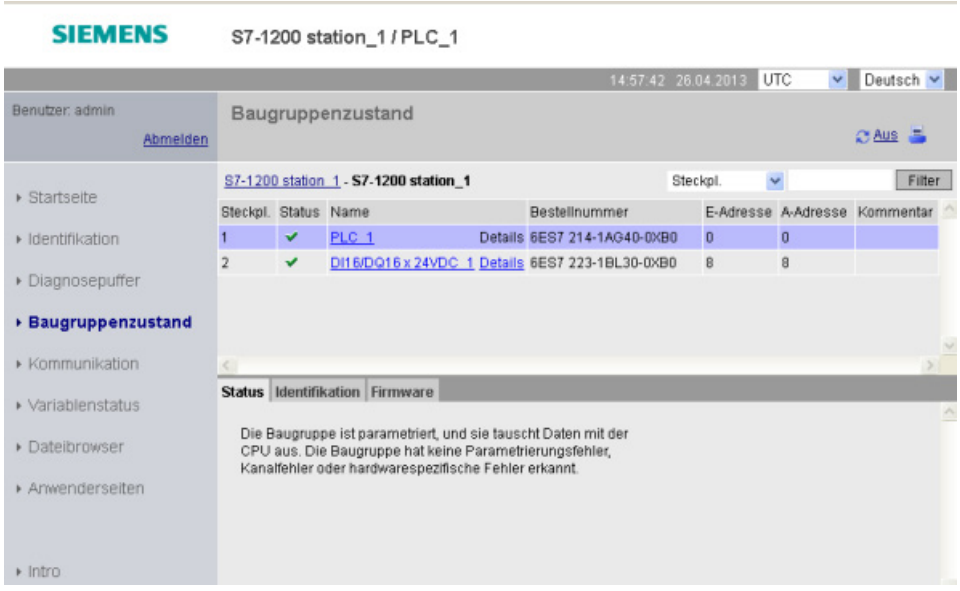
12.6.7 Modulinformationen

Die Modulinformationsseite zeigt Informationen zu allen Modulen im lokalen Baugruppenträger an. Im oberen Abschnitt der Seite wird ein Überblick über die Module angezeigt, im unteren Abschnitt werden Zustand, Identifikation und Firmware-Informationen des ausgewählten Moduls angezeigt. Auf der Modulinformationsseite besteht auch die Möglichkeit, eine Firmware-Aktualisierung durchzuführen.

Zum Anzeigen der Modulinformationsseite ist das Recht (Seite 821) für den Diagnoseabruf erforderlich.

Modulinformationen: Register "Status"

Das Register "Status" im unteren Abschnitt der Modulinformationsseite zeigt eine Beschreibung des aktuellen Zustands des Moduls an, das im oberen Abschnitt ausgewählt ist.



Hinweis

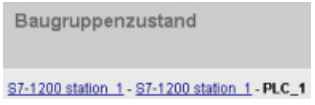
Die Modulinformationsseite auf Mobilgeräten zeigt die Informationen "E-Adresse", "A-Adresse" und "Kommentar" statt in den Spalten der Hauptmodulinformationstabelle im Register "Identifikation" an.

Weitere Einzelheiten anzeigen

Sie können im oberen Abschnitt einen Link auswählen, um weitere Einzelheiten der Modulinformationen des spezifischen Moduls anzuzeigen. Module mit Untermodulen haben für jedes Untermodul einen eigenen Link. Die Art der angezeigten Informationen richtet sich nach dem jeweils ausgewählten Modul. Zum Beispiel werden anfänglich in den Modulinformationen der Name der SIMATIC 1200 Station, eine Statusanzeige und ein Kommentar angezeigt. Wenn Sie den Link bis zur CPU-Ebene erweitern, zeigen die Modulinformationen die Namen der digitalen und analogen Eingänge und Ausgänge des CPU-Modells an (z. B. "DI14/DO10", "AI2") sowie Adressierungsinformationen für die E/A, Statusanzeigen, Steckplatznummern und Kommentare.

Steckpl.	Status	Name		Bestellnummer	E-Adresse	A-Adresse	Kommentar
1.1	✔	DI14/DO10_1	Details	6ES7 214-1AG40-0XB0	0	0	
1.2	✔	AI2_1	Details	6ES7 214-1AG40-0XB0	64	---	
1.16	✔	HSC_1	Details	6ES7 214-1AG40-0XB0	1000	---	
1.17	✔	HSC_2	Details	6ES7 214-1AG40-0XB0	1004	---	
1.18	✔	HSC_3	Details	6ES7 214-1AG40-0XB0	1008	---	
1.19	✔	HSC_4	Details	6ES7 214-1AG40-0XB0	1012	---	
1.20	✔	HSC_5	Details	6ES7 214-1AG40-0XB0	1016	---	
1.21	✔	HSC_6	Details	6ES7 214-1AG40-0XB0	1020	---	
1.32	✔	Pulse_1	Details	6ES7 214-1AG40-0XB0	---	1000	

Beim Erweitern der Anzeige gibt die Modulinformationsseite den entsprechenden Pfad an. Sie können auf jeden Link in diesem Pfad klicken, um zu einer höheren Ebene zurückzukehren.



Felder zum Sortieren

Wenn in der Liste mehrere Module angezeigt werden, können Sie auf den Spaltentitel eines Felds klicken, um anhand dieses Felds auf- oder absteigend zu sortieren.

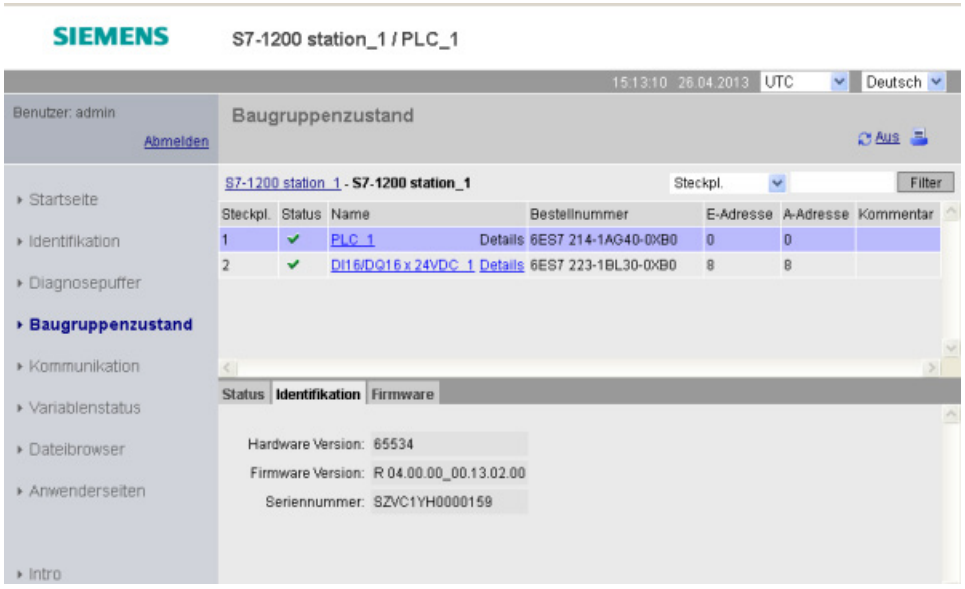
Hinweis: Diese Funktion ist auf der chinesischen Modulinformationsseite noch nicht verfügbar.

Modulinformationen filtern

Sie können jedes Feld in der Liste der Modulinformationen filtern. Wählen Sie in der Klappliste den Namen des Felds, dessen Daten Sie filtern möchten. Geben Sie Text in das zugehörige Textfeld ein und klicken Sie auf den Link "Filter". Die Liste wird aktualisiert und zeigt nur die Module an, die Ihren Filterkriterien entsprechen.

Modulinformationen: Register "Identifikation"

Das Register "Identifikation" zeigt die Seriennummer und die Versionsnummer des ausgewählten Moduls an.



Modulinformationen: Register "Firmware"

Im Register "Firmware" der Modulinformationsseite werden Informationen über die Firmware des ausgewählten Moduls angezeigt. Wenn Sie das Recht (Seite 821) zur Durchführung eines Firmware-Updates haben, können Sie auch die Firmware der CPU oder anderer Module im lokalen Baugruppenträger, die Firmware-Updates unterstützen, aktualisieren.

Hinweis

Sie können nur S7-1200 CPUs ab Version 3.0 mit der Firmware-Update-Funktion aktualisieren.

Die CPU muss sich im Betriebszustand STOP befinden, um ein Firmware-Update durchzuführen. Befindet sich die CPU im Betriebszustand STOP, klicken Sie auf die Schaltfläche "Durchsuchen", um eine Firmware-Datei auszuwählen. Firmware-Updates stehen auf der Kunden-Support-Website (http://support.automation.siemens.com) zur Verfügung.

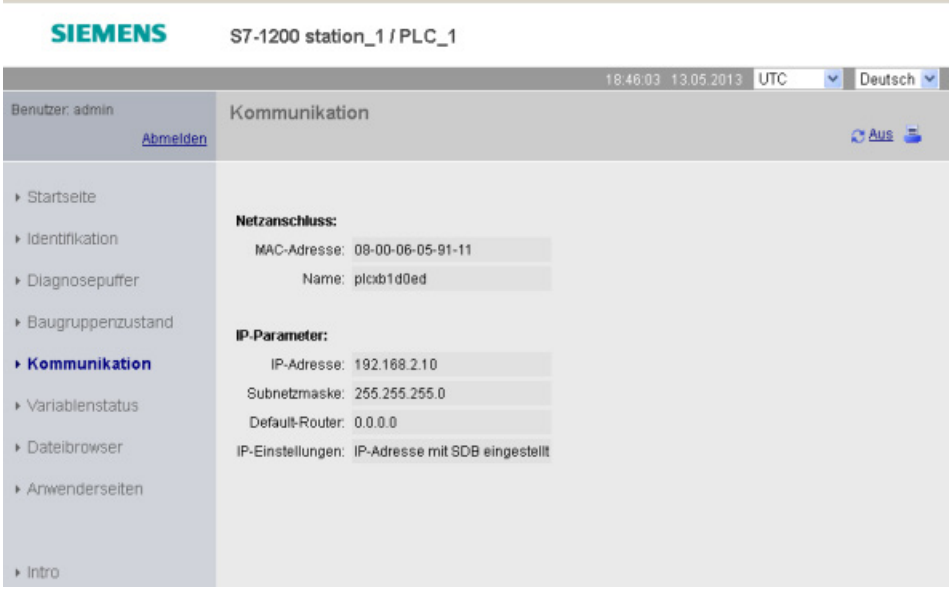
Während der Aktualisierung wird auf der Seite eine Meldung angezeigt, dass die Aktualisierung gerade durchgeführt wird. Nach Abschluss der Aktualisierung werden auf der Seite die Artikelnummer und die Versionsnummer der aktualisierten Firmware angezeigt. Wenn Sie die Firmware der CPU oder eines Signalboards aktualisiert haben, startet der Webserver die CPU neu.

Hinweis

Ein Firmware-Update können Sie auch über STEP 7 (Seite 1124) oder mittels einer Memory Card (Seite 146) durchführen.

12.6.8 Kommunikation

Die Kommunikationsseite zeigt die Parameter der angeschlossenen CPU einschließlich MAC-Adresse, IP-Adresse und IP-Einstellungen der CPU an.



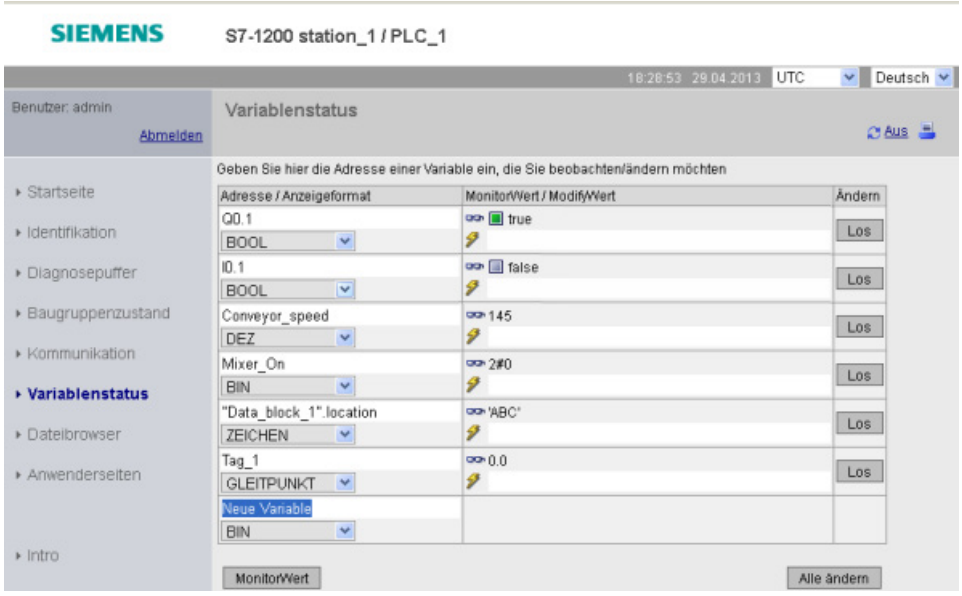
Zum Anzeigen der Kommunikationsseite ist das Recht für den Diagnoseabruf erforderlich.

12.6.9 Variablenzustand

Auf der Variablenzustandsseite können Sie alle E/A oder Speicherdaten in Ihrer CPU anzeigen. Sie können eine direkte Adresse (wie E0.0), einen PLC-Variablennamen oder eine Variable aus einem spezifischen Datenbaustein eingeben. Bei Datenbausteinvariablen setzen Sie den Bausteinnamen in doppelte Anführungszeichen. Für jeden überwachten Wert können Sie ein Anzeigeformat für die Daten auswählen. Sie können so lange im Rahmen der Seiteneinschränkung Werte eingeben, bis Sie alle gewünschten Werte spezifiziert haben. Die überwachten Werte werden automatisch angezeigt und standardmäßig aktualisiert, sofern Sie diese Aktualisierung nicht über "Off" oben rechts auf der Seite ausschalten. Ist die automatische Aktualisierung deaktiviert, können Sie sie über "On" wieder aktivieren.

Zum Anzeigen der Variablenzustandsseite ist das Recht zum Lesen des Variablenstatus erforderlich. Damit eine Variable auf der Variablenstatusseite angezeigt werden kann, müssen Sie sie in STEP 7 als "Erreichbar aus HMI" konfigurieren.

Wenn Sie sich als Benutzer mit dem Recht (Seite 829) zum Schreiben des Variablenstatus anmelden, können Sie die Datenwerte auch ändern. Geben im entsprechenden Feld "Modify Value" die Werte ein, die Sie festlegen möchten. Klicken Sie neben einem Wert auf die Schaltfläche "Go", um den Wert in die CPU zu schreiben. Sie können auch mehrere Werte eingeben und auf "Modify All Values" klicken, um alle Werte in die CPU zu schreiben. Die Schaltflächen und Spaltenbeschriftungen zum Ändern werden nur angezeigt, wenn Sie das Recht zum Schreiben des Variablenstatus haben.



Wenn Sie die Variablenzustandsseite verlassen und wieder zurückkehren, sind Ihre Eingaben auf der Variablenzustandsseite nicht gespeichert. Sie können die Seite als Lesezeichen speichern und das Lesezeichen erneut aufrufen, damit die gleichen Einträge angezeigt werden. Wenn Sie die Seite nicht als Lesezeichen speichern, müssen Sie die Variablen erneut eingeben.

Hinweis

Beachten Sie bei der Nutzung der Standardseite für den Variablenzustand Folgendes:

- Setzen Sie alle Zeichenkettenänderungen in einzelne Hochkommata.
- Die Variablenzustandsseite kann keine Variablen beobachten oder ändern, die eines der folgenden Zeichen enthalten: &, <, (, +, ,(Komma), ., [,], $ oder %. Die Variable "Takt_2,5Hz" beispielsweise können Sie nicht beobachten.
- Auf der Variablenzustandsseite können Sie keine Zeichenketten mit mehr als 198 Zeichen ändern.
- Gehen Sie bei der Eingabe eines Werts vom Datentyp Real oder LReal in Exponentialdarstellung auf der Variablenzustandsseite wie folgt vor:
  - Um einen Realzahlenwert (Real oder LReal) mit positivem Exponenten (wie +3.402823e+25) einzugeben, geben Sie den Wert in einem der folgenden Formate ein:

    +3.402823e25

    +3.402823e+25
  - Um einen Realzahlenwert (Real oder LReal) mit negativem Exponenten (wie +3.402823e-25) einzugeben, geben Sie den Wert wie folgt ein:

    +3.402823e-25
  - Achten Sie darauf, dass der Anteil der Mantisse des Realzahlenwerts in der Exponentialdarstellung einen Dezimalpunkt enthält. Wird der Dezimalpunkt weggelassen, führt dies dazu, dass der Wert zu einem unerwarteten ganzzahligen Wert verändert wird. Beispiel: Geben Sie -1.0e8 und nicht -1e8 ein.
- Die Variablenzustandsseite unterstützt nur 15 Ziffern für einen Wert vom Typ LReal (unabhängig von der Position des Dezimalpunkts). Wenn Sie mehr als 15 Ziffern eingeben, entsteht ein Rundungsfehler.

Einschränkungen auf der Variablenzustandsseite:

- Die maximale Anzahl von Variableneinträgen pro Seite beträgt 50.
- Die maximale Anzahl von Zeichen für die URL der Variablenzustandsseite beträgt 2083. Sie sehen die URL, die Ihrer aktuellen Variablenseite entspricht, in der Adressleiste Ihres Browsers.
- Hinsichtlich des Zeichenanzeigeformats zeigt die Seite Hexadezimalwerte an, wenn es sich nach Auswertung des Browsers bei den tatsächlichen CPU-Werten nicht um gültige ASCII-Zeichen handelt.

Hinweis

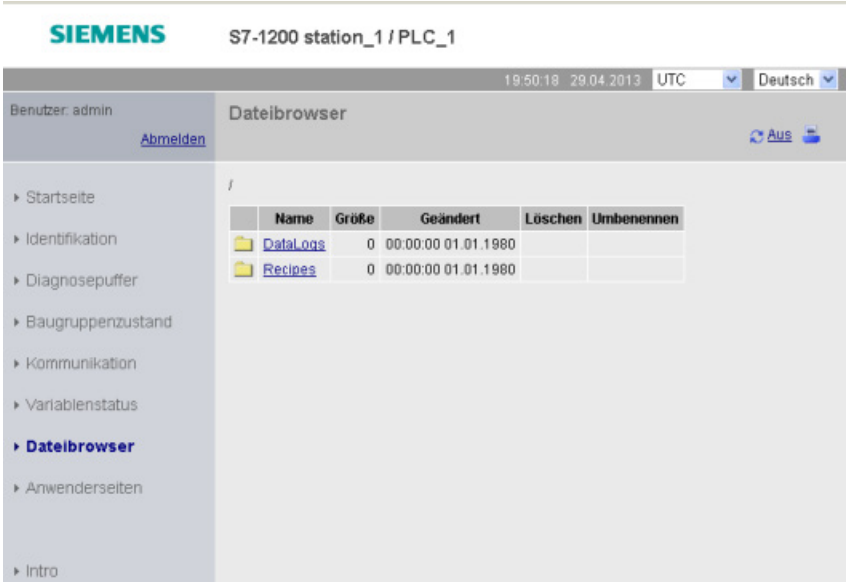
Wenn ein Variablenname Sonderzeichen enthält, so dass der Name als Eintrag auf der Variablenzustandsseite abgelehnt wird, können Sie den Variablennamen in doppelte Anführungszeiten setzen. In den meisten Fällen erkennt dann die Variablenzustandsseite den Variablennamen.

12.6.10 Dateibrowser

Die Dateibrowser-Seite bietet Zugriff auf Dateien im internen Ladespeicher der CPU und auf der Memory Card (externer Ladespeicher). Die Dateibrowser-Seite zeigt zunächst den Stammordner des Ladespeichers an, in dem sich die Ordner "DataLogs" und "Recipes" befinden, doch es werden bei Verwendung einer Memory Card auch alle anderen von Ihnen angelegten Ordner angezeigt.

Ihre Art des Dateizugriffs auf Dateien und Ordner hängt von Ihren Benutzerrechten (Seite 821) ab. Alle Benutzer mit Rechten zum Lesen von Dateien können die Dateien und Ordner im Dateibrowser anzeigen. Unabhängig von Ihren Login-Rechten können Sie die Ordner "DataLogs" und "Recipes" nicht löschen, doch wenn Sie auf der Memory Card benutzerspezifische Ordner angelegt haben, können Sie diese Ordner löschen, sofern Sie sich mit Rechten zum Schreiben und Löschen von Dateien angemeldet haben.

Klicken Sie auf einen Ordner, um auf die einzelnen Dateien in dem Ordner zuzugreifen.



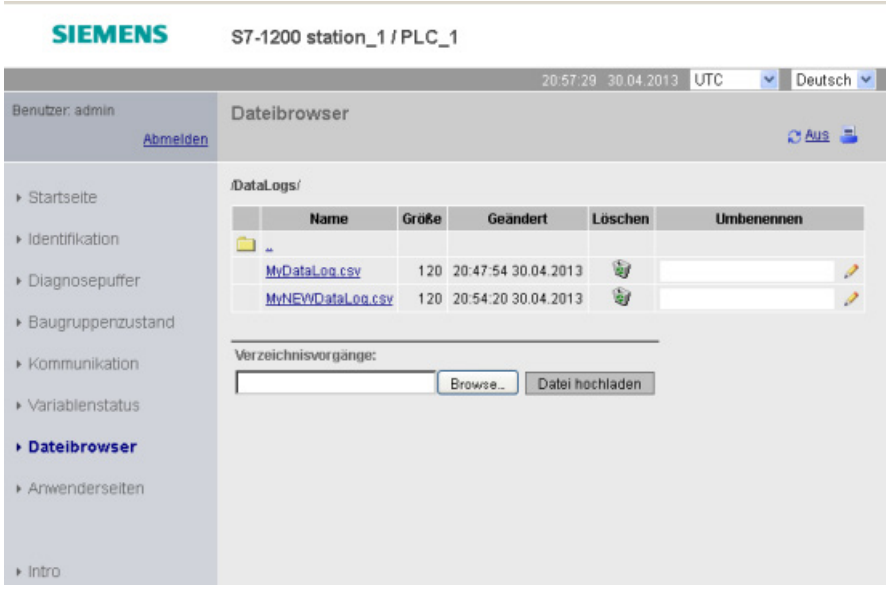
Datenprotokolle

Über den Ordner "DataLogs" können Sie alle Datenprotokolldateien öffnen. Wenn Sie sich mit dem Recht (Seite 821) zum Schreiben und Löschen von Dateien angemeldet haben, können Sie Dateien auch löschen, umbenennen oder hochladen. Die Datenprotokolldateien sind im CSV-Dateiformat (durch Komma getrennte Werte) gespeichert. Sie können sie auf Ihrem Computer speichern oder in Microsoft Excel (Standard) oder einem anderen Programm öffnen.

Hinweis

Zeitstempel für Datenprotokolle

Der Webserver zeigt die Zeitstempel für die Datenprotokolle je nach Ihrer Auswahl im oberen Bereich der Seite entweder entsprechend der UTC-Zeit oder entsprechend der Ortszeit des PLCs an.



Hinweis: Die Optionen zum Löschen und Umbenennen sind nur verfügbar, wenn Sie mit Rechten zum Schreiben und Löschen von Dateien angemeldet sind.

Hinweis

Verwaltung von Datenprotokollen

Speichern Sie maximal 1000 Datenprotokolle in einem Dateisystem. Bei Überschreiten dieser Anzahl kann es passieren, dass der Webserver nicht mehr genügend CPU-Ressourcen zur Verfügung hat, um die Datenprotokolle anzuzeigen.

Wenn Sie bemerken, dass die Webseite des Dateibrowsers die Datenprotokolle nicht anzeigen kann, müssen Sie die CPU in den Betriebszustand STOP versetzen, um Datenprotokolle anzeigen und löschen zu können.

Stellen Sie über die Verwaltung Ihrer Datenprotolle sicher, dass Sie lediglich die erforderliche Anzahl Datenprotokolle speichern und die maximale Anzahl von 1000 Datenprotokollen nicht überschreiten.

Mit einem Datenprotokoll in Excel arbeiten

Die Protokolldatei steht im amerikanischen/englischen CSV-Format (durch Komma getrennte Werte). Um dieses Format in Excel auf nicht amerikanischen/englischen Systemen zu öffnen, müssen Sie die Datei mit spezifischen Einstellungen nach Excel importieren (Seite 896).

Rezeptdateien

Wie im DataLogs-Ordner werden im Recipe-Ordner alle Rezeptdateien angezeigt, die im Ladespeicher vorhanden sind. Auch die Rezeptdateien sind im CSV-Format und können in Microsoft Excel oder einem anderen Programm geöffnet werden. Wie bei den Datenprotokollen benötigen Sie Änderungsrechte, um Rezeptdateien zu löschen, zu ändern und zu speichern, umzubenennen oder hochzuladen.

Hochladen von Dateien und automatisches Aktualisieren von Seiten

Wenn Sie mit dem Hochladen einer Datei beginnen, wird der Ladevorgang so lange fortgesetzt, wie Sie auf der Webseite des Dateibrowsers bleiben. Wenn Sie die automatische Aktualisierung der Webserverseiten im Abstand von zehn Sekunden aktiviert haben, sehen Sie den Fortschritt des Datei-Uploads bei jeder Seitenaktualisierung. Wenn Sie beispielsweise eine Datei von 2 MB hochladen, werden Ihnen Aktualisierungen der Dateigröße möglicherweise bei 2500, 5000, 10.000, 15.000 und 20.000 Byte des Ladevorgangs angezeigt.

Wenn Sie die Seite des Dateibrowsers verlassen, bevor der Ladevorgang abgeschlossen ist, erhalten Sie keine vollständige Datei. Wenn Sie auf die Seite zurückkehren, zeigt der Dateibrowser den Dateinamen und die Größe der Datei zu dem Zeitpunkt an, zu dem der Ladevorgang gestoppt wurde. Ansonsten erhalten Sie keinen weiteren Hinweis darauf, dass die Datei unvollständig ist. Um sicherzugehen, dass die Datei vollständig hochgeladen wird, bleiben Sie auf der Seite im Dateibrowser, bis die angezeigte Dateigröße der tatsächlichen Dateigröße entspricht.

Weitere Informationen

Weitere Informationen zur Programmierung mit den Datenprotokollanweisungen und zum Importieren (Seite 426) und Exportieren (Seite 424) von Rezepten finden Sie im Kapitel Rezepte und Datenprotokolle (Seite 419).

12.7 Benutzerdefinierte Webseiten

Für den S7-1200 Webserver können Sie auch eigene anwendungsspezifische HTML-Seiten mit Daten des Zielsystems anlegen.

⚠ WARNUNG

Nicht berechtigter Zugriff auf die CPU über benutzerdefinierte Webseiten

Nicht berechtigter Zugriff auf die CPU über benutzerdefinierte Webseiten kann den Prozessbetrieb unterbrechen, was zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen kann.

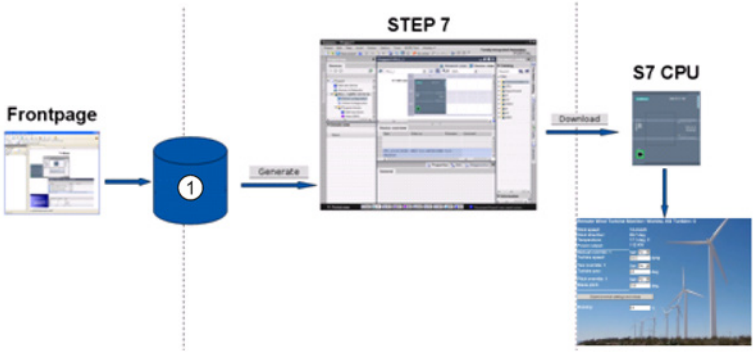
Die nicht sichere Codierung von benutzerdefinierten Webseiten führt zu Sicherheitslücken, über die Angriffe mittels Cross-Site-Scripting (XSS), Code Injection oder andere Methoden möglich sind.

Schützen Sie Ihre S7-1200 CPU durch sichere Installation vor unberechtigtem Zugriff. Informationen dazu finden Sie in den "Operational Guidelines" auf der Industrial Security-Website (http://www.siemens.com/industrialsecurity).

Sie erstellen benutzerdefinierte Webseiten mit einem HTML-Editor Ihrer Wahl und laden die Seiten in die CPU, von wo aus sie über das Menü der Standard-Webseiten aufrufbar sind. Hierfür sind mehrere Tätigkeiten durchzuführen:

- HTML-Seiten mit einem HTML-Editor wie Microsoft Frontpage anlegen (Seite 848)
- AWP-Befehle in HTML-Kommentare im HTML-Code einfügen (Seite 849):Bei den AWP-Befehlen handelt es sich um einen fest vorgegebenen Satz Befehle, den Siemens für den Zugriff auf CPU-Informationen bereitstellt.
- STEP 7 zum Lesen und Verarbeiten von HTML-Seiten konfigurieren (Seite 865)
- Bausteine aus den HTML-Seiten generieren (Seite 865)
- STEP 7 für die Steuerung der Verwendung der HTML-Seiten programmieren (Seite 866)
- Bausteine übersetzen und in die CPU laden (Seite 868)
- Über Ihren PC auf die benutzerdefinieren Webseiten zugreifen (Seite 868)

Dieser Prozess wird im Folgenden dargestellt:



① HTML-Dateien mit eingebetteten AWP-Befehlen

12.7.1 HTML-Seiten anlegen

Um Ihre eigenen HTML-Seiten zur Verwendung mit dem Webserver zu erstellen, können Sie ein Softwarepaket Ihrer Wahl verwenden. Achten Sie darauf, dass Ihr HTML-Code mit den HTML-Standards des W3C (World Wide Web Consortium) konform ist. STEP 7 führt keine Überprüfung Ihrer HTML-Syntax durch.

Sie können eine Software verwenden, bei der Sie im WYSIWYG- oder Design-Layout-Modus programmieren können, doch Sie müssen Ihren HTML-Code auch im reinen HTML-Format bearbeiten können. Die meisten Web-Authoring-Tools bieten diese Art der Bearbeitung. Falls nicht, können Sie den HTML-Code einfach in einem Texteditor bearbeiten. Nehmen Sie die folgende Zeile in Ihre HTML-Seite auf, um UTF-8 als Zeichensatz der Seite festzulegen:

```
<meta http-equiv="content-type" content="text/html; charset=utf-8">
```

Achten Sie auch darauf, dass Sie die Datei im Editor ebenfalls in der UTF-8-Zeichencodierung speichern:

Sie können in STEP 7 die Inhalte Ihrer HTML-Seiten in STEP 7-Datenbausteine übersetzen. Diese Datenbausteine bestehen aus einem Steuerdatenbaustein, der die Anzeige der Webseiten regelt, und einem oder mehreren Datenbausteinfragmenten mit den übersetzten Webseiten. Beachten Sie, dass umfangreiche HTML-Seiten, insbesondere Seiten mit vielen Bildern, einen beträchtlichen Platz im Ladespeicher (Seite 869) für die DB-Fragmente belegen. Wenn der interne Ladespeicher Ihrer CPU für Ihre benutzerdefinierten Webseiten nicht ausreicht, stellen Sie über eine Memory Card (Seite 136) externen Ladespeicher zur Verfügung.

Um Ihren HTML-Code so zu programmieren, dass Daten aus der S7-1200 verwendet werden, können Sie AWP-Befehle (Seite 849) als HTML-Kommentare einfügen. Speichern Sie abschließend Ihre HTML-Seiten auf Ihrem PC und notieren Sie sich den Speicherpfad.

Hinweis

Die maximale Dateigröße für HTML-Dateien mit AWP-Befehlen beträgt 64 KB. Ihre Dateien dürfen diesen Grenzwert nicht überschreiten.

Benutzerdefinierte Webseiten aktualisieren

Benutzerdefinierte Webseiten werden nicht automatisch aktualisiert. Sie können wählen, ob Sie den HTML-Code so programmieren, dass die Seite aktualisiert wird oder nicht. Bei Seiten, die PLC-Daten anzeigen, bleiben die Daten durch regelmäßige Aktualisierung auf dem aktuellen Stand. Bei HTML-Seiten, die als Formulare zur Dateneingabe dienen, kann eine Aktualisierung die Dateneingabe durch den Benutzer beeinträchtigen. Wenn Ihre gesamte Seite automatisch aktualisiert werden soll, können Sie diese Zeile in Ihre HTML-Kopfzeile aufnehmen, wobei "10" für die Anzahl der Sekunden zwischen zwei Aktualisierunsgvorgängen steht:

```
<meta http-equiv="Refresh" content="10">
```

Sie können für die Seiten- und Datenaktualisierung auch JavaScripts oder andere HTML-Techniken nutzen. Hierfür ziehen Sie bitte eine Dokumentation zu HTML oder JavaScripts hinzu.

12.7.2 Vom S7-1200 Webserver unterstützte AWP-Befehle

Der S7-1200 Webserver bietet AWP-Befehle, die Sie zu folgenden Zwecken als HTML-Befehle in Ihre benutzerdefinierten Webseiten einfügen können:

- Variablen lesen (Seite 851)
- Variablen schreiben (Seite 852)
- Sondervariablen lesen (Seite 854)
- Sondervariablen schreiben (Seite 855)
- Enum-Typen definieren (Seite 858)
- Variablen zu Enum-Typen zuweisen (Seite 858)
- Datenbausteinfragmente erstellen (Seite 860)

Allgemeine Syntax

Mit Ausnahme des Befehls zum Lesen einer Variablen haben die AWP-Befehle die folgende Syntax:

```
<!-- AWP_ <Befehlsname und Parameter> -->
```

Sie nutzen die AWP-Befehle in Verbindung mit typischen HTML-Formularbefehlen, um Variablen in die CPU zu schreiben.

In den Beschreibungen der AWP-Befehle auf den folgenden Seiten werden die folgenden Konventionen verwendet:

- In eckigen Klammern [ ] angegebene Elemente sind optional.
- In spitzen Klammern < > dargestellte Elemente sind anzugebende Parameterwerte.
- Fragezeichen sind ein tatsächlicher Teil des Befehls. Sie müssen wie angezeigt vorhanden sein.
- Sonderzeichen in Variablen- oder Datenbausteinnamen müssen, je nach Verwendung, durch eine Escape-Zeichenfolge gekennzeichnet oder in Anführungszeichen gesetzt werden (Seite 862).

Fügen Sie die AWP-Befehle mit einem Texteditor oder im HTML-Bearbeitungsmodus in Ihre Webseiten ein.

Hinweis

Erwartete Syntax von AWP-Befehlen

Das Leerzeichen nach "<!--" und das Leerzeichen vor "-->" in der Formulierung eines AWP-Befehls sind für die korrekte Übersetzung des Befehls entscheidend. Ein Weglassen der Leerzeichen kann dazu führen, dass die Übersetzung des richtigen Codes nicht möglich ist. In diesem Fall wird keine Fehlermeldung angezeigt.

Überblick über die AWP-Befehle

Genauere Erläuterungen zu den einzelnen AWP-Befehlen finden Sie in den nächsten Abschnitten, hier zunächst ein Überblick über die Befehle:

Variablen lesen

```
:=<Varname>:
```

Variablen schreiben

```
<!-- AWP_In_Variable Name='<Varname1>' [Use='<Varname2>'] ... -->
```

Dieser AWP-Befehl deklariert lediglich, dass in die Variable der Namensklausel geschrieben werden kann. Ihr HTML-Code führt namentliche Schreibvorgänge in die Variable aus <input>, <select> oder einer anderen HTML-Anweisung innerhalb eines HTML-Formulars durch.

Sondervariablen lesen

```
<!-- AWP_Out_Variable Name='<Typ>:<Name>' [Use='<Varname>'] -->
```

Sondervariablen schreiben

```
<!-- AWP_In_Variable Name='<Typ>:<Name>' [Use='<Varname>']-->
```

Enum-Typen definieren

```
<!-- AWP_Enum_Def Name='<Name Enum-
Typ>' Values='<Wert>, <Wert>,... ' -->
```

Enum-Typen definieren

```
<!-- AWP_In_Variable Name='<Varname>' Enum="<Name Enum-Typ>" -->
<!-- AWP_Out_Variable Name='<Varname>' Enum="<Name Enum-Typ>" -->
```

Fragmente erstellen

```
<!-- AWP_Start_Fragment Name='<Name>' [Type=<Typ>][ID=<ID>] -->
```

Fragmente importieren

```
<!-- AWP_Import_Fragment Name='<Name>' -->
```

12.7.2.1 Variablen lesen

Benutzerdefinierte Webseiten können Variablen (PLC-Variablen) und Datenbausteinvariablen aus der CPU lesen, vorausgesetzt, Sie haben die Variablen als "Erreichbar aus HMI" konfiguriert.

Syntax

```
:=<Varname>:
```

Parameter

<Varname>	Die zu lesende Variable, bei der es sich um eine PLC-Variable aus Ihrem STEP 7-Programm, eine Datenbausteinvariable, E/A oder eine Adresse im Speicher handeln kann. Setzen Sie bei Speicher- und E/A-Adressen oder Aliasnamen (Seite 862) den Variablennamen nicht in Anführungszeichen. Setzen Sie bei PLC-Variablen den Variablennamen in doppelte Anführungszeichen. Bei Datenbausteinvariablen setzen Sie nur den Bausteinnamen in doppelte Anführungszeichen. Der Variablenname befindet sich außerhalb der Anführungszeichen. Beachten Sie, dass Sie den Namen des Datenbausteins und nicht die Nummer des Datenbausteins verwenden.

Beispiele

```
:="Fördergeschwindigkeit"::="Mein_Datenbaustein".Merker1:
:=I0.0:
:=MW100:
```

Beispiel für das Lesen einer über Alias angegebenen Variable

```
<!-- AWP_Out_Variable Name='Merker1'
Use='"Mein_Datenbaustein".Merker1' -->
:=Merker1:
```

Hinweis

Wie Sie Aliasnamen für PLC-Variablen und Datenbausteinvariablen definieren, wird unter Alias für einen Variablenverweis nutzen (Seite 857) beschrieben.

Wenn ein Variablenname oder Datenbausteinname Sonderzeichen enthält, müssen Sie zusätzlich Anführungszeichen oder Escape-Zeichen verwenden. Dies wird unter Handhabung von Variablennamen mit Sonderzeichen (Seite 862) beschrieben.

12.7.2.2 Variablen schreiben

Benutzerdefinierte Webseiten können Daten in die CPU schreiben. Hierfür geben Sie über einen AWP-Befehl eine Variable in der CPU an, in die über die HTML-Seite geschrieben werden soll. Die Variable muss über den PLC-Variablennamen oder den Variablennamen des Datenbausteins angegeben werden. Sie können in einer Anweisung mehrere Variablennamen deklarieren. Um die Daten in die CPU zu schreiben, verwenden Sie den Standard-HTTP-Befehl POST.

Eine typische Verwendung ist die Gestaltung eines Formulars auf Ihrer HTML-Seite mit Texteingabefeldern oder Listenauswahlfeldern, die den schreibbaren CPU-Variablen entsprechen. Wie bei allen benutzerdefinierten Webseiten generieren Sie dann die Bausteine in STEP 7, damit diese in Ihr STEP 7-Programm aufgenommen werden. Wenn ein Benutzer mit Rechten zum Ändern von Variablen anschließend diese Seite aufruft und Daten in die Eingabefelder einträgt oder ein Feld in einer Auswahlliste auswählt, konvertiert der Webserver die Eingaben in den entsprechenden Datentyp für die Variable und schreibt den Wert in die Variable in der CPU. Beachten Sie, dass die Namensklausel für HTML-Eingabefelder und HTML-Auswahllisten eine Syntax verwendet, die für die Namensklausel des Befehls AWP_In_Variable typisch ist. Üblicherweise setzen Sie den Namen in einfache Anführungszeichen und beim Verweis auf einen Datenbaustein setzen Sie den Datenbausteinnamen in doppelte Anführungszeichen.

Weitere Informationen zur Formularverwaltung finden Sie in der HTML-Dokumentation.

Syntax

```
<!-- AWP_In_Variable Name='<Varname1>' [Use='<Varname2>'] ... -->
```

Parameter

<Varname1>	Wenn keine Use-Klausel angegeben ist, ist Varname1 die Variable, in die geschrieben wird. Hierbei kann es sich um eine PLC-Variable aus Ihrem STEP 7-Programm oder um eine Variable aus einem spezifischen Datenbaustein handeln. Ist eine Use-Klausel angegeben, ist Varname1 ein alternativer Name für die in <Varname2> referenzierte Variable (Seite 857). Es handelt sich um einen lokalen Namen innerhalb der HTML-Seite.
<Varname2>	Wenn eine Use-Klausel angegeben ist, ist Varname2 die Variable, in die geschrieben wird. Hierbei kann es sich um eine PLC-Variable aus Ihrem STEP 7-Programm oder um eine Variable aus einem spezifischen Datenbaustein handeln.

Sowohl bei Namens- als auch bei Verwendungsklauseln ist der vollständige Name in einfache Anführungszeichen zu setzen. Innerhalb der einfachen Anführungszeichen setzen Sie eine PLC-Variable sowie einen Datenbausteinnamen in doppelte Anführungszeichen. Der Datenbausteinname befindet sich in doppelten Anführungszeichen, jedoch nicht der Variablenname des Datenbausteins. Beachten Sie, dass Sie für Datenbausteinvariablen den Namen des Bausteins und nicht die Nummer des Datenbausteins verwenden.

Beispiele mit HTML-Eingabefeldern

```
<!-- AWP_In_Variable Name='"Zielstufe"' -->
<form method="post">
<p>Eingabe Zielstufe: <input name='"Zielstufe"' type="text" />
</p>
</form>

<!-- AWP_In_Variable Name='"Datenbaustein_1".Bremsen' -->
<form method="post">
<p>Bremsen: <input name='"Datenbaustein_1".Bremsen' type="text" />
%</p>
</form>

<!-- AWP_In_Variable Name='"Bremsen"'
Use='"Datenbaustein_1".Bremsen' -->
<form method="post">
<p>Bremsen: <input name='"Bremsen"' type="text" /> %</p>
</form>
```

Beispiel mit einer HTML-Auswahlliste

```
<!-- AWP_In_Variable
Name='"Datenbaustein_1".FreigabeManuelleÜbersteuerung'-->
<form method="post">
<select name='"Datenbaustein_1".FreigabeManuelleÜbersteuerung'>
<option value=:"Datenbaustein_1".FreigabeManuelleÜbersteuerung:>
</option>
<option value=1>Ja</option>
<option value=0>Nein</option>
</select><input type="submit" value="Einstellung absenden" /></form>
```

Hinweis

Nur ein Benutzer mit Rechten zum Ändern von Variablen kann Daten in die CPU schreiben. Der Webserver ignoriert die Befehle, wenn der Benutzer keine Änderungsrechte hat.

Wenn ein Variablenname oder Datenbausteinname Sonderzeichen enthält, müssen Sie zusätzliche Anführungszeichen oder Escape-Zeichen verwenden. Dies wird unter "Handhabung von Variablennamen mit Sonderzeichen (Seite 862)" beschrieben.

12.7.2.3 Sondervariablen lesen

Der Webserver bietet die Möglichkeit, Werte aus dem PLC-Gerät zu lesen und diese in Sondervariablen in der HTTP-Antwortkopfzeile zu speichern. Sie können z. B. einen Pfadnamen aus einer PLC-Variable auslesen, um die URL über die Sondervariable HEADER:Speicherort zu einem anderen Speicherort umzuleiten.

Syntax

```
<!-- AWP_Out_Variable Name='<Typ>:<Name>' [Use='<Varname>'] -->
```

Parameter

<Typ>	Der Typ der Sondervariablen; möglich sind: HEADER COOKIE_VALUE COOKIE_EXPIRES
<Name>	In der HTTP-Dokumentation finden Sie eine Liste aller Namen der HEADER-Variablen. Hier werden einige Beispiele aufgeführt: Status: Antwortcode Location: Pfad für die Umleitung Retry-After: Zeitdauer, über die der Dienst dem anfordernden Client voraussichtlich nicht zur Verfügung steht. Bei den Typen COOKIE_VALUE und COOKIE_EXPIRES, ist <Name> der Name eines bestimmten Cookies. COOKIE_VALUE:name: Wert des genannten Cookies COOKIE_EXPIRES:name: Ablaufzeit in Sekunden des genannten Cookies Die Namensklausel muss in einfache oder doppelte Anführungszeichen gesetzt werden. Ist keine Verwendungsklausel angegeben, entspricht der Sondervariablenname einem PLC-Variablennamen. Setzen Sie die vollständige Namensklausel in einfache Anführungszeichen und die PLC-Variable in doppelte Anführungszeichen. Der Sondervariablenname und der PLC-Variablenname müssen sich exakt entsprechen.
<Varname>	Name der PLC-Variablen oder der Datenbausteinvariablen für die auszulesende Variable Der Variablenname muss in einfache Anführungszeichen gesetzt werden. Innerhalb der einfachen Anführungszeichen setzen Sie eine PLC-Variable sowie einen Datenbausteinnamen in doppelte Anführungszeichen. Der Datenbausteinname steht in doppelten Anführungszeichen, jedoch nicht der Variablenname des Datenbausteins. Beachten Sie, dass Sie für Datenbausteinvariablen den Namen des Bausteins und nicht die Nummer des Datenbausteins verwenden.

Wenn ein Variablenname oder Datenbausteinname Sonderzeichen enthält, müssen Sie zusätzliche Anführungszeichen oder Escape-Zeichen verwenden. Dies wird unter Handhabung von Variablennamen mit Sonderzeichen (Seite 862) beschrieben.

Beispiel: Sondervariable ohne Use-Klausel lesen

```
<!-- AWP_Out_Variable Name='"HEADER:Status"' -->
```

In diesem Beispiel empfängt die HTTP-Sondervariable "HEADER:Status" den Wert der PLC-Variable "HEADER:Status". Der Name in der PLC-Variablentabelle muss dem Namen der Sondervariable exakt entsprechen, wenn keine Use-Klausel angegeben ist.

Beispiel: Sondervariable mit einer Use-Klausel lesen

```
<!-- AWP_Out_Variable Name='HEADER:Status' Use='"Status"' -->
```

In diesem Beispiel empfängt die HTTP-Sondervariable "HEADER:Status" den Wert der PLC-Variable "Status".

12.7.2.4 Sondervariablen schreiben

Der Webserver bietet die Möglichkeit, Werte aus Sondervariablen der HTTP-Antwortkopfzeile in die CPU zu schreiben. Sie können beispielsweise in STEP 7 Informationen zu dem Cookie einer benutzerdefinierten Webseite, zu dem Benutzer, der auf eine Seite zugreift, oder Header-Informationen speichern. Der Webserver bietet Zugriff auf bestimmte Sondervariablen, die Sie, wenn Sie als Benutzer mit Rechten zum Ändern von Variablen angemeldet sind, in die CPU schreiben können.

Syntax

```
<!-- AWP_In_Variable Name='<Typ>:<Name>' [Use='<Varname>']-->
```

Parameter

<Typ>	Der Typ der Sondervariable. Möglich sind: HEADER SERVER COOKIE_VALUE
<Name>	Sondervariable innerhalb der oben definierten Typen, wie in diesen Beispielen gezeigt: HEADER:Accept: akzeptable Inhaltstypene HEADER:User-Agent: Informationen zum Benutzeragenten, von dem die Anforderung stammt. SERVER:current_user_id: ID des aktuellen Benutzers; 0, wenn kein Benutzer angemeldet ist SERVER:current_user_name: Name des aktuellen Benutzers COOKIE_VALUE:<name>: Wert des genannten Cookies Setzen Sie die Namensklausel in einfache Anführungszeichen. Ist keine Verwendungsklausel angegeben, entspricht der Sondervariablenname einem PLC-Variablennamen. Setzen Sie die vollständige Namensklausel in einfache Anführungszeichen und die PLC-Variable in doppelte Anführungszeichen. Der Sondervariablenname muss dem PLC-Variablennamen exakt entsprechen. In der HTTP-Dokumentation finden Sie eine Liste aller Namen der HEADER-Variablen.
<Varname>	Der Variablenname in Ihrem STEP 7-Programm, in das Sie die Sondervariable schreiben möchten. Es kann sich um eine PLC-Variable oder eine Datenbausteinvariable handeln. Der Variablenname muss in einfache Anführungszeichen gesetzt werden. Innerhalb der einfachen Anführungszeichen setzen Sie eine PLC-Variable sowie einen Datenbausteinnamen in doppelte Anführungszeichen. Der Datenbausteinname steht in doppelten Anführungszeichen, jedoch nicht der Variablenname des Datenbausteins. Beachten Sie, dass Sie für Datenbausteinvariablen den Namen des Bausteins und nicht die Nummer des Datenbausteins verwenden.

Beispiele

```
<!-- AWP_In_Variable Name='"SERVER:current_user_id"' -->
```

In diesem Beispiel schreibt die Webseite den Wert der HTTP-Sondervariablen "SERVER:current_user_id" in die PLC-Variable "SERVER:current_user_id".

```
<!-- AWP_In_Variable Name=SERVER:current_user_id'
Use='"Meine_Benutzer-ID"' -->
```

In diesem Beispiel schreibt die Webseite den Wert der HTTP-Sondervariablen "SERVER:current_user_id" in die PLC-Variable "Meine_Benutzer-ID".

Hinweis

Nur ein Benutzer mit Rechten zum Ändern von Variablen kann Daten in die CPU schreiben. Der Webserver ignoriert die Befehle, wenn der Benutzer keine Änderungsrechte hat.

Wenn ein Variablenname oder Datenbausteinname Sonderzeichen enthält, müssen Sie zusätzliche Anführungszeichen oder Escape-Zeichen verwenden. Dies wird unter "Handhabung von Variablennamen mit Sonderzeichen (Seite 862)" beschrieben.

12.7.2.5 Alias für einen Variablenverweis nutzen

Für eine In_Variable oder eine Out_Variable können Sie in Ihren benutzerdefinierten Webseiten einen Alias verwenden. Sie können beispielsweise einen anderen symbolischen Namen in Ihrer HTML-Seite verwenden als den in der CPU, oder Sie können eine Variable in der CPU einer Sondervariablen gleichsetzen. Die AWP-Verwendungsklausel bietet diese Möglichkeit.

Syntax

```
<-- AWP_In_Variable Name='<Varname1>' Use='<Varname2>' -->
<-- AWP_Out_Variable Name='<Varname1>' Use='<Varname2>' -->
```

Parameter

<Varname1>	Der Aliasname oder der Sondervariablenname Varname1 muss in einfachen oder doppelten Anführungszeichen angegeben werden.
<Varname2>	Name der PLC-Variable, der Sie einen Aliasnamen zuweisen möchten. Bei der Variable kann es sich um eine PLC-Variable, eine Datenbausteinvariable oder eine Sondervariable handeln. Varname2 muss in einfache Anführungszeichen gesetzt werden. Innerhalb der einfachen Anführungszeichen setzen Sie eine PLC-Variable, Sondervariable oder einen Datenbausteinnamen in doppelte Anführungszeichen. Der Datenbausteinname steht in doppelten Anführungszeichen, jedoch nicht der Variablenname des Datenbausteins. Beachten Sie, dass Sie für Datenbausteinvariablen den Namen des Bausteins und nicht die Nummer des Datenbausteins verwenden.

Beispiele

```
<-- AWP_In_Variable Name='SERVER:current_user_id'
Use='"Data_Block_10".server_user' -->
```

In diesem Beispiel wird die Sondervariable SERVER:current_user_id in die Variable "server_user" im Datenbaustein "Data_Block_10" geschrieben.

```
<-- AWP_Out_Variable Name='Weight'
Use='"Data_Block_10".Tank_data.Weight' -->
```

In diesem Beispiel kann der Wert in Datenbaustein-Strukturelement Data_Block_10.Tank_data.Weight innerhalb der übrigen benutzerdefinierten Webseite einfach als "Weight" referenziert werden.

```
<-- AWP_Out_Variable Name='Gewicht' Use='"Gewicht_Rohmilchbehälter"'
-->
```

In diesem Beispiel kann der Wert der PLC-Variable "Gewicht_Rohmilchbehälter" in der übrigen benutzerdefinierten Webseite einfach als "Gewicht" angegeben werden.

Wenn ein Variablenname oder Datenbausteinname Sonderzeichen enthält, müssen Sie zusätzlich Anführungszeichen oder Escape-Zeichen verwenden. Dies wird unter Handhabung von Variablennamen mit Sonderzeichen (Seite 862) beschrieben.

12.7.2.6 Enum-Typen definieren

Sie können in Ihren benutzerdefinieren Seiten Enum-Typen definieren und die Elemente in einem AWP-Befehl zuweisen.

Syntax

```
<!-- AWP_Enum_Def Name='<Name Enum-Typ>' Values='<Wert>, <Wert>,...
' -->
```

Parameter

<Name Enum-Typ>	Name des Aufzählungstyps, in einfache oder doppelte Anführungszeichen gesetzt.
<Wert>	<Konstante>:<Name> Die Konstante kennzeichnet den numerischen Wert für die Zuweisung des Enum-Typs. Die Gesamtzahl ist unbegrenzt. Der Name ist der dem Enum-Element zugewiesene Wert.

Beachten Sie, dass die gesamte Zeichenkette der Enum-Wertzuweisungen in einzelne Hochkommata eingeschlossen ist, und jede einzelne Elementzuweisung zu einem Enum-Typ steht in doppelten Anführungszeichen. Die Definition eines Enum-Typs hat für die benutzerdefinierten Webseiten globale Gültigkeit. Wenn Sie Ihre benutzerdefinierten Webseiten in sprachspezifischen Ordnern (Seite 883) abgelegt haben, gilt die Enum-Typ-Definition global für alle Seiten in dem Sprachordner.

Beispiel

```
<!-- AWP_Enum_Def Name='AlarmEnum' Values='0:"Keine Alarme",
1:"Behälter ist voll", 2:"Tank is empty"' -->
```

12.7.2.7 CPU-Variablen mit einem Enum-Typ referenzieren

Sie können eine Variable in der CPU einem Enum-Typ zuweisen. Diese Variable kann an anderer Stelle in Ihren benutzerdefinierten Webseiten in einer Leseoperation (Seite 851) oder einer Schreiboperation (Seite 852) verwendet werden. Bei einer Leseoperation ersetzt der Webserver den aus der CPU gelesenen numerischen Wert durch den entsprechenden Enum-Textwert. Bei einer Schreiboperation ersetzt der Webserver den Textwert mit dem ganzzahligen Wert der Aufzählung, der dem Text entspricht, bevor der Wert in die CPU geschrieben wird.

Syntax

```
<!-- AWP_In_Variable Name='<Varname>' Enum="<Enum-Typ>" -->
<!-- AWP_Out_Variable Name='<Varname>' Enum="<Enum-Typ>" -->
```

Parameter

<Varname>	Name der PLC-Variable oder Datenbausteinvariable, die dem Enum-Typ zugewiesen werden soll, oder Name des Aliasnamens einer PLC-Variable (Seite 857), sofern deklariert. Varname muss in einfache Anführungszeichen gesetzt werden. Innerhalb der einfachen Anführungszeichen setzen Sie eine PLC-Variable sowie einen Datenbausteinnamen in doppelte Anführungszeichen. Beachten Sie, dass Sie für Datenbausteinvariablen den Namen des Bausteins und nicht die Nummer des Datenbausteins verwenden. Der Datenbausteinname steht in doppelten Anführungszeichen, jedoch nicht der Variablenname des Datenbausteins.
<Enum-Typ>	Name des Aufzählungstyps, in einfache oder doppelte Anführungszeichen gesetzt

Die Referenz eines Enum-Typs gilt für das aktuelle Fragment.

Beispiel für die Verwendung beim Lesen einer Variablen

```
<!-- AWP_Out_Variable Name='"Alarm"' Enum="AlarmEnum" -->...
<p>Der aktuelle Wert von "Alarm" ist :="Alarm":</p>
```

Wenn der Wert von "Alarm" in der CPU gleich 2 ist, zeigt die HTML-Seite 'Der aktuelle Wert von "Alarm" ist Behälter ist leer' an, weil dieDefinition des Enum-Typs (Seite 858) dem numerischen Wert 2 die Zeichenfolge "Behälter ist leer" zuweist.

Beispiel für die Verwendung beim Schreiben einer Variablen

```
<!-- AWP_Enum_Def Name='AlarmEnum' Values='0:"Keine Alarme",
1:"Behälter ist voll", 2:"Behälter ist leer"' -->
<!-- AWP_In_Variable Name='"Alarm"' Enum='AlarmEnum' -->...
<form method="POST">
<p><input type="hidden" name='"Alarm"' value="Behälter ist voll"
/></p>
<p><input type="submit" value='Behälter ist voll setzen' /><p>
</form>
```

Weil die Definition des Enum-Typs (Seite 858) dem numerischen Wert 1 den Text "Behälter ist voll" zuweist, wird der Wert 1 in die PLC-Variable "Alarm" in der CPU geschrieben.

Beachten Sie, dass die Enum-Klausel in der Deklaration AWP_In_Variable exakt der Namensklausel in der Deklaration AWP_Enum_Def entsprechen muss.

Beispiel für das Schreiben von Variablen unter Verwendung von Aliasnamen

```
<!-- AWP_Enum_Def Name='AlarmEnum' Values='0:"Keine Alarme",
1:"Behälter ist voll", 2:"Behälter ist leer"' -->
<!-- AWP_In_Variable Name='"Alarm"' Enum='AlarmEnum'
Use='"Data_block_4".Motor1.Alarm'-->...
<form method="POST">
<p><input type="hidden" name='"Alarm"' value="Behälter ist voll"
/></p>
<p><input type="submit" value='Behälter ist voll setzen' /><p>
</form>
```

Weil die Definition des Enum-Typs (Seite 858) dem numerischen Wert 1 den Text "Behälter ist voll" zuweist, wird der Wert 1 in den Aliasnamen "Alarm" geschrieben, welcher der PLC-Variablen "Motor1.Alarm" im Datenbaustein "Data_Block_4" der CPU entspricht.

Wenn ein Variablenname oder Datenbausteinname Sonderzeichen enthält, müssen Sie zusätzliche Anführungszeichen oder Escape-Zeichen verwenden. Dies wird unter Handhabung von Variablennamen mit Sonderzeichen (Seite 862) beschrieben.

Hinweis

In Vorgängerversionen war eine getrennte AWP_Enum_Ref-Deklaration erforderlich, um einer Variable einen definierten Enum-Typ zuzuordnen. STEP 7 und die S7-1200 unterstützen bestehenden Code mit AWP_Enum_Ref-Deklarationen. Dieser Befehl ist jedoch nicht mehr erforderlich.

12.7.2.8 Fragmente erstellen

STEP 7 konvertiert und speichert benutzerdefinierte Webseiten als Steuerungs-DB und DB-Fragmente, wenn Sie in den CPU-Eigenschaften für den Webserver auf "Bausteine generieren" klicken. Sie können spezifische Fragmente für spezifische Seiten oder für Abschnitte spezifischer Seiten einrichten. Sie können diese Fragmente mit dem AWP-Befehl "Start_Fragment" durch einen Namen und eine Nummer kennzeichnen. Alles auf einer Seite, was auf den Befehl AWP_Start_Fragment folgt, gehört zu dem Fragment, bis ein nachfolgender Befehl AWP_Start_Command abgesetzt wird oder bis das Ende der Datei erreicht ist.

Syntax

```
<!-- AWP_Start_Fragment Name='<Name>'
[Type=<Typ>][ID=<ID>][Mode=<Modus>] -->
```

Parameter

<Name>	Text-Zeichenkette: Name des DB-Fragments Fragmentnamen müssen mit einem Buchstaben oder einem Unterstrich beginnen und sich aus Buchstaben, Ziffern und Unterstrichen zusammensetzen. Der Fragmentname ist ein regulärer Ausdruck der Form: [a-zA-Z_][a-zA-Z_0-9]*
<Typ>	"Manuell" oder "Automatisch" Manuell: Das STEP 7-Programm muss dieses Fragment anfordern und kann entsprechend reagieren. Die Funktionsweise des Fragments muss mit STEP 7 und den Variablen des Steuerungs-DBs gesteuert werden. Automatisch: Der Webserver verarbeitet das Fragment automatisch. Wenn Sie den Parameter für den Typ nicht angeben, ist die Voreinstellung "Automatisch".

<ID>	Ganzzahlige Identifikationsnummer. Wenn Sie den Parameter ID nicht angeben, weist der Webserver standardmäßig eine Nummer zu. Geben Sie bei manuellen Fragmenten für die ID eine niedrige Nummer an. Die ID ist der Weg, über den das STEP 7-Programm ein manuelles Fragment steuert.
<Modus>	"sichtbar" oder "ausgeblendet" sichtbar: Inhalte des Fragments werden auf der benutzerdefinierten Webseite angezeigt. ausgeblendet: Inhalte des Fragments werden nicht auf der benutzerdefinierten Webseite angezeigt. Wenn Sie den Parameter für den Typ nicht angeben, ist die Voreinstellung "sichtbar".

Manuelle Fragmente

Wenn Sie ein manuelles Fragment für eine benutzerdefinierte Webseite oder einen Teil einer Seite anlegen, muss Ihr STEP 7-Programm steuern, wann das Fragment gesendet wird. Das STEP 7-Programm muss im Steuer-DB für eine benutzerdefinierte Seite unter manueller Steuerung entsprechende Parameter festlegen und dann die WWW-Anweisung mit dem geänderten Steuer-DB aufrufen. Nähere Erläuterungen zur Struktur des Steuer-DBs und dazu, wie Sie einzelne Seiten und Fragmente ändern, finden Sie unter "Erweiterte Steuerung von benutzerdefinierten Webseiten (Seite 887)".

12.7.2.9 Fragmente importieren

Sie können von einem Teil Ihres HTML-Codes ein benanntes Fragment erstellen und dann dieses Fragment an anderer Stelle in Ihren Satz benutzerdefinierter Webseiten importieren. Stellen Sie sich z. B. einen Satz benutzerdefinierter Webseiten vor, die aus einer Startseite und verschiedenen anderen HTML-Seiten bestehen, die über Links auf der Startseite aufgerufen werden. Angenommen, auf jeder dieser einzelnen Seiten soll das Firmenlogo angezeigt werden. Dafür könnten Sie ein Fragment erstellen (Seite 860), das das Bild des Firmenlogos lädt. Jede einzelne HTML-Seite kann dann dieses Fragment importieren, um das Firmenlogo anzuzeigen. Für diesen Zweck verwenden Sie den AWP-Befehl Import_Fragment. Der HTML-Code für das Fragment ist nur in einem Fragment vorhanden, doch Sie können dieses DB-Fragment so oft wie nötig in so viele Webseiten, wie Sie wünschen, importieren.

Syntax

```
<!-- AWP_Import_Fragment Name='<Name>' -->
```

Parameter

<Name>	Text-Zeichenkette: Name des zu importierenden DB-Fragments

Beispiel

Auszug aus dem HTML-Code, der ein Fragment zum Anzeigen eines Bilds erstellt:

```
<!-- AWP_Start_Fragment Name='Mein_Firmenlogo' --><p><img
src="Firmenlogo.jpg"></p>
```

Auszug aus dem HTML-Code in einer anderen *.html-Datei, der das Fragment mit dem Logobild importiert:

```
<!-- AWP_Import_Fragment Name='Mein_Firmenlogo' -->
```

Beide *.html-Dateien (die Datei, die das Fragment erstellt und die Datei, die das Fragment importiert) befinden sich in der Ordnerstruktur, die Sie festlegen, wenn Sie die benutzerdefinierten Seiten in STEP 7 konfigurieren (Seite 865).

12.7.2.10 Definitionen verbinden

Wenn Sie die Variablen für die Verwendung in Ihren benutzerdefinierten Webseiten deklarieren, können Sie eine Variablendeklaration mit einem Alias für die Variable (Seite 857) verbinden. Sie können außerdem mehrere In_Variables in einer Anweisung und mehrere Out_Variables in einer Anweisung verbinden.

Beispiele

```
<!-- AWP_In_Variable Name='"Füllstand", Name='"Gewicht"',
Name='"Temp"' -->
<--! AWP_Out_Variable Name='HEADER:Status', Use='"Status"',
         Name='HEADER:Location', Use="Speicherort",
         Name='COOKIE_VALUE:name', Use="Mein_Cookie" -->
<!-- AWP_In_Variable Name='Alarm' Use='"Datenbaustein_10".Alarm' -->
```

12.7.2.11 Handhabung von Variablennamen mit Sonderzeichen

Bei der Angabe von Variablennamen in benutzerdefinierten Webseiten müssen Sie mit besonderer Vorsicht vorgehen, wenn die Variablennamen Zeichen mit besonderer Bedeutung enthalten.

Variablen lesen

Um eine Variable zu lesen (Seite 851), verwenden Sie die folgende Syntax:
:=<Varname>:

Die folgenden Regeln gelten beim Lesen von Variablen:

- Bei Variablennamen aus der PLC-Variablentabelle setzen Sie den Variablennamen in doppelte Anführungszeichen.
- Bei Variablennamen, bei denen es sich um Datenbausteinvariablen handelt, ist der Datenbausteinname in doppelte Anführungszeichen zu setzen. Die Variable befindet sich außerhalb der Anführungszeichen.
- Bei Variablennamen, bei denen es sich um direkte E/A-Adressen, Speicheradressen oder Aliasnamen handelt, setzen Sie die gelesene Variable nicht in Anführungszeichen.
- Bei Variablennamen oder Datenbausteinvariablennamen, die einen nach links geneigten Schrägstrich ("Backslash") enthalten, stellen Sie dem Backslash einen weiteren Backslash voran.
- Wenn ein Variablenname oder Datenbausteinvariablenname einen Doppelpunkt, ein Kleiner-als-Zeichen, Größer-als-Zeichen oder ein Kaufmanns-Und enthält, definieren Sie für die gelesene Variable einen Alias ohne Sonderzeichen und lesen die Variable über diesen Aliasnamen. Stellen Sie in Variablennamen enthaltenen Doppelpunkten in Verwendungsklauseln einen Backslash voran.

Tabelle 12- 1 Beispiele für zu lesende Variablen

Datenbausteinname	Variablenname	Lesebefehl
-/-	ABC:DEF	`<!--AWP_Out_Variable Name='Sondervariable' Use ='"ABC:DEF"' --> :=Sondervariable:`
nicht zutreffend	T\	`:="T\\":`
nicht zutreffend	A \B 'C :D	`<!--AWP_Out_Variable Name='Weitere_Sondervariable' Use='"A \\B \'C :D"' --> :=Weitere_Sondervariable:`
nicht zutreffend	a<b	`<!--AWP_Out_Variable Name='a_kleiner_b' Use='"a<b"' --> :=a_kleiner_b:`
Datenbaustein_1	Variable_1	`:="Datenbaustein_1".Tag_1:`
Datenbaustein_1	ABC:DEF	`<!-- AWP_Out_Variable Name='Sondervariable' Use='"Datenbaustein_1".ABC\:DEF'--> :=Sondervariable:`
DB A' B C D$ E	Variable	`:="DB A' B C D$ E".Tag:`
DB:DB	Variable:Variable	`<!--AWP_Out_Variable Name='meine_Variable' Use ='"DB:DB".Tag\:Tag' --> :=meine_Variable:`

Namens- und Verwendungsklauseln

Die AWP-Befehle AWP_In_Variable, AWP_Out_Variable, AWP_Enum_Def, AWP_Enum_Ref, AWP_Start_Fragment und AWP_Import_Fragment haben Namensklauseln. HTML-Formularbefehle wie <input> und <select> haben ebenfalls Namensklauseln. AWP_In_Variable und AWP_Out_Variable können zusätzlich Verwendungsklauseln haben. Unabhängig vom Befehl ist die Syntax von Namens- und Verwendungsklauseln hinsichtlich der Handhabung von Sonderzeichen die gleiche:

- Der Text, den Sie für eine Namens- oder Verwendungsklausel angeben, ist in einfache Anführungszeichen zu setzen. Handelt es sich bei dem Namen in Anführungszeichen um eine PLC-Variable oder einen Datenbausteinnamen, setzen Sie die gesamte Klausel in einfache Anführungszeichen.
- Innerhalb einer Namens- oder Verwendungsklausel sind Datenbausteinnamen und PLC-Variablennamen in doppelte Anführungszeichen zu setzen.
- Wenn ein Variablenname oder Datenbausteinname ein einfaches Anführungszeichen oder einen rückwärts gerichteten Schrägstrich ("Backslash") enthält, stellen Sie dem Zeichen einen Backslash als Escape-Zeichen voran. Der rückwärts gerichtete Schrägstrich "\" dient bei der Übersetzung der AWP-Befehle als Escape-Zeichen.

Tabelle 12- 2 Beispiele für Namensklauseln

Datenbausteinname	Variablenname	Möglichkeiten für Namensklauseln
-/-	ABC'DEF	`Name='"ABC\'DEF"'`
nicht zutreffend	A \B 'C :D	`Name='"A \\B \'C :D"'`
Datenbaustein_1	Variable_1	`Name='"Datenbaustein_1".Tag_1'`
Datenbaustein_1	ABC'DEF	`Name='"Datenbaustein_1".ABC\'DEF'`
Datenbaustein_1	A \B 'C :D	`Name='"Datenbaustein_1".A \\B \'C :D'`
DB A' B C D$ E	Variable	`Name='"DB A\' B C D$ E".Variable'`

Für Verwendungsklauseln gelten die gleichen Konventionen wie für Namensklauseln.

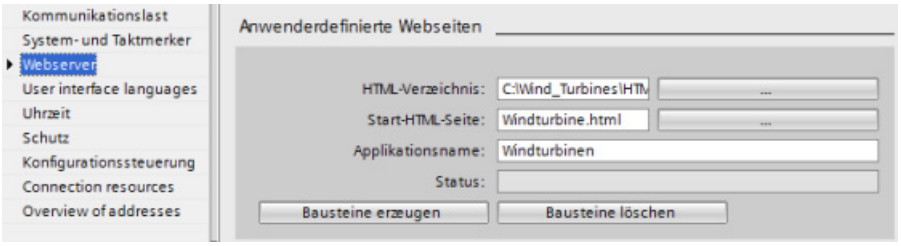
Hinweis

Unabhängig von den Zeichen, die Sie in Ihrer HTML-Seite verwenden, legen Sie als Zeichensatz der HTML-Seite UTF-8 fest und speichern die Seite im Editor mit der Zeichenverschlüsselung UTF-8.

12.7.3 Verwendung von benutzerdefinierten Webseiten konfigurieren

Um benutzerdefinierte Webseiten in STEP 7 zu konfigurieren, gehen Sie folgendermaßen vor:

1. Wählen Sie die CPU in der Gerätekonfiguration aus.
2. Rufen Sie im Inspektorfenster der CPU die Eigenschaften des Webservers auf.



3. Sofern nicht bereits geschehen, aktivieren Sie das Kontrollkästchen für "Webserver auf diesem Modul aktivieren".
4. Wählen Sie "Erlaubt nur HTTPS-Zugriff" aus, um sicherzustellen, dass der Webserver die Kommunikation verschlüsselt, und um die Sicherheit Ihrer über das Web zugänglichen CPU zu erhöhen.
5. Geben Sie den Namen des Ordners auf Ihrem PC ein, in dem Sie die HTML-Standardseite (Startseite) gespeichert haben.
6. Geben Sie den Namen der Standardseite ein.
7. Geben Sie einen Namen für Ihre Anwendung ein (optional). Der Webserver verwendet den Anwendungsnamen zur weiteren Unterteilung bzw. Gruppierung der Webseiten. Wenn Sie einen Anwendungsnamen eingeben, erstellt der Webserver eine URL für Ihre benutzerdefinierte Seite in folgendem Format: http[s]://ww.xx.yy.zz/awp/<Anwendungsname>/<Seitenname>.html.

   Vermeiden Sie Sonderzeichen im Anwendungsnamen. Einige Zeichen können verursachen, dass der Webserver die benutzerdefinierten Seiten nicht anzeigen kann.
8. Geben Sie Dateinamenerweiterungen von Dateien ein, die AWP-Befehle enthalten. Standardmäßig analysiert STEP 7 Dateien mit den Erweiterungen *.htm, *.html und *.js. Wenn Sie andere Dateierweiterungen nutzen, fügen Sie sie ein.
9. Übernehmen Sie die Standardnummer für den Web-DB oder geben Sie eine Nummer Ihrer Wahl ein. Dies ist die DB-Nummer des Steuer-DBs, der die Anzeige der Webseiten steuert.
10. Übernehmen Sie die standardmäßige Anfangsnummer für das DB-Fragment oder geben Sie eine Nummer Ihrer Wahl ein. Dies ist das erste der DB-Fragmente mit den Webseiten.

Programmbausteine generieren

Wenn Sie auf die Schaltfläche "Bausteine generieren" klicken, generiert STEP 7 aus den HTML-Seiten im von Ihnen angegebenen HTML-Quellverzeichnis Datenbausteine und einen Steuerdatenbaustein für den Betrieb Ihrer Webseiten. Sie können diese Attribute nach Bedarf für Ihre Anwendung festlegen (Seite 866). STEP 7 generiert außerdem einen Satz Datenbausteinfragmente, um die Darstellung aller Ihrer HTML-Seiten zu speichern. Wenn Sie die Datenbausteine generieren, aktualisiert STEP 7 die Eigenschaften, um die Nummer des Steuerdatenbausteins und die Nummer des ersten Datenbausteinfragments anzuzeigen. Nachdem Sie die Datenbausteine generiert haben, sind Ihre benutzerdefinierten Webseiten Teil Ihres STEP 7-Programms. Die diesen Seiten entsprechenden Bausteine erscheinen im Webserverordner, der sich in der Projektnavigation unter den Programmbausteinen im Ordner "Systembausteine" befindet.

Programmbausteine löschen

Um Datenbausteine zu löschen, die Sie zuvor generiert haben, klicken Sie auf die Schaltfläche "Datenbausteine löschen". STEP 7 löscht den Steuerdatenbaustein und alle Datenbausteinfragmente aus Ihrem Projekt, in dem sich die benutzerdefinierten Webseiten befinden.

12.7.4 WWW-Anweisung für benutzerdefinierte Webseiten programmieren

Die WWW-Anweisung muss in Ihrem STEP 7-Anwenderprogramm enthalten sein und ausgeführt werden, damit die benutzerdefinierten Webseiten über die Standard-Webseiten aufrufbar sind. Der Steuerdatenbaustein ist der Eingangsparameter für die WWW-Anweisung und gibt den Inhalt der Seiten wie in den Datenbausteinfragmenten dargestellt sowie Zustands- und Steuerinformationen an. STEP 7 erstellt den Steuerdatenbaustein, wenn Sie in der Konfiguration der benutzerdefinierten Webseiten (Seite 865) auf die Schaltfläche "Bausteine erstellen" klicken.

WWW-Anweisung programmieren

Das STEP 7-Programm muss die Anweisung WWW ausführen, damit die benutzerdefinierten Webseiten über die Standard-Webseiten aufrufbar sind. Sie können festlegen, dass die benutzerdefinierten Webseiten nur unter bestimmten Bedingungen entsprechend den Anwendungsvoraussetzungen und Einstellungen verfügbar sind. Dann kann Ihre Programmlogik steuern, wann die Anweisung WWW aufzurufen ist.

Tabelle 12- 3 Anweisung WWW

KOP/FUP	SCL	Beschreibung
	`ret_val := WWW(` `  ctrl_db:=_uint_in_);`	Zugriff auf die benutzerdefinierten Webseiten über die Standard-Webseiten

Sie müssen den Eingangsparameter des Steuerdatenbausteins (CTRL_DB), der der ganzzahligen DB-Nummer des Steuer-DBs entspricht, eingeben. Sie finden diese Bausteinnummer des Steuer-DBs (als Web-DB-Nummer bezeichnet) in den Webserver-Eigenschaften der CPU, nachdem Sie die Bausteine für die benutzerdefinierten Webseiten erstellt haben. Geben Sie die ganzzahlige DB-Nummer als Parameter CTRL_DB der Anweisung WWW an. Der Rückgabewert (RET_VAL) enthält das Funktionsergebnis. Beachten Sie, dass die Anweisung WWW asynchron ausgeführt wird und dass der Ausgang RET_VAL einen Anfangswert von 0 hat, auch wenn später ein Fehler auftreten kann. Das Programm kann den Zustand des Steuer-DBs abfragen, um sicherzustellen, dass die Anwendung erfolgreich gestartet wurde, oder es kann mit einem nachfolgenden Aufruf von WWW den Parameter RET_VAL abfragen.

Tabelle 12- 4 Rückgabewert

RET_VAL	Beschreibung
0	Kein Fehler
16#00yx	x: Die vom entsprechenden Bit dargestellte Anforderung ist im Wartezustand: x=1: Anforderung 0 x=2: Anforderung 1 x=4: Anforderung 2 x=8: Anforderung 3 Die x-Werte können logisch durch ODER verknüpft werden, um die Wartezustände mehrerer Anforderungen darzustellen. Wenn z. B. x = 6 ist, sind die Anforderungen 1 und 2 im Wartezustand. y: 0: kein Fehler; 1: Fehler vorhanden und "last_error" wurde im Steuer-DB gesetzt (Seite 887)
16#803a	Der Steuer-DB ist nicht geladen.
16#8081	Datentyp, Format oder Version des Steuer-DBs ist falsch.
16#80C1	Für die Initialisierung der Webanwendung sind keine Ressourcen verfügbar.

Verwendung des Steuer-DBs

STEP 7 erstellt den Steuerdatenbaustein, wenn Sie auf die Schaltfläche "Bausteine generieren" klicken. Die Nummer des Steuer-DBs wird in den Eigenschaften der benutzerdefinierten Webseiten angezeigt. Sie finden den Steuer-DB auch im Ordner "Programmbausteine" in der Projektnavigation.

Typischerweise verwendet Ihr STEP 7-Programm den Steuer-DB direkt wie über den Vorgang "Bausteine generieren" angelegt, ohne weitere Änderungen. Das STEP 7-Anwenderprogramm kann jedoch globale Befehle im Steuer-DB festlegen, um den Webserver zu deaktivieren oder ihn nachfolgend wieder zu aktivieren. Außerdem muss das STEP 7-Anwenderprogramm das Verhalten von benutzerdefinierten Webseiten, die Sie als manuelle DB-Fragmente anlegen (Seite 865), über eine Anforderungstabelle im Steuer-DB steuern. Informationen zu diesen erweiterten Aufgaben finden Sie unter Erweiterte Steuerung von benutzerdefinierten Webseiten (Seite 887).

12.7.5 Programmbausteine in die CPU laden

Nachdem Sie die Bausteine für die benutzerdefinierten Webseiten generiert haben, sind diese Teil Ihres STEP 7-Programms, genau wie alle anderen Programmbausteine. Um die Programmbausteine in die CPU zu laden, gehen Sie wie üblich vor. Beachten Sie, dass Sie Programmbausteine für benutzerdefinierte Webseiten nur in die CPU laden können, wenn sich diese im Betriebszustand STOP befindet.

12.7.6 Zugriff auf die benutzerdefinierten Webseiten

Sie rufen die benutzerdefinierten Webseiten über die Standard-Webseiten (Seite 823) auf. Die Standard-Webseiten zeigen einen Link für die Benutzerseiten auf der linken Seite an, wo sich auch die Links auf die anderen Seiten befinden. Die Navigationsseite auf Mobilgeräten enthält außerdem einen Link zu den "Benutzerseiten". Wenn Sie auf den Link "Benutzerseiten" klicken, ruft Ihr Webbrowser die Seite auf, die den Link auf Ihre Standardseite enthält. Innerhalb der benutzerdefinierten Seiten entspricht die Navigation dem Aufbau Ihrer spezifischen Seiten.

12.7.7 Einschränkungen bei benutzerdefinierten Webseiten

Die Einschränkungen bei Standard-Webseiten (Seite 892) gelten auch bei benutzerdefinierten Webseiten. Zudem gibt es bei benutzerdefinierten Webseiten einige spezifische Aspekte.

Kapazität des Ladespeichers

Ihre benutzerdefinierten Webseiten werden, wenn Sie auf "Bausteine generieren" klicken, zu Datenbausteinen, die Platz im Ladespeicher benötigen. Wenn Sie eine Memory Card gesteckt haben, steht die Kapazität Ihrer Memory Card als externer Ladespeicher für die benutzerdefinierten Webseiten zur Verfügung.

Wenn Sie keine Memory Card gesteckt haben, belegen diese Bausteine Platz im internen Ladespeicher, der je nach CPU-Modell begrenzt ist.

Mit den Online- und Diagnosefunktionen in STEP 7 können Sie den belegten und den freien Platz im Ladespeicher abfragen. Sie können zudem in den Eigenschaften der einzelnen Bausteine, die STEP 7 für Ihre benutzerdefinierten Webseiten generiert, den benötigten Platz im Ladespeicher prüfen.

Hinweis

Wenn Sie den Platz für Ihre benutzerdefinierten Webseiten verringern müssen, entfernen Sie ggf. einige der eingefügten Bilder.

Anführungszeichen in Textzeichenketten

In Datenbausteinvariablen, die in benutzerdefinierten Webseiten benutzt werden, sind keine Textzeichenketten mit eingebetteten einfachen oder doppelten Anführungszeichen zu verwenden. Da die HTML-Syntax oft einfache oder doppelte Anführungszeichen als Trennzeichen verwendet, können Anführungszeichen in Textzeichenfolgen die Anzeige benutzerdefinierter Webseiten stören.

Für Datenbausteinvariablen des Typs String, die in benutzerdefinierten Webseiten verwendet werden, sind die folgenden Regeln zu beachten:

- Im String-Wert der Datenbausteinvariablen in STEP 7 keine einfachen oder doppelten Anführungszeichen verwenden.
- Darauf achten, dass das Anwenderprogramm diesen Datenbausteinvariablen keine Zeichenketten mit Anführungszeichnen zuweist.

12.7.8 Beispiel für eine benutzerdefinierte Webseite

12.7.8.1 Webseite zum Beobachten und Steuern einer Windturbine

Stellen Sie sich als Beispiel für eine benutzerdefinierte Webseite eine Webseite vor, die dazu dient, eine Windturbine entfernt zu beobachten und zu steuern:

Beschreibung

In dieser Anwendung ist jede Windturbine des Windparks mit einer S7-1200 zur Steuerung der Turbine ausgestattet. Im STEP 7-Programm hat jede Windturbine einen Datenbaustein mit Daten, die für die jeweilige Windturbine spezifisch sind.

Die benutzerdefinierte Webseite bietet dezentralen Zugriff auf die Turbine über einen PC. Ein Benutzer kann die Standard-Webseiten der CPU einer bestimmten Windturbine aufrufen und auf die benutzerdefinierte "Remote Wind Turbine Monitor"-Webseite (Webseite für die dezentrale Beobachtung der Windturbine) zugreifen, um die Daten der Turbine einzusehen. Ein Benutzer mit Rechten zum Ändern von Variablen kann zudem die Turbine in den manuellen Modus versetzen und die Variablen für Drehzahl, Ausrichtung und Anstellwinkel der Turbine über die Webseite steuern. Ein Benutzer mit Rechten zum Ändern von Variablen kann auch einen Bremswert festlegen, unabhängig davon, ob die Turbine manuell oder automatisch gesteuert wird.

Das STEP 7-Programm prüft die Booleschen Werte für Übersteuerung der automatischen Steuerung und verwendet, sofern diese eingestellt ist, die vom Anwender eingegebenen Werte für Drehzahl, Ausrichtung und Anstellwinkel der Turbine. Andernfalls ignoriert das Programm diese Werte.

Verwendete Dateien

Dieses Beispiel für eine benutzerdefinierte Webseite besteht aus drei Dateien:

- **Windturbine.html**: Dies ist die HTML-Seite mit der oben dargestellten Anzeige. Über AWP-Befehle wird auf Steuerungsdaten zugegriffen.
- **Windturbine.css**: Dies ist das Cascading Style Sheet, das die Formatierungsvorgaben für die HTML-Seite enthält. Die Verwendung eines Cascading Style Sheet ist optional, doch es kann die Entwicklung von HTML-Seiten vereinfachen.
- **Wind_turbine.jpg**: Dies ist das Hintergrundbild, das auf der HMTL-Seite angezeigt wird. Die Verwendung von Bildern in benutzerdefinierten Webseiten ist natürlich freigestellt, und Bilder benötigen zusätzlichen Speicherplatz in der CPU.

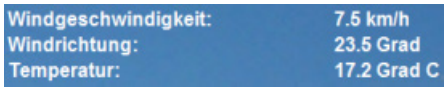
Diese Dateien sind nicht in Ihrer Installation enthalten, werden jedoch als Beispiel beschrieben.

Implementierung

Die HTML-Seite verwendet AWP-Befehle zum Auslesen von Werten aus dem PLC-Gerät (Seite 851) für die Anzeigefelder und sie verwendet AWP-Befehle zum Schreiben von Werten in das PLC-Gerät (Seite 852) für die Daten der Benutzereingabe. Diese Seite nutzt zudem AWP-Befehle für die Definition von Enum-Typen (Seite 858) und für die Referenz (Seite 858) zur Handhabung von EIN/AUS-Einstellungen.

Der erste Teil der Seite zeigt eine Kopfzeile mit der Nummer der Windturbine an.

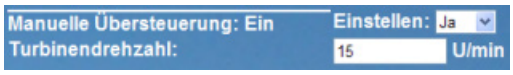
Der nächste Teil der Seite zeigt die atmosphärischen Bedingungen an der Windturbine an. Die E/A am Turbinenstandort liefern die Windgeschwindigkeit, Windrichtung und die aktuelle Temperatur.



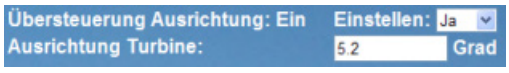
Dann zeigt die Seite die aus der S7-1200 ausgelesene Leistungsabgabe der Turbine an.

Die folgenden Abschnitte ermöglichen die manuelle Steuerung der Turbine, eine Übersteuerung der normalen Automatiksteuerung der S7-1200. Die folgenden Übersteuerungen sind möglich:

- Manuelle Übersteuerung: Aktiviert die manuelle Übersteuerung der Turbine. Das STEP 7-Anwenderprogramm verlangt, dass die Einstellung für manuelle Übersteuerung wahr ist, damit die manuellen Einstellungen für Drehzahl, Ausrichtung oder Anstellwinkel der Turbine verwendet werden können.



- Übersteuerung der Ausrichtung: Aktiviert die manuelle Übersteuerung der Turbinenausrichtung und die manuelle Einstellung der Ausrichtung. Das STEP 7-Anwenderprogramm verlangt, dass die Einstellung für manuelle Übersteuerung und für Übersteuerung der Ausrichtung wahr ist, damit die Ausrichtungseinstellung angewendet werden kann.



- Übersteuerung des Anstellwinkels: Aktiviert die manuelle Übersteuerung des Anstellwinkels der Rotorblätter. Das STEP 7-Anwenderprogramm verlangt, dass die Einstellung für manuelle Übersteuerung und für Übersteuerung des Anstellwinkels wahr ist, damit die Einstellung des Anstellwinkels der Rotorblätter angewendet werden kann.

Die HTML-Seite enthält eine Schaltfläche "Senden", um die Übersteuerungseinstellungen an die Steuerung zu senden.

Das Eingabefeld für die Bremsung bietet eine manuelle Einstellung eines Prozentwerts für die Bremsung. Das STEP 7-Anwenderprogramm verlangt keine manuelle Übersteuerung, um einen Bremswert zu übernehmen.

Außerdem nutzt die HTML-Seite einen AWP-Befehl, um die Sondervariable, die die Benutzer-ID des Benutzers enthält, der auf die Seite zugreift, in eine Variable in der PLC-Variablentabelle zu schreiben (Seite 855).

12.7.8.2 Steuerungsdaten lesen und anzeigen

Die HTML-Seite "Remote Wind Turbine Monitor" nutzt zahlreiche AWP-Befehle zum Lesen von Daten aus der Steuerung (Seite 851), um diese Daten auf der Seite anzuzeigen. Beachten Sie beispielsweise den HTML-Code zum Anzeigen der Leistungsausgabe in diesem Teil der Beispiel-Webseite:

Beispiel für einen HTML-Code

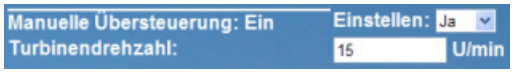
Der folgende Auszug aus der HTML-Seite "Remote Wind Turbine Monitor" zeigt den Text "Leistungsabgabe:" in der linken Zelle einer Tabellenzeile an und liest die Variable für die Leistungsabgabe und zeigt sie in der rechten Zelle der Tabellenzeile zusammen mit dem Text für die Einheiten, kW, an.

Der AWP-Befehl :="Datenbaustein_1".Leistungsabgabe: führt die Leseoperation durch. Beachten Sie, dass Datenbausteine über den Namen und nicht über die Nummer des Datenbausteins referenziert werden (d.h. über "Datenbaustein_1" und nicht über "DB1").

```
<tr style="height:2%;">
<td>
<p>Leistungsabgabe:</p>
</td>
<td>
<p style="margin-bottom:5px;"> :="Datenbaustein_1".Leistungsabgabe:
kW</p>
</td>
</tr>
```

12.7.8.3 Enum-Typ verwenden

Die HTML-Seite "Remote Wind Turbine Monitor" verwendet Enum-Typen für die drei Stellen, an denen die HTML-Seite "ON" oder "OFF" für einen Booleschen Wert anzeigt und an der der Benutzer einen Booleschen Wert eingibt. Der Enum-Typ für "ON" führt zu einem Wert von 1, und der Enum-Typ für "OFF" führt zu einem Wert von 0. Beispiel: Betrachten Sie den HTML-Code zum Lesen und Schreiben der Einstellung zum Aktivieren der manuellen Übersteuerung im Wert "Datenbaustein_1".FreigabeManuelleÜbersteuerung mittels eines Enum-Typs:



Beispiel für einen HTML-Code

Die folgenden Auszüge aus der HTML-Seite "Remote Wind Turbine Monitor" zeigen die Deklaration eines Enum-Typs mit dem Namen "Übersteuerungsstatus" mit den Werten für "Aus" und "Ein" von 0 und 1 und das nachfolgende Festlegen einer Enum-Typreferenz von "Übersteuerungsstatus" für die Boolesche Variable "FreigabeManuelleÜbersteuerung" im Datenbaustein "Datenbaustein_1".

```
<!-- AWP_In_Variable
Name='"Datenbaustein_1".FreigabeManuelleÜbersteuerung'
Enum="Übersteuerungsstatus" -->

<!-- AWP_Enum_Def Name="Übersteuerungsstatus"
Values='0:"Aus",1:"Ein"' -->
```

Wenn die HTML-Seite ein Anzeigefeld in einer Tabellenzelle für den aktuellen Zustand von "FreigabeManuelleÜbersteuerung" beinhaltet, wird lediglich ein normaler Lesebefehl für Variablen verwendet, doch dank des zuvor deklarierten und referenzierten Enum-Typs zeigt die Seite "Off" oder "On" und nicht 0 oder 1 an.

```
<td style="width:24%; border-top-style: Solid; border-top-width:
2px; border-top-color: #ffffff;">
<p>Manuelle Übersteuerung:
:="Datenbaustein_1".FreigabeManuelleÜbersteuerung:</p>
</td>
```

Die HTML-Seite enthält eine Klappliste, damit der Anwender den Wert von "FreigabeManuelleÜbersteuerung" ändern kann. Die Auswahlliste zeigt den Text "Ja" und "Nein" als Optionen an. Durch den Enum-Typ ist der Text "Ja" mit dem Wert "Ein" und der Text "Nein" mit dem Wert "Aus" verbunden. Wird keine Auswahl getroffen, bleibt der Wert von "FreigabeManuelleÜbersteuerung" unverändert.

```
<select name='"Datenbaustein_1".FreigabeManuelleÜbersteuerung'>
<option value=':"Datenbaustein_1".FreigabeManuelleÜbersteuerung:'>
</option>
<option value="Ein">Ja</option>
<option selected value="Aus">Nein</option>
</select>
```

Die Auswahlliste ist in einem Formular auf der HTML-Seite enthalten. Wenn der Anwender auf die Schaltfläche "Senden" klickt, lädt die Seite das Formular hoch. Dadurch wird der Wert "1" in die Boolesche Variable "FreigabeManuelleÜbersteuerung" in Datenbaustein_1 geschrieben, sofern der Anwender "Ja" ausgewählt hat, bzw. es wird "0" geschrieben, sofern der Anwender "Nein" ausgewählt hat.

12.7.8.4 Benutzereingaben in die Steuerung schreiben

Die HTML-Seite "Remote Wind Turbine Monitor" umfasst verschiedene AWP-Befehle zum Schreiben von Daten in die Steuerung (Seite 852). Die HTML-Seite deklariert AWP_In_Variables für Boolesche Variablen, so dass ein Benutzer mit Rechten zum Ändern von Variablen die Windturbine manuell steuern kann und die manuelle Übersteuerung für die Turbinendrehzahl, die Übersteuerung der Turbinenausrichtung und/oder des Anstellwinkels der Rotorblätter aktivieren kann. Die Seite umfasst zudem AWP_In_Variables, damit ein Benutzer mit Rechten zum Ändern von Variablen nachfolgend Gleitpunktwerte für Turbinendrehzahl, Ausrichtung, Anstellwinkel und Prozentwert für die Bremsung festlegen kann. Die Seite nutzt einen HTTP-Befehl zum Einstellen eines Formulars (Form POST), um die AWP_In_Variables in die Steuerung zu schreiben.

Beachten Sie z. B. den HTML-Code für die manuelle Einstellung des Bremswerts:

Beispiel für einen HTML-Code

Der folgende Ausschnitt aus der HTML-Seite "Remote Wind Turbine Monitor" deklariert zunächst eine AWP_In_Variable für "Datenbaustein_1", die es der HTML-Seite ermöglicht, in beliebige Variablen im Datenbaustein "Datenbaustein_1" zu schreiben. Die Seite zeigt den Text "Bremsen:" in der linken Zelle der Tabellenzeile an. In der rechten Zelle der Tabellenzeile befindet sich das Feld, in dem Benutzereingaben für die Variable "Bremsen" von "Datenbaustein_1" möglich sind. Dieser Benutzereingabewert befindet sich in einem HTML-Formular, das die eingegebenen Textdaten über die HTTP-Methode "POST" in die CPU schreibt. Die Seite liest dann den tatsächlichen Verzögerungswert aus der Steuerung aus und zeigt ihn im Dateneingabefeld an.

Ein Benutzer mit Rechten zum Ändern von Variablen kann anschließend über diese Seite einen Bremswert in den Datenbaustein der CPU schreiben, der die Bremsung steuert.

```
<!-- AWP_In_Variable Name='"Datenbaustein_1"' -->
...
<tr style="vertical-align: top; height: 2%;">
<td style="width: 22%;"><p>Bremsen:</p></td>
<td>
<form method="POST">
<p><input name='"Datenbaustein_1".Bremsen' size="10" type="text">
%</p>
</form>
</td>
</tr>
```

Hinweis

Beachten Sie Folgendes: Wenn eine benutzerdefinierte Seite ein Dateneingabefeld für eine schreibbare Datenbausteinvariable vom Datentyp "String" enthält, muss der Anwender die Zeichenkette bei der Eingabe des Textes in das Feld in einfache Anführungszeichen setzen.

Hinweis

Beachten Sie, dass, wenn Sie einen gesamten Datenbaustein in einer Deklaration AWP_In_Variable deklarieren, z. B. <!-- AWP_In_Variable Name="'Datenbaustein_1"' -->, dann kann jede Variable in dem Datenbaustein über die benutzerdefinierte Webseite geschrieben werden. Gehen Sie auf diese Weise vor, wenn Sie möchten, dass alle Variablen in einem Datenbaustein schreibbar sein sollen. Andernfalls, wenn Sie möchten, dass nur spezifische Datenbausteinvariablen über die benutzerdefinierte Webseite beschreibbar sein sollen, deklarieren Sie dies spezifisch anhand einer Deklaration wie <!-- AWP_In_Variable Name="'Datenbaustein_1".Bremsung' -->

12.7.8.5 Sondervariablen schreiben

Die Webseite "Remote Wind Turbine Monitor" schreibt die Sondervariable SERVER:current_user_id in eine PLC-Variable in der CPU, wenn der Benutzer über Änderungsberechtigungen verfügt. In diesem Fall enthält der PLC-Variablenwert die Benutzer-ID des Benutzers, der auf die Webseite "Remote Wind Turbine Monitor" zugreift.

Die Sondervariable wird von der Webseite in das PLC-Gerät geschrieben und benötigt keine Benutzeroberfläche.

Beispiel für einen HTML-Code

```
<!-- AWP_In_Variable Name="SERVER:current_user_id" Use="Benutzer-
ID"-->
```

12.7.8.6 Referenz: HTML-Code der Webseite "Remote Wind Turbine Monitor"

Windturbine.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<!--
Dieses Testprogramm simuliert eine Webseite zur Überwachung und
Bedienung einer Windturbine.
Erforderliche PLC-Variablen und Datenbausteinvariablen in STEP 7:

PLC-Variable:
Benutzer-ID: Int

Datenbausteine:
Datenbaustein_1

Variablen in Datenbaustein_1:

Turbinennummer: Int
Windgeschwindigkeit: Real
Windrichtung: Real
Temperatur: Real
Leistungsabgabe: Real
FreigabeManuelleÜbersteuerung: Bool
Turbinendrehzahl: Real
ÜbersteuerungAusrichtung: Bool
Ausrichtung: Real
ÜbersteuerungAnstellwinkel: Bool
Anstellwinkel: Real
Bremsen: Real
Die benutzerdefinierte Webseite zeigt aktuelle Werte für die PLC-
Daten und bietet eine Auswahlliste, um die drei Booleschen Werte mit
zugewiesenem Aufzählungstyp festzulegen. Über die Schaltfläche
"Senden" werden die ausgewählten Booleschen Werte ebenso wie die
Dateneingabefelder für Drehzahl, Ausrichtung und Anstellwinkel der
Turbine hochgeladen. Der Bremswert kann ohne die Schaltfläche
"Senden" festgelegt werden.

Für die Verwendung dieser Seite ist kein tatsächliches STEP 7-
Programm erforderlich. Theoretisch würde das STEP 7-Programm nur auf
die Werte für Drehzahl, Ausrichtung und Anstellwinkel der Turbine
reagieren, wenn die zugewiesenen Booleschen Werte festgelegt wären.
Die einzige Anforderung an STEP 7 ist, die WWW-Anweisung mit der DB-
Nummer der generierten Datenbausteine für diese Seite aufzurufen.
-->
<!-- AWP_In_Variable Name='"Datenbaustein_1"' -->
<!-- AWP_In_Variable
Name='"Datenbaustein_1".FreigabeManuelleÜbersteuerung'
Enum="Übersteuerungsstatus" -->
<!-- AWP_In_Variable
Name='"Datenbaustein_1".ÜbersteuerungAnstellwinkel'
Enum="Übersteuerungsstatus" -->
```

```
<!-- AWP_In_Variable
Name='"Datenbaustein_1".ÜbersteuerungAusrichtung'
Enum="Übersteuerungsstatus" -->
<!-- AWP_In_Variable Name="SERVER:current_user_id" Use="Benutzer-
ID"-->
<!-- AWP_Enum_Def Name="Übersteuerungsstatus"
Values='0:"Aus",1:"Ein"' -->

<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-
8"><link rel="stylesheet" href="Windturbine.css">
<title>Fernüberwachung von Windturbinen</title>
</head>
<body>
<table cellpadding="0" cellspacing="2">
<tr style="height: 2%;">
<td colspan="2">
<h2>Fernüberwachung von Windturbinen: Turbine
Nr.:="Datenbaustein_1".Turbinennummer:</h2>
</td>

<tr style="height: 2%;"><td style="width:
25%;"><p>Windgeschwindigkeit:</p></td>
<td><p> :="Datenbaustein_1".Windgeschwindigkeit: km/h</p></td>
</tr>

<tr style="height: 2%;">
<td style="width: 25%;"><p>Windrichtung:</p></td>
<td><p> :="Datenbaustein_1".Windrichtung: Grad</p></td>
</tr>

<tr style="height: 2%;"><td style="width:
25%;"><p>Temperatur:</p></td>
<td><p> :="Datenbaustein_1".Temperatur: Grad C</p></td>
</tr>

<tr style="height: 2%;">
<td style="width: 25%;"><p>Leistungsabgabe:</p></td>
<td><p style="margin-bottom:5px;">
:="Datenbaustein_1".Leistungsabgabe: kW</p>
</td>
</tr>

<form method="POST" action="">
<tr style="height: 2%;" >
<td style="width=25%; border-top-style: Solid; border-top-width:
2px; border-top-color: #ffffff;">
<p>Manuelle Übersteuerung:
:="Datenbaustein_1".FreigabeManuelleÜbersteuerung:</p>
</td>
<td class="Text">Einstellen:

<select name='"Datenbaustein_1".FreigabeManuelleÜbersteuerung'>
```

```
<option value=':="Datenbaustein_1".FreigabeManuelleÜbersteuerung:'>
</option>
<option value="Ein">Ja</option>
<option value="Aus">Nein</option>
</select>

</td>
</tr>

<tr style="vertical-align: top; height: 2%;"><td style="width:
25%;"><p>Turbinendrehzahl:</p></td>
<td>
<p style="margin-bottom:5px;"><input
name='"Datenbaustein_1".Turbinendrehzahl' size="10"
value=':="Datenbaustein_1".Turbinendrehzahl:' type="text"> U/min</p>
</td>
</tr>

<tr style="vertical-align: top; height: 2%;">
<td style="width: 25%;">
<p>Übersteuerung Ausrichtung:
:="Datenbaustein_1".ÜbersteuerungAusrichtung: </p>
</td>
<td class="Text">Einstellen:

<select name='"Datenbaustein_1".ÜbersteuerungAusrichtung'>
<option value=':="Datenbaustein_1".ÜbersteuerungAusrichtung:'>
</option>
<option value="Ein">Ja</option>
<option value="Aus">Nein</option>
</select>

</td>
</tr>

<tr style="vertical-align: top; height: 2%;">
<td style="width: 25%;">
<p>Ausrichtung Turbine:</p>
</td>
<td>
<p style="margin-bottom:5px;"><input
name='"Datenbaustein_1".Ausrichtung' size="10"
value=':="Datenbaustein_1".Ausrichtung:' type="text"> Grad</p>
</td>
</tr>

<tr style="vertical-align: top; height: 2%;">
<td style="width: 25%;">
<p>Übersteuerung Anstellwinkel:
:="Datenbaustein_1".ÜbersteuerungAnstellwinkel: </p>
</td>
<td class="Text">Einstellen:

<select name='"Datenbaustein_1".ÜbersteuerungAnstellwinkel'>
```

```
<option value=':="Datenbaustein_1".ÜbersteuerungAnstellwinkel:'>
</option>
<option value="Ein">Ja</option>
<option value="Aus">Nein</option>
</select>

</td>
</tr>

<tr style="vertical-align: top; height: 2%;">
<td style="width=25%; border-bottom-style: Solid; border-bottom-
width: 2px; border-bottom-color: #ffffff;">
<p>Anstellwinkel Rotorblätter:</p>
</td>
<td>
<p style="margin-bottom:5px;"><input
name='"Datenbaustein_1".Anstellwinkel' size="10"
value=':="Datenbaustein_1".Anstellwinkel:' type="text"> Grad</p>
</td>

</tr>
<tr style="height: 2%;">
<td colspan="2">
<input type="submit" value="Einstellungen und Werte für
Übersteuerung absenden">
</td>
</tr>
</form>

<tr style="vertical-align: top; height: 2%;">
<td style="width: 25%;"><p>Bremsen:</p></td>
<td>
<form method="POST" action="">
<p> <input name='"Datenbaustein_1".Bremsen' size="10"
value=':="Datenbaustein_1".Bremsen:' type="text"> %</p>
</form>
</td>
</tr>
<tr><td></td></tr>

</table>
</body>
</html>
```

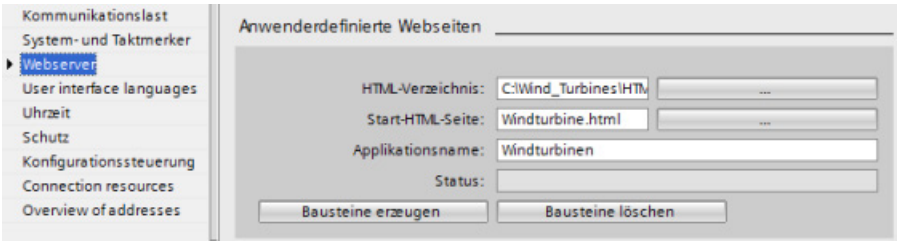
Windturbine.css

```
BODY {
 background-image: url('./Wind_turbine.jpg');
 background-position: 0% 0%;
 background-repeat: no-repeat;
 background-size: cover;
}
H2 {
 font-family: Arial;
 font-weight: bold;
 font-size: 14.0pt;
 color: #FFFFFF;
 margin-top:0px;
 margin-bottom:10px;
}
P {
 font-family: Arial;
 font-weight: bold;
 color: #FFFFFF;
 font-size: 12.0pt;
 margin-top:0px;
 margin-bottom:0px;
}
TD.Text {
 font-family: Arial;
 font-weight: bold;
 color: #FFFFFF;
 font-size: 12.0pt;
 margin-top:0px;
 margin-bottom:0px;
}
```

12.7.8.7 Konfiguration der Beispiel-Webseite in STEP 7

Um die HTML-Seite "Remote Wind Turbine Monitor" als benutzerdefinierte Webseite für die S7-1200 einzufügen, konfigurieren Sie die Daten für die HTML-Seite in STEP 7 und erstellen aus der HTML-Seite Datenbausteine.

Sie greifen auf die CPU-Eigenschaften der S7-1200, die die Windturbine steuert, zu und geben die Konfigurationsinformationen in die Eigenschaften der benutzerdefinierten Webseiten des Webservers ein:



Konfigurationsfelder

- HTML-Verzeichnis: Dieses Feld gibt den vollständig qualifizierten Pfadnamen auf den Ordner an, in dem sich die Standardseite (Startseite) auf dem Computer befindet. Über die Schaltfläche "..." können Sie zu dem gewünschten Ordner blättern.
- Standard-HTML-Seite: Dieses Feld gibt den Dateinamen der Standardseite bzw. der Startseite der HTML-Anwendung an. Über die Schaltfläche "..." können Sie die gewünschte Datei auswählen. In diesem Beispiel ist Windturbine.html die Standard-HTML-Seite. Das Beispiel "Remote Wind Turbine Monitor" besteht lediglich aus einer einzigen Seite, doch in anderen benutzerdefinierten Anwendungen können von der Standardseite weitere Seiten über Links aufgerufen werden. Im HTML-Code muss die Standardseite andere Seiten relativ zum HTML-Quellordner referenzieren.
- Anwendungsname: Dieses Feld ist optional und enthält den Namen, den der Webbrowser im Adressfeld anzeigt, wenn die Seite aufgerufen wird. In diesem Beispiel lautet der Name "Remote Wind Turbine Monitor", Sie können jedoch einen beliebigen Namen eingeben.

Es müssen keine weiteren Felder konfiguriert werden.

Abschließende Schritte

Um die HTML-Seite "Remote Wind Turbine Monitor" wie konfiguriert nutzen zu können, erstellen Sie die Bausteine, programmieren die WWW-Anweisung (Seite 866) mit der Nummer des generierten Steuer-DBs als Eingangsparameter, laden die Programmbausteine in die CPU und versetzen die CPU in den Betriebszustand RUN.

Wenn ein Bediener danach die Standard-Webseiten für die S7-1200 aufruft, von der die Windturbine gesteuert wird, kann er über den Link "Benutzerseiten" in der Navigationsleiste die Webseite "Remote Wind Turbine Monitor" anzeigen. Diese Seite bietet nun die Mittel, um die Windturbine zu überwachen und zu steuern.

12.7.9 Benutzerdefinierte Webseiten in mehreren Sprachen einrichten

Der Webserver bietet Ihnen die Möglichkeit, benutzerdefinierte Webseiten in den folgenden Sprachen anzulegen:

- Deutsch (de)
- Englisch (en)
- Spanisch (es)
- Französisch (fr)
- Italienisch (it)
- Vereinfachtes Chinesisch (zh)

Hierfür legen Sie Ihre HTML-Seiten in einer Ordnerstruktur (Seite 883) entsprechend den Sprachen an und richten das spezifische Cookie "siemens_automation_language" für Ihre Seiten ein (Seite 883). Der Webserver reagiert auf dieses Cookie und schaltet zu der Standardseite in dem jeweiligen Sprachordner um.

12.7.9.1 Ordnerstruktur anlegen

Um benutzerdefinierte Webseiten in mehreren Sprachen bereitzustellen, richten Sie in Ihrem HTML-Verzeichnis eine Ordnerstruktur ein. Die aus zwei Buchstaben bestehenden Ordnernamen sind spezifisch und müssen wie im Folgenden gezeigt vergeben werden:

de: Deutsch
en: Englisch
es: Spanisch
fr: Französisch
it: Italienisch
zh: Vereinfachtes Chinesisch

Auf der gleichen Ebene können Sie auch andere Ordner für Ihre Seiten anordnen, z. B. Ordner für Bilder oder Skripte.

Sie können beliebige Sprachordner einfügen. Sie brauchen nicht alle sechs Sprachen aufzunehmen. In den Sprachordnern erstellen und programmieren Sie Ihre HTML-Seiten in der jeweiligen Sprache.

12.7.9.2 Sprachumschaltung programmieren

Der Webserver führt die Sprachumschaltung anhand des Cookies "siemens_automation_language" durch. Dieses Cookie wird in den HTML-Seiten definiert und eingerichtet und vom Webserver ausgewertet, um eine Seite in der jeweiligen Sprache aus dem Sprachordner mit dem Namen der entsprechenden Sprache anzuzeigen. Die HTML-Seite muss JavaScript enthalten, um für dieses Cookie eine der vordefinierten Sprachkennungen einzurichten: "de", "en", "es", "fr", "it" oder "zh".

Setzt die HTML-Seite z. B. das Cookie auf "de", schaltet der Webserver zum Ordner "de" um und zeigt die Seite mit dem HTML-Standardseitennamen gemäß der STEP 7-Konfiguration (Seite 887) an.

Beispiel

Im folgenden Beispiel gibt es eine HTML-Standardseite mit dem Namen "langswitch.html" in jedem der Sprachordner. Im HTML-Verzeichnis befindet sich außerdem der Ordner "script". Der Ordner "script " enthält eine JavaScript-Datei mit dem Namen "lang.js". Jede Seite "langswitch.html " nutzt dieses JavaScript, um das Sprachcookie "siemens_automation_language" festzulegen.

HTML für "langswitch.html" im Ordner "en"

Die Kopfzeile der HTML-Seite richtet die Sprache "Englisch" ein, legt den Zeichensatz "UTF-8" fest und gibt den Pfad der JavaScript-Datei "lang.js" an.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Language" content="en">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Language switching english page</title>
<script type="text/javascript" src="script/lang.js" ></script>
```

Der Inhalt der Datei enthält eine Auswahlliste, über die der Anwender zwischen Deutsch und Englisch wählen kann. Englisch ("en") ist die vorausgewählte Sprache. Wenn der Anwender die Sprache wechselt, ruft die Seite die JavaScript-Funktion DoLocalLanguageChange() mit dem Wert der ausgewählten Option auf.

```
<!-- Language Selection -->
<table>
   <tr>
      <td align="right" valign="top" nowrap>
       <!-- change language immediately on selection change -->
          <select name="Language"
                    onchange="DoLocalLanguageChange(this)"
                    size="1">
            <option value="de" >German</option>
            <option value="en" selected >English</option>
           </select>
      </td>
   </tr>
</table><!-- Language Selection End-->
```

HTML für "langswitch.html" im Ordner "de"

Die Kopfzeile für die deutsche Seite "langswitch.html" ist mit der Kopfzeile der englischen Seite identisch, außer dass die deutsche Sprache eingestellt ist.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Language" content="de"><meta http-
equiv="Content-Type" content="text/html; charset=utf-8">
<title>Sprachumschaltung Deutsche Seite</title>
<script type="text/javascript" src="script/lang.js" ></script>
</head>
```

Der HTML-Code auf der deutschen Seite ist mit dem der englischen Seite identisch, außer dass der Standardwert der ausgewählten Sprache Deutsch ("de") ist.

```
<!-- Language Selection -->
<table>
   <tr>
     <td align="right" valign="top" nowrap>
       <!-- change language immediately on change of the selection -
->
       <select name="Language"
               onchange="DoLocalLanguageChange(this)"
               <size="1">
            <option value="de" selected >Deutsch</option>
            <option value="en" >Englisch</option>
       </select>
      </td>
   </tr>
</table><!-- Language Selection End-->
```

JavaScript "lang.js" im Ordner "script"

Die Funktion "DoLocalLanguageChange()" befindet sich in der Datei "lang.js". Diese Funktion ruft die Funktion "SetLangCookie()" auf und aktualisiert dann das Fenster, in dem die HTML-Seite angezeigt wird.

Die Funktion "SetLangCookie()" weist dem Cookie "siemens_automation_language" des Dokuments den Wert aus der Auswahlliste zu. Die Funktion legt auch den Pfad der Anwendung fest, so dass die umgeschaltete Seite und nicht die anfordernde Seite den Wert des Cookies empfängt.

Optional kann die Seite im kommentierten Abschnitt einen Wert für die Gültigkeit des Cookies angeben.

```
function DoLocalLanguageChange(oSelect) {
        SetLangCookie(oSelect.value);
        top.window.location.reload();
    }
function SetLangCookie(value) {
        var strval = "siemens_automation_language=";
        // Dies ist das Cookie, über das der Webserver
        // die gewünschte Sprache erkennt.
        // Dieser Name ist für den Webserver erforderlich.
        strval = strval + value;
        strval = strval + "; path=/ ;";
        // Geben Sie den Pfad für die Anwendung an, weil ansonsten
        // der Pfad auf die anfordernde Seite gelegt werden würde
        // und diese Seite das Cookie nicht erhalten würde.
        /* OPTIONAL
           Geben Sie eine Ablaufzeit an, wenn dieses Cookie länger
existieren soll
           als die aktuelle Browser-Sitzung:
           var now = new Date();
           var endttime = new Date(now.getTime() + expiration);
           strval = strval + "; expires=" +
                    endttime.toGMTString() + ";";
        */
        document.cookie = strval;
    }
```

Hinweis

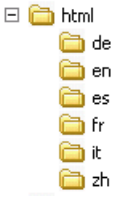
Wenn Ihre benutzerdefinierte Webseitenimplementierung sowohl HTML-Dateien in sprachspezifischen Ordnern (z. B. en, de) als auch HTML-Dateien enthält, die sich nicht in sprachspezifischen Ordnern befinden, können Sie mit dem Befehl AWP_Enum_Def Enum-Typen nicht in Dateien an beiden Speicherorten definieren. Stattdessen müssen Sie bei der Verwendung von Enum-Typen diese entweder in Dateien in den sprachspezifischen Ordnern oder in Dateien außerhalb der sprachspezifischen Ordner definieren. Sie können Enum-Deklarationen nicht in Dateien in beiden Speicherorten vornehmen.

12.7.9.3 STEP 7 für die Verwendung einer mehrsprachigen Seitenstruktur konfigurieren

Um mehrsprachige benutzerdefinierte Webseiten zu konfigurieren, gehen Sie ähnlich vor wie beim Konfigurieren von benutzerdefinierten Webseiten (Seite 865). Wenn Sie jedoch Ordner für Sprachen eingerichtet haben, geben Sie in der Einstellung des HTML-Verzeichnisses den Ordner an, der die einzelnen Spachordner enthält. Sie richten das HTML-Verzeichnis nicht als einen der Sprachordner ein.

Wenn Sie die HTML-Standardseite auswählen, navigieren Sie in den Sprachordner und wählen die HTML-Seite aus, die die Startseite sein soll. Wenn Sie anschließend Bausteine generieren und die Bausteine in die CPU laden, zeigt der Webserver die Startseite aus dem konfigurierten Sprachordner an.

Beispiel: Befindet sich die hier gezeigte Ordnerstruktur auf C:\, ist die Einstellung für das HTML-Verzeichnis C:\html. Und wenn zunächst die englische Seite angezeigt werden soll, navigieren Sie für die Einstellung der HTML-Standardseite zum Pfad en\langswitch.html.



12.7.10 Erweiterte Steuerung von benutzerdefinierten Webseiten

Wenn Sie für Ihre benutzerdefinierten Webseiten Datenbausteine generieren, erstellt STEP 7 einen Steuer-DB, der die Anzeige von und die Interaktion mit den benutzerdefinierten Webseiten steuert. STEP 7 erstellt zudem einen Satz DB-Fragmente, die die einzelnen Seiten darstellen. Unter normalen Umständen müssen Sie die Struktur des Steuer-DBs nicht kennen und brauchen auch nicht zu wissen, wie Sie ihn ändern.

Wenn Sie beispielsweise eine Webanwendung ein- und ausschalten möchten oder einzelne Fragmente manuell ändern möchten, können Sie dies über die Variablen des Steuer-DBs und die WWW-Anweisung erreichen.

Struktur des Steuer-DBs

Der Steuer-DB besitzt eine umfangreiche Datenstruktur und ist zugänglich, wenn Sie Ihr STEP 7-Anwenderprogramm programmieren. Hier werden nur einige der Variablen des Steuerdatenbausteins beschrieben.

Commandstate-Struktur

"Commandstate" ist eine Struktur, die globale Befehle und globale Zustände für den Webserver enthält.

Globale Befehle in der "Commandstate"-Struktur

Die globalen Befehle gelten für den Webserver im Allgemeinen. Sie können den Webserver über die Parameter des Steuer-DBs deaktivieren oder neu starten.

Bausteinvariable	Datentyp	Beschreibung
init	BOOL	Auswertung des Steuer-DB und Initialisierung der Webanwendung
deactivate	BOOL	Deaktivierung der Webanwendung

Globale Zustände in der Commandstate-Struktur

Die globalen Zustände gelten für den Webserver im Allgemeinen und enthalten Statusinformationen über die Webanwendung.

Bausteinvariable	Datentyp	Beschreibung
initializing	BOOL	Webanwendung liest den Steuer-DB
error	BOOL	Webanwendung konnte nicht initialisiert werden
deactivating	BOOL	Webanwendung wird beendet
deactivated	BOOL	Webanwendung ist beendet
initialized	BOOL	Webanwendung ist initialisiert
last_error	INT	Letzter Fehler, der von einem WWW-Anweisungsaufruf (Seite 866) zurückgegeben wurde, wenn der Rückgabecode für WWW 16#0010 lautet: 16#0001: Struktur DB-Fragment inkonsistent 16#0002: Name der Anwendung existiert bereits 16#0003: keine Ressourcen (Speicher) 16#0004: Struktur Steuer-DB inkonsistent 16#0005: DB-Fragment nicht verfügbar 16#0006: DB-Fragment nicht für AWP 16#0007: Aufzählungsdaten inkonsistent 16#000D: Konflikt durch Größe des Steuer-DBs

Anforderungstabelle

Die Anforderungstabelle ist ein Array aus Strukturen mit Befehlen und Zuständen, die für einzelne DB-Fragmente gelten. Wenn Sie mit dem Befehl AWP_Start_Fragment (Seite 860) Fragmente des "manuellen" Typs angelegt haben, muss das STEP 7-Anwenderprogramm diese Seiten über den Steuer-DB steuern. Die Anforderungszustände sind schreibgeschützt und liefern Informationen zum aktuellen Fragment. Die Anforderungsbefehle nutzen Sie, um das aktuelle Fragment zu steuern.

Bausteinvariable	Datentyp	Beschreibung
requesttab	ARRAY [1 .. 4] OF STRUCT	Array aus Strukturen für die Steuerung einzelner DB-Fragmente Der Webserver kann bis zu vier Fragmente gleichzeitig bearbeiten. Der Array-Index für ein bestimmtes Fragment ist beliebig, wenn der Webserver mehrere Fragmente oder Fragmente aus mehreren Browser-Sessions verarbeitet.

Struct-Elemente der Requesttab-Struktur

Bausteinvariable	Datentyp	Beschreibung
page_index	UINT	Nummer der aktuellen Webseite
fragment_index	UINT	Nummer des aktuellen Fragments - Festlegung eines anderen Fragments möglich
// Anforderungsbefehle		
continue	BOOL	Gibt die aktuelle Seite bzw. das aktuelle Fragment zum Senden frei und fährt mit dem nächsten Fragment fort
repeat	BOOL	Gibt die aktuelle Seite bzw. das aktuelle Fragment zum erneuten Senden frei und fährt mit demselben Fragment fort
abort	BOOL	HTTP-Verbindung ohne Senden schließen
finish	BOOL	Dieses Fragment senden; Seite ist vollständig - keine weiteren Fragmente bearbeiten
// Anforderungszustände		Die Anforderungszustände sind schreibgeschützt.
idle	BOOL	Nichts durchzuführen, jedoch aktiv
waiting	BOOL	Fragment wartet auf Freigabe
sending	BOOL	Fragment sendet
aborting	BOOL	Benutzer hat die aktuelle Anforderung abgebrochen

Funktionsweise

Immer wenn Ihr Programm den Steuer-DB ändert, muss es die WWW-Anweisung mit der Nummer des geänderten Steuer-DBs als Parameter aufrufen. Die globalen Befehle und Anforderungsbefehle werden wirksam, wenn das STEP 7-Anwenderprogramm die WWW-Anweisung (Seite 866) ausführt.

Das STEP 7-Anwenderprogramm kann fragment_index explizit festlegen und bewirkt dadurch, dass der Webserver das angegebene Fragment mit einem Anforderungsbefehl verarbeitet. Andernfalls verarbeitet der Webserver das aktuelle Fragment für die aktuelle Seite, wenn die WWW-Anweisung ausgeführt wird.

Mögliche Techniken zur Verwendung von fragment_index sind u.a.:

- Aktuelles Fragment verarbeiten: Fragment_index unverändert lassen und den Fortsetzungsbefehl setzen.
- Aktuelles Fragment überspringen: fragment_index auf 0 setzen und den Fortsetzungsbefehl setzen
- Aktuelles Fragment durch ein anderes Fragment ersetzen: Für fragment_index die neue Fragment-ID angeben und den Fortsetzungsbefehl setzen

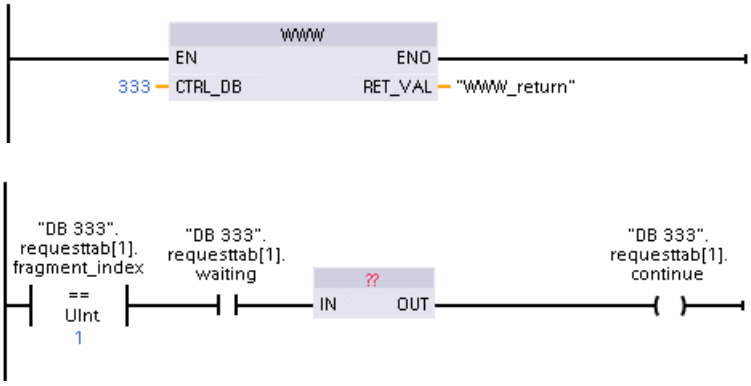
Um sich möglicherweise verändernde globale Zustände oder Anforderungszustände abzufragen, muss das STEP 7-Anwenderprogramm die WWW-Anweisung aufrufen, um die aktuellen Werte dieser Zustände auszuwerten. Eine typische Verwendung könnte es sein, die WWW-Anweisung regelmäßig aufzurufen, bis ein bestimmter Zustand eintritt.

Hinweis

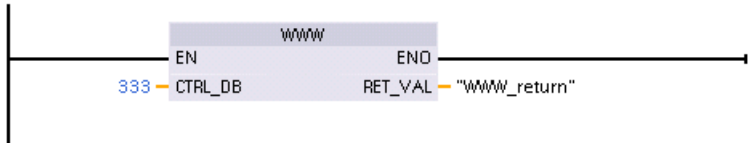
Wenn das STEP 7-Anwenderprogramm mehrere Anforderungsbefehle setzt, verarbeitet die WWW-Anweisung nur einen davon und hält dabei diese Reihenfolge ein: abbrechen, beenden, wiederholen, fortfahren. Die WWW-Anweisung löscht alle Anforderungsbefehle nach der Verarbeitung.

Beispiele

Das folgende Beispiel zeigt ein STEP 7-Anwenderprogramm, das auf ein Fragment mit der ID 1 im Wartezustand prüft, nachdem zuvor die WWW-Anweisung aufgerufen wurde. Es wartet möglicherweise auch, dass andere anwendungsspezifische Bedingungen auftreten. Dann führt es die für das Fragment erforderliche Verarbeitung durch, wobei es z. B. Datenbausteinvariablen setzt, Berechnungen durchführt oder andere anwendungsspezifische Aufgaben verarbeitet. Danach setzt es den Fortsetzungsmerker, damit der Webserver dieses Fragment ausführt.



Wenn das Programm die WWW-Anweisung mit diesem geänderten Steuer-DB aufruft, kann die benutzerdefinierte Webseite mit diesem Fragment vom Webbrowser angezeigt werden.



Beachten Sie, dass es sich hier um ein vereinfachtes Beispiel handelt. Das zu prüfende Fragment kann sich in jeder der vier requesttab-Strukturen des Arrays befinden.

12.8 Einschränkungen

Die folgenden IT-Aspekte können sich auf Ihre Nutzung des Webservers auswirken:

- Für den Zugriff auf die Standard-Webseiten oder auf benutzerdefinierte Webseiten muss im Allgemeinen die IP-Adresse der CPU oder die IP-Adresse eines Wireless-Routers mit einer Portnummer verwendet werden. Wenn Ihr Webbrowser keine direkte Verbindung mit einer IP-Adresse gestattet, wenden Sie sich an Ihren IT-Administrator. Wenn Ihre lokalen Richtlinien DNS unterstützen, können Sie über einen DNS-Eintrag eine Verbindung zu der IP-Adresse herstellen.
- Firewalls, Proxy-Einstellungen und andere standortspezifische Einschränkungen können ebenfalls den Zugriff auf die CPU begrenzen. Um solche Probleme zu beheben, wenden Sie sich an Ihren IT-Administrator.
- Die Standard-Webseiten verwenden JavaScript und Cookies. Sind JavaScript oder Cookies in Ihrem Webbrowser deaktiviert, aktivieren Sie sie. Wenn Sie sie nicht aktivieren können, sind einige Funktionen eingeschränkt (Seite 893). Die Verwendung von JavaScript und Cookies in benutzerdefinierten Webseiten ist optional. Wenn Sie sie verwenden, müssen sie in Ihrem Browser aktiviert sein.
- Der Webserver unterstützt Secure Sockets Layer (SSL). Sie können die Standard-Webseiten und benutzerdefinierten Webseiten über eine der beiden URLs http://ww.xx.yy.zz oder https://ww.xx.yy.zz aufrufen. Dabei steht "ww.xx.yy.zz" für die IP-Adresse der CPU.
- Siemens bietet für den sicheren Zugriff auf den Webserver ein Sicherheitszertifikat. Auf der Standard-Webseite "Einführung" (Seite 833) können Sie das Zertifikat herunterladen und in die Internetoptionen Ihres Webbrowsers importieren (Seite 894). Wenn Sie das Zertifikat nicht importieren, wird Ihnen bei jedem Aufruf des Webservers über die https://-Adresse eine Sicherheitsabfrage angezeigt.

Anzahl der Verbindungen

Der Webserver unterstützt maximal 30 aktive HTTP-Verbindungen. Diese 30 Verbindungen können auf verschiedene Weise genutzt werden, je nach dem von Ihnen verwendeten Webbrowser und der Anzahl der unterschiedlichen Objekte pro Seite (CSS-Dateien, Bilder, weitere HTML-Dateien). Einige Verbindungen bleiben bestehen, während der Webserver eine Seite anzeigt, andere werden nach der Erstverbindung freigegeben.

Wenn Sie beispielsweise den Browser Mozilla Firefox 8 nutzen, der maximal sechs beständige Verbindungen unterstützt, könnten Sie fünf Browser oder Browserregister verwenden, bevor der Webserver beginnt, Verbindungen abzubrechen. Sollte eine Seite nicht alle sechs Verbindungen nutzen, können Sie weitere Browser oder Browserregister verwenden.

Beachten Sie auch, dass die Anzahl aktiver Verbindungen die Leistung der Seite beeinflussen kann.

Hinweis

Vor dem Schließen des Webservers abmelden

Wenn Sie sich am Webserver angemeldet haben, sollten Sie sich vor dem Schließen des Webbrowsers unbedingt abmelden. Der Webserver unterstützt maximal sieben gleichzeitig angemeldete Benutzer.

12.8.1 Funktionseinschränkungen bei Deaktivierung von JavaScript in den Internetoptionen

Die Standard-Webseiten verwenden HTML, JavaScript und Cookies. Wenn an Ihrem Standort die Verwendung von JavaScript und Cookies eingeschränkt ist, aktivieren Sie diese, damit die Webseiten einwandfrei funktionieren. Wenn Sie in Ihrem Webbrowser JavaScript nicht aktivieren können, können Funktionen mit JavaScript-Steuerung nicht ausgeführt werden.

Allgemein

Die Seiten werden nicht dynamisch aktualisiert. Sie müssen die Seiten manuell über das Symbol zum Aktualisieren (Seite 827) aktualisieren, um die aktuellen Daten anzuzeigen.

Diagnosepufferseite

Die Diagnosepufferseite verwendet JavaScript wie folgt:

- Einzelheiten zu Ereignissen anzeigen: Mit JavaScript wählen Sie eine Zeile im Diagnosepuffer aus, um im unteren Abschnitt Einzelheiten dazu anzuzeigen. Ohne JavaScript müssen Sie auf den Hyperlink des Ereignisfelds eines Diagnosepuffereintrags klicken, um die Ereignisdaten im unteren Bereich anzuzeigen.
- Bereich der anzuzeigenden Diagnosepuffereinträge ändern: Mit JavaScript können Sie über die Klappliste oben auf der Seite den Bereich der anzuzeigenden Diagnosepuffereinträge auswählen, woraufhin die Seite automatisch aktualisiert wird. Ohne JavaScript können Sie über die Klappliste oben auf der Seite den Bereich der anzuzeigenden Diagnosepuffereinträge auswählen, doch Sie müssen dann auf den Link "Go" klicken, um die Diagnosepufferseite entsprechend zu aktualisieren.

Beachten Sie, dass der Hyperlink "Go" und die Hyperlinks der Ereignisfelder nur angezeigt werden, wenn JavaScript nicht aktiviert ist. Diese Links sind nicht erforderlich und deshalb auch nicht vorhanden, wenn JavaScript aktiviert ist.

Modulinformationsseite

Ohne aktiviertes JavaScript gelten die folgenden Einschränkungen:

- Sie können keine Daten filtern.
- Sie können keine Felder sortieren.

Variablenzustandsseite

Ohne aktiviertes JavaScript gelten die folgenden Einschränkungen:

- Nachdem Sie eine Variable eingegeben haben, müssen Sie den Fokus manuell in die Zeile "New variable" setzen, um eine neue Variable einzugeben.
- Wenn Sie ein Anzeigeformat auswählen, wird der Datenwert nicht automatisch im ausgewählten Format angezeigt. Sie müssen auf die Schaltfläche "Monitor value" klicken, um die Anzeige mit dem neuen Format zu aktualisieren.

12.8.2 Eingeschränkte Funktionen, wenn Cookies in den Internetoptionen nicht erlaubt sind

Wenn Cookies in Ihrem Webbrowser deaktiviert sind, gelten die folgenden Einschränkungen:

- Sie können sich nicht anmelden.
- Sie können die Sprache nicht wechseln.
- Sie können nicht von UTC- auf PLC-Zeit umschalten. Ohne Cookies sind alle Zeiten UTC-Zeiten.

12.8.3 Siemens-Sicherheitszertifikat importieren

Sie können das Siemens-Sicherheitszertifikat in Ihre Internetoptionen importieren, damit Ihnen nicht bei jedem Aufruf von https://ww.xx.yy.zz (wobei "ww.xx.yy.zz" die IP-Adresse des Geräts ist) in Ihrem Webbrowser die Sicherheitabfrage angezeigt wird. Wenn Sie eine http://-Adresse und keine https://-URL eingeben, brauchen Sie das Zertifikat nicht herunterzuladen und zu installieren.

Herunterladen des Zertifikats

Klicken Sie auf der Einführungsseite (Seite 833) auf den Link "Zertifikat herunterladen", um das Siemens-Sicherheitszertifikat auf Ihren PC herunterzuladen. Die Vorgehensweise unterscheidet sich dabei je nach Webbrowser:

Zertifikat im Internet Explorer importieren

1. Klicken Sie auf der Einführungsseite auf den Link "Zertifikat herunterladen". Es wird der Dialog "Dateidownload - Sicherheitswarnung" angezeigt.
2. Klicken Sie im Dialog "Dateidownload - Sicherheitswarnung" auf "Öffnen", um die Datei zu öffnen. Der Dialog "Zertifikat" wird angezeigt.
3. Klicken Sie im Dialog "Zertifikat" auf die Schaltfläche "Zertifikat installieren", um den Assistenten zum Importieren des Zertifikats aufzurufen.
4. Folgen Sie den Dialogen im "Zertifikatimport-Assistent", um das Zertifikat zu importieren. Dabei wählt das Betriebssystem automatisch den Zertifikatspeicher aus.

Zertifikat in Mozilla Firefox importieren

1. Klicken Sie auf der Einführungsseite auf den Link "Zertifikat herunterladen". Der Dialog "MiniWebCA_Cer.crt öffnen" wird angezeigt.
2. Klicken Sie im Dialog "MiniWebCA_Cer.crt öffnen" auf "Datei speichern". Der Dialog "Downloads" wird angezeigt.
3. Doppelklicken Sie im Dialog "Downloads" auf "MiniWebCA_Cer.crt". Wenn Sie den Download mehrmals gestartet haben, werden mehrere Kopien angezeigt. Doppelklicken Sie auf einen beliebigen Eintrag "MiniWebCA_Cer.crt".
4. Klicken Sie auf "OK", wenn Sie bestätigen sollen, dass eine ausführbare Datei geöffnet wird.
5. Klicken Sie ggf. im Dialog "Datei öffnen - Sicherheitswarnung" auf "Öffnen". Der Dialog "Zertifikat" wird angezeigt.
6. Klicken Sie im Dialog "Zertifikat" auf die Schaltfläche "Zertifikat installieren".
7. Folgen Sie den Dialogen im "Zertifikatimport-Assistent", um das Zertifikat zu importieren. Dabei wählt das Betriebssystem automatisch den Zertifikatspeicher aus.
8. Wenn der Dialog "Sicherheitswarnung" angezeigt wird, klicken Sie auf "Ja", um die Installation des Zertifikats zu bestätigen.

Andere Browser

Folgen Sie den Konventionen Ihres Webbrowsers, um das Siemens-Zertifikat zu importieren und zu installieren.

Nachdem Sie das Siemens-Sicherheitszertifikat "S7-1200 Controller Family" in den Internetoptionen für die Inhalte Ihres Webbrowsers installiert haben, brauchen Sie keine Sicherheitsabfrage zu bestätigen, wenn Sie über https:// ww.xx.yy.zz auf den Webserver zugreifen.

Hinweis

Das Sicherheitszertifikat bleibt auch bei einem Neustart der CPU gültig. Ändern Sie jedoch die IP-Adresse des Geräts, müssen Sie ein neues Zertifikat laden, sofern Sie einen anderen Browser als Internet Explorer oder Mozilla Firefox nutzen.

12.8.4 Datenprotokolle im CSV-Format in nicht amerikanische/englische Versionen von Microsoft Excel importieren

Die Protokolldateien sind im CSV-Dateiformat (durch Komma getrennte Werte) gespeichert. Sie können diese Dateien von der Datenprotokollseite direkt in Excel öffnen, wenn auf Ihrem System eine amerikanische oder englische Version von Excel installiert ist. In anderen Ländern ist dieses Format jedoch nicht so weit verbreitet, weil Kommas häufig in numerischer Notation auftreten.

Wenn Sie nicht eine amerikanische/englische Version von Excel verwenden, gehen Sie folgendermaßen vor, um eine gespeicherte Protokolldatei zu öffnen:

1. Öffnen Sie Microsoft Excel und legen Sie eine leere Arbeitsmappe an.
2. Wählen Sie im Menü "Daten > Externe Daten importieren" den Befehl "Daten importieren".
3. Navigieren Sie zu der Protokolldatei, die Sie öffnen möchten, und wählen Sie sie aus. Der Textimport-Assistent wird aufgerufen.
4. Ändern Sie im Textimport-Assistenten die Standardoption für "Ursprünglicher Datentyp" von "Feste Breite" in "Getrennt".
5. Wählen Sie die Schaltfläche "Weiter".
6. Aktivieren Sie im Dialog "Schritt 2" das Kontrollkästchen "Komma", um den Trennzeichentyp von "Tab" in "Komma" zu ändern.
7. Wählen Sie die Schaltfläche "Weiter".
8. Im Dialog "Schritt 3" können Sie optional das Datenformat von MDJ (Monat/Tag/Jahr) in ein anderes Format ändern.
9. Führen Sie die übrigen Schritt im Textimport-Assistent durch, um die Datei zu importieren.

Kommunikationsprozessor und Modbus-TCP 13

13.1 Mit den seriellen Kommunikationsschnittstellen arbeiten

Zwei Kommunikationsmodule (CMs) und ein Kommunikationsboard (CB) bieten die Schnittstelle für die PtP-Kommunikation:

- CM 1241 RS232 (Seite 1324)
- CM 1241 RS422/485 (Seite 1325)
- CB 1241 RS485 (Seite 1322)

Sie können bis zu drei CMs (jeden Typs) plus ein CB, insgesamt vier Kommunikationsschnittstellen anschließen. Bauen Sie das CM links von der CPU oder eines anderen CMs ein. Bauen Sie das CB auf der Vorderseite der CPU ein. Weitere Informationen zum Ein- und Ausbau von Modulen finden Sie in den Einbaurichtlinien (Seite 66).

Die seriellen Kommunikationsschnittstellen haben die folgenden Eigenschaften:

- Potentialgetrennter Anschluss
- Unterstützung von Punkt-zu-Punkt-Protokollen
- Konfiguration und Programmierung über die Anweisungen des Punkt-zu-Punkt-Kommunikationsprozessors
- Anzeige der Sende- und Empfangsaktivität über LEDs
- Diagnose-LED (nur CMs)
- Spannungsversorgung über die CPU: Keine externe Stromversorgung erforderlich.

Siehe technische Daten der Kommunikationsschnittstellen (Seite 1311).

LED-Anzeigen

Die Kommunikationsmodule haben drei LED-Anzeigen:

- Diagnose-LED (DIAG): Diese LED blinkt rot, bis sie von der CPU angesprochen wird. Nach dem Anlauf der CPU prüft diese auf CMs und adressiert diese. Die Diagnose-LED beginnt, grün zu blinken. Das bedeutet, dass die CPU das CM adressiert, ihm jedoch noch keine Konfiguration zugewiesen hat. Die CPU lädt die Konfiguration in die konfigurierten CMs, wenn das Programm in die CPU geladen wird. Nach dem Laden in die CPU muss die Diagnose-LED am Kommunikationsmodul dauerhaft grün leuchten.
- Sende-LED (Tx): Die Sende-LED leuchtet, wenn Daten über den Kommunikationsport gesendet werden.
- Empfangs-LED (Rx): Diese LED leuchtet, wenn Daten über den Kommunikationsport empfangen werden.

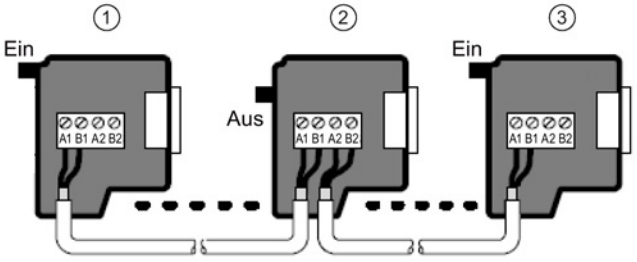
Das Kommunikationsboard bietet LEDs für Senden (TxD) und Empfangen (RxD). Es hat keine Diagnose-LED.

13.2 Abschließen eines RS485-Busanschlusssteckers

Siemens bietet einen RS485-Busanschlussstecker (Seite 1342), mit dem Sie problemlos mehrere Geräte an ein RS485-Netz anschließen können. Der Stecker verfügt über zwei Klemmensätze, mit denen Sie die Eingangs- und Ausgangskabel für das Netz befestigen können. Der Stecker verfügt zudem über Schalter, um das Netz selektiv abschließen zu können.

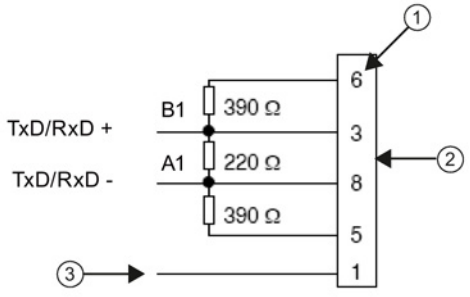
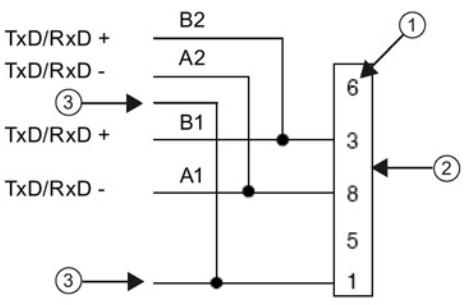
Hinweis

Sie schließen nur die zwei Enden des RS485-Netzes ab. Die Geräte zwischen den beiden Endgeräten werden nicht abgeschlossen. Blanker Kabelschirm: Ca. 12 mm muss blank auf der Metallführung liegen.



① Schalterstellung = Ein: Abschlusswiderstand zugeschaltet
② Schalterstellung = Aus: Abschlusswiderstand nicht zugeschaltet
③ Schalterstellung = Ein: Abschlusswiderstand zugeschaltet

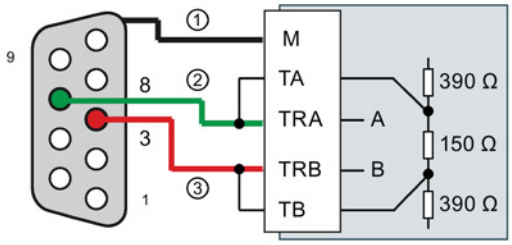
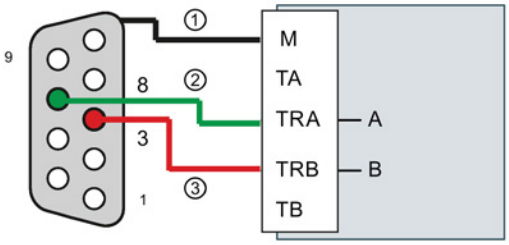
Tabelle 13- 1 Abschluss des RS485-Steckers

Abschlussgerät (Abschluss EIN)	Kein Abschlussgerät (Abschluss AUS)
	

① Pinnummer
② Busanschlussstecker
③ Kabelschirm

Das CB 1241 bietet interne Widerstände für den Netzabschluss. Um die Verbindung abzuschließen, verbinden Sie TRA mit TA und TRB mit TB, um die internen Widerstände in den Schaltkreis aufzunehmen. CB 1241 hat keinen 9-poligen Steckverbinder. Die folgende Tabelle zeigt die Anschlüsse an einen 9-poligen Steckverbinder am Kommunikationspartner.

Tabelle 13- 2 Abschluss des CB 1241

Abschlussgerät (Abschluss EIN)	Kein Abschlussgerät (Abschluss AUS)
	

① M an den Kabelschirm anschließen

① A = TxD/RxD - (grüne Ader/Pol 8)

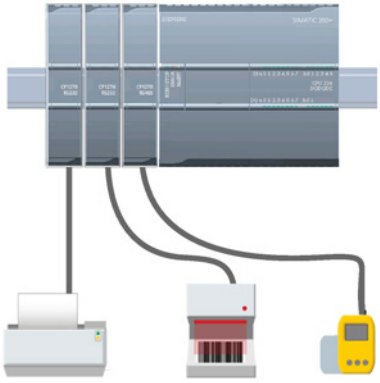
② B = TxD/RxD + (rote Ader/Pol 3)

13.3 Punkt-zu-Punkt-Kommunikation (PtP)

Die CPU unterstützt die folgende Punkt-zu-Punkt-Kommunikation (PtP) für zeichenbasierte serielle Protokolle.

- PtP
- USS (Seite 952)
- Modbus (Seite 973)

Das Punkt-zu-Punkt-Protokoll bietet maximale Freiheit und Flexibilität, erfordert jedoch eine aufwändige Implementierung im Anwenderprogramm.



PtP bietet eine Vielzahl an Möglichkeiten:

- Direktes Senden von Informationen an ein externes Gerät, wie z. B. einen Drucker
- Empfangen von Informationen von anderen Geräten, wie z. B. Strichcodelesern, RFID-Lesern, Kamera- oder Überwachungssystemen anderer Hersteller und vielen anderen Gerätearten
- Informationsaustausch, Senden und Empfangen von Daten mit anderen Geräten, wie z. B. GPS-Geräten, Kamera- oder Überwachungssystemen anderer Hersteller, Funkmodems und vielen anderen

Diese PtP-Kommunikation ist eine serielle Kommunikation mit Standard-UARTs, die verschiedene Baudraten und Paritäten unterstützen. Die RS232- und RS422/485-Kommunikationsmodule (CM 1241) sowie das RS485-Kommunikationsboard (CB 1241) stellen die elektrischen Schnittstellen für die PtP-Kommunikation zur Verfügung.

PtP über PROFIBUS oder PROFINET

Version V4.1 der S7-1200 CPU in Verbindung mit STEP 7 V13 SP1 erweitert die PtP-Fähigkeit zur Nutzung eines dezentralen PROFINET- oder PROFIBUS-Peripheriebaugruppenträgers für die Kommunikation mit verschiedenen Geräten (RFID-Lesegeräte, GPS und andere):

- PROFINET (Seite 636): Die Ethernet-Schnittstelle der S7-1200 CPU wird mit einem PROFINET Schnittstellenmodul verbunden. Über PtP-Kommunikationsmodule im Baugruppenträger mit dem Schnittstellenmodul ist dann die serielle Kommunikation zu den PtP-Geräten möglich.
- PROFIBUS (Seite 782): Ein PROFIBUS-Kommunikationsmodul wird an der linken Seite des Baugruppenträgers mit der S7-1200 CPU eingesteckt. Das PROFIBUS-Kommunikationsmodul wird mit einem Baugruppenträger verbunden, der ein PROFIBUS-Schnittstellenmodul enthält. Über PtP-Kommunikationsmodule im Baugruppenträger mit dem Schnittstellenmodul ist dann die serielle Kommunikation zu den PtP-Geräten möglich.

Die S7-1200 unterstützt daher zwei Sätze PtP-Anweisungen:

- Ältere Punkt-zu-Punkt-Anweisungen (Seite 1022): Diese Anweisungen wurden vor der Version V4.0 der S7-1200 benutzt und ermöglichen nur die serielle Kommunikation mit Hilfe eines Kommunikationsmoduls CM 1241 oder eines Kommunikationsboards CB 1241.
- Punkt-zu-Punkt-Anweisungen (Seite 915): Diese Anweisungen stellen alle Funktionen der alten Anweisungen bereit und zusätzlich die Möglichkeit zur Verbindung mit der dezentralen PROFINET- und PROFIBUS-Peripherie. Die PtP-Anweisungen ermöglichen das Konfigurieren der Kommunikation zwischen den PtP-Kommunikationsmodulen im Baugruppenträger mit der dezentralen Peripherie und den PtP-Geräten.

Hinweis

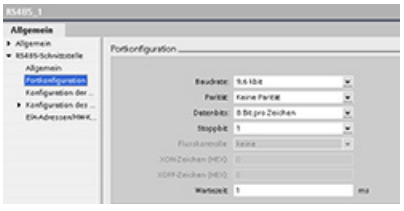
Ab der Version V4.1 der S7-1200 können PtP-Anweisungen für alle Arten der PtP-Kommunikation verwendet werden: seriell, seriell über PROFINET und seriell über PROFIBUS. STEP 7 stellt die alten PtP-Anweisungen nur zur Unterstützung vorhandener Programme bereit. Die alten Anweisungen funktionieren jedoch noch mit V4.1-CPUs sowie mit CPU-Versionen 4.0 und früher. Es ist nicht nötig, ältere Programme auf die neuen Anweisungen zu konvertieren.

13.3.1 Kommunikationsanschlüsse konfigurieren

Zur Konfiguration der Kommunikationsschnittstellen sind die folgenden Vorgehensweisen möglich:

- Konfigurieren Sie in der Gerätekonfiguration in STEP 7 die Portparameter (Baudrate und Parität), die Sendeparameter und die Empfangsparameter. Die CPU speichert die Einstellungen in der Gerätekonfiguration und übernimmt die Einstellungen nach einem Neustart und einem Wechsel von RUN zu STOP.
- Richten Sie die Parameter über die Anweisungen Port_Config (Seite 918), Send_Config (Seite 920) und Receive_Config (Seite 923) ein. Die von den Anweisungen festgelegten Porteinstellungen sind gültig, wenn sich die CPU in der Betriebsart RUN befindet. Die Porteinstellungen kehren nach einem Wechsel in STOP oder nach einem Neustart zu den Einstellungen der Gerätekonfiguration zurück.

Nach der Konfiguration der Hardwaregeräte (Seite 151) parametrieren Sie die Kommunikationsschnittstellen durch Auswahl eines der CMs in Ihrem Baugruppenträger bzw. des CBs, sofern konfiguriert.



Im Inspektorfenster werden im Register "Eigenschaften" die Parameter des ausgewählten CMs oder CB angezeigt. Wählen Sie "Portkonfiguration", um die folgenden Parameter zu bearbeiten:

- Baudrate
- Parität
- Datenbits pro Zeichen
- Anzahl Stoppbits
- Flusskontrolle (nur RS232)
- Wartezeit

Beim CM 1241 RS232 und beim CB RS485 (außer für die Flusskontrolle (Seite 904), die nur vom CM 1241 RS232 unterstützt wird) haben die RS232- und RS485-Kommunikationsmodule und das RS485-Kommunikationsboard die gleichen Parameter für die Portkonfiguration. Die Parameterwerte können verschieden sein.

Beim CM 1241 RS422/485 stehen zusätzliche Optionen für die Portkonfiguration, wie unten dargestellt, zur Verfügung. Der 422-Modus des CM 1241 RS422/485 unterstützt auch die Software-Flusskontrolle.

Wählen Sie "Portkonfiguration", um die folgenden RS422/485-Parameter zu bearbeiten:

- "Betriebsart":
  - Vollduplex (RS422), Vierdrahtmodus (Punkt-zu-Punkt-Verbindung)
  - Vollduplex (RS422), Vierdrahtmodus (Multipoint-Master)
  - Vollduplex (RS422), Vierdrahtmodus (Multipoint-Slave)
  - Halbduplex (RS485), Zweidrahtmodus
- "Ausgangszustand Empfangsleitung":
  - Ohne
  - Vorspannung in Vorwärtsrichtung (Signal R(A) 0 V, Signal R(B) 5 V)

Das STEP 7 Anwenderprogramm kann auch den Port konfigurieren oder die vorhandene Konfiguration mit der Anweisung Port_Config (Seite 918) ändern. Die Beschreibung der Anweisung enthält genauere Angaben zu Betriebsart und Ausgangszustand der Leitung sowie weiteren Parametern.

Parameter	Definition
Baudrate	Die Voreinstellung für die Baudrate ist 9,6 kBit/s. Gültige Werte sind: 300 Baud, 600 Baud, 1,2 kBit/s, 2,4 kBit/s, 4,8 kBit/s, 9,6 kBit/s, 19,2 kBit/s, 38,4 kBit/s, 57,6 kBit/s, 76,8 kBit/s und 115,2 kBit/s.
Parität	Die Voreinstellung für die Parität ist "Keine". Gültige Werte sind: Keine Parität, gerade, ungerade, Mark (Paritätsbit immer auf 1) und Space (Paritätsbit immer auf 0).
Datenbits pro Zeichen	Die Anzahl der Datenbits in einem Zeichen. Gültig sind die Einstellungen 7 oder 8.
Anzahl Stoppbits	Die Anzahl der Stoppbits kann eins oder zwei sein. Die Voreinstellung ist eins.
Flusskontrolle	Für das RS232-Kommunikationsmodul kann die Hardware- oder Software-Flusskontrolle (Seite 904) gewählt werden. Ist die Hardware-Flusskontrolle eingestellt, so können Sie wählen, ob das RTS-Signal immer ein ist oder ob RTS geschaltet wird. Ist die Software-Flusskontrolle eingestellt, können Sie die Zeichen XON und XOFF definieren. Die RS485-Kommunikationsschnittstellen unterstützen keine Flusskontrolle. Der 422-Modus des CM 1241 RS422/485 unterstützt die Software-Flusskontrolle.
Wartezeit	Die Wartezeit ist die Zeit, während der das CM oder CB nach dem Absetzen von RTS auf den Empfang von CTS oder nach dem Empfang von XOFF auf den Empfang von XON, je nach Art der Flusskontrolle, wartet. Ist die Wartezeit beendet, bevor die Kommunikationsschnittstelle ein erwartetes CTS oder XON empfängt, so wird die Sendung durch das CM oder CB abgebrochen und eine Fehlermeldung an das Anwenderprogramm ausgegeben. Geben Sie die Wartezeit in Millisekunden ein. Der zulässige Wertebereich ist 0 bis 65535 Millisekunden.
Betriebszustand	Hier wählen Sie den Betriebszustand RS422 oder RS485 und die Netzwerkkonfigurationen aus.
Ausgangszustand Empfangsleitung	Hier wählen Sie die Vorspannung aus. Gültig sind die Werte Ohne, Vorspannung in Vorwärtsrichtung und Vorspannung in Sperrrichtung. Vorspannung in Sperrrichtung ermöglicht die Drahtbrucherkennung.

13.3.1.1 Steuerung der Flusskontrolle

Die Flusskontrolle sorgt für den Ausgleich zwischen unterschiedlichen Kapazitäten von Absender und Empfänger der Daten, so dass keine Daten verlorengehen. Mit der Flusskontrolle ist gewährleistet, dass der Absender nicht mehr Informationen überträgt als der Empfänger verarbeiten kann. Die Flusskontrolle ist auf der Hardware- oder der Softwareebene möglich. Das RS232-Kommunikationsmodul unterstützt die Hardware- und Software-Flusskontrolle. Die RS485-CM und -CB unterstützen keine Flusskontrolle. Der 422-Modus des CM 1241 RS422/485 unterstützt die Software-Flusskontrolle. Welche Flusskontrolle eingerichtet werden soll, geben Sie beim Konfigurieren der Schnittstelle (Seite 902) oder über die Anweisung PORT_CFG (Seite 1022) ein.

Die Hardware-Flusskontrolle beruht auf den Kommunikationssignalen RTS (Request-to-send) und CTS (Clear-to-send). Beim RS232-Kommunikationsmodul wird das RTS-Signal an Pin 7 und das CTS-Signal an Pin 8 abgenommen. Das RS232-Kommunikationsmodul ist ein DTE-Gerät (Data Terminal Equipment, Datenendgerät), das RTS als Ausgang sicherstellt und CTS als Eingang überwacht.

Hardware-Flusskontrolle: RTS-geschaltet

Ist die RTS-geschaltete Hardware-Flusskontrolle für ein RS232-Kommunikationsmodul aktiviert, setzt das Modul das RTS-Signal auf 1, damit Daten gesendet werden. Es überwacht das CTS-Signal und erkennt damit, ob das Empfangsgerät empfangsbereit ist. Ist das CTS-Signal aktiv, kann das Modul Daten übertragen, solange das CTS-Signal aktiv bleibt. Wird das CTS-Signal inaktiv, muss die Übertragung angehalten werden.

Sie wird fortgesetzt, wenn das CTS-Signal erneut eingeschaltet wird. Wird das CTS-Signal nicht innerhalb der eingestellten Wartezeit erneut eingeschaltet, bricht das Modul die Übertragung ab und gibt einen Fehler an das Anwenderprogramm zurück. Die Wartezeit müssen Sie in der Schnittstellenkonfiguration (Seite 902) eingeben.

Die RTS-geschaltete Flusskontrolle ist bei Geräten nützlich, die ein Signal benötigen, dass der Sendevorgang aktiv ist. Ein Beispiel hierfür ist ein Funkmodem, bei dem RTS als "Schlüssel"signal den Funksender aktiviert. Die RTS-geschaltete Flusskontrolle funktioniert nicht bei herkömmlichen Telefonmodems. Verwenden Sie für Telefonmodems die Option "RTS immer ein".

Hardware-Flusskontrolle: RTS immer ein

Bei der Option "RTS immer ein" aktiviert das CM 1241 das RTS-Signal standardmäßig. Ein Gerät, wie z. B. ein Telefonmodem, überwacht das RTS-Signal vom CM und nutzt dieses Signal als Mitteilung der Sendebereitschaft. Das Modem sendet nur dann an das CM, wenn RTS aktiv ist, d. h. wenn das Telefonmodem ein aktives CTS erkennt. Ist RTS inaktiv, überträgt das Telefonmodem keine Daten zum CM.

Damit das Modem jederzeit Daten an das CM senden kann, ist für die Hardware-Flusskontrolle "RTS immer ein" einzustellen. Das CM setzt das RTS-Signal dann immer auf 1. Das CM schaltet RTS auch dann nicht aus, wenn das Modul keine Zeichen empfangen kannn. Das Sendegerät muss gewährleisten, dass es keinen Überlauf im Empfangspuffer des CMs verursacht.

Signale Data Terminal Ready (DTR) und Data Set Ready (DSR)

Das CM setzt DTR für jede Art der Hardware-Flusskontrolle auf 1. Es überträgt erst, wenn das Signal DSR eingeschaltet wird. Der Zustand von DSR wird nur am Anfang der Sendefunktion ausgewertet. Wenn DSR nach Beginn der Übertragung inaktiv wird, wird die Übertragung nicht angehalten.

Software-Flusskontrolle

Bei der Software-Flusskontrolle werden spezielle Zeichen in den Meldungen für die Flusskontrolle verwendet. Sie können Hexadezimalzeichen konfigurieren, die XON und XOFF darstellen.

XOFF zeigt an, dass eine Übertragung angehalten werden muss. XON zeigt an, dass eine Übertragung fortgesetzt werden kann. XOFF und XON dürfen nicht das gleiche Zeichen sein.

Empfängt das Sendegerät ein Zeichen XOFF vom Empfänger, wird die Übertragung angehalten. Sie wird fortgesetzt, wenn das Sendegerät das Zeichen XON empfängt. Empfängt das CM innerhalb der in der Schnittstellenkonfiguration (Seite 902) angegebenen Wartezeit kein XON-Zeichen, bricht es die Übertragung ab und gibt eine Fehlermeldung an das Anwenderprogramm aus.

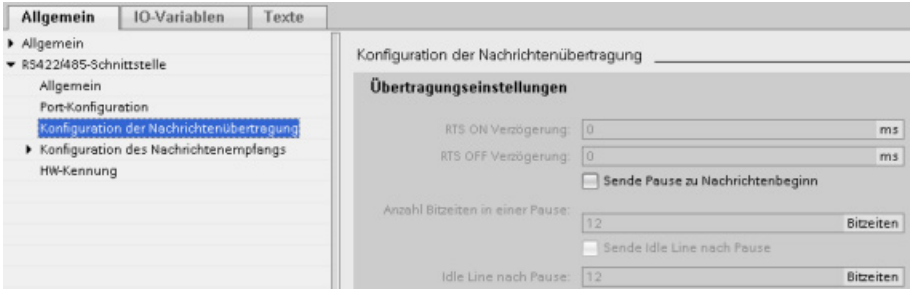
Die Software-Flusskontrolle setzt die Vollduplex-Kommunikation voraus, da der Empfänger in der Lage sein muss, während einer laufenden Übertragung XOFF an den Sender zu senden. Die Software-Flusskontrolle ist nur bei Meldungen möglich, die nur aus ASCII-Zeichen bestehen. Binäre Protokolle können keine Software-Flusskontrolle nutzen.

13.3.2 Sende- und Empfangsparameter konfigurieren

Damit die Punkt-zu-Punkt-Kommunikation aufgenommen werden kann, müssen die Parameter zum Senden und Empfangen von Meldungen eingerichtet werden. Diese Parameter legen fest, wie die Kommunikation abläuft, wenn Meldungen von einem Zielgerät gesendet oder von diesem empfangen werden.

13.3.2.1 Sendeparameter konfigurieren

In der Gerätekonfiguration der CPU richten Sie ein, wie eine Kommunikationsschnittstelle Daten sendet. Hierfür geben Sie die Eigenschaften "Konfiguration Sendemeldung" für die ausgewählte Schnittstelle an.



Sie können ferner die Parameter für das Senden einer Meldung im Anwenderprogramm mit der Anweisung Send_Config (Seite 920) dynamisch konfigurieren oder ändern.

Hinweis

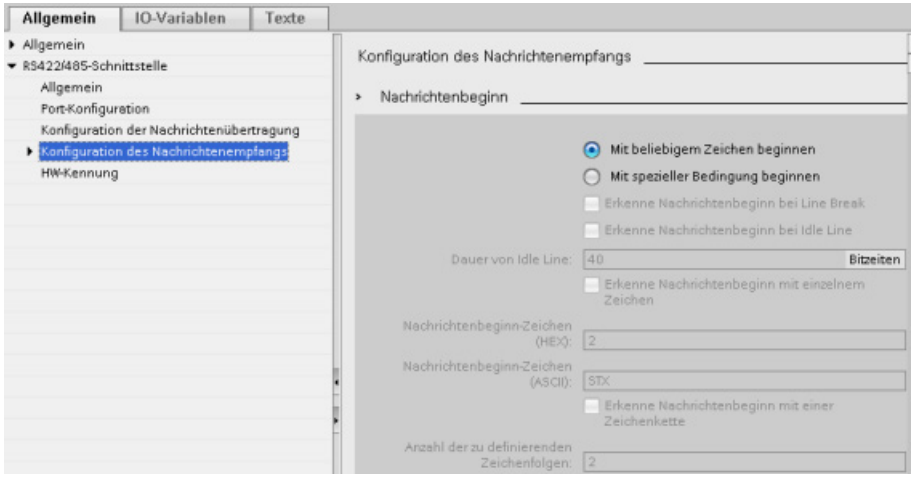
Parameterwerte, die mit Send_Config im Anwenderprogramm eingegeben wurden, überschreiben die Eigenschaften "Konfiguration Sendemeldung". Parameter, die mit der Anweisung Send_Config eingegeben wurden, werden bei einer Abschaltung nicht in der CPU gespeichert.

Parameter	Definition
RTS-Einschaltverzögerung	Wartezeit nach dem Aktivieren von RTS, bis die Sendung ausgelöst wird. Der gültige Bereich ist 0 bis 65535 ms, voreingestellt ist 0. Dieser Parameter ist nur gültig, wenn in der Schnittstellenkonfiguration (Seite 902) die Hardware-Flusskontrolle eingestellt ist. CTS wird nach dem Ablauf der RTS-Einschaltverzögerung bearbeitet. Dieser Parameter gilt nur für RS232-Module.
RTS-Ausschaltverzögerung	Wartezeit nach der beendeten Übertragung, bis RTS deaktiviert wird. Der gültige Bereich ist 0 bis 65535 ms, voreingestellt ist 0. Dieser Parameter ist nur gültig, wenn in der Schnittstellenkonfiguration (Seite 902) die Hardware-Flusskontrolle eingestellt ist. Dieser Parameter gilt nur für RS232-Module.

Parameter	Definition
Sendepause bei Meldungsbeginn Anzahl der Bittakte in einer Pause	Damit wird festgelegt, dass beim Start einer Meldung eine Pause gesendet wird, wenn die RTS -inschaltverzögerung (sofern eingerichtet) abgelaufen ist und CTS aktiv ist. Wie viele Bittakte eine Pause darstellen, während der die Übertragung angehalten ist, können Sie einstellen. Die Voreinstellung ist 12 und das Maximum ist 65535 bis zu einem Grenzwert von acht Sekunden.
Leerzeile nach Pause senden Leerzeile nach Pause	Gibt an, dass eine Leerzeile vor Meldungsbeginn gesendet wird. Sie wird nach der Pause gesendet, sofern eine Pause konfiguriert ist. Der Parameter "Leerzeile nach Pause" gibt an, wie viele Bittakte einer Leerzeile entsprechen. Die Voreinstellung ist 12 und das Maximum ist 65535 bis zu einem Grenzwert von acht Sekunden.

13.3.2.2 Empfangsparameter konfigurieren

In der Gerätekonfiguration der CPU legen Sie fest, wie Daten über eine Kommunikationsschnittstelle empfangen und wie Meldungsbeginn und Meldungsende erkannt werden sollen. Sie geben diese Parameter in den Eigenschaften "Konfiguration Empfangsmeldung" für die ausgewählte Schnittstelle an.



Sie können ferner die Meldungsparameter im Anwenderprogramm mit der Anweisung Receive_Config (Seite 923) dynamisch konfigurieren oder ändern.

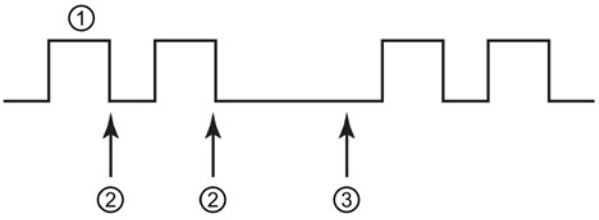
Hinweis

Parameterwerte, die mit Receive_Config im Anwenderprogramm eingegeben wurden, überschreiben die Eigenschaften "Konfiguration Empfangsmeldung". Parameter, die mit der Anweisung RCV_CFG eingegeben wurden, werden bei einer Abschaltung oder einem Wechsel in STOP nicht in der CPU gespeichert.

Bedingungen für den Meldungsbeginn

Sie können festlegen, wie die Kommunikationsschnittstelle den Meldungsbeginn erkennen soll. Startzeichen und Meldungsinhalt werden im Empfangspuffer abgelegt, bis eine konfigurierte Endebedingung erfüllt ist.

Sie können mehrere Startbedingungen angeben. Wenn Sie mehrere Startbedingungen angeben, müssen alle Startbedingungen erfüllt sein, damit die Meldung als gestartet betrachtet wird. Wenn Sie z. B. eine Zeit für den Leitungsleerlauf und ein bestimmtes Startzeichen konfigurieren, wartet das CM bzw. CB zunächst auf die Erfüllung der Zeit für den Leitungsleerlauf und sucht dann nach dem spezifischen Startzeichen. Wenn ein anderes Zeichen empfangen wird (nicht das angegebene Startzeichen), startet das CM bzw. CB die Suche nach dem Meldungsstart erneut, indem es auf die Zeit für den Leitungsleerlauf wartet.

Parameter	Definition
Start mit beliebigem Zeichen	Die Bedingung "Beliebiges Zeichen" gibt an, dass durch erfolgreichen Empfang eines beliebigen Zeichens ein Meldungsbeginn angezeigt wird. Dieses Zeichen ist das erste Zeichen in einer Meldung.
Leitungspause	Die Bedingung "Leitungspause" gibt an, dass eine Operation zum Meldungsempfang beginnt, nachdem ein Pausezeichen empfangen wurde.
Leitung im Leerlauf	Die Bedingung "Leitung im Leerlauf" gibt an, dass der Meldungsempfang beginnt, wenn die Empfangsleitung für die Anzahl der angegebenen Bitzeiten im Leerlauf war. Nachdem diese Bedingung aufgetreten ist, beginnt der Empfang einer Meldung.  ① Zeichen ② Startet die Zeit für den Leitungsleerlauf neu ③ Leitungsleerlauf wird erkannt und Meldungsempfang gestartet
Sonderbedingung: Meldungsbeginn durch einzelnes Zeichen erkennen	Gibt an, dass ein bestimmtes Zeichen den Beginn einer Meldung anzeigt. Dieses Zeichen ist dann das erste Zeichen in einer Meldung. Jedes Zeichen, das vor diesem bestimmten Zeichen empfangen wird, wird ignoriert. Das Standardzeichen ist STX.
Sonderbedingung: Meldungsbeginn durch Zeichenfolge erkennen	Gibt an, dass eine bestimmte Zeichenfolge aus bis zu vier konfigurierten Folgen den Beginn einer Meldung anzeigt. Sie können bis zu fünf Zeichen für jede Zeichenfolge eingeben. Für jede Zeichenposition kann ein bestimmtes Hex-Zeichen oder das Ignorieren des Zeichens in der Zeichenfolge (Platzhalterzeichen) eingegeben werden. Das letzte spezifische Zeichen einer Zeichenfolge beendet die Startbedingungsfolge. Eingehende Zeichenfolgen werden mit den konfigurierten Startbedingungen verglichen, bis eine Startbedingung erfüllt ist. Ist die Startbedingung erfüllt, beginnt die Überwachung der Endebedingung. Sie können bis zu vier spezifische Zeichenfolgen konfigurieren. Sie verwenden eine Startbedingung mit mehreren Folgen, wenn verschiedene Zeichenfolgen den Beginn einer Meldung anzeigen können. Die Meldung beginnt, wenn eine der eingerichteten Zeichenfolgen erfüllt ist.

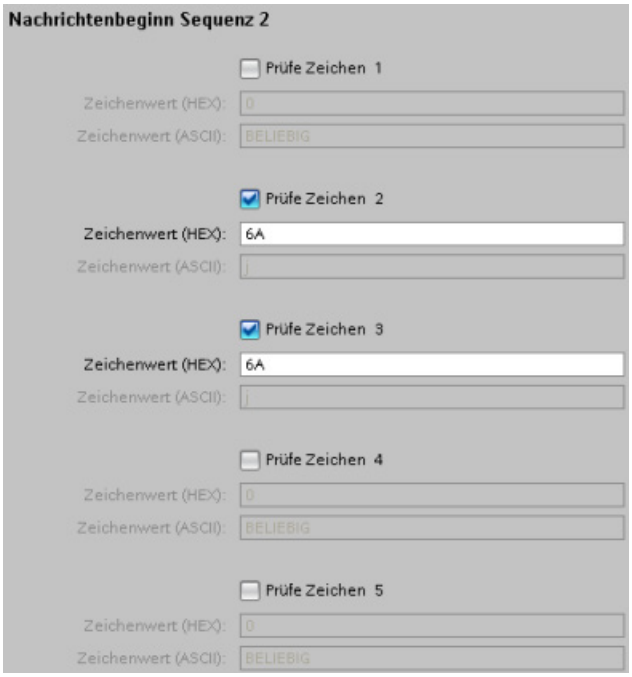
Die Startbedingungen werden in der folgenden Reihenfolge geprüft:

- Leerzeile
- Zeilenumbruch
- Zeichen oder Zeichenfolgen

Wenn mehrere Startbedingungen geprüft werden und eine der Bedingungen nicht erfüllt wird, beginnt das CM bzw. CB die Prüfung erneut mit der ersten geforderten Bedingung. Nachdem das CM bzw. CB ermittelt hat, dass die Startbedingungen erfüllt wurden, beginnt es mit dem Auswerten der Endebedingungen.

Beispielkonfiguration: Meldungsbeginn bei einer von zwei Zeichenfolgen

Beachten Sie die folgende Konfiguration der Bedingungen für den Meldungsbeginn:



Bei dieser Konfiguration ist die Startbedingung erfüllt, wenn eines der folgenden Bitmuster vorliegt:

- Empfang einer Zeichenfolge mit 5 Zeichen, bei der das erste Zeichen 0x6A und das fünfte Zeichen 0x1C ist. Die Zeichen 2, 3 und 4 können hierbei beliebig sein. Nach dem Empfang des fünften Zeichens beginnt die Überwachung auf die Endebedingung.
- Empfang von zwei aufeinanderfolgenden Zeichen 0x6A nach einem beliebigen anderen Zeichen. In diesem Fall beginnt die Überwachung auf die Endebedingung nach dem Empfang der zweiten 0x6A-Folge (3 Zeichen). Das Zeichen vor dem ersten 0x6A ist in der Startbedingung enthalten.

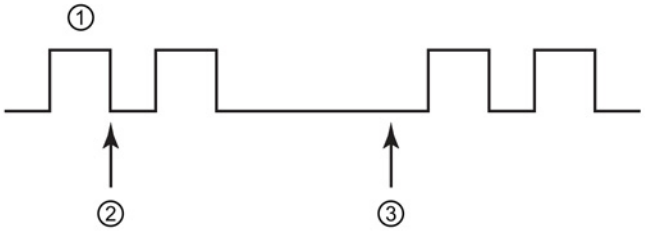
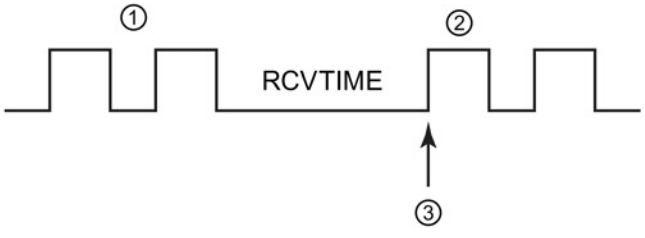
Beispiele für Zeichenfolgen, die diese Startbedingung erfüllen:

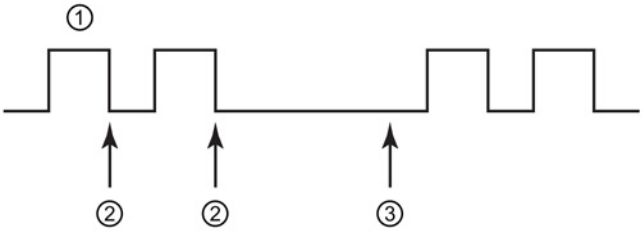
- <beliebiges Zeichen> 6A 6A
- 6A 12 14 18 1C
- 6A 44 A5 D2 1C

Bedingungen für das Meldungsende

Sie legen ferner fest, wie die Kommunikationsschnittstelle das Meldungsende erkennen soll. Sie können mehrere Endebedingungen für Meldungen einrichten. Die Meldung endet, sobald eine der eingerichteten Bedingungen erfüllt ist.

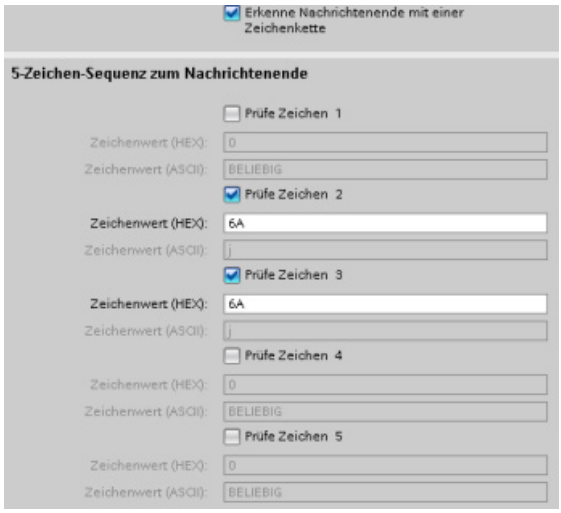
Sie können z. B. eine Endebedingung mit einem Meldungs-Timeout von 300 ms, einem Zeichenabstands-Timeout von 40 Bitzeiten und eine maximale Länge von 50 Bytes angeben. Die Meldung endet, wenn der Meldungsempfang länger als 300 ms dauert oder wenn der Abstand zwischen zwei beliebigen Zeichen 40 Bitzeiten überschreitet oder wenn 50 Bytes empfangen wurden.

Parameter	Definition
Meldungsende erkennen durch Meldungs-Timeout	Das Meldungsende ist erreicht, wenn die konfigurierte Wartezeit auf das Meldungsende abgelaufen ist. Die Timeout-Zeit der Meldung beginnt, wenn eine Startbedingung erfüllt wurde. Die Voreinstellung ist 200 ms, der Wertebereich ist 0 bis 65535 ms.  ① Empfangene Zeichen ②Bedingung für den Meldungsbeginn erfüllt: Meldungs-Timer beginnt zu laufen ③ Meldungs-Timer läuft ab und beendet die Meldung
Meldungsende erkennen durch Antwort-Timeout	Das Meldungsende ist erreicht, wenn die konfigurierte Wartezeit auf eine Antwort erreicht ist, bevor eine gültige Startzeichenfolge empfangen wurde. Die Timeout-Zeit für die Antwort beginnt, wenn eine Übertragung endet und das CM bzw. CB die Empfangsanweisung startet. Die Voreinstellung für die Timeout-Zeit der Antwort beträgt 200 ms, der Wertebereich ist 0 bis 65535 ms. Wenn nicht innerhalb des von RCVTIME angegebenen Antwortzeitraums ein Zeichen empfangen wird, wird der entsprechenden Anweisung RCV_PTP ein Fehler gemeldet. Das Antwort-Timeout definiert keine spezifische Endebedingung. Das Timeout gibt nur an, dass ein Zeichen innerhalb der angegebenen Zeit erfolgreich empfangen werden muss. Sie müssen eine andere Endebedingung konfigurieren, um das tatsächliche Ende einer Meldung anzugeben.  ① Gesendete Zeichen ② Empfangene Zeichen ③ Erstes Zeichen muss zu diesem Zeitpunkt erfolgreich empfangen worden sein.

Parameter	Definition
Meldungsende durch Zeichenabstand erkennen	Das Meldungsende ist erreicht, wenn die konfigurierte maximale Zeit zwischen zwei beliebigen, aufeinanderfolgenden Zeichen einer Meldung abgelaufen ist. Die Voreinstellung für den Zeichenabstand ist 12 Bittakte und die maximale Zahl ist 65535 Bittakte, bis maximal acht Sekunden.  ① Empfangene Zeichen ② Startet den Timer für die Zeit zwischen den Zeichen neu ③ Der Timer für die Zeit zwischen den Zeichen läuft ab und beendet die Meldung.
Meldungsende durch Empfang einer festen Anzahl von Zeichen erkennen	Das Meldungsende ist erreicht, wenn die konfigurierte Anzahl von Zeichen empfangen wurde. Der gültige Bereich für die feste Länge liegt zwischen 1 und 4096. Beachten Sie, dass diese Bedingung für das Meldungsende bei der S7-1200 erst ab CPUs der Version V4.0 gültig ist.
Meldungsende durch maximale Meldungslänge erkennen	Das Meldungsende ist erreicht, wenn die konfigurierte maximale Zeichenzahl empfangen wurde. Der gültige Bereich für die maximale Länge liegt zwischen 1 und 1023. Diese Bedingung kann dafür genutzt werden, den Fehler "Überlauf Meldungspuffer" zu verhindern. Wird diese Endebedingung mit Timeout-Endebedingungen verknüpft und tritt die Timeout-Endebedingung auf, werden alle gültigen empfangenen Zeichen ausgegeben, auch wenn die maximale Länge nicht erreicht wurde. So können Protokolle unterschiedlicher Länge unterstützt werden, auch wenn nur die maximale Länge bekannt ist.
Meldungslänge aus Meldung lesen	Die Meldungslänge ist in der Meldung selbst angegeben. Das Meldungsende ist erreicht, wenn eine Meldung mit der angegebenen Länge empfangen wurde. Wie die Meldungslänge angegeben und ausgewertet wird, ist nachstehend beschrieben.
Meldungsende mit einem Zeichen erkennen	Das Meldungsende ist erreicht, wenn ein bestimmtes Zeichen empfangen wird.
Meldungsende mit einer Zeichenfolge erkennen	Das Meldungsende ist erreicht, wenn eine bestimmte Zeichenfolge empfangen wird. Sie können eine Zeichenfolge mit bis zu 5 Zeichen angeben. Für jede Zeichenposition kann ein bestimmtes Hex-Zeichen oder dass das Zeichen in der Zeichenfolge ignoriert werden soll eingegeben werden. Führende Zeichen, die ignoriert werden, gehören nicht zur Endebedingung. Nachgestellte Zeichen, die ignoriert werden, gehören zur Endebedingung.

Beispielkonfiguration: Meldung mit einer Zeichenfolge beenden

Beachten Sie die folgende Konfiguration der Bedingungen für das Meldungsende:



Die Endebedingung ist in diesem Fall erfüllt, wenn zwei aufeinanderfolgende Zeichen 0x6A, gefolgt von zwei beliebigen Zeichen, empfangen werden. Das Zeichen vor 0x6A 0x6A gehört nicht zur Endezeichenfolge. Nach 0x6A 0x6A sind zwei Zeichen erforderlich, um die Endezeichenfolge abzuschließen. Die von Zeichen 4 und 5 empfangenen Werte sind irrelevant, sie müssen jedoch empfangen werden, damit die Endebedingung erfüllt ist.

Hinweis

Soll Ihre Zeichenfolge das Ende der Meldung anzeigen, setzen Sie sie an die letzte Zeichenposition. Im Beispiel oben würde 0x6A an den Zeichenpositionen 4 und 5 konfiguriert werden, wenn 0x6A 0x6A die Meldung ohne nachgestellte Zeichen beenden soll.

Angabe der Meldungslänge in der Meldung

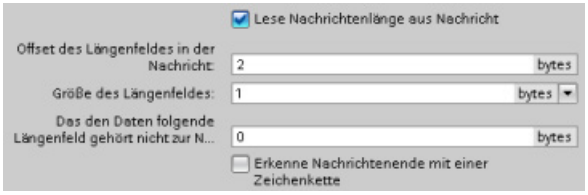
Bei der Auswahl der Sonderbedingung, bei der die Meldungslänge in der Meldung enthalten ist, müssen drei Parameter angegeben werden, die Informationen über die Meldungslänge enthalten.

Die tatsächliche Meldungsstruktur hängt vom verwendeten Protokoll ab. Die drei Parameter sind:

- n: Zeichenposition (Basis 1) innerhalb der Meldung, an der die Längenangabe beginnt
- Längenwert: Anzahl der Bytes (eins, zwei oder vier) der Längenangabe
- Länge m: Anzahl der Zeichen nach der Längenangabe, die nicht in der Längenzählung berücksichtigt werden

Die Endezeichen müssen nicht aufeinander folgen. Der Wert "Länge m" kann genutzt werden, um die Länge eines Prüfsummenfelds anzugeben, dessen Größe nicht im Längenfeld enthalten ist.

Diese Felder erscheinen in der Konfiguration des Meldungsempfangs in den Geräteeigenschaften:



Beispiel 1: Es wird eine Meldung betrachtet, die nach dem folgenden Protokoll strukturiert ist:

STX	Len (n)	Zeichen 3 bis 14, nach Länge gezählt											
		ADR	PKE		INDEX		PWD		STW		HSW		BCC
1	2	3	4	5	6	7	8	9	10	11	12	13	14
STX	0x0C	xx	xxxx		xxxx		xxxx		xxxx		xxxx		xx

Die Parameter für die Länge des Meldungsempfangs konfigurieren Sie wie folgt:

- n = 2 (Meldungslänge startet in Byte 2.)
- Längenwert = 1 (Meldungslänge ist in einem Byte enthalten.)
- Länge m = 0 (Auf die Längenangabe folgen keine weiteren Zeichen, die nicht in der Längenzählung berücksichtigt werden. Nach der Längenangabe folgen zwölf Zeichen.)

In diesem Beispiel werden die Zeichen von 3 bis einschließlich 14 von Len (n) gezählt.

Beispiel 2: Es wird eine weitere Meldung betrachtet, die nach dem folgenden Protokoll strukturiert ist:

SD1	Len (n)	Len (n)	SD2	Zeichen 5 bis 10, nach Länge gezählt						FCS	ED
				DA	SA	FA	Dateneinheit=3 Bytes				
1	2	3	4	5	6	7	8	9	10	11	12
xx	0x06	0x06	xx	xx	xx	xx	xx	xx	xx	xx	xx

Die Parameter für die Länge des Meldungsempfangs konfigurieren Sie wie folgt:

- n = 3 (Meldungslänge startet in Byte 3.)
- Längenwert = 1 (Meldungslänge ist in einem Byte enthalten.)
- Länge m = 3 (Auf die Längenangabe folgen drei Zeichen, die nicht in der Längenzählung berücksichtigt werden. Im Protokoll dieses Beispiels werden Zeichen SD2, FCS und ED bei der Längenzählung nicht beachtet. Die sechs anderen Zeichen werden in der Längenzählung berücksichtigt; die Gesamtzahl der Zeichen nach der Längenangabe ist somit neun.)

In diesem Beispiel werden die Zeichen von 5 bis einschließlich 10 von Len (n) gezählt.

13.3.3 Punkt-zu-Punkt-Anweisungen

13.3.3.1 Gemeinsame Parameter für Punkt-zu-Punkt-Operationen

Tabelle 13- 3 Allgemeine Eingangsparameter der PTP-Anweisungen

Parameter	Beschreibung
REQ	Viele der PtP-Anweisungen haben einen Eingang REQ, der die Anweisung bei einer steigenden Flanke (0 nach 1) initiiert. Der Eingang REQ muss während der Ausführung einer Anweisung 1 (WAHR) sein, doch der Eingang REQ kann so lange wie gewünscht WAHR bleiben. Die Anweisung stößt erst dann eine andere Anweisung an, wenn sie mit Eingang REQ = FALSCH aufgerufen wird, so dass die Anweisung den Zustand des Eingangs REQ zurücksetzen kann. Dies ist erforderlich, damit die Anweisung die steigende Flanke zum Starten der nächsten Anweisung erkennen kann. Wenn Sie eine PtP-Anweisung in Ihr Programm einfügen, werden Sie von STEP 7 aufgefordert, den Instanz-DB anzugeben. Verwenden Sie für jeden PtP-Anweisungsaufruf einen eindeutigen DB. Dadurch wird sichergestellt, dass jede Anweisung die Eingänge wie REQ ordnungsgemäß verarbeitet.
PORT	Eine Portadresse wird während der Konfiguration des Kommunikationsgeräts zugewiesen. Nach der Konfiguration kann aus der Klappliste des Parameters ein symbolischer Name für den Standardport ausgewählt werden. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Konstanten" der PLC-Variablentabelle zugewiesen.
Auflösung von Bitzeiten	Für verschiedene Parameter wird die Anzahl der Bitzeiten bei der konfigurierten Baudrate angegeben. Durch Angabe des Parameters in Bitzeiten ist der Parameter unabhängig von der Baudrate. Alle Parameter mit Einheiten von Bitzeiten können mit einer maximalen Anzahl von 65535 angegeben werden. Der maximale Zeitraum jedoch, der von einem CM oder CB gemessen werden kann, beträgt 8 Sekunden.

Die Ausgangsparameter DONE, NDR, ERROR und STATUS der PtP-Anweisungen zeigen den Ausführungsstatus der PtP-Funktionen an.

Tabelle 13- 4 Ausgangsparameter DONE, NDR, ERROR und STATUS

Parameter	Datentyp	Standard	Beschreibung
DONE	Bool	FALSCH	Wird eine Ausführung lang auf WAHR gesetzt, um anzuzeigen, dass die letzte Anforderung mit Fehler abgeschlossen wurde; andernfalls FALSCH.
NDR	Bool	FALSCH	Wird eine Ausführung lang auf WAHR gesetzt, um anzuzeigen, dass die angeforderte Aktion fehlerfrei abgeschlossen und neue Daten empfangen wurden; andernfalls FALSCH.
ERROR	Bool	FALSCH	Wird eine Ausführung lang auf WAHR gesetzt, um anzuzeigen, dass die letzte Anforderung mit Fehlern abgeschlossen wurde, der entsprechende Fehlercode befindet sich in STATUS; andernfalls FALSCH.
STATUS	Word	0	Ergebniszustand: • Wird das Bit DONE oder NDR gesetzt, wird STATUS auf 0 oder auf einen Informationscode gesetzt. • Wird das Bit ERROR gesetzt, wird STATUS auf einen Fehlercode gesetzt. • Wird keines der oben aufgeführten Bits gesetzt, kann die Anweisung Statusergebnisse zurückgeben, die den aktuellen Zustand der Funktion beschreiben. Der Wert in STATUS wird während der Ausführung der Funktion gehalten.

Hinweis

Beachten Sie, dass die Parameter DONE, NDR und ERROR nur eine Ausführung lang gesetzt sind. Ihre Programmlogik muss temporäre Ausgangszustandswerte in Datenpuffern speichern, damit Sie in nachfolgenden Programmzyklen Zustandsänderungen erkennen können.

Tabelle 13- 5 Gemeinsame Bedingungscodes

STATUS (W#16#....)	Beschreibung
0000	Kein Fehler
7000	Funktion nicht besetzt
7001	Funktion ist beim ersten Aufruf aktiv.
7002	Funktion ist bei Folgeaufrufen aktiv (Abfragen nach dem ersten Aufruf).
8x3A	Unzulässiger Pointer in Parameter x
8070	Gesamter interner Speicher ausgelastet, zu viele Anweisungen gleichzeitig in Bearbeitung
8080	Portnummer ist unzulässig.
8081	Zeitüberschreitung, Modulfehler oder anderer interner Fehler
8082	Parametrierung fehlgeschlagen, weil Parametrierung im Hintergrund läuft.

STATUS (W#16#....)	Beschreibung
8083	Pufferüberlauf: Das CM oder CB hat eine Empfangsmeldung mit einer Länge, die größer ist als vom Längenparameter zugelassen, zurückgegeben.
8090	Interner Fehler: Falsche Meldungslänge, falsches Submodul oder unzulässige Meldung Wenden Sie sich an den Kunden-Support.
8091	Interner Fehler: Falsche Version in Parametrierungsmeldung Wenden Sie sich an den Kunden-Support.
8092	Interner Fehler: Falsche Datensatzlänge in Parametrierungsmeldung Wenden Sie sich an den Kunden-Support.

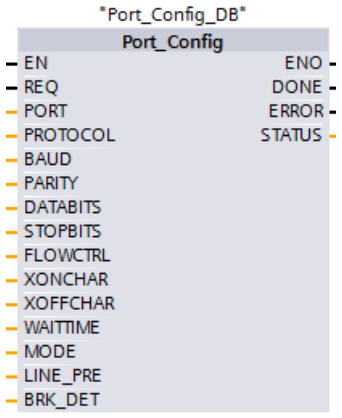
Tabelle 13- 6 Gemeinsame Fehlerklassen

Beschreibung der Klasse	Fehlerklassen	Beschreibung
Portkonfiguration	16#81Ax	Für die Beschreibung häufiger Fehler in der Schnittstellenkonfiguration
Sendekonfiguration	16#81Bx	Für die Beschreibung häufiger Fehler in der Sendekonfiguration
Empfangskonfiguration	16#81Cx 16#82Cx	Für die Beschreibung häufiger Fehler in der Empfangskonfiguration
Sende-Laufzeit	16#81Dx	Für die Beschreibung häufiger Laufzeitfehler beim Senden
Empfangs-Laufzeit	16#81Ex	Für die Beschreibung häufiger Laufzeitfehler beim Empfangen
Signalverarbeitung	16#81Fx	Für die Beschreibung häufiger Fehler in Verbindung mit der Signalverarbeitung
Pointer-Fehler	16#8p01 bis 16#8p51	Für ANY-Pointer-Fehler, wobei "p" die Parameternummer der Anweisung ist
Embedded-Protokoll-Fehler	16#848x 16#858x	Für Embedded-Protokoll-Fehler

13.3.3.2 Anweisung Port_Config (Kommunikationsparameter dynamisch konfigurieren)

Tabelle 13- 7 Anweisung Port_Config (Portkonfiguration)

KOP/FUP	SCL	Beschreibung

	`"Port_Config_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`PROTOCOL:=_uint_in_,`
	`BAUD:=_uint_in_,`
	`PARITY:=_uint_in_,`
	`DATABITS:=_uint_in_,`
	`STOPBITS:=_uint_in_,`
	`FLOWCTRL:=_uint_in_,`
	`XONCHAR:=_char_in_,`
	`XOFFCHAR:=_char_in_,`
	`WAITTIME:=_uint_in_,`
	`MODE:=_uint_in_,`
	`LINE_PRE:=_uint_in_,`
	`BRK_DET:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Mit Port_Config können Sie Portparameter wie die Baudrate über Ihr Programm ändern.
		Sie können die statische Anfangskonfiguration des Ports in den Eigenschaften der Gerätekonfiguration einrichten oder einfach die Standardwerte nutzen. Dann können Sie mit Port_Config in Ihrem Programm die Konfiguration ändern.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Die CPU speichert die von Ihnen in der Anweisung Port_Config festgelegten Werte nicht dauerhaft. Die in der Gerätekonfiguration gespeicherten Parameter werden von der CPU wiederhergestellt, wenn diese von RUN in STOP wechselt oder neu eingeschaltet wird. Weitere Informationen finden Sie unter Kommunikationsports konfigurieren (Seite 902) und Flusskontrolle verwalten (Seite 904).

Tabelle 13- 8 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert die Konfigurationsänderung bei steigender Flanke dieses Eingangs. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
PROTOCOL	IN	UInt	0 - Protokoll der Punkt-zu-Punkt-Kommunikation (Standardwert) 1..n - künftige Definition spezifischer Protokolle
BAUD	IN	UInt	Baudrate des Ports (Standardwert: 6): 1 = 300 Baud, 2 = 600 Baud, 3 = 1200 Baud, 4 = 2400 Baud, 5 = 4800 Baud, 6 = 9600 Baud, 7 = 19200 Baud, 8 = 38400 Baud, 9 = 57600 Baud, 10 = 76800 Baud, 11 = 115200 Baud
PARITY	IN	UInt	Parität des Ports (Standardwert: 1): 1 = Keine Parität, 2 = Gerade Parität, 3 = Ungerade Parität, 4 = Mark-Parität, 5 = Space-Parität

Parameter und Datentyp		Datentyp	Beschreibung
DATABITS	IN	UInt	Bits pro Zeichen (Standardwert: 1): 1 = 8 Datenbits, 2 = 7 Datenbits
STOPBITS	IN	UInt	Stoppbits (Standardwert: 1): 1 = 1 Stoppbit, 2 = 2 Stoppbits
FLOWCTRL	IN	UInt	Flusskontrolle (Standardwert: 1): 1 = Keine Flusskontrolle, 2 = XON/XOFF, 3 = Hardware-RTS immer EIN, 4 = Hardware-RTS geschaltet
XONCHAR	IN	Char	Gibt das Zeichen an, das als XON-Zeichen dient. Dies ist typischerweise ein DC1-Zeichen (16#11). Dieser Parameter wird nur ausgewertet, wenn die Flusskontrolle aktiviert ist. (Standardwert: 16#11)
XOFFCHAR	IN	Char	Gibt das Zeichen an, das als XOFF-Zeichen dient. Dies ist typischerweise ein DC3-Zeichen (16#13). Dieser Parameter wird nur ausgewertet, wenn die Flusskontrolle aktiviert ist. (Standardwert: 16#13)
WAITTIME	IN	UInt	Gibt an, wie lange nach dem Empfang eines XON-Zeichens auf ein XOFF-Zeichen gewartet werden soll bzw. wie lange nach Aktivierung von RTS auf das CTS-Signal gewartet werden soll (0 bis 65535 ms). Dieser Parameter wird nur ausgewertet, wenn die Flusskontrolle aktiviert ist. (Standardwert: 2000)
MODE	IN	UInt	Gibt die gewählte Betriebsart des Moduls an. • RS232-Betrieb (voreingestellt für RS232-CM oder -CB) • RS422 Point-to-Point, Sender immer aktiviert • RS422 Multi-Point Master, Sender immer aktiviert • RS422 Multi-Point Slave, Sender beim Senden aktiviert • RS485-Betrieb (Halbduplex, 2-Leiteranschluss) (voreingestellt für RS422/RS485-CM oder -CB)
LINE_PRE	IN	UInt	Gibt den inaktiven Leitungszustand (Leerlauf) an. Für RS422- und RS485-Module wird der inaktive Leitungszustand durch Anlegen einer Vorspannung an die Signale R(A) und R(B) hergestellt. Die folgenden Einstellungen sind möglich: • Keine Vorspannung (keine Voreinstellung) (Standard) • Vorspannung R(A) > R(B) ≥ 0V; nur RS422 • Vorspannung R(B) > R(A) ≥ 0V; RS422 und RS485
BRK_DET	IN	UInt	Aktiviert/deaktiviert Kommunikationsleitungsbrucherkennung. Bei aktivierter Leitungsbrucherkennung meldet das Modul einen Fehler, wenn das Kommunikationskabel nicht am Modul angeschlossen ist. Im RS422 Point-to-Point-Betrieb ist die Leitungsbrucherkennung nur möglich, wenn die Voreinstellung der Empfangsleitung mit einer Vorspannung verwendet wird, sodass R(A) > R(B) ≥ 0 V ist. • Keine Leitungsbrucherkennung (Standard) • Leitungsbrucherkennung aktiviert
DONE	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

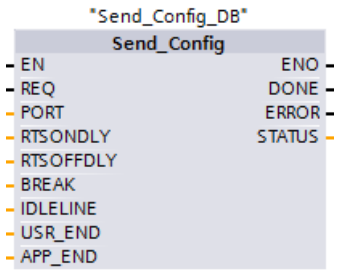
Tabelle 13- 9 Bedingungscodes

STATUS (W#16#....)	Beschreibung
81A0	Angegebenes Protokoll ist nicht vorhanden.
81A1	Angegebene Baudrate ist nicht vorhanden.
81A2	Angegebene Paritätsoption ist nicht vorhanden.
81A3	Angegebene Anzahl Datenbits ist nicht vorhanden.
81A4	Angegebene Anzahl Stoppbits ist nicht vorhanden.
80A5	Angegebene Art der Flusskontrolle ist nicht vorhanden.
81A6	Wartezeit ist 0 und Flusskontrolle ist aktiviert
81A7	XON und XOFF sind unzulässige Werte (z. B. der gleiche Wert)
81A8	Fehler im Baustein-Header (zum Beispiel falscher Bausteintyp oder falsche Bausteinlänge)
81A9	Rekonfiguration abgelehnt, weil ein Konfigurierungsvorgang läuft
81AA	Ungültiger RS422/RS485-Betrieb
81AB	Ungültige Voreinstellung der Empfangsleitung für Leitungsbrucherkennung
81AC	Ungültige RS232-Pausenverarbeitung
8280	Negative Quittierung beim Lesen aus Modul
8281	Negative Quittierung beim Schreiben in Modul
8282	DP-Slave oder Modul nicht verfügbar

13.3.3.3 Anweisung Send_Config (Serielle Übertragungsparameter dynamisch konfigurieren)

Tabelle 13- 10 Anweisung Send_Config (Sendekonfiguration)

KOP/FUP	SCL	Beschreibung

	`"Send_Config_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`RTSONDLY:=_uint_in_,`
	`RTSOFFDLY:=_uint_in_,`
	`BREAK:=_uint_in_,`
	`IDLELINE:=_uint_in_,`
	`USR_END:=_string_in_,`
	`APP_END:=_string_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Send_Config ermöglicht das dynamische Konfigurieren serieller Übertragungsparameter für eine PtP-Kommunikationsschnittstelle. In einem CM oder CB anstehende Meldungen werden gelöscht, wenn Send_Config ausgeführt wird.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Sie können die statische Anfangskonfiguration des Ports in den Eigenschaften der Gerätekonfiguration einrichten oder einfach die Standardwerte nutzen. Dann können Sie mit Send_Config in Ihrem Programm die Konfiguration ändern.

Die CPU speichert die von Ihnen in der Anweisung Send_Config festgelegten Werte nicht dauerhaft. Die in der Gerätekonfiguration gespeicherten Parameter werden von der CPU wiederhergestellt, wenn diese von RUN in STOP wechselt oder neu eingeschaltet wird. Siehe Sendeparameter konfigurieren (Seite 906).

Tabelle 13- 11 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert die Konfigurationsänderung bei steigender Flanke an diesem Eingang. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie ein CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
RTSONDLY	IN	UInt	Anzahl der Millisekunden, die nach Aktivierung von RTS gewartet werden soll, bevor eine Übertragung von Tx-Daten durchgeführt wird. Dieser Parameter ist nur gültig, wenn die Hardwareflusskontrolle aktiviert ist. Der gültige Bereich ist 0 bis 65535 ms. Der Wert 0 deaktiviert die Funktion. (Standardwert: 0)
RTSOFFDLY	IN	UInt	Anzahl der Millisekunden, die nach der Übertragung von Tx-Daten gewartet werden soll, bevor RTS deaktiviert wird: Dieser Parameter ist nur gültig, wenn die Hardwareflusskontrolle aktiviert ist. Der gültige Bereich ist 0 bis 65535 ms. Der Wert 0 deaktiviert die Funktion. (Standardwert: 0)
BREAK	IN	UInt	Dieser Parameter gibt an, dass beim Start jeder Meldung für die angegebene Anzahl von Bitzeiten eine Pause gesendet wird. Maximal sind 65535 Bitzeiten bis maximal 8 Sekunden möglich. Der Wert 0 deaktiviert die Funktion. (Standardwert: 12)
IDLELINE	IN	UInt	Dieser Parameter gibt an, dass vor dem Start jeder Meldung für die angegebene Anzahl von Bitzeiten die Leitung im Leerlauf bleibt. Maximal sind 65535 Bitzeiten bis maximal 8 Sekunden möglich. Der Wert 0 deaktiviert die Funktion. (Standardwert: 0)
USR_END*	IN	STRING[2]	Gibt Anzahl und Art der Zeichen im Abschlusstrennzeichen an. Das Abschlusstrennzeichen ist in den Sendepuffer eingebettet (nur Zeichen) und kennzeichnet das Ende der gesendeten Meldung (Zeichen werden bis zum Abschlusstrennzeichen übertragen). Das Abschlusstrennzeichen wird am Ende der Meldung angehängt. • STRING[2,0,xx,yy] – Abschlusstrennzeichen nicht benutzt (Standard) • STRING[2,1,xx,yy] – Abschlusstrennzeichen besteht aus einem Zeichen • STRING[2,2,xx,yy] – Abschlusstrennzeichen besteht aus zwei Zeichen Entweder USR_END oder APP_END müssen eine Länge von null haben.

Parameter und Datentyp		Datentyp	Beschreibung
APP_END*	IN	STRING[5]	Gibt Anzahl und Art der Zeichen an, die an die gesendete Meldung anzuhängen sind (nur die Zeichen werden angehägt). STRING[5,0,aa,bb,cc,dd,ee] – Endezeichen nicht benutzt (Standard) • STRING[5,1,aa,bb,cc,dd,ee] – Ein Endezeichen senden • STRING[5,2,aa,bb,cc,dd,ee] – Zwei Endezeichen senden • STRING[5,3,aa,bb,cc,dd,ee]– Drei Endezeichen senden • STRING[5,4,aa,bb,cc,dd,ee] – Vier Endezeichen senden • STRING[5,5,aa,bb,cc,dd,ee] – Fünf Endezeichen senden
DONE	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

* Für CM und CB 1241 nicht unterstützt; für den Parameter ist eine leere Zeichenkette ("") zu verwenden.

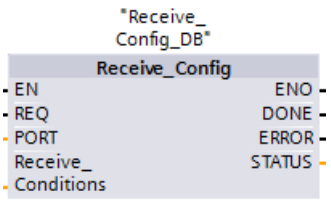
Tabelle 13- 12 Bedingungscodes

STATUS (W#16#....)	Beschreibung
81B0	Konfiguration eines Sendealarms ist nicht zulässig. Wenden Sie sich an den Kunden-Support.
81B1	Pausenzeit ist größer als der maximal zulässige Wert.
81B2	Leerlaufzeit ist größer als der maximal zulässige Wert.
81B3	Fehler im Baustein-Header, zum Beispiel falscher Bausteintyp oder falsche Bausteinlänge
81B4	Rekonfiguration abgelehnt, weil ein Konfigurierungsvorgang läuft
81B5	Angegebene Anzahl der Abschlusstrennzeichen ist größer als zwei oder Anzahl der Endezeichen ist größer als fünf
81B6	Sendekonfiguration abgelehnt, wenn in der Firmware eingebettete Protokolle betroffen sind
8280	Negative Quittierung beim Lesen aus Modul
8281	Negative Quittierung beim Schreiben in Modul
8282	DP-Slave oder Modul nicht verfügbar

13.3.3.4 Anweisung Receive_Config (Serielle Empfangsparameter dynamisch konfigurieren)

Tabelle 13- 13 Anweisung Receive_Config (Empfangskonfiguration)

KOP/FUP	SCL	Beschreibung

	`"Receive_Config_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`Re-`
	`ceive_Conditions:=_struct_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Receive_Config führt die dynamische Konfiguration serieller Empfangsparameter für eine PtP-Kommunikationsschnittstelle durch. Diese Anweisung konfiguriert die Bedingungen, die den Beginn und das Ende einer empfangenen Meldung kennzeichnen. In einem CM oder CB anstehende Meldungen werden gelöscht, wenn Receive_Config ausgeführt wird.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Sie können die statische Anfangskonfiguration des Kommunikationsanschlusses in den Eigenschaften der Gerätekonfiguration einrichten oder einfach die Standardwerte nutzen. Dann können Sie mit Receive_Config in Ihrem Programm die Konfiguration ändern.

Die CPU speichert die von Ihnen in der Anweisung Receive_Config festgelegten Werte nicht dauerhaft. Die in der Gerätekonfiguration gespeicherten Parameter werden von der CPU wiederhergestellt, wenn diese von RUN in STOP wechselt oder neu eingeschaltet wird. Weitere Informationen dazu finden Sie unter "Empfangsparameter konfigurieren (Seite 907)".

Tabelle 13- 14 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert die Konfigurationsänderung bei steigender Flanke an diesem Eingang. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
CONDITIONS	IN	CONDITIONS	Die Datenstruktur von CONDITIONS gibt die Anfangs- und Endbedingungen der Meldung wie im Folgenden beschrieben an.
DONE	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Startbedingungen für die Anweisung Receive_P2P

Die Anweisung Receive_P2P verwendet die von der Anweisung Receive_Config angegebene Konfiguration, um Anfang und Ende von Punkt-zu-Punkt-Kommunikationsmeldungen zu bestimmen. Der Meldungsbeginn wird von den Startbedingungen festgelegt. Der Meldungsbeginn kann anhand von einer oder mehreren Startbedingungen ermittelt werden. Sind mehrere Startbedingungen angegeben, müssen alle Bedingungen erfüllt sein, damit die Meldung gestartet wird.

Eine Beschreibung der Bedingungen für den Meldungsbeginn finden Sie unter "Empfangsparameter konfigurieren (Seite 907)".

Datentypstruktur des Parameters CONDITIONS, Teil 1 (Startbedingungen)

Tabelle 13- 15 Struktur von CONDITIONS für Startbedingungen

Parameter und Datentyp		Datentyp	Beschreibung
STARTCOND	IN	UInt	Angabe der Startbedingung (Standardwert: 1) • 01H - Startzeichen • 02H - Beliebiges Zeichen • 04H - Leitungspause • 08H - Leitung im Leerlauf • 10H - Zeichenfolge 1 • 20H - Zeichenfolge 2 • 40H - Zeichenfolge 3 • 80H - Zeichenfolge 4
IDLETIME	IN	UInt	Die Anzahl der erforderlichen Bitzeiten für Timeout des Leitungsleerlaufs. (Standardwert: 40). Nur in Verbindung mit der Bedingung "Leitung im Leerlauf". 0 bis 65535
STARTCHAR	IN	Byte	Das Startzeichen für die Bedingung "Startzeichen". (Standardwert: B#16#2)
STRSEQ1CTL	IN	Byte	Zeichenfolge 1, Steuerung für jedes Zeichen ignorieren/vergleichen: (Standardwert: B#16#0) Dies sind die Aktivierungsbits für jedes Zeichen der Startzeichenfolge. • 01H - Zeichen 1 • 02H - Zeichen 2 • 04H - Zeichen 3 • 08H - Zeichen 4 • 10H - Zeichen 5 Wird ein Bit für ein bestimmtes Zeichen deaktiviert, bedeutet dies, dass an dieser Position der Zeichenfolge jedes Zeichen eine Übereinstimmung darstellt.
STRSEQ1	IN	Char[5]	Zeichenfolge 1, Startzeichen (5 Zeichen). Standardwert: 0
STRSEQ2CTL	IN	Byte	Zeichenfolge 2, Steuerung für jedes Zeichen ignorieren/vergleichen. Standardwert: B#16#0)
STRSEQ2	IN	Char[5]	Zeichenfolge 2, Startzeichen (5 Zeichen). Standardwert: 0

Parameter und Datentyp		Datentyp	Beschreibung
STRSEQ3CTL	IN	Byte	Zeichenfolge 3, Steuerung für jedes Zeichen ignorieren/vergleichen. Standardwert: B#16#0
STRSEQ3	IN	Char[5]	Zeichenfolge 3, Startzeichen (5 Zeichen). Standardwert: 0
STRSEQ4CTL	IN	Byte	Zeichenfolge 4, Steuerung für jedes Zeichen ignorieren/vergleichen. Standardwert: B#16#0
STRSEQ4	IN	Char[5]	Zeichenfolge 4, Startzeichen (5 Zeichen), Standardwert: 0

Beispiel

Sehen Sie sich die folgende empfangene hexadezimal-codierte Meldung an: "**68** 10 aa **68** bb 10 aa 16". Die konfigurierten Startzeichenfolgen finden Sie in der folgenden Tabelle. Startzeichenfolgen werden ausgewertet, nachdem das erste Zeichen 68H erfolgreich empfangen wurde. Nach erfolgreichem Empfang des vierten Zeichens (das zweite 68H) ist die Startbedingung 1 erfüllt. Wenn die Startbedingungen erfüllt sind, beginnt die Auswertung der Endebedingungen.

Die Verarbeitung der Startzeichenfolge kann aufgrund verschiedener Fehler bei Parität, Framing oder Zeitabständen zwischen den Zeichen beendet werden. Diese Fehler führen dazu, dass die Meldung nicht empfangen wird, weil die Startbedingung nicht erfüllt wurde.

Tabelle 13- 16 Startbedingungen

Startbedingung	Erstes Zeichen	Erstes Zeichen +1	Erstes Zeichen +2	Erstes Zeichen +3	Erstes Zeichen +4
1	**68**H	xx	xx	**68**H	xx
2	10H	aaH	xx	xx	xx
3	dcH	aaH	xx	xx	xx
4	e5H	xx	xx	xx	xx

Endebedingungen für die Anweisung Receive_P2P

Das Ende einer Meldung wird von den angegebenen Endebedingungen festgelegt. Das Ende einer Meldung wird durch das erste Auftreten einer oder mehrerer konfigurierter Endebedingungen festgelegt. Im Abschnitt "Bedingungen für den Meldungsbeginn" unter "Empfangsparameter konfigurieren (Seite 907)" werden die Endebedingungen beschrieben, die Sie für die Anweisung Receive_Config konfigurieren können.

Sie können die Endebedingungen entweder in den Eigenschaften der Kommunikationsschnittstelle in der Gerätekonfiguration oder über die Anweisung Receive_Config konfigurieren. Immer wenn die CPU von STOP in RUN wechselt, werden die Empfangsparameter (Start- und Endebedingungen) wieder auf die Einstellungen der Gerätekonfiguration gesetzt. Wenn das STEP 7 Anwenderprogramm Receive_Config ausführt, werden die Einstellungen auf die Bedingungen von Receive_Config gesetzt.

Datentypstruktur des Parameters CONDITIONS, Teil 2 (Endebedingungen)

Tabelle 13- 17 Struktur von CONDITIONS für Endebedingungen

Parameter	Parametertyp	Datentyp	Beschreibung
ENDCOND	IN	UInt 0	Dieser Parameter gibt die Bedingung für das Meldungsende an: • 01H - Antwortzeit • 02H - Meldungszeit • 04H - Zeichenabstand • 08H - Maximale Länge • 10H - N + LEN + M • 20H - Zeichenfolge
MAXLEN	IN	UInt 1	Maximale Meldungslänge: Wird nur verwendet, wenn die Endebedingung "Maximale Länge" ausgewählt ist. 1 bis 1024 Bytes
N	IN	UInt 0	Byteposition des Längenfelds in der Meldung. Wird nur bei der Endebedingung N + LEN + M verwendet. 1 bis 1022 Bytes
LENGTHSIZE	IN	UInt 0	Größe des Bytefelds (1, 2 oder 4 Byte). Wird nur bei der Endebedingung N + LEN + M verwendet.
LENGTHM	IN	UInt 0	Geben Sie die Anzahl der Zeichen nach dem Längenfeld an, die nicht im Wert des Längenfelds enthalten sind. Diese Angabe wird nur bei der Endebedingung N + LEN + M verwendet. 0 bis 255 Bytes
RCVTIME	IN	UInt 200	Geben Sie an, wie lange auf das erste empfangene Zeichen gewartet werden soll. Die Empfangsanweisung wird mit einem Fehler beendet, wenn nicht innerhalb der angegebenen Zeit ein Zeichen erfolgreich empfangen wird. Diese Angabe wird nur bei der Bedingung "Antwortzeit" verwendet. (0 bis 65535 Bitzeiten, maximal 8 Sekunden) Bei diesem Parameter handelt es sich nicht um eine Endebedingung, weil die Auswertung beendet wird, wenn das erste Zeichen einer Antwort empfangen wird. Es handelt sich nur in dem Sinn um eine Endebedingung, als dass eine Empfangsanweisung beendet wird, weil bei Erwartung einer Antwort keine Antwort empfangen wird. Sie müssen eine getrennte Endebedingung definieren.
MSGTIME	IN	UInt 200	Geben Sie an, wie lange nach dem Empfang des ersten Zeichens auf den vollständigen Empfang der gesamten Meldung gewartet werden soll. Dieser Parameter wird nur verwendet, wenn die Bedingung "Timeout der Meldung" ausgewählt ist. (0 bis 65535 ms)
CHARGAP	IN	UInt 12	Geben Sie die Anzahl der Bitzeiten zwischen den Zeichen an. Wenn die Anzahl der Bitzeiten zwischen den Zeichen den angegebenen Wert überschreitet, ist die Endebedingung erfüllt. Diese Angabe wird nur bei der Bedingung "Zeichenabstand" verwendet. (0 bis 65535 Bitzeiten, maximal 8 Sekunden)

Parameter	Parametertyp	Datentyp	Beschreibung
ENDSEQ1CTL	IN	Byte B#16#0	Zeichenfolge 1, Steuerung für jedes Zeichen ignorieren/vergleichen: Dies sind die Aktivierungsbits für jedes Zeichen der Endezeichenfolge. Zeichen 1 ist Bit 0, Zeichen 2 ist Bit 1, ..., Zeichen 5 ist Bit 4. Wird ein Bit für ein bestimmtes Zeichen deaktiviert, bedeutet dies, dass an dieser Position der Zeichenfolge jedes Zeichen eine Übereinstimmung darstellt.
ENDSEQ1	IN	Char[5] 0	Zeichenfolge 1, Startzeichen (5 Zeichen)

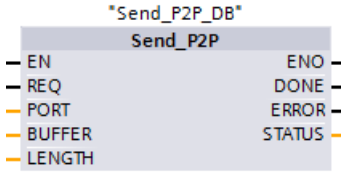
Tabelle 13- 18 Bedingungscodes

STATUS (W#16#....)	Beschreibung
81C0	Unzulässige Startbedingung ausgewählt
81C1	Unzulässige Endebedingung ausgewählt, keine Endebedingung ausgewählt
81C2	Empfangsalarm aktiviert und dies ist nicht möglich.
81C3	Endebedingung "Maximale Länge" ist aktiviert und die maximale Länge ist 0 oder > 1024.
81C4	Berechnete Länge ist aktiviert und N ist >= 1023.
81C5	Berechnete Länge ist aktiviert und Länge ist nicht 1, 2 oder 4.
81C6	Berechnete Länge ist aktiviert und der Wert von M ist > 255.
81C7	Berechnete Länge ist aktiviert und die berechnete Länge ist > 1024.
81C8	Timeout der Antwort ist aktiviert und das Antwort-Timeout ist null.
81C9	Timeout für den Zeichenabstand ist aktiviert und das Timeout ist null.
81CA	Timeout für den Leitungsleerlauf ist aktiviert und das Timeout ist null.
81CB	Endezeichenfolge ist aktiviert, doch alle Zeichen sind "nicht relevant".
81CC	Startzeichenfolge (eine von 4) ist aktiviert, doch alle Zeichen sind "nicht relevant".
81CD	Fehler Überschreibschutz Empfangsmeldung ungültig
81CE	Fehler Pufferverarbeitung für Empfangsmeldungen beim Übergang von STOP zu RUN ungültig
81CF	Fehler im Baustein-Header, zum Beispiel falscher Bausteintyp oder falsche Bausteinlänge
8281	Negative Quittierung beim Schreiben in Modul
8282	DP-Slave oder Modul nicht verfügbar
82C0	Rekonfiguration abgelehnt, weil ein Konfigurierungsvorgang läuft
82C1	Der angegebene Wert für die Anzahl der Meldungen, die das Modul puffern kann, ist größer als der maximal zulässige Wert.
82C2	Empfangskonfiguration abgelehnt, wenn in der Firmware eingebettete Protokolle betroffen sind
8351	Datentyp für diesen Variant-Pointer nicht zulässig

13.3.3.5 Anweisung Send_P2P (Sendepufferdaten übertragen)

Tabelle 13- 19 Anweisung Send_P2P (Punkt-zu-Punkt-Daten senden)

KOP/FUP	SCL	Beschreibung

	`"Send_P2P_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`BUFFER:=_variant_in_,`
	`LENGTH:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Send_P2P startet die Übertragung der Daten und überträgt den zugewiesenen Puffer zur Kommunikationsschnittstelle. Das Programm der CPU wird weiterhin ausgeführt, während das CM oder CB die Daten mit der zugewiesenen Baudrate sendet. Es darf zu jeder Zeit nur eine Sendeanweisung anstehen. Das CM oder CB gibt einen Fehler aus, wenn eine zweite Anweisung Send_P2P ausgeführt wird, während das CM oder CB bereits eine Meldung sendet.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

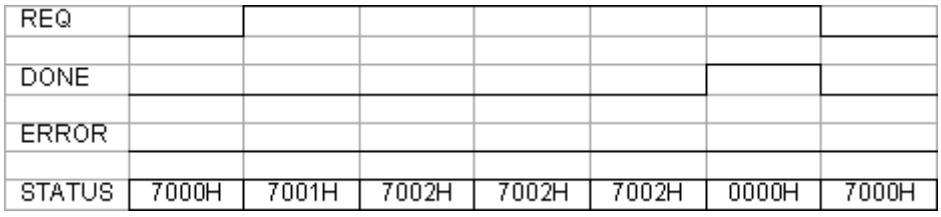
Tabelle 13- 20 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Freigabe der angeforderten Übertragung bei steigender Flanke dieses Freigabeeingangs. Dadurch wird der Inhalt des Puffers zur Punkt-zu-Punkt-Kommunikationsschnittstelle übertragen. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
BUFFER	IN	Variante	Dieser Parameter zeigt auf die Anfangsadresse des Sendepuffers. (Standardwert: 0) **Hinweis:** Boolesche Daten oder Boolesche Felder werden nicht unterstützt.
LENGTH	IN	UInt	Übertragene Rahmenlänge in Byte (Standardwert: 0) Verwenden Sie beim Senden einer komplexen Struktur immer die Länge 0. Bei einer Länge 0 sendet die Anweisung den gesamten Rahmen.
DONE	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

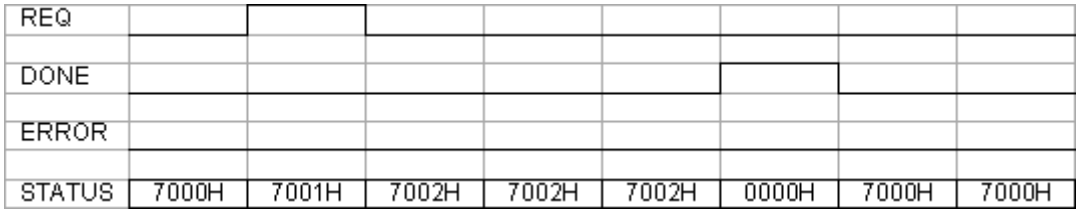
Wenn eine Sendeanweisung in Bearbeitung ist, sind die Ausgänge DONE und ERROR im Zustand FALSCH. Nach dem Ende einer Sendeanweisung wird einer der Ausgänge DONE oder ERROR auf WAHR gesetzt, um den Zustand der Sendeanweisung zu melden. Während DONE oder ERROR im Zustand WAHR ist, ist der Ausgang STATUS gültig.

Die Anweisung gibt den Status 16#7001 aus, wenn die Kommunikationsschnittstelle die Sendedaten annimmt. Nachfolgende Ausführungen von Send_P2P geben den Wert 16#7002 aus, wenn das CM oder CB immer noch sendet. Nach dem Ende der Sendeanweisung gibt das CM oder CB den Status 16#0000 für die Sendeanweisung aus (sofern kein Fehler aufgetreten ist). Nachfolgende Ausführungen von Send_P2P mit REQ = 0 geben den Status 16#7000 (frei) aus.

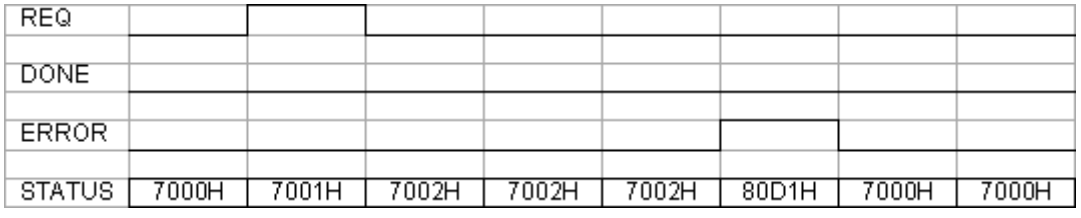
Das folgende Diagramm zeigt die Beziehung zwischen den Ausgangswerten und REQ. Hierbei wird vorausgesetzt, dass die Anweisung regelmäßig aufgerufen wird, um den Status des Sendevorgangs zu prüfen. In der folgenden Abbildung wird davon ausgegangen, dass die Anweisung in jedem Zyklus aufgerufen wird (dargestellt durch die STATUS-Werte).



Die folgende Abbildung zeigt, wie die Parameter DONE und STATUS nur einen Zyklus lang gültig sind, wenn an der REQ-Leitung (einen Zyklus lang) ein Impuls anliegt, um die Sendeanweisung anzustoßen.



Die folgende Abbildung zeigt die Beziehung der Parameter DONE, ERROR und STATUS im Fehlerfall.



Die Werte DONE, ERROR und STATUS sind nur solange gültig, bis Send_P2P erneut mit dem gleichen Instanz-DB ausgeführt wird.

Tabelle 13- 21 Bedingungscodes

STATUS (W#16#....)	Beschreibung
81D0	Neue Anforderung bei aktivem Sender
81D1	Sendung abgebrochen, weil kein CTS innerhalb der Wartezeit
81D2	Sendung abgebrochen, weil kein DSR vom DCE-Gerät
81D3	Sendung wegen Überlauf der Warteschlange abgebrochen (mehr als 1024 Bytes senden)
81D5	Umgekehrtes Biassignal (Drahtbruch)
81D6	Sendeanforderung abgelehnt, weil im Sendepuffer kein Abschlusstrennzeichen gefunden wurde.
81D7	Interner Fehler / Synchronisationsfehler zwischen FB und CM
81D8	Sendeversuch abgelehnt, weil der Anschluss nicht konfiguriert ist.
81DF	CM hat Schnittstelle zum FB aus einem der folgenden Gründe zurückgesetzt • Modul neu gestartet (Aus-/Einschalten) • CPU hat einen Haltepunkt erreicht und ODIS (Ausgang deaktivieren) gesetzt • Modul wurde neu parametriert Das Modul zeigt den entsprechenden Code in jedem Fall im Status-Parameter an. Das Modul setzt Status und Error nach dem ersten empfangenen Datensatz für SEND_P2P zurück.
8281	Negative Quittierung beim Schreiben in Modul
8282	DP-Slave oder Modul nicht verfügbar
8301	Unzulässige Syntax-ID an einem ANY-Pointer
8322	Bereichslängenfehler beim Lesen eines Parameters
8324	Bereichsfehler beim Lesen eines Parameters
8328	Ausrichtungsfehler beim Lesen eines Parameters
8332	Der Parameter enthält eine DB-Nummer, die höher als die maximal zulässige Nummer ist (DB-Nummernfehler).
833A	Der DB für den Parameter BUFFER ist nicht vorhanden.

Interaktion der Parameter LENGTH und BUFFER

Die Mindestdatengröße, die von der Anweisung SEND_P2P gesendet werden kann, ist ein Byte. Der Parameter BUFFER legt die Größe der zu sendenden Daten fest. Sie können für den Parameter BUFFER weder den Datentyp Bool noch Arrays vom Typ Bool verwenden.

Sie können den Parameter LENGTH jederzeit auf 0 setzen und damit sicherstellen, dass SEND_P2P die gesamte Datenstruktur des Parameters BUFFER sendet. Wenn Sie nur einen Teil der Datenstruktur des Parameters BUFFER senden möchten, können Sie LENGTH wie folgt setzen:

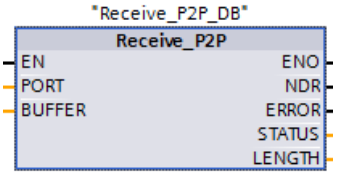
Tabelle 13- 22 Parameter LENGTH und BUFFER

LENGTH	BUFFER	Beschreibung
= 0	Nicht verwendet	Die vollständigen Daten werden wie vom Parameter BUFFER definiert gesendet. Wenn LENGTH = 0, brauchen Sie die Anzahl der übertragenen Bytes nicht anzugeben.
> 0	Elementarer Datentyp	Der LENGTH-Wert muss die Bytezahl dieses Datentyps enthalten. Beispiel: Bei einem Word-Wert muss LENGTH zwei sein. Bei einem Dword- oder Real-Wert muss LENGTH vier sein. Andernfalls erfolgt keine Übertragung, und der Fehler 8088H wird ausgegeben.
	Struktur	Der LENGTH-Wert kann eine Bytezahl enthalten, die kleiner als die vollständige Bytelänge der Struktur ist. In diesem Fall werden von der Anweisung nur die ersten n Bytes der Struktur aus BUFFER, gesendet, wobei n = LENGTH ist. Weil die interne Byteanordnung einer Struktur nicht immer bestimmt werden kann, kann dies zu unerwarteten Ergebnissen führen. Verwenden Sie dann eine LENGTH = 0, um die vollständige Struktur zu senden.
	Array	Der LENGTH-Wert muss eine Bytezahl kleiner als oder gleich der vollständigen Bytelänge des Arrays enthalten, wobei es sich um ein Vielfaches der Bytezahl des Datenelements handeln muss. Beispiel: Der Parameter LENGTH eines Arrays vom Typ Word muss ein Vielfaches von zwei sein und bei einem Array vom Typ Real ein Vielfaches von vier. Wenn LENGTH angegeben ist, überträgt die Anweisung die Anzahl der Arrayelemente, die dem LENGTH-Wert in Bytes entspricht. Wenn beispielsweise BUFFER ein Array mit 15 Dword-Elementen (insgesamt 60 Bytes) enthält und Sie geben LENGTH = 20 an, werden die ersten fünf Dword-Elemente aus dem Array übertragen. Der LENGTH-Wert muss ein Vielfaches der Bytezahl der Datenelemente sein. Andernfalls ist STATUS = 8088H, ERROR = 1 und keine Übertragung erfolgt.
	String	Der Parameter LENGTH enthält die Anzahl der zu sendenden Zeichen. Nur die Zeichen des String werden übertragen. Die Bytes mit der maximalen und der tatsächlichen Länge des String werden nicht gesendet.

13.3.3.6 Anweisung Receive_P2P (Empfangsmeldungen aktivieren)

Tabelle 13- 23 Anweisung Receive_P2P (Punkt-zu-Punkt-Daten empfangen)

KOP/FUP	SCL	Beschreibung

	`"Receive_P2P_DB"(`
	`PORT:=_word_in_,`
	`BUFFER:=_variant_in_,`
	`NDR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`LENGTH=>_uint_out_);`
		Receive_P2P prüft die Meldungen, die im CM oder CB empfangen wurden. Wenn eine Meldung verfügbar ist, wird sie vom CM oder CB zur CPU übertragen. Ein Fehler gibt den entsprechenden STATUS-Wert aus.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 13- 24 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
BUFFER	IN	Variante	Dieser Parameter zeigt auf die Anfangsadresse des Empfangspuffers. Dieser Puffer muss groß genug sein, um die maximale Meldungslänge zu empfangen. Boolesche Daten oder Boolesche Felder werden nicht unterstützt. (Standardwert: 0)
NDR	OUT	Bool	Eine Ausführung lang WAHR, wenn neue Daten bereit sind und die Anweisung fehlerfrei beendet wurde.
ERROR	OUT	Bool	Eine Ausführung lang WAHR, nachdem die Anweisung mit Fehler beendet wurde.
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)
LENGTH	OUT	UInt	Länge der ausgegebenen Meldung in Byte (Standardwert: 0)

Der STATUS-Wert ist gültig, wenn NDR oder ERROR im Zustand WAHR ist. Der STATUS-Wert liefert den Grund für die Beendigung der Empfangsoperation im CM oder CB. Dies ist typischerweise ein positiver Wert, der angibt, dass die Empfangsoperation erfolgreich war und dass der Empfangsvorgang normal beendet wurde. Ist der STATUS-Wert negativ (das höchstwertige Bit des Hexadezimalwerts ist gesetzt), wurde die Empfangsoperation wegen einer Fehlerbedingung wie Paritäts-, Framing- oder Überlauffehler beendet.

Jede PtP-Kommunikationsschnittstelle kann maximal 1024 Bytes puffern. Hierbei kann es sich um eine große oder mehrere kleinere Meldungen handeln. Sind mehrere Meldungen im CM oder CB verfügbar, gibt die Anweisung Receive_P2P die älteste verfügbare Meldung aus. Eine nachfolgend ausgeführte Anweisung Receive_P2P gibt die zweitälteste Meldung aus.

Tabelle 13- 25 Bedingungscodes

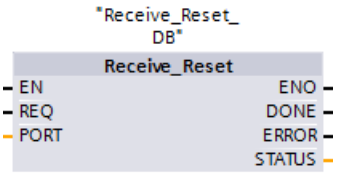
STATUS (W#16#...)	Beschreibung
0000	Kein Puffer vorhanden
0094	Meldung wurde beendet, weil die maximale Zeichenlänge empfangen wurde
0095	Meldung wurde wegen Meldungs-Timeout beendet
0096	Meldung wurde wegen Zeichenabstands-Timeout beendet
0097	Meldung wurde wegen Antwort-Timeout beendet
0098	Meldung wurde beendet, weil die Längenbedingung "N+LEN+M" erfüllt war
0099	Meldung wurde beendet, weil die Endezeichenfolge erfüllt war
8085	Parameter LENGTH hat einen Wert 0 oder ist größer als 4 kB.
8088	Parameter LENGTH oder die empfangene Länge ist größer als der in BUFFER angegebene Bereich oder die empfangene Länge ist größer als der in BUFFER angegebene Bereich.
8090	Fehlerhafte Konfigurationsmeldung, falsche Meldungslänge, falsches Submodul, unzulässige Meldung
81E0	Meldung wurde beendet, weil der Empfangspuffer voll ist
81E1	Meldung wurde wegen Paritätsfehler beendet
81E2	Meldung wurde wegen Framingfehler beendet
81E3	Meldung wurde wegen Überlauffehler beendet
81E4	Meldung wurde beendet, weil die berechnete Länge die Puffergröße überschreitet
81E5	Umgekehrtes Biassignal (Drahtbruch)
81E6	Meldungswarteschlange voll. Dieser Fehler wird ohne Daten gemeldet. Tritt er auf, schaltet das Modul zwischen einer fehlerfreien Datenübertragung und dem Fehler hin und her.
81E7	Interner Fehler, Synchronisationsfehler zwischen Anweisung und CM: wird bei einem Sequenzfehler gesetzt
81E8	Meldung beendet, Zeitüberschreitung zwischen Zeichen erreicht, bevor Meldungsende erkannt wurde
81E9	Modbus-CRC-Fehler erkannt (nur für Module, die die CRC-Generierung/Prüfung für das Modbus-Protokoll unterstützen)
81EA	Modbus-Telegramm zu kurz (nur für Module, die die CRC-Generierung/Prüfung für das Modbus-Protokoll unterstützen)
81EB	Meldung abgebrochen, da maximale Meldungsgröße überschritten
8201	Unzulässige Syntax-ID an einem ANY-Pointer
8223	Bereichslängenfehler beim Schreiben eines Parameters. Der Parameter liegt ganz oder teilweise außerhalb eines Adressbereichs oder die Länge eines Bitbereichs ist kein Vielfaches von 8 bei einem ANY-Pointer.
8225	Bereichsfehler beim Schreiben eines Parameters. Der Parameter liegt in einem Bereich, der für die Systemfunktion nicht zulässig ist.
8229	Ausrichtungsfehler beim Schreiben eines Parameters. Der referenzierte Parameter befindet sich an einer Bitadresse, die nicht gleich 0 ist.
8230	Der Parameter befindet sich in einem schreibgeschützten globalen DB

STATUS (W#16#...)	Beschreibung
8231	Der Parameter befindet sich in einem schreibgeschützten Instanz-DB
8232	Der Parameter enthält eine DB-Nummer, die höher als die maximal zulässige Bausteinnummer ist (DB-Nummernfehler).
823A	Der DB für den Parameter BUFFER ist nicht vorhanden.
8280	Negative Quittierung beim Lesen aus Modul
8282	DP-Slave oder Modul nicht verfügbar

13.3.3.7 Anweisung Receive_Reset (Empfangspuffer löschen)

Tabelle 13- 26 Anweisung Receive_Reset (Empfänger zurücksetzen)

KOP/FUP	SCL	Beschreibung

	`"Receive_Reset_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Receive_Reset löscht den Empfangspuffer im CM oder CB.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

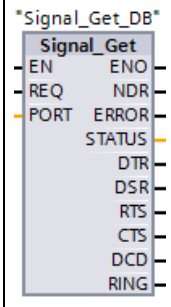
Tabelle 13- 27 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert das Löschen des Empfangspuffers bei steigender Flanke dieses Freigabeeingangs (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
DONE	OUT	Bool	Einen Zyklus lang WAHR bedeutet, dass die letzte Anforderung fehlerfrei ausgeführt wurde.
ERROR	OUT	Bool	WAHR bedeutet, dass die letzte Anforderung mit Fehlern ausgeführt wurde. Ist dieser Ausgang WAHR, enthält Ausgang STATUS die zugehörigen Fehlercodes.
STATUS	OUT	Word	Fehlercode (Standardwert: 0)

13.3.3.8 Anweisung Signal_Get (RS232-Signale abfragen)

Tabelle 13- 28 Anweisung Signal_Get (RS232-Signale abrufen)

KOP/FUP	SCL	Beschreibung

	`"Signal_Get_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`NDR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`DTR=>_bool_out_,`
	`DSR=>_bool_out_,`
	`RTS=>_bool_out_,`
	`CTS=>_bool_out_,`
	`DCD=>_bool_out_,`
	`RING=>_bool_out_);`
		Signal_Get liest die aktuellen Zustände der RS232-Kommunikationssignale.
		Diese Funktion gilt nur beim RS232-CM.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 13- 29 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	RS232-Signalzustände werden bei der steigenden Flanke an diesem Eingang abgerufen (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen.
NDR	OUT	Bool	Einen Zyklus lang WAHR, wenn neue Daten bereit sind und die Anweisung fehlerfrei beendet wurde
ERROR	OUT	Bool	Einen Zyklus lang WAHR, nachdem die Anweisung mit Fehler beendet wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)
DTR	OUT	Bool	Datenterminal bereit, Modul bereit (Ausgang). Standardwert: False
DSR	OUT	Bool	Datensatz bereit, Kommunikationsteilnehmer bereit (Eingang). Standardwert: False
RTS	OUT	Bool	Sendeanforderung, Modul sendebereit (Ausgang). Standardwert: False
CTS	OUT	Bool	Sendebereit, Kommunikationsteilnehmer kann Daten empfangen (Eingang). Standardwert: False
DCD	OUT	Bool	Datenträger erkannt, Signalpegel empfangen (immer False, nicht unterstützt)
RING	OUT	Bool	Rufanzeige, Meldung eines eingehenden Rufs (immer False, nicht unterstützt)

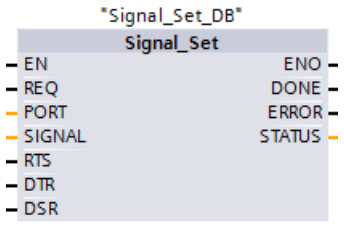
Tabelle 13- 30 Bedingungscodes

STATUS (W#16#....)	Beschreibung
81F0	CM oder CB ist RS485 und es sind keine Signale verfügbar
81F4	Fehler im Baustein-Header, zum Beispiel falscher Bausteintyp oder falsche Bausteinlänge
8280	Negative Quittierung beim Lesen aus Modul
8282	DP-Slave oder Modul nicht verfügbar

13.3.3.9 Anweisung Signal_Set (RS232-Signale setzen)

Tabelle 13- 31 Anweisung Signal_Set (RS232-Signale setzen)

KOP/FUP	SCL	Beschreibung

	`"Signal_Set_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`SIGNAL:=_byte_in_,`
	`RTS:=_bool_in_,`
	`DTR:=_bool_in_,`
	`DSR:=_bool_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Signal_Set setzt die Zustände der RS232-Kommunikationssignale.
		Diese Funktion gilt nur beim RS232-CM.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 13- 32 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung zum Setzen der RS232-Signale wird bei der steigenden Flanke an diesem Eingang gestartet (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
SIGNAL	IN	Byte	Auswahl des zu setzenden Signals: (mehrere zulässig). Standardwert: 0 • 01H = RTS • 02H = DTR • 04H = DSR
RTS	IN	Bool	Sendeanforderung, Modul bereit, zu setzenden Wert zu senden (wahr oder falsch), Standardwert: Falsch
DTR	IN	Bool	Datenterminal bereit, Modul bereit, zu setzenden Wert zu senden (wahr oder falsch). Standardwert: Falsch
DSR	IN	Bool	Datensatz bereit (gilt nur für Schnittstellentyp DCE), nicht verwendet.

Parameter und Datentyp		Datentyp	Beschreibung
DONE	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

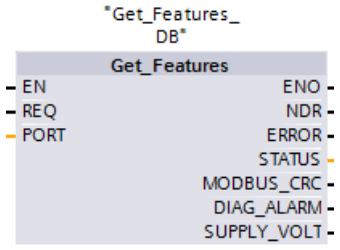
Tabelle 13- 33 Bedingungscodes

STATUS (W#16#....)	Beschreibung
81F0	CM oder CB ist RS485 und es können keine Signale gesetzt werden
81F1	Signale können wegen Hardwareflusskontrolle nicht gesetzt werden
81F2	DSR kann nicht gesetzt werden, weil das Modul ein DTE-Gerät ist
81F3	DTR kann nicht gesetzt werden, weil das Modul ein DCE-Gerät ist
81F4	Fehler im Baustein-Header, zum Beispiel falscher Bausteintyp oder falsche Bausteinlänge
8280	Negative Quittierung beim Lesen aus Modul
8281	Negative Quittierung beim Schreiben in Modul
8282	DP-Slave oder Modul nicht verfügbar

13.3.3.10 Get_Features

Tabelle 13- 34 Anweisung Get_Features (Erweiterte Funktionen abrufen)

KOP/FUP	SCL	Beschreibung

	`"Get_Features_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`NDR:=_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MODBUS_CRC=>_bool_out_,`
	`DIAG_ALARM=>_bool_out_,`
	`SUPPLY_VOLT=>_bool_out);`
		Get_Features liest die erweiterten Funktionsmöglichkeiten eines Moduls.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Mit Anweisung Get_Features werden die erweiterten Funktionsmöglichkeiten eines Moduls gelesen.

Tabelle 13- 35 Datentypen für die Parameter

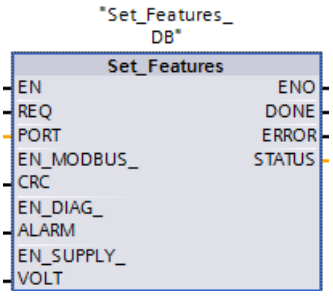
Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert die Konfigurationsänderung bei steigender Flanke an diesem Eingang. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
NDR	OUT	Bool	Zeigt an, dass neue Daten bereit sind.
ERROR	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)
MODBUS_CRC*	OUT	Bool	MODBUS CRC-Generierung und -Prüfung
DIAG_ALARM*	OUT	Bool	Erzeugung Diagnosealarm
SUPPLY_VOLT*	OUT	Bool	Diagnose für Ausfall Versorgungsspannung L+ verfügbar

*Get_Features gibt WAHR (1) zurück, wenn die Funktion verfügbar ist und FALSCH (0), wenn die Funktion nicht verfügbar ist

13.3.3.11 Set_Features

Tabelle 13- 36 Anweisung Set_Features (Erweiterte Funktionen setzen)

KOP/FUP	SCL	Beschreibung

	`"Set_Features_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_word_in_,`
	`EN_MODBUS_CRC:=_bool_in_,`
	`EN_DIAG_ALARM:=_bool_in_,`
	`EN_SUPPLY_VOLT:=_bool_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Set_Features aktiviert die erweiterten Funktionen, die ein Modul unterstützt.

1 STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Mit Anweisung Get_Features werden die erweiterten Funktionsmöglichkeiten eines Moduls gelesen.

Tabelle 13- 37 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert die Konfigurationsänderung bei steigender Flanke an diesem Eingang. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
EN_MODBUS_CRC	IN	Bool	MODBUS CRC-Generierung und -Prüfung aktivieren: • 0: CRC-Berechnung AUS (Standard) • 1: CRC-Berechnung EIN Hinweis: Dieser Parameter wird nur von CMs der Version 2.1, CPUs der Version 4.1 mit CBs und CM-PtP-Modulen für dezentrale E/A unterstützt.
EN_DIAG_ALARM	IN	Bool	Diagnosealarm aktivieren • 0: Diagnosealarm AUS • 1: Diagnosealarm EIN (Standard)
EN_SUPPLY_VOLT	IN	Bool	Diagnose für Ausfall Versorgungsspannung L+ aktivieren: • 0: Diagnose Versorgungsspannung deaktiviert (Standard) • 1: Diagnose Versorgungsspannung aktiviert
DONE	OUT	Bool	Gibt an, dass Set_Features ausgeführt ist
ERROR	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

13.3.4 Programmieren der PtP-Kommunikation

STEP 7 bietet erweiterte Anweisungen, mit denen das Anwenderprogramm die Punkt-zu-Punkt-Kommunikation über ein im Anwenderprogramm konzipiertes und vorgegebenes Protokoll durchführen kann. Diese Anweisungen lassen sich in zwei Kategorien einteilen:

- Konfigurationsanweisungen
- Kommunikationsanweisungen

Konfigurationsanweisungen

Bevor das Anwenderprogramm die PtP-Kommunikation starten kann, müssen die Kommunikationsschnittstelle und die Parameter zum Senden und Empfangen der Daten konfiguriert werden.

Die Schnittstellenkonfiguration und die Meldungskonfiguration können für jedes CM oder CB in der Gerätekonfiguration oder mit den folgenden Anweisungen Ihres Anwenderprogramms durchgeführt werden:

- Port_Config (Seite 918)
- Send_Config (Seite 920)
- Receive_Config (Seite 923)

Kommunikationsanweisungen

Mit den Anweisungen für die Punkt-zu-Punkt-Kommunikation kann das Anwenderprogramm Meldungen an die Kommunikationsschnittstellen senden und von diesen Meldungen empfangen. Beachten Sie für weitere Informationen zum Übertragen von Daten mit diesen Anweisungen den Abschnitt zur Datenkonsistenz (Seite 191).

Alle PtP-Anweisungen funktionieren asynchron. Mit Hilfe einer Abfragearchitektur kann das Anwenderprogramm den Sende- und Empfangsstatus feststellen. Send_P2P und Receive_P2P können gleichzeitig ausgeführt werden. Die Kommunikationsmodule und das Kommunikationsboard puffern die Sende- und Empfangsmeldungen je nach Bedarf bis zu einer maximalen Puffergröße von 1024 Bytes.

Die CMs und das CB senden und empfangen Meldungen an die bzw. von den Kommunikationsteilnehmern. Das Meldungsprotokoll befindet sich in einem Puffer, der von einer bestimmten Kommunikationsschnittstelle empfangen oder an diese gesendet wird. Puffer und Port sind Parameter der Sende- und Empfangsanweisungen:

- Send_P2P (Seite 928)
- Receive_P2P (Seite 932)

Mit zusätzlichen Anweisungen kann der Empfangspuffer zurückgesetzt und es können spezielle RS232-Signale abgefragt und gesetzt werden:

- Receive_Reset (Seite 934)
- Signal_Get (Seite 935)
- Signal_Set (Seite 936)

13.3.4.1 Abfragearchitektur

Das STEP 7-Anwenderprogramm muss die Punkt-zu-Punkt-Anweisungen der S7-1200 zyklisch/regelmäßig aufrufen, um empfangene Meldungen abzufragen. Durch Abfragen des Sendevorgangs ermittelt das Anwenderprogramm, wann die Übertragung beendet ist.

Abfragearchitektur: Master

Die typische Sequenz für einen Master ist wie folgt:

1. Die Anweisung Send_P2P (Seite 928) veranlasst eine Übertragung zum CM oder CB.
2. Die Anweisung Send_P2P wird in aufeinanderfolgenden Zyklen ausgeführt, um den Status des Übertragungsvorgangs abzufragen.
3. Wenn die Anweisung Send_P2P meldet, dass die Übertragung beendet ist, kann der Anwendercode den Empfang der Antwort vorbereiten.
4. Die Anweisung Receive-P2P (Seite 932) wird wiederholt ausgeführt, um auf eine Antwort abzufragen. Wenn das CM oder CB eine Antwortmeldung erfasst hat, kopiert die Anweisung Receive_P2P die Antwort in die CPU und meldet, dass neue Daten empfangen wurden.
5. Das Anwenderprogramm kann die Antwort verarbeiten.
6. Zurück zu Schritt 1 und Wiederholung des Zyklus.

Abfragearchitektur: Slave

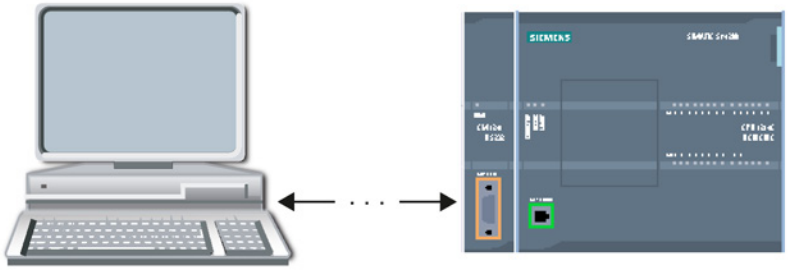
Die typische Sequenz für einen Slave ist wie folgt:

1. Das Anwenderprogramm führt die Anweisung Receive_P2P in jedem Zyklus aus.
2. Wenn das CM oder CB eine Anforderung empfangen hat, meldet die Anweisung Receive_P2P , dass neue Daten bereit sind, und die Anforderung wird in die CPU kopiert.
3. Das Anwenderprogramm verarbeitet die Anforderung und erzeugt eine Antwort.
4. Mit der Anweisung Send_P2P wird die Antwort an den Master zurückgesendet.
5. Führen Sie Send_P2P wiederholt aus, um sicherzustellen, dass der Sendevorgang stattfindet.
6. Zurück zu Schritt 1 und Wiederholung des Zyklus.

Der Slave muss dafür sorgen, dass Receive_P2P oft genug aufgerufen wird, um eine Übertragung vom Master empfangen zu können, bevor dieser beim Warten auf eine Antwort wegen Zeitüberschreitung den Vorgang abbricht. Um diese Aufgabe zu erfüllen, kann das Anwenderprogramm RCV_PTP aus einem Zyklus-OB heraus aufrufen, dessen Zykluszeit ausreichend lang ist, um eine Übertragung vom Master vor dem Ablauf der Timeout-Einstellung zu empfangen. Wird die Zykluszeit für den OB so eingestellt, dass zwei Ausführungen innerhalb der Timeout-Einstellung des Masters erfolgen, kann das Anwenderprogramm alle Übertragungen ohne Verlust empfangen.

13.3.5 Beispiel: Punkt-zu-Punkt-Kommunikation

In diesem Beispiel kommuniziert eine S7-1200 CPU mit einem PC mit einem Terminalemulator über ein CM 1241 RS232-Modul. Die Punkt-zu-Punkt-Konfiguration und das STEP 7-Programm in diesem Beispiel zeigen, wie die CPU eine Meldung vom PC empfangen und das Echo der Meldung an den PC zurückgeben kann.



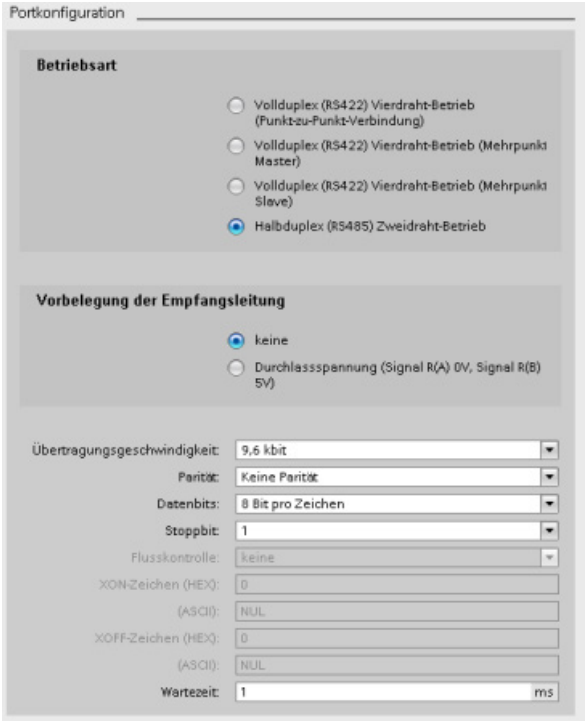
Sie müssen die Kommunikationsschnittstelle des CM 1241 RS232-Moduls an die RS232-Schnittstelle des PC anschließen. Dies ist üblicherweise COM1. Weil es sich bei beiden Ports um Datenendgeräte (Data Terminal Equipment, DTE) handelt, müssen Sie die Empfangs- und Sendepins (2 und 3) schalten, wenn Sie die beiden Ports anschließen. Hierzu gehen Sie auf eine der folgenden Arten vor:

- Verwenden Sie einen NULL-Modemadapter, um die Pins 2 und 3 zu tauschen, zusammen mit einem herkömmlichen RS232-Kabel.
- Verwenden Sie ein NULL-Modemkabel, bei dem die Pins 2 und 3 bereits getauscht sind. Sie erkennen ein NULL-Modemkabel üblicherweise an den zwei 9-poligen D-Buchsen.

13.3.5.1 Kommunikationsmodul konfigurieren

Sie können das CM 1241 in der Gerätekonfiguration in STEP 7 oder mit Anweisungen im Anwenderprogramm konfigurieren. In diesem Beispiel wird die Gerätekonfiguration verwendet.

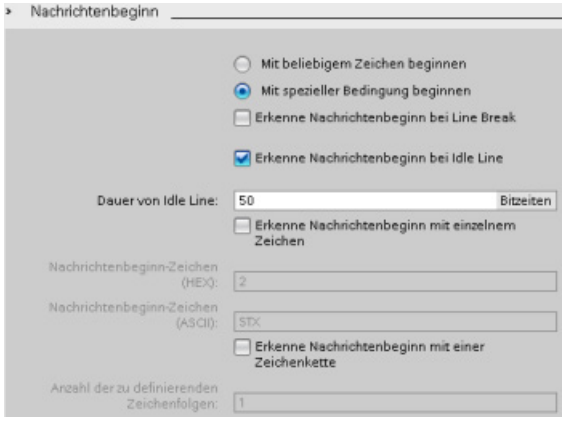
- Schnittstellenkonfiguration: Klicken Sie auf den Kommunikationsport des CM in der Gerätekonfiguration und konfigurieren Sie den Port wie folgt:



Hinweis

Die Konfigurationseinstellungen für "Betriebsart" und "Ausgangszustand Empfangsleitung" gelten nur für das Modul CM 1241 (RS422/RS485). Die anderen CM 1241-Module haben diese Portkonfigurationseinstellungen nicht. Weitere Informationen finden Sie unter RS422 und RS485 konfigurieren (Seite 946).

- Konfiguration der Sendemeldung: Übernehmen Sie die Voreinstellung für die Konfiguration zum Senden von Meldungen. Bei Meldungsbeginn wird keine Pause gesendet.
- Beginn des Meldungsempfangs konfigurieren: Konfigurieren Sie das CM 1241 so, dass der Meldungsempfang beginnt, wenn die Kommunikationsleitung mindestens 50 Bitzeiten (ca. 5 ms bei 9600 Baud = 50 * 1/9600) inaktiv ist:



- Ende des Meldungsempfangs konfigurieren: Konfigurieren Sie das CM 1241 so, dass der Meldungsempfang beendet wird, wenn maximal 100 Byte oder ein Zeilenschaltungszeichen (10 dezimal oder a hexadezimal) empfangen wurden. Als Endezeichenfolge sind maximal fünf Endezeichen in Folge zulässig. An der fünften Stelle der Folge befindet sich das Zeilenschaltungszeichen. Die vorhergehenden vier Endezeichen der Folge sind "nicht relevant" oder nicht ausgewählte Zeichen. Das CM 1241 wertet die Zeichen "nicht relevant" nicht aus, erwartet jedoch ein Zeilenschaltungszeichen mit vorhergehender 0 oder weiteren Zeichen "nicht relevant" zur Kennzeichnung des Meldungsendes.

13.3.5.2 Betriebsarten RS422 und RS485

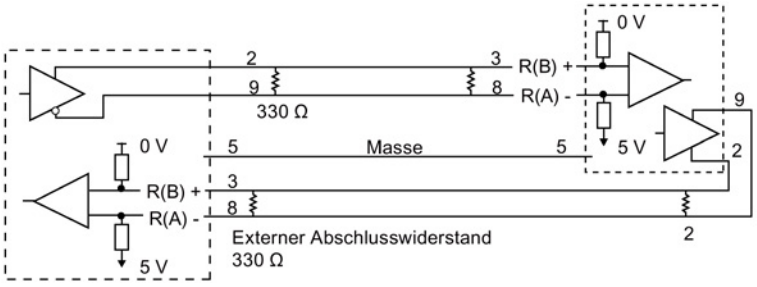
RS422 konfigurieren

Im RS422-Modus gibt es, abhängig von Ihrer Netzwerkkonfiguration, drei Betriebsarten. Wählen Sie je nach den Geräten in Ihrem Netzwerk eine dieser Betriebsarten aus. Die verschiedenen Einstellungen für "Ausgangszustand Empfangsleitung" beziehen sich auf die im Folgenden dargestellten Fälle.

- Vollduplex (RS422), Vierdrahtmodus (Punkt-zu-Punkt-Verbindung): Wählen Sie diese Option aus, wenn Ihr Netzwerk zwei Geräte umfasst. Für "Ausgangszustand Empfangsleitung":
  - Wählen Sie "Ohne", wenn Sie die Vorspannung und den Abschluss vorgeben (Fall 3).
  - Wählen Sie "Vorspannung in Vorwärtsrichtung", um interne Vorspannung und Abschluss zu verwenden (Fall 2).
  - Wählen Sie "Vorspannung in Sperrrichtung", um interne Vorspannung und Abschluss zu verwenden und die Kabelbrucherkennung für beide Geräte zu aktivieren (Fall 1).
- Vollduplex (RS422), Vierdrahtmodus (Multipoint-Master): Wählen Sie diese Option für das Mastergerät aus, wenn Ihr Netzwerk einen Master und mehrere Slaves umfasst. Für "Ausgangszustand Empfangsleitung":
  - Wählen Sie "Ohne", wenn Sie die Vorspannung und den Abschluss vorgeben (Fall 3).
  - Wählen Sie "Vorspannung in Vorwärtsrichtung", um interne Vorspannung und Abschluss zu verwenden (Fall 2).
  - Kabelbrucherkennung ist in dieser Betriebsart nicht möglich.
- Vollduplex (RS422), Vierdrahtmodus (Multipoint-Slave): Wählen Sie diese Option für alle Slavegeräte aus, wenn Ihr Netzwerk einen Master und mehrere Slaves umfasst. Für "Ausgangszustand Empfangsleitung":
  - Wählen Sie "Ohne", wenn Sie die Vorspannung und den Abschluss vorgeben (Fall 3).
  - Wählen Sie "Vorspannung in Vorwärtsrichtung", um interne Vorspannung und Abschluss zu verwenden (Fall 2).
  - Wählen Sie "Vorspannung in Sperrrichtung", um interne Vorspannung und Abschluss zu verwenden und die Kabelbrucherkennung für die Slaves zu aktivieren (Fall 1).

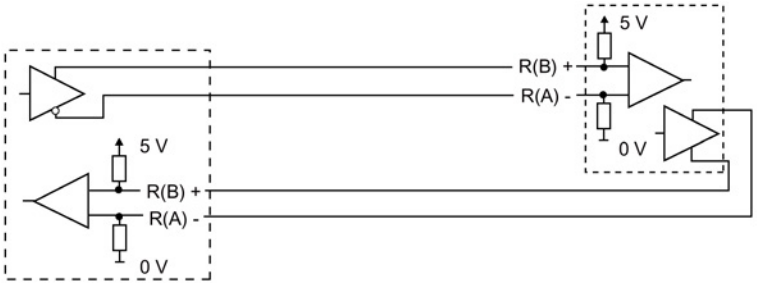
Fall 1: RS422 mit Kabelbrucherkennung

- Betriebsart: RS422
- Ausgangszustand Empfangsleitung: Vorspannung in Sperrrichtung (Vorspannung mit R(A) > R(B) > 0 V)
- Kabelbruch: Kabelbrucherkennung aktiviert (Sender immer aktiv)



Fall 2: RS422 ohne Kabelbrucherkennung, Vorspannung in Vorwärtsrichtung

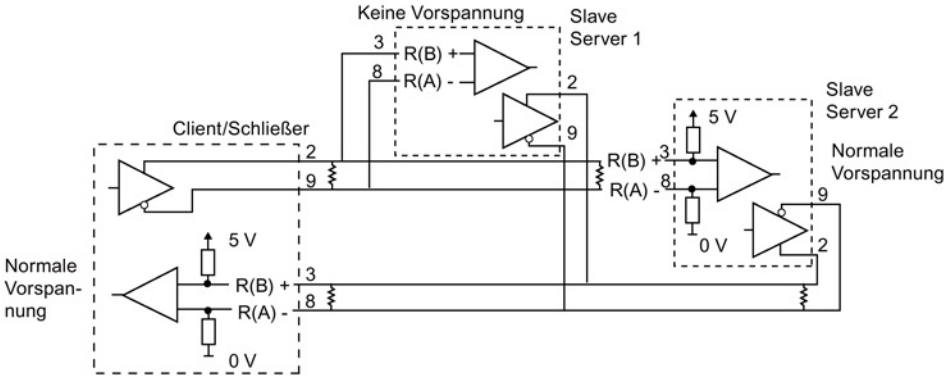
- Betriebsart: RS422
- Ausgangszustand Empfangsleitung: Vorspannung in Vorwärtsrichtung (Vorspannung mit R(B) > R(A) > 0 V)
- Kabelbruch: Keine Kabelbrucherkennung (Sender nur beim Senden aktiviert)



Fall 3: RS422: Keine Kabelbrucherkennung, keine Vorspannung

- Betriebsart: RS422
- Ausgangszustand Empfangsleitung: Keine Vorspannung
- Kabelbruch: Keine Kabelbrucherkennung (Sender nur beim Senden aktiviert)

Vorspannung und Abschluss werden vom Anwender an den Endknoten des Netzwerks bereitgestellt.



RS485 konfigurieren

Im RS485-Modus gibt es nur eine Betriebsart. Die verschiedenen Einstellungen für "Ausgangszustand Empfangsleitung" beziehen sich auf die im Folgenden dargestellten Fälle.

- Halbduplex (RS485), Zweidrahtmodus. Für "Ausgangszustand Empfangsleitung":
  - Wählen Sie "Ohne", wenn Sie die Vorspannung und den Abschluss vorgeben (Fall 5).
  - Wählen Sie "Vorspannung in Vorwärtsrichtung", um interne Vorspannung und Abschluss zu verwenden (Fall 4).

Fall 4: RS485: Vorspannung in Vorwärtsrichtung

- Betriebsart: RS485
- Ausgangszustand Empfangsleitung: Vorspannung in Vorwärtsrichtung (Vorspannung mit R(B) > R(A) > 0 V)

Fall 5: RS485: Keine Vorspannung (externe Vorspannung)

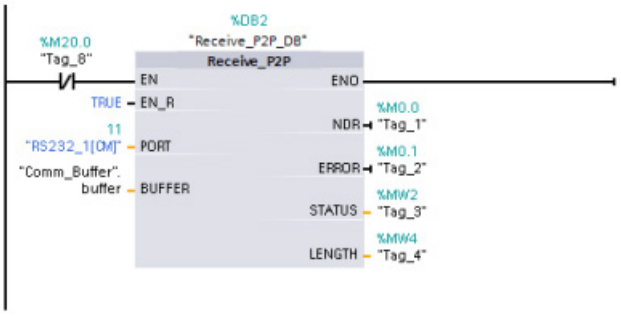
- Betriebsart: RS485
- Ausgangszustand Empfangsleitung: Keine Vorspannung (externe Vorspannung erforderlich)

13.3.5.3 STEP 7-Programm programmieren

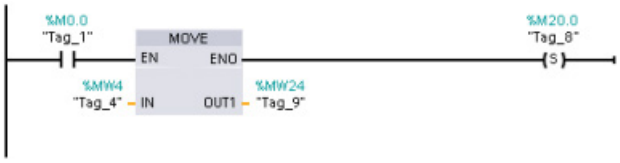
Das Beispielprogramm verwendet einen globalen Datenbaustein für den Kommunikationspuffer, eine Anweisung RCV_PTP (Seite 1033) für den Empfang von Daten vom Terminalemulator und eine Anweisung SEND_PTP (Seite 1030) zum Zurücksenden des Pufferechos an den Terminalemulator. Um das Beispiel zu programmieren, fügen Sie die Datenbausteinkonfiguration und den Hauptprogrammbaustein OB 1 wie im Folgenden beschrieben ein.

Globaler Datenbaustein "Comm_Buffer": Erstellen Sie einen globalen Datenbaustein (DB) und nennen Sie ihn "Comm_Buffer". Erstellen Sie einen Wert im Datenbaustein, nämlich "buffer", mit dem Datentyp "Array [0 .. 99] of byte".

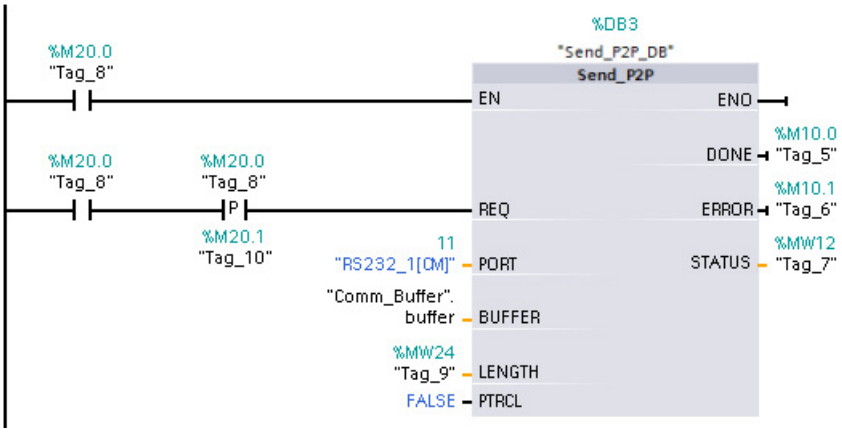
Netzwerk 1: Aktivieren Sie die Anweisung RCV_PTP immer, wenn SEND_PTP nicht aktiv ist. Tag_8 in MW20.0 zeigt in Netzwerk 4 an, wenn der Sendevorgang abgeschlossen ist und wenn das Kommunikationsmodul damit für den Meldungsempfang bereit ist.



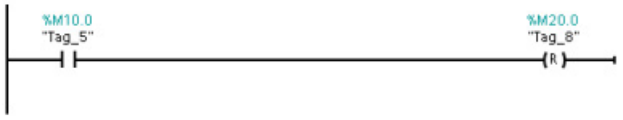
Netzwerk 2: Erstellen Sie mit dem von der Anweisung RCV_PTP gesetzten NDR Wert (Tag_1 in M0.0) eine Kopie von der empfangenen Anzahl Bytes und setzen Sie einen Merker (Tag_8 in M20.0), um die Anweisung SEND_PTP auszulösen.



Netzwerk 3: Aktivieren Sie die Anweisung SEND_PTP, wenn der Merker M20.0 gesetzt ist. Mit diesem Merker setzen Sie auch den Eingang REQ einen Zyklus lang auf WAHR. Der Eingang REQ teilt der Anweisung SEND_PTP mit, dass eine neue Anforderung zu übertragen ist. Der Eingang REQ darf nur während einer Ausführung von SEND_PTP auf WAHR gesetzt sein. Die Anweisung SEND_PTP wird in jedem Zyklus ausgeführt, bis die Übertragung beendet ist. TDie Übertragung ist beendet, wenn das letzte Byte der Meldung vom CM 1241 übertragen wurde. Wenn die Übertragung beendet ist, wird der Ausgang DONE (Tag_5 in M10.0) während einer Ausführung von SEND_PTP auf WAHR gesetzt.



Netzwerk 4: Überwachen Sie den Ausgang DONE von SEND_PTP und setzen Sie den Übertragungsmerker (Tag_8 in M20.0) zurück, wenn der Übertragungsvorgang beendet ist. Wenn der Übertragungsmerker zurückgesetzt wird, wird die Anweisung RCV_PTP in Netzwerk 1 aktiviert, um die nächste Meldung zu empfangen.



13.3.5.4 Terminalemulator konfigurieren

Sie müssen den Terminalemulator einrichten, um das Beispielprogramm auszuführen. Sie können nahezu jeden Terminalemulator an Ihrem PC verwenden, z. B. HyperTerminal. Achten Sie darauf, dass der Terminalemulator ausgeschaltet ist, bevor Sie die Einstellungen wie folgt ändern:

1. Legen Sie fest, dass der Terminalemulator den RS232-Anschluss am PC verwendet (normalerweise COM1).
2. Konfigurieren Sie den Port für 9600 Baud, 8 Datenbits, keine Parität, 1 Stoppbit und keine Flusskontrolle.
3. Ändern Sie die Einstellungen des Terminalemulators, um ein ANSI-Terminal zu emulieren.
4. Konfigurieren Sie die ASCII-Einrichtung des Terminalemulators so, dass nach jeder Zeile eine Zeilenschaltung gesendet wird (nachdem der Anwender die Eingabetaste drückt).
5. Geben Sie ein lokales Echo der Zeichen zurück, so dass der Terminalemulator anzeigt, was eingegeben wird.

13.3.5.5 Beispielprogramm ausführen

Um das Beispielprogramm auszuführen, gehen Sie folgendermaßen vor:

1. Laden Sie das STEP 7-Programm in die CPU und achten Sie darauf, dass diese im Betriebszustand RUN ist.
2. Klicken Sie im Terminalemulator auf die Schaltfläche zum Verbinden, um die Konfigurationsänderungen zu übernehmen, und öffnen Sie eine Terminalsitzung für das CM 1241.
3. Geben Sie am PC Zeichen ein und drücken Sie die Eingabetaste.

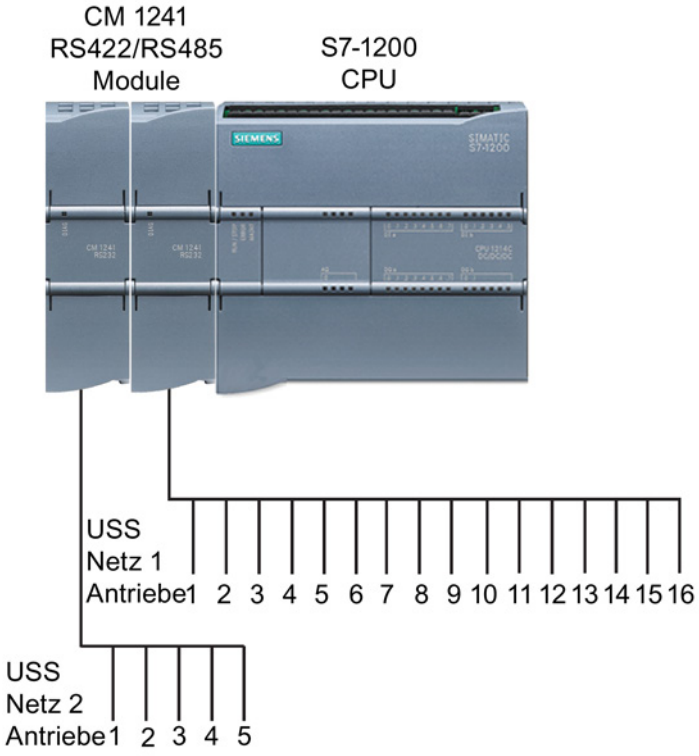
Der Terminalemulator sendet die Zeichen zum CM 1241 und zur CPU. Das CPU-Programm gibt dann ein Echo der Zeichen zum Terminalemulator zurück.

13.4 Kommunikation über die universelle serielle Schnittstelle (USS)

13.4.1 Übersicht

Die USS-Anweisungen steuern den Betrieb von Motorantrieben, die das Protokoll der universellen seriellen Schnittstelle (USS) unterstützen. Mit den USS-Anweisungen können Sie über RS485-Verbindungen mit mehreren Antrieben mit CM 1241 RS485-Kommunikationsmodulen oder einem CB 1241 RS485-Kommunikationsboard kommunizieren. In einer S7-1200 CPU können bis zu drei CM 1241 RS422/RS485-Module und ein CB 1241 RS485-Board eingebaut werden. Jeder RS485-Port kann bis zu sechzehn Antriebe betreiben.

Das USS-Protokoll nutzt ein Master/Slave-Netzwerk für die Kommunikation über einen seriellen Bus. Der Master verwendet einen Adressparameter, um eine Meldung an einen ausgewählten Slave zu senden. Ein Slave selbst kann niemals senden, ohne dafür zuvor eine Anforderung zu erhalten. Die direkte Meldungsübertragung zwischen den einzelnen Slaves ist nicht möglich. Die USS-Kommunikation funktioniert im Halbduplex-Betrieb. Die folgende USS-Abbildung zeigt ein Netzwerkdiagramm für eine Beispielanwendung eines Antriebs.



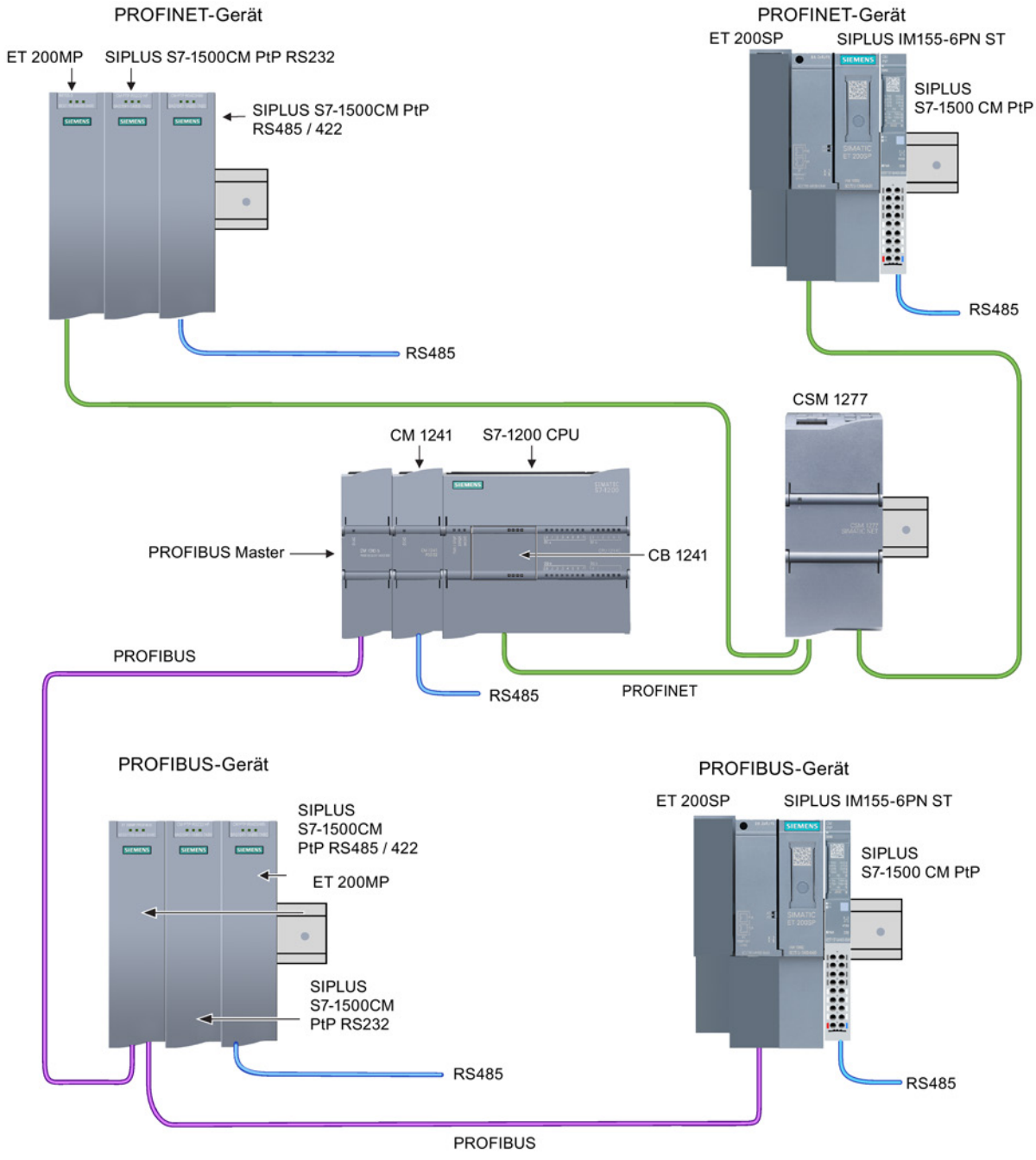
USS-Kommunikation über PROFIBUS oder PROFINET

Version V4.1 der S7-1200 CPU in Verbindung mit STEP 7 V13 SP1 erweitert die USS-Fähigkeit zur Nutzung eines dezentralen PROFINET- oder PROFIBUS-Peripheriebaugruppenträgers für die Kommunikation mit verschiedenen Geräten (RFID-Lesegeräte, GPS und andere):

- PROFINET (Seite 636): Die Ethernet-Schnittstelle der S7-1200 CPU wird mit einem PROFINET Schnittstellenmodul verbunden. Über PtP-Kommunikationsmodule im Baugruppenträger mit dem Schnittstellenmodul ist dann die serielle Kommunikation zu den PtP-Geräten möglich.
- PROFIBUS (Seite 782): Ein PROFIBUS-Kommunikationsmodul wird an der linken Seite des Baugruppenträgers mit der S7-1200 CPU eingesteckt. Das PROFIBUS-Kommunikationsmodul wird mit einem Baugruppenträger verbunden, der ein PROFIBUS-Schnittstellenmodul enthält. Über PtP-Kommunikationsmodule im Baugruppenträger mit dem Schnittstellenmodul ist dann die serielle Kommunikation zu den PtP-Geräten möglich.

Die S7-1200 unterstützt daher zwei Sätze PtP-Anweisungen:

- Ältere USS-Anweisungen (Seite 1042): Diese USS-Anweisungen wurden vor der Version V4.0 der S7-1200 benutzt und ermöglichen nur die serielle Kommunikation mit Hilfe eines Kommunikationsmoduls CM 1241 oder eines Kommunikationsboards CB 1241.
- USS-Anweisungen (Seite 959): Diese USS-Anweisungen stellen alle Funktionen der alten Anweisungen bereit und zusätzlich die Möglichkeit zur Verbindung mit der dezentralen PROFINET- und PROFIBUS-Peripherie. Die PtP-Anweisungen ermöglichen das Konfigurieren der Kommunikation zwischen den PtP-Kommunikationsmodulen im Baugruppenträger mit der dezentralen Peripherie und den PtP-Geräten.



Hinweis

Ab der Version V4.1 der S7-1200 können PtP-Anweisungen für alle Arten der PtP-Kommunikation verwendet werden: seriell, seriell über PROFINET und seriell über PROFIBUS. STEP 7 stellt die alten PtP-Anweisungen nur zur Unterstützung vorhandener Programme bereit. Die alten Anweisungen funktionieren noch bei allen S7-1200 CPUs. Es ist nicht nötig, ältere Programme auf die neuen Anweisungen zu konvertieren.

13.4.2 Version der USS-Anweisungen auswählen

Es gibt zwei Versionen der USS-Anweisungen in STEP 7:

- Version 2.0 (alte Anweisungen) war anfangs in STEP 7 Basic/Professional V13 verfügbar.
- Version 2.1 ist in STEP 7 Basic/Professional V13, SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Sie können nicht beide Versionen der Anweisungen bei demselben Modul verwenden, Sie können jedoch für zwei verschiedene Module unterschiedliche Versionen der Anwendungen einsetzen.

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.

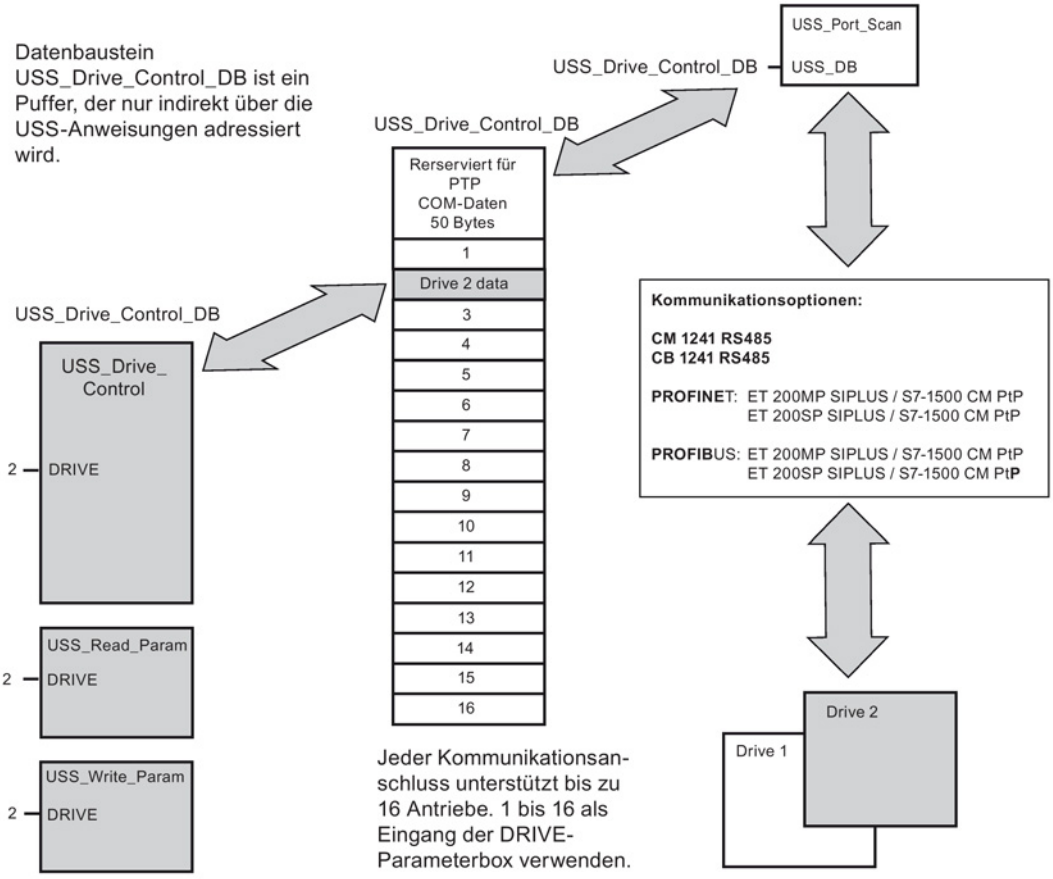
Um die Version einer USS-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine USS-Anweisung über das Anweisungsverzeichnis in Ihr Programm platzieren, wird je nach der ausgewählten USS-Anweisung eine neue FB- oder FC-Instanz in der Projektnavigation angelegt. Die neue FB- oder FC-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer USS-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation die Instanz eines USS-FBs oder FCs aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der USS-Anweisung anzuzeigen.

13.4.3 Voraussetzungen für den Einsatz des USS-Protokolls

Die vier USS-Anweisungen nutzen zwei Funktionsbausteine (FB) und zwei Funktionen (FC) für die Unterstützung des USS-Protokolls. Für jedes USS-Netzwerk wird ein Instanz-Datenbaustein (DB) USS_Port_Scan verwendet. Der Instanz-Datenbaustein USS_Port_Scan enthält temporäre Speicher und Puffer für alle Antriebe im USS-Netzwerk. Die USS-Anweisungen nutzen die Informationen in diesem Datenbaustein gemeinsam.



Alle Antriebe (max. 16), die an einen RS485-Port angeschlossen sind, sind Teil desselben USS-Netzwerks. Alle Antriebe, die an einen anderen RS485-Port angeschlossen sind, sind Teil eines anderen USS-Netzwerks. Jedes USS-Netzwerk wird mithilfe eines eindeutigen Datenbausteins verwaltet. Alle Anweisungen, die zu einem USS-Netzwerk gehören, müssen diesen Datenbaustein gemeinsam nutzen. Dies umfasst alle Anweisungen USS_Drive_Control, USS_Port_Scan, USS_Read_Param und USS_Write_Param für die Steuerung aller Antriebe in einem USS-Netzwerk.

Die Anweisung USS_Drive_Control ist ein Funktionsbaustein (FB). Wenn Sie die Anweisung USS_Drive_Control in den Programmiereditor einfügen, werden Sie im Dialog "Aufrufoptionen" aufgefordert, einen DB für diesen FB zuzuweisen. Wenn es sich um die erste Anweisung USS_Drive_Control in diesem Programm für dieses USS-Netzwerk handelt, können Sie die DB-Standardzuweisung übernehmen (oder ggf. den Namen ändern), und der neue DB wird für Sie erstellt. Wenn es sich jedoch nicht um die erste Anweisung USS_Drive_Control für diesen Kanal handelt, müssen Sie im Dialog "Aufrufoptionen" in der Klappliste den DB auswählen, der diesem USS-Netzwerk bereits zuvor zugewiesen wurde.

Die Anweisung USS_Port_Scan ist ein Funktionsbaustein (FB) und steuert die Kommunikation zwischen der CPU und den Antrieben über den Punkt-zu-Punkt(PtP)-RS485-Kommunikationsport. Bei jedem Aufruf dieses FB wird eine Kommunikation mit einem Antrieb bearbeitet. Ihr Programm muss diesen FB schnell genug aufrufen, so dass die Antriebe keine Zeitüberschreitung melden. Dieser FB kann aus dem Zyklus-OB des Hauptprogramms oder aus einem beliebigen Alarm-OB aufgerufen werden.

Anweisungen USS_Read_Param und USS_Write_Param sind beide Funktionen (FCs). Wenn Sie diese FCs im Editor einfügen, wird kein DB zugewiesen. Stattdessen müssen Sie dem Eingang USS_DB dieser Anweisungen den jeweiligen DB zuweisen. Doppelklicken Sie auf das Parameterfeld und klicken Sie dann auf das Symbol, um die verfügbaren DBs anzuzeigen.

In der Regel wird der FB USS_Port_Scan aus einem Weckalarm-OB aufgerufen. Stellen Sie die Zykluszeit des Weckalarm-OBs etwa auf die Hälfte eines Mindestaufrufintervalls ein (beispielsweise sollte für die Kommunikation mit 1200 Baud eine Zykluszeit von maximal 350 ms verwendet werden).

Der Funktionsbaustein USS_Drive_Control gibt Ihrem Programm Zugriff auf einen angegebenen Antrieb im USS-Netzwerk. Seine Ein- und Ausgänge entsprechen den Zuständen und den Bedienfunktionen des Antriebs. Sind 16 Antriebe im Netzwerk vorhanden, so muss USS_Drive_Control in Ihrem Programm mindestens 16 mal aufgerufen werden, also jeweils einmal für jeden Antrieb. Wie schnell diese Bausteine aufgerufen werden, hängt von der erforderlichen Geschwindigkeit für die Steuerung des Antriebsbetriebs ab.

Sie können den FB USS_Drive_Control nur aus dem Zyklus-OB eines Hauptprogramms aufrufen.

⚠ VORSICHT

Beim Aufruf von USS-Anweisungen aus OBs zu beachten

Rufen Sie USS_Drive_Control, USS_Read_Param und USS_Write_Param nur aus einem Zyklus-OB des Hauptprogramms auf. Der FB USS_Port_Scan kann aus einem beliebigen OB aufgerufen werden, üblicherweise wird er aus einem Weckalarm-OB aufgerufen.

Verwenden Sie die Anweisungen USS_Drive_Control, USS_Read_Param und USS_Write_Param nicht in einem OB mit einer höheren Priorität als die entsprechende Anweisung USS_Port_Scan. Fügen Sie beispielsweise USS_Port_Scan nicht in einen OB des Hauptprogramms und USS_Read_Param in einen Weckalarm-OB ein. Wird die Unterbrechung der Ausführung von USS_Port_Scan nicht verhindert, kann es zu unerwarteten Fehlern kommen, die zu Verletzungen führen können.

Mit den Funktionen USS_Read_Param und USS_Write_Param werden die Betriebsparameter des entfernten Antriebs gelesen und geschrieben. Diese Parameter steuern die interne Funktionsweise des Antriebs. Eine Definition dieser Parameter finden Sie im Handbuch des Antriebs. Ihr Programm kann eine beliebige Anzahl dieser Funktionen enthalten, es kann jedoch immer nur eine Lese- oder Schreibanforderung für einen Antrieb aktiv sein. Sie können die FCs USS_Read_Param und USS_Write_Param nur aus dem Zyklus-OB eines Hauptprogramms aufrufen.

Zeit für die Kommunikation mit dem Antrieb berechnen

Die Kommunikation mit dem Antrieb läuft asynchron zum Zyklus der S71200 ab. Die S7-1200 durchläuft üblicherweise mehrere Zyklen, bevor die Kommunikation mit einem Antrieb beendet ist.

Das Intervall USS_Port_Scan ist die Zeit, die für eine Transaktion des Antriebs erforderlich ist. Die folgende Tabelle zeigt die Mindestintervalle für USS_Port_Scan für jede Baudrate der Kommunikation. Wenn Sie den FB USS_Port_Scan häufiger aufrufen, als es das USS_Port_Scan-Intervall vorgibt, wird die Anzahl der Transaktionen nicht erhöht. Das Timeout-Intervall des Antriebs ist die Zeitdauer, die für eine Transaktion zur Verfügung steht, wenn zur Fertigstellung der Transaktion aufgrund von Kommunikationsfehlern 3 Versuche nötig sind. Standardmäßig führt die Bibliothek für das USS-Protokoll bei jeder Transaktion bis zu 2 Wiederholungen durch.

Tabelle 13- 38 Zeitbedarf berechnen

Baudrate	Berechnetes Mindestintervall für Aufruf von USS_Port_Scan (ms)	Intervall-Timeout für Antriebsmeldung pro Antrieb (ms)
1200	790	2370
2400	405	1215
4800	212.5	638
9600	116.3	349
19200	68.2	205
38400	44.1	133
57600	36.1	109
115200	28.1	85

13.4.4 USS-Anweisungen

13.4.4.1 Anweisung USS_Port_Scan (Kommunikation über USS-Netzwerk bearbeiten)

Tabelle 13- 39 Anweisung USS_Port_Scan

KOP/FUP	SCL	Beschreibung

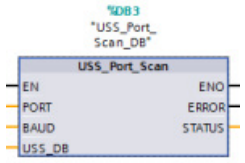
	`USS_Port_Scan(`
	`  PORT:=_uint_in_,`
	`  BAUD:=_dint_in_,`
	`  ERROR=>_bool_out_,`
	`  STATUS=>_word_out_,`
	`  USS_DB:=_fbtref_inout_);`
		Die Anweisung USS_Port_Scan bearbeitet die Kommunikation über ein USS-Netzwerk.

Tabelle 13- 40 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
PORT	IN	Port	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen.
BAUD	IN	DInt	Die Baudrate für die USS-Kommunikation.
USS_DB	INOUT	USS_BASE	Der Name des Instanz-DBs, der erstellt und initialisiert wird, wenn eine Anweisung USS_Drive_Control in Ihr Programm eingefügt wird.
ERROR	OUT	Bool	Wenn WAHR, weist dieser Ausgang darauf hin, dass ein Fehler aufgetreten und Ausgang STATUS gültig ist.
STATUS	OUT	Word	Der Zustandswert der Anforderung zeigt des Ergebnis des Zyklus oder der Initialisierung an. Weitere Informationen stehen für einige Statuscodes in der Variablen "USS_Extended_Error" zur Verfügung.

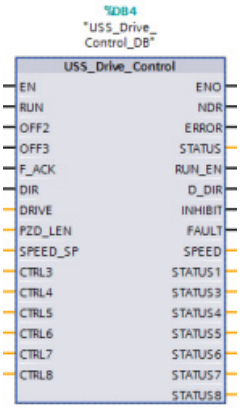
Normalerweise ist nur eine Anweisung USS_Port_Scan pro PtP-Kommunikationsport im Programm vorhanden und jeder Aufruf dieses Funktionsbausteins (FB) steuert eine Übertragung zu oder von einem einzigen Antrieb. Alle USS-Funktionen, die einem USS-Netzwerk und einem PtP-Kommunikationsport zugewiesen sind, müssen den gleichen Instanz-DB nutzen.

Ihr Programm muss die Anweisung USS_Port_Scan so oft ausführen, dass keine Zeitüberschreitung im Antrieb auftritt. USS_Port_Scan wird üblicherweise aus einem Weckalarm-OB aufgerufen, um Zeitüberschreitungen der Antriebe zu verhindern und die letzten USS-Datenaktualisierungen für Aufrufe von USS_Drive_Control verfügbar zu halten.

13.4.4.2 Anweisung USS_Drive_Control (Daten mit Antrieb tauschen)

Tabelle 13- 41 Anweisung USS_Drive_Control

KOP/FUP	SCL	Beschreibung

	`"USS_Drive_Control_DB"(`
	`RUN:=_bool_in_,`
	`OFF2:=_bool_in_,`
	`OFF3:=_bool_in_,`
	`F_ACK:=_bool_in_,`
	`DIR:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PZD_LEN:=_usint_in_,`
	`SPEED_SP:=_real_in_,`
	`CTRL3:=_word_in_,`
	`CTRL4:=_word_in_,`
	`CTRL5:=_word_in_,`
	`CTRL6:=_word_in_,`
	`CTRL7:=_word_in_,`
	`CTRL8:=_word_in_,`
	`NDR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`RUN_EN=>_bool_out_,`
	`D_DIR=>_bool_out_,`
	`INHIBIT=>_bool_out_,`
	`FAULT=>_bool_out_,`
	`SPEED=>_real_out_,`
	`STATUS1=>_word_out_,`
	`STATUS3=>_word_out_,`
	`STATUS4=>_word_out_,`
	`STATUS5=>_word_out_,`
	`STATUS6=>_word_out_,`
	`STATUS7=>_word_out_,`
	`STATUS8=>_word_out_);`
		Die Anweisung USS_Drive_Control tauscht Daten mit einem Antrieb aus, indem Anfragemeldungen erzeugt und die Antwortmeldungen des Antriebs ausgewertet werden. Es sollte für jeden Antrieb ein eigener Funktionsbaustein verwendet werden, jedoch müssen alle USS-Funktionen, die einem USS-Netzwerk und einem PtP-Kommunikationsport zugewiesen sind, den gleichen Instanz-Datenbaustein verwenden. Sie müssen den DB-Namen eingeben, wenn Sie die erste Anweisung USS_Drive_Control einfügen. Dann verweisen Sie auf diesen DB, der beim Einfügen der ersten Anweisung angelegt wurde.
		STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

[1] KOP und FUP: Erweitern Sie die Box, um alle Parameter anzuzeigen. Klicken Sie dazu auf den unteren Bereich der Box. Die Parameteranschlüsse, die grau dargestellt sind, sind optional, eine Parametrierung ist nicht erforderlich.

Tabelle 13- 42 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
RUN	IN	Bool	Startbit des Antriebs: Ist dieser Parameter WAHR, so ermöglicht dieser Eingang den Betrieb des Antriebs mit der voreingestellten Drehzahl. Wenn RUN im Betrieb des Antriebs nach Falsch wechselt, läuft der Motor bis zum Stillstand aus. Dieses Verhalten unterscheidet sich von der Abschaltung der Spannungsversorgung (OFF2) und vom Bremsen des Motors (OFF3).
OFF2	IN	Bool	Bit "Zum Stillstand auslaufen": Ist dieser Parameter FALSCH, so veranlasst dieses Bit das Auslaufen des Antriebs, ohne zu bremsen.

Parameter und Datentyp		Datentyp	Beschreibung
OFF3	IN	Bool	Schnelles Stoppbit: Ist dieser Parameter FALSCH, so verursacht dieses Bit einen schnellen Halt durch Abbremsen des Antriebs.
F_ACK	IN	Bool	Fehlerquittierungsbit: Mit diesem Bit wird das Fehlerbit eines Antriebs zurückgesetzt. Das Bit wird nach dem Löschen des Fehlers gesetzt und der Antrieb erkennt damit, dass der vorherige Fehler nicht mehr gemeldet werden muss.
DIR	IN	Bool	Richtungssteuerung des Antriebs: Dieses Bit wird gesetzt, wenn der Antrieb in Vorwärtsrichtung laufen soll (wenn SPEED_SP positiv ist).
DRIVE	IN	USInt	Adresse des Antriebs: Dieser Eingang ist die Adresse des USS-Antriebs. Der gültige Bereich liegt zwischen Antrieb 1 und Antrieb 16.
PZD_LEN	IN	USInt	Wortlänge: Dies ist die Anzahl der PZD-Datenwörter. Gültige Werte sind 2, 4, 6 oder 8 Wörter. Der Standardwert ist 2.
SPEED_SP	IN	Real	Drehzahlsollwert: Dies ist die Drehzahl des Antriebs prozentual zur konfigurierten Frequenz. Ein positiver Wert bedeutet, dass der Antrieb vorwärts läuft (wenn DIR wahr ist). Gültig ist der Bereich von 200,00 bis -200,00.
CTRL3	IN	Word	Steuerwort 3: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL4	IN	Word	Steuerwort 4: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL5	IN	Word	Steuerwort 5: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL6	IN	Word	Steuerwort 6: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL7	IN	Word	Steuerwort 7: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL8	IN	Word	Steuerwort 8: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
NDR	OUT	Bool	Neue Daten bereit: Ist dieser Parameter wahr, so meldet das Bit, dass am Ausgang Daten einer neuen Kommunikationsanforderung bereitstehen.
ERROR	OUT	Bool	Fehler aufgetreten: Wenn WAHR, weist dies darauf hin, dass ein Fehler aufgetreten und Ausgang STATUS gültig ist. Alle anderen Ausgänge werden bei einem Fehler auf Null gesetzt. Kommunikationsfehler werden nur an den Ausgängen ERROR und STATUS der Anweisung USS_Port_Scan gemeldet.
STATUS	OUT	Word	Der Zustandswert der Anforderung zeigt des Ergebnis des Zyklus an. Dies ist kein vom Antrieb ausgegebenes Zustandswort.
RUN_EN	OUT	Bool	Betrieb freigegeben: Dieses Bit meldet, ob der Antrieb läuft.
D_DIR	OUT	Bool	Antriebsrichtung: Dieses Bit meldet, ob der Antrieb vorwärts läuft.
INHIBIT	OUT	Bool	Antrieb gesperrt: Dieses Bit meldet den Zustand des Sperrbits für den Antrieb.

Parameter und Datentyp		Datentyp	Beschreibung
FAULT	OUT	Bool	Antriebsfehler: Dieses Bit meldet, dass im Antrieb ein Fehler aufgetreten ist. Sie müssen die Störung beheben und Bit F_ACK setzen, um dieses Bit zu löschen.
SPEED	OUT	Real	Istwert Antriebsdrehzahl (skalierter Wert von Zustandswort 2 des Antriebs): Dies ist die Drehzahl des Antriebs prozentual zur konfigurierten Drehzahl.
STATUS1	OUT	Word	Zustandswort 1 des Antriebs: Dieser Wert enthält feste Zustandsbits eines Antriebs.
STATUS3	OUT	Word	Zustandswort 3 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS4	OUT	Word	Zustandswort 4 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS5	OUT	Word	Zustandswort 5 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS6	OUT	Word	Zustandswort 6 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS7	OUT	Word	Zustandswort 7 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS8	OUT	Word	Zustandswort 8 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.

Wenn die erste Ausführung von USS_Drive_Control erfolgt, wird der von der USS-Adresse (Parameter DRIVE) angegebene Antrieb im Instanz-DB initialisiert. Nach dieser Initialisierung können nachfolgende Anweisungen USS_Port_Scan die Kommunikation mit dem Antrieb an dieser Antriebsnummer beginnen.

Wenn Sie die Antriebsnummer ändern, muss die CPU zunächst in STOP und dann wieder in RUN versetzt werden, damit der Instanz-DB initialisiert wird. Die Eingangsparameter werden im USS-Sendepuffer konfiguriert und die Ausgänge werden, sofern vorhanden, aus einem "vorherigen" gültigen Antwortpuffer gelesen. Während der Ausführung der Anweisung USS_Drive_Control findet keine Datenübertragung statt. Nach der Ausführung von USS_Port_Scan kommunizieren die Antriebe. USS_Drive_Control konfiguriert nur die zu sendenden Meldungen und wertet Daten aus, die möglicherweise mit einer vorherigen Anforderung empfangen wurden.

Sie können die Drehrichtung des Antriebs entweder über den Eingang DIR (Bool) oder über das Vorzeichen (positiv oder negativ) am Eingang SPEED_SP (Real) steuern. Die folgende Tabelle erläutert, wie diese Eingänge zusammen funktionieren, um die Drehrichtung des Antriebs zu bestimmen, vorausgesetzt der Motor dreht vorwärts.

Tabelle 13- 43 Interaktion der Parameter SPEED_SP und DIR

SPEED_SP	DIR	Drehrichtung des Antriebs
Wert > 0	0	Rückwärts
Wert > 0	1	Vorwärts
Wert < 0	0	Vorwärts
Wert < 0	1	Rückwärts

13.4.4.3 Anweisung USS_Read_Param (Parameter aus dem Antrieb auslesen)

Tabelle 13- 44 Anweisung USS_Read_Param

KOP/FUP	SCL	Beschreibung

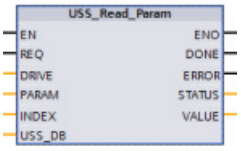
	`USS_Read_Param(REQ:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PARAM:=_uint_in_,`
	`INDEX:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`VALUE=>_variant_out_,`
	`USS_DB:=_fbtref_inout_);`
		Die Anweisung USS_Read_Param liest einen Parameter aus einem Antrieb. Alle USS-Funktionen, die einem USS-Netzwerk und einem PtP-Kommunikationsport zugewiesen sind, müssen den gleichen Datenbaustein verwenden.
		USS_Read_Param muss aus einem Zyklus-OB des Hauptprogramms aufgerufen werden.

Tabelle 13- 45 Datentypen für die Parameter

Parametertyp		Datentyp	Beschreibung
REQ	IN	Bool	Sendeanforderung: Ist REQ WAHR, so wird eine neue Leseanforderung benötigt. Dies wird ignoriert, wenn die Anforderung für diesen Parameter bereits ansteht.
DRIVE	IN	USInt	Adresse des Antriebs: DRIVE ist die Adresse des USS-Antriebs. Der gültige Bereich liegt zwischen Antrieb 1 und Antrieb 16.
PARAM	IN	UInt	Parameternummer: PARAM gibt an, welcher Antriebsparameter geschrieben wird. Der Bereich für diesen Parameter liegt zwischen 0 und 2047. Bei einigen Antrieben kann das höchstwertige Byte auf PARAM-Werte größer als 2047 zugreifen. Weitere Informationen für den Zugriff auf einen erweiterten Bereich finden Sie im Handbuch zu Ihrem Antrieb.
INDEX	IN	UInt	Parameterindex: INDEX gibt an, in welchen Antriebsparameterindex geschrieben werden soll. Es handelt sich um einen 16-Bit-Wert, bei dem das niederwertigste Byte der tatsächliche Indexwert ist, mit einem Bereich von (0 bis 255). Das höchstwertige Byte kann ebenfalls von dem Antrieb verwendet werden und ist antriebsspezifisch. Weitere Informationen finden Sie im Handbuch Ihres Antriebs.
USS_DB	INOUT	USS_BASE	Der Name des Instanz-DBs, der erstellt und initialisiert wird, wenn eine Anweisung USS_Drive_Control in Ihr Programm eingefügt wird.
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	Dies ist der Wert des Parameters, der gelesen wurde und er ist nur gültig, wenn das Bit DONE wahr ist.
DONE[1]	OUT	Bool	Ist dieser Parameter WAHR, so steht am Ausgang VALUE der zuvor angeforderte Wert des Leseparameters an. Dieses Bit wird gesetzt, wenn die Anweisung USS_Drive_Control die Leseantwort des Antriebs erkennt. Dieses Bit wird zurückgesetzt, wenn: Sie die Antwortdaten über eine andere Abfrage USS_Read_Param anfordern oder beim zweiten der nächsten beiden Aufrufe von USS_Drive_Control.

Parametertyp		Datentyp	Beschreibung
ERROR	OUT	Bool	Fehler aufgetreten: Wenn WAHR, weist ERROR darauf hin, dass ein Fehler aufgetreten und Ausgang STATUS gültig ist. Alle anderen Ausgänge werden bei einem Fehler auf Null gesetzt. Kommunikationsfehler werden nur an den Ausgängen ERROR und STATUS der Anweisung USS_Port_Scan gemeldet.
STATUS	OUT	Word	STATUS gibt das Ergebnis der Leseanforderung an. Weitere Informationen stehen für einige Statuscodes in der Variablen "USS_Extended_Error" zur Verfügung.

[1] Das Bit DONE weist darauf hin, dass gültige Daten aus dem referenzierten Motorantrieb ausgelesen und an die CPU geliefert wurden. Es weist nicht darauf hin, dass die USS-Bibliothek in der Lage ist, sofort einen weiteren Parameter auszulesen. Eine leere PKW-Anforderung muss an den Motorantrieb gesendet und auch von der Anweisung quittiert werden, bevor der Parameterkanal zur Verwendung durch den jeweiligen Antrieb frei wird. Der sofortige Aufruf eines USS_Read_Param- oder USS_Write_Param -FC für den spezifischen Motorantrieb führt zu dem Fehler 0x818A.

13.4.4.4 Anweisung USS_Write_Param (Parameter im Antrieb ändern)

Hinweis

EEPROM-Schreibanweisungen (für den EEPROM in einem USS-Antrieb)

Übertreiben Sie die Verwendung der EEPROM-Schreiboperation nicht. Halten Sie die Anzahl der EEPROM-Schreiboperationen möglichst gering, um die Lebensdauer des EEPROM zu verlängern.

Tabelle 13- 46 Anweisung USS_Write_Param

KOP/FUP	SCL	Beschreibung

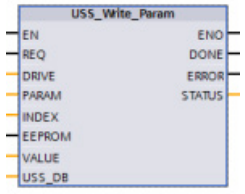
	`USS_Write_Param(REQ:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PARAM:=_uint_in_,`
	`INDEX:=_uint_in_,`
	`EEPROM:=_bool_in_,`
	`VALUE:=_variant_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`USS_DB:=_fbtref_inout_);`
		Die Anweisung USS_Write_Param ändert einen Parameter im Antrieb. Alle USS-Funktionen, die einem USS-Netzwerk und einem PtP-Kommunikationsport zugewiesen sind, müssen den gleichen Datenbaustein nutzen.
		USS_Write_Param muss aus dem Zyklus-OB eines Hauptprogramms aufgerufen werden.

Tabelle 13- 47 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Sendeanforderung: Ist REQ WAHR, so wird eine neue Schreibanforderung benötigt. Dies wird ignoriert, wenn die Anforderung für diesen Parameter bereits ansteht.
DRIVE	IN	USInt	Adresse des Antriebs: DRIVE ist die Adresse des USS-Antriebs. Der gültige Bereich liegt zwischen Antrieb 1 und Antrieb 16.
PARAM	IN	UInt	Parameternummer: PARAM gibt an, welcher Antriebsparameter geschrieben wird. Der Bereich für diesen Parameter liegt zwischen 0 und 2047. Bei einigen Antrieben kann das höchstwertige Byte auf PARAM-Werte größer als 2047 zugreifen. Weitere Informationen für den Zugriff auf einen erweiterten Bereich finden Sie im Handbuch zu Ihrem Antrieb.
INDEX	IN	UInt	Parameterindex: INDEX gibt an, in welchen Antriebsparameterindex geschrieben werden soll. Es handelt sich um einen 16-Bit-Wert, bei dem das niederwertigste Byte der tatsächliche Indexwert ist, mit einem Bereich von (0 bis 255). Das höchstwertige Byte kann ebenfalls von dem Antrieb verwendet werden und ist antriebsspezifisch. Weitere Informationen finden Sie im Handbuch Ihres Antriebs.
EEPROM	IN	Bool	Im EEPROM des Antriebs speichern: Wenn WAHR, wird die Transaktion eines Parameters zum Schreiben in den Antrieb im EEPROM des Antriebs gespeichert. Wenn FALSCH, so wird der geschriebene Wert nur temporär gespeichert und geht beim nächsten Einschalten des Antriebs verloren.
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	Wert des Parameters, in den geschrieben werden soll. Er muss beim Zustandswechsel von REQ gültig sein.
USS_DB	INOUT	USS_BASE	Der Name des Instanz-DBs, der erstellt und initialisiert wird, wenn eine Anweisung USS_Drive_Control in Ihr Programm eingefügt wird.
DONE[1]	OUT	Bool	Wenn WAHR, weist DONE darauf hin, dass der Eingang VALUE in den Antrieb geschrieben wurde. Dieses Bit wird gesetzt, wenn die Anweisung USS_Drive_Control die Schreibantwort des Antriebs erkennt. Dieses Bit wird zurückgesetzt, entweder wenn Sie die Antwortdaten über eine weitere Abfrage USS_Drive_Control anfordern, oder beim zweiten der nächsten beiden Aufrufe von USS_Drive_Control.
ERROR	OUT	Bool	Wenn WAHR, weist ERROR darauf hin, dass ein Fehler aufgetreten und Ausgang STATUS gültig ist. Alle anderen Ausgänge werden bei einem Fehler auf Null gesetzt. Kommunikationsfehler werden nur an den Ausgängen ERROR und STATUS der Anweisung USS_Port_Scan gemeldet.
STATUS	OUT	Word	STATUS gibt das Ergebnis der Schreibanforderung an. Weitere Informationen stehen für einige Statuscodes in der Variablen "USS_Extended_Error" zur Verfügung.

[1] Das Bit DONE weist darauf hin, dass gültige Daten aus dem referenzierten Motorantrieb ausgelesen und an die CPU geliefert wurden. Es weist nicht darauf hin, dass die USS-Bibliothek in der Lage ist, sofort einen weiteren Parameter auszulesen. Eine leere PKW-Anforderung muss an den Motorantrieb gesendet und auch von der Anweisung quittiert werden, bevor der Parameterkanal zur Verwendung durch den jeweiligen Antrieb frei wird. Der sofortige Aufruf eines USS_Read_Param- oder USS_Write_Param-FC für den spezifischen Motorantrieb führt zu dem Fehler 0x818A.

13.4.5 USS-Zustandscodes

Statuscodes der USS-Anweisung werden im Ausgang STATUS der USS-Funktionen ausgegeben.

Tabelle 13- 48 STATUS-Codes [1]

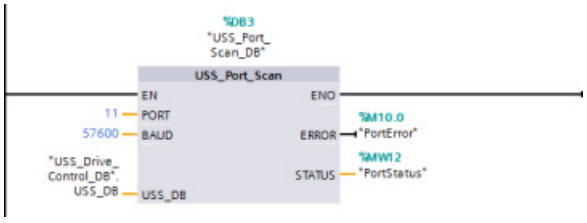
STATUS (W#16#....)	Beschreibung
0000	Kein Fehler
8180	Die Länge der Antwort des Antriebs entsprach nicht den vom Antrieb empfangenen Zeichen. Die Nummer des Antriebs, in dem der Fehler aufgetreten ist, wird in der Variablen "USS_Extended_Error" ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
8181	Der Parameter VALUE gehört nicht zum Datentyp Wort, Real oder Doppelwort.
8182	Der Anwender hat einen Parameterwert vom Typ Wort eingegeben und die Antwort vom Antrieb im Format Doppelwort oder Real empfangen.
8183	Der Anwender hat einen Parameterwert vom Typ Doppelwort oder Real eingegeben und die Antwort vom Antrieb im Format Wort empfangen.
8184	Das Antworttelegramm des Antriebs hatte eine falsche Prüfsumme. Die Nummer des Antriebs, in dem der Fehler aufgetreten ist, wird in der Variablen "USS_Extended_Error" ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
8185	Unzulässige Antriebsadresse (gültiger Adressbereich für Antriebe: 1 bis16)
8186	Der Drehzahlsollwert liegt außerhalb des gültigen Bereichs (gültiger Sollwertbereich für die Drehzahl: -200 % bis 200 %).
8187	Die falsche Antriebsnummer hat auf die gesendete Anforderung geantwortet. Die Nummer des Antriebs, in dem der Fehler aufgetreten ist, wird in der Variablen "USS_Extended_Error" ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
8188	Unzulässige PZD-Wortlänge angegeben (gültiger Bereich = 2, 4, 6 oder 8 Wörter)
8189	Unzulässige Baudrate angegeben.
818A	Der Anforderungskanal für Parameter wird von einer anderen Anforderung für diesen Antrieb verwendet.
818B	Der Antrieb hat nicht auf Anforderungen und Wiederholungen reagiert. Die Nummer des Antriebs, in dem der Fehler aufgetreten ist, wird in der Variablen "USS_Extended_Error" ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
818C	Der Antrieb hat einen erweiterten Fehler zu einer Parameteranforderung ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
818D	Der Antrieb hat einen Fehler "Unzulässiger Zugriff" zu einer Parameteranforderung ausgegeben. Weitere Informationen dazu, weshalb der Parameterzugriff begrenzt sein kann, finden Sie im Handbuch zu Ihrem Antrieb.
818E	Der Antrieb wurde nicht initialisiert. Dieser Fehlercode wird an USS_Read_Param oder USS_Write_Param ausgegeben, wenn die Anweisung USS_Drive_Control nicht mindestens einmal für diesen Antrieb aufgerufen wurde. Dadurch wird verhindert, dass die Initialisierung im ersten Zyklus von USS_Drive_Control eine anstehende Anforderung zum Lesen oder Schreiben von Parametern überschreibt, weil dabei der Antrieb als neuer Eintrag initialisiert wird. Um diesen Fehler zu beheben, rufen Sie die Anweisung USS_Drive_Control für diesen Antrieb auf.
80Ax-80Fx	Spezifische Fehler, die von den von der USS-Bibliothek aufgerufenen FBs für die PtP-Kommunikation zurückgegeben werden - Diese Fehlercodes werden von der USS-Bibliothek nicht verändert und sind in den Beschreibungen der PtP-Anweisung definiert.

[1] Neben den oben aufgeführten Fehlern der USS-Anweisungen können auch die zugrunde liegenden PtP-Kommunikationsanweisungen (Seite 915) Fehler zurückgeben.

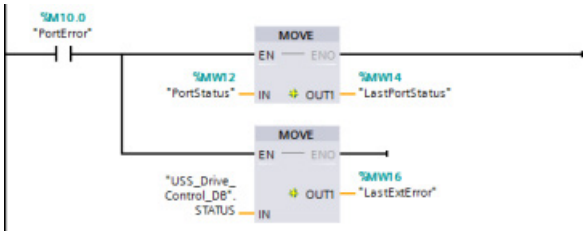
Für verschiedene STATUS-Codes werden weitere Informationen in der Variable "USS_Extended_Error" des Instanz-DB USS_Drive_Control zur Verfügung gestellt. Für die STATUS-Codes hexadezimal 8180, 8184, 8187 und 818B, enthält USS_Extended_Error die Antriebsnummer des Antriebs, bei dem der Kommunikationsfehler auftrat. Für STATUS-Code hexadezimal 818C enthält USS_Extended_Error einen Antriebsfehlercode, der bei der Verwendung einer Anweisung USS_Read_Param oder USS_Write_Param vom Antrieb ausgegeben wird.

Beispiel: Meldung von Kommunikationsfehlern

Kommunikationsfehler (STATUS = 16#818B) werden nur bei der Anweisung USS_Port_Scan und nicht bei der Anweisung USS_Drive_Control gemeldet. Beispiel: Wenn das Netzwerk nicht ordnungsgemäß beendet wird, kann ein Antrieb in RUN wechseln, doch die Anweisung USS_Drive_Control zeigt an allen Ausgangsparametern 0 an. In diesem Fall können Sie den Kommunikationsfehler nur über die Anweisung USS_Port_Scan erkennen. Weil dieser Fehler nur einen Zyklus lang sichtbar ist, müssen Sie entsprechende Erfassunglogik einfügen. Dies wird in dem folgenden Beispiel dargestellt. In diesem Beispiel werden, wenn das Fehlerbit der Anweisung USS_Port_Scan WAHR ist, die Werte STATUS und USS_Extended_Error im Speicherbereich der Merker abgelegt. Die Antriebsnummer wird in der Variablen USS_Extended_Error abgelegt, wenn der Wert des STATUS-Codes hexadezimal 8180, 8184, 8187 oder 818B ist.



Netzwerk 1 Der Portzustand "PortStatus" und die erweiterten Fehlercodewerte "USS_Drive_Control_DB".USS_Extended_Error sind nur einen Programmzyklus lang gültig. Die Werte müssen zur späteren Bearbeitung erfasst werden.



Netzwerk 2 Der "PortError"-Kontakt löst die Speicherung des "PortStatus"-Werts in "LastPortStatus" und des "USS_Drive_Control_DB".USS_Extended_Error -Werts in "LastExtError" aus.

Lese- und Schreibzugriff auf die internen Parameter eines Antriebs

USS-Antriebe unterstützen Lese- und Schreibzugriff auf die internen Parameter eines Antriebs. Diese Funktion ermöglicht die dezentrale Steuerung und Konfiguration des Antriebs. Zugriffe der Antriebsparameter können aufgrund von Fehlern wie Wert außerhalb des Bereichs oder unzulässige Anforderungen in der aktuellen Betriebsart des Antriebs fehlschlagen. Der Antrieb erzeugt einen Fehlercde, der in der Variablen "USS_Extended_Error" ausgegeben wird. Dieser Fehlercode gilt nur für die letzte Ausführung der Anweisung USS_Read_Param oder USS_Write_Param. Der Fehlercode des Antriebs wird in der Variablen "USS_Extended_Error" abgelegt, wenn der Wert von STATUS code hexadezimal 818C ist. Der Fehlercodewert von USS_Extended_Error richtet sich nach dem Antriebsmodell. Eine Beschreibung der erweiterten Fehlercodes von Lese- und Schreibfunktionen für Parameter finden Sie im Handbuch des Antriebs.

13.4.6 Allgemeine Voraussetzungen für die USS-Antriebseinrichtung

Für die USS-Antriebseinrichtung gelten die folgenden allgemeinen Voraussetzungen:

- Für die Antriebe muss die Verwendung von 4 PKW-Wörtern eingerichtet werden.
- Die Antriebe können für 2, 4, 6 oder 8 PZD-Wörter konfiguriert werden.
- Die Anzahl der PZD-Wörter im Antrieb muss dem Eingang PZD_LEN der Anweisung USS_Drive_Control des Antriebs entsprechen.
- Die Baudrate aller Antriebe muss dem Eingang BAUD der Anweisung USS_Port_Scan entsprechen.
- Der Antrieb muss für die Fernsteuerung eingerichtet werden.
- Für den Frequenzsollwert an der COM-Verbindung des Antriebs muss USS festgelegt werden.
- Für die Antriebsadresse muss 1 bis 16 festgelegt sein. Diese Adresse muss dem Eingang DRIVE am Baustein USS_Drive_Control des Antriebs entsprechen.
- Für die Richtungssteuerung des Antriebs muss die Verwendung der Polarität des Antriebssollwerts eingerichtet werden.
- Das RS485-Netzwerk muss ordnungsgemäß abgeschlossen sein.

13.4.7 Beispiel: Allgemeine USS-Antriebsverbindung und -einrichtung

MicroMaster-Antrieb anschließen

Diese Informationen zu SIEMENS MicroMaster-Antriebe dienen als Beispiel. Bei anderen Antrieben finden Sie die Einrichtungsanleitung im Handbuch des Antriebs.

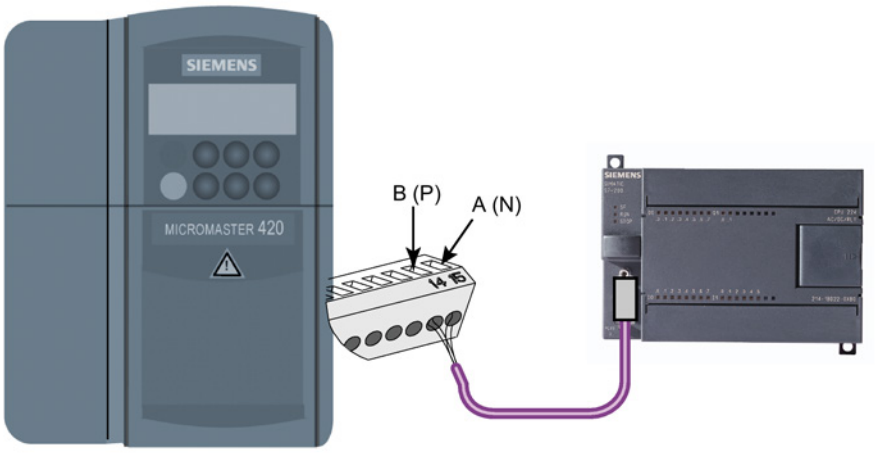
Wenn Sie einen MicroMaster-Antrieb der Serie 4 (MM4) anschließen möchten, stecken Sie die Enden des RS485-Kabels in die beiden schraubenlosen Druckklemmen für den USS-Betrieb. Die S7-1200 kann mit dem herkömmlichen PROFIBUS-Kabel und den Steckverbindern an den MicroMaster-Antrieb angeschlossen werden.

⚠ VORSICHT

Wenn Sie Geräte miteinander verbinden, die nicht die gleiche Bezugsspannung haben, kann dies unerwünschte Ströme im Verbindungskabel hervorrufen.

Diese unerwünschten Ströme können Kommunikationsfehler verursachen oder Sachschaden in den Geräten hervorrufen. Stellen Sie sicher, dass alle Geräte, die Sie über ein Kommunikationskabel miteinander verbinden, entweder den gleichen Bezugsleiter im Stromkreis haben oder elektrisch getrennt sind, damit keine unerwünschten Ströme auftreten. Die Schirmung muss mit Masse oder Pin 1 des 9-poligen Steckverbinders verbunden werden. Die Verdrahtungsklemme 2-0 V des MicroMaster-Antriebs sollte mit Masse verbunden werden.

Stecken Sie die beiden Drähte am gegenüberliegenden Ende des RS485-Kabels in den Klemmenblock des MM4-Antriebs. Zum Herstellen der Kabelverbindung am MM4-Antrieb entfernen Sie die Abdeckung(en) des Antriebs, damit Sie Zugriff auf die Klemmenblöcke haben. Im Benutzerhandbuch des MM4-Antriebs finden Sie eine ausführliche Beschreibung, wie Sie die Abdeckung(en) an Ihrem spezifischen Antrieb entfernen.

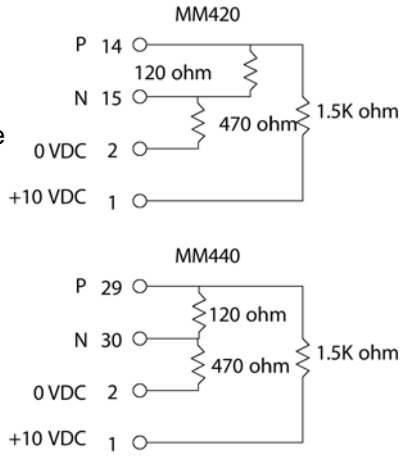


Die Anschlüsse am Klemmenblock sind durchnummeriert. Verwenden Sie auf der Seite der S7-1200 einen PROFIBUS-Anschlussstecker und schließen Sie die Klemme A des Kabels an Klemme 15 des Antriebs (bei einem MM420) oder an Klemme 30 des Antriebs (bei einem MM440) an. Schließen Sie die Klemme B (P) A (N) des Kabelverbinders an Klemme 14 (bei einem MM420) oder an Klemme 29 (bei einem MM440) an.

Handelt es sich bei der S71200 um einen abschließenden Teilnehmer im Netz, oder handelt es sich um eine Punkt-zu-Punkt-Verbindung, müssen Sie die Klemmen A1 und B1 (nicht A2 und B2) des Steckverbinders verwenden, weil diese Klemmen die Abschlusseinstellungen ermöglichen (z. B. beim DP-Steckverbinder 6ES7 972--0BA40--0X40).

⚠ VORSICHT
Bauen Sie die Abdeckungen des Antriebs sorgfältig ein, bevor Sie das Gerät wieder mit Spannung versorgen Achten Sie darauf, dass Sie die Abdeckungen des Antriebs sorgfältig wieder einbauen, bevor Sie das Gerät mit Spannung versorgen.

Ist der Antrieb als abschließender Teilnehmer im Netz konfiguriert, dann müssen außerdem Abschlusswiderstände mit den entsprechenden Klemmen verdrahtet werden. Diese Abbildung zeigt Beispiele für die Anschlüsse des MM4-Antriebs für den Abschluss.



Einrichten des MM4-Antriebs

Bevor Sie einen Antrieb an die S7-1200 anschließen, müssen Sie sicherstellen, dass der Antrieb über folgende Systemparameter verfügt. Sie stellen die Parameter mit der Tastatur des Antriebs ein:

1. Setzen Sie den Antrieb auf die Werkseinstellungen zurück (optional).	P0010 = 30 P0970 = 1
Wenn Sie Schritt 1 überspringen, müssen Sie darauf achten, dass diese Parameter auf die angegebenen Werte gesetzt sind:	USS PZD-Länge = P2012 Index 0 = (2, 4, 6 oder 8) USS PKW-Länge = P2013 Index 0 = 4
2. Aktivieren Sie den Lese-/Schreibzugriff für alle Parameter (Expertenmodus).	P0003 = 3
3. Prüfen Sie die Motoreinstellungen Ihres Antriebs. Die Einstellungen richten sich nach dem jeweiligen Motor. Damit Sie die Parameter P304, P305, P307, P310 und P311 einstellen können, müssen Sie zunächst den Parameter P010 auf 1 setzen (Modus für Schnellinbetriebsetzung). Nachdem Sie die Parameter eingestellt haben, setzen Sie den Parameter P010 auf 0. Die Parameter P304, P305, P307, P310 und P311 können nur im Modus für Schnellinbetriebsetzung geändert werden.	P0304 = Motornennspannung (V) P0305 = Motornennstrom (A) P0307 = Motornennleistung (W) P0310 = Motornennfrequenz (Hz) P0311 = Motornenndrehzahl
4. Stellen Sie den lokalen/entfernten Modus ein.	P0700 Index 0 = 5
5. Stellen Sie den Frequenzsollwert der COM-Verbindung auf USS ein.	P1000 Index 0 = 5
6. Hochlaufzeit (optional) Zeit in Sekunden, die der Motor braucht, um auf maximale Frequenz zu beschleunigen.	P1120 = (0 bis 650,00)
7. Auslaufzeit (optional) Zeit in Sekunden, die der Motor braucht, um bis zum vollständigen Halt abzubremsen.	P1121 = (0 bis 650,00)
8. Stellen Sie die Referenzfrequenz der seriellen Verbindung ein:	P2000 = (1 bis 650 Hz)
9. Stellen Sie die USS-Normalisierung ein:	P2009 Index 0 = 0

10. Stellen Sie die Baudrate der seriellen Schnittstelle RS485 ein:	P2010 Index 0 = 4 (2400 Baud) 5 (4800 Baud) 6 (9600 Baud) 7 (19200 Baud 8 (38400 Baud) 9 (57600 Baud) 12 (115200 Baud)
11. Geben Sie die Slave-Adresse ein. Jeder Antrieb (maximal 31) kann über den Bus betrieben werden.	P2011 Index 0 = (0 bis 31)
12. Legen Sie den Timeout-Wert für die serielle Verbindung fest. Dies ist der maximal zulässige Zeitraum zwischen zwei eingehenden Datentelegrammen. Diese Funktion schaltet den Inverter bei Kommunikationsausfall aus. Die Zeit wird gemessen, nachdem ein gültiges Telegramm empfangen wurde. Wird kein weiteres Datentelegramm innerhalb der angegebenen Zeit empfangen, schaltet der Inverter ab und zeigt Fehlercode F0070 an. Wenn Sie den Wert auf Null setzen, wird die Steuerung ausgeschaltet.	P2014 Index 0 = (0 bis 65.535 ms) 0 = Timeout deaktiviert
13. Übertragen Sie die Daten vom RAM zum EEPROM:	P0971 = 1 (Übertragung starten). Speichern Sie die Änderungen an den Parametereinstellungen im EEPROM.

13.5 Modbus-Kommunikation

13.5.1 Überblick über die Modbus RTU- und TCP-Kommunikation, Modbus-TCP-Anweisungen V13

Modbus-Funktionscodes

- Eine CPU, die als Modbus RTU-Master (oder Modbus TCP-Client) betrieben wird, kann Daten und E/A-Zustände in einem dezentralen Modbus RTU-Slave (bzw. Modbus TCP-Server) lesen und schreiben. Dezentrale Daten können von Ihrer Programmlogik gelesen und dann verarbeitet werden.
- Eine CPU, die als Modbus RTU-Slave (oder Modbus TCP-Server) betrieben wird, ermöglicht es einem übergeordneten Gerät, Daten und E/A-Zustände im Speicher der CPU zu lesen und zu schreiben. Ein RTU-Master (oder Modbus TCP-Client) kann neue Werte in den Speicher der Slave/Server-CPU schreiben, der für Ihre Programmlogik verfügbar ist.

⚠ WARNUNG

Wenn sich ein Angreifer physischen Zugriff auf Ihre Netzwerke verschaffen kann, kann er möglicherweise Daten lesen und schreiben.

Das TIA Portal, die CPU und HMI-Geräte (außer HMI-Geräten über GET/PUT) nutzen die sichere Kommunikation, die Schutz vor Replay- und Man-in-the-Middle-Angriffen bietet. Nach der Aktivierung der Kommunikation erfolgt der Austausch signierter Meldungen in Klartext, wodurch ein Angreifer Daten lesen kann, die Daten jedoch vor unberechtigtem Schreiben geschützt sind. Das TIA Portal, nicht der Kommunikationsprozess, verschlüsselt die Daten von Knowhow-geschützten Bausteinen.

Alle anderen Arten der Kommunikation (E/A-Austausch über PROFIBUS, PROFINET, AS-i oder einen anderen E/A-Bus, GET/PUT, T-Block und Kommunikationsmodule (CM)) haben keine Sicherheitsfunktionen. Sie müssen diese Arten der Kommunikation durch Begrenzung des physischen Zugriffs schützen. Wenn sich ein Angreifer über diese Arten der Kommunikation physischen Zugriff auf Ihre Netzwerke verschaffen kann, kann er möglicherweise Daten lesen und schreiben.

Informationen und Empfehlungen zur Sicherheit finden Sie in unseren Operational Guidelines für Industrial Security (http://www.industry.siemens.com/topics/global/en/industrial-security/Documents/operational_guidelines_industrial_security_en.pdf) auf der Website "Service & Support" von Siemens.

Tabelle 13- 49 Funktionen zum Lesen von Daten: Dezentrale E/A und Programmdaten lesen

Modbus-Funktionscode	Funktionen zum Lesen des Slave (Server) - Standardadressierung
01	Ausgangsbits lesen: 1 bis 2000 Bits pro Anforderung
02	Eingangsbits lesen: 1 bis 2000 Bits pro Anforderung
03	Halteregister lesen: 1 bis 125 Bits pro Anforderung
04	Eingangswörter lesen: 1 bis 125 Bits pro Anforderung

Tabelle 13- 50 Funktionen zum Schreiben von Daten: Dezentrale E/A schreiben und Programmdaten ändern

Modbus-Funktionscode	Funktionen zum Schreiben in den Slave (Server) - Standardadressierung
05	Ein Ausgangsbit schreiben: 1 Bit pro Anforderung
06	Ein Halteregister schreiben: 1 Wort pro Anforderung
15	Ein oder mehrere Ausgangsbits schreiben: 1 bis 1968 Bits pro Anforderung
16	Ein oder mehrere Halteregister schreiben: 1 bis 123 Bits pro Anforderung

- Die Modbus-Funktionscodes 08 und 11 bieten Diagnoseinformationen für die Kommunikation des Slavegeräts.
- Modbus-Funktionscode 0 sendet eine Broadcast-Meldung an alle Slaves (ohne Slaveantwort). Die Broadcast-Funktion ist bei Modbus TCP nicht verfügbar, weil die Kommunikation verbindungsbasiert abläuft.

Tabelle 13- 51 Stationsadressen im Modbus-Netzwerk

Station		Adresse
RTU-Station	Standardstationsadresse	1 bis 247
	Erweiterte Stationsadresse	1 bis 65535
TCP-Station	Stationsadresse	IP-Adresse und Portnummer

Modbus-Speicheradressen

Die tatsächlich verfügbare Anzahl von Modbus-Speicheradressen richtet sich nach der CPU-Variante, nach dem verfügbaren Arbeitsspeicher und danach, wie viel CPU-Speicher durch andere Programmdaten belegt ist. Die folgende Tabelle zeigt den Nennwert des Adressbereichs.

Tabelle 13- 52 Modbus-Speicheradressen

Station		Adressbereich
RTU-Station	Adresse im Standardspeicher	10 K
	Adresse im erweiterten Speicher	64 K
TCP-Station	Adresse im Standardspeicher	10 K

Modbus RTU-Kommunikation

Modbus RTU (Remote Terminal Unit) ist ein Standardprotokoll für die Kommunikation im Netzwerk und verwendet die elektrische RS232- oder RS485-Verbindung für die serielle Datenübertragung zwischen Modbus-Geräten im Netzwerk. Eine CPU mit einem RS232 oder RS485 CM oder einem RS485 CB können Sie um PtP-Netzwerkports (Punkt zu Punkt) erweitern.

Modbus RTU nutzt ein Master/Slave-Netzwerk, in dem die gesamte Kommunikation von einem einzigen Master-Gerät ausgelöst wird, während die Slaves lediglich auf die Anforderung des Masters reagieren können. Der Master sendet eine Anforderung an eine Slave-Adresse und nur die Slave-Adresse antwortet auf den Befehl.

Modbus TCP-Kommunikation

Modbus TCP (Transmission Control Protocol) ist ein Standardprotokoll für die Kommunikation im Netzwerk und verwendet den PROFINET-Anschluss an der CPU für die TCP/IP-Kommunikation. Es ist kein zusätzliches Hardwaremodul für die Kommunikation erforderlich.

Modbus TCP nutzt Open User Communications (OUC)-Verbindungen als Modbus-Kommunikationspfad. Neben der Verbindung zwischen STEP 7 und der CPU kann es mehrere Client/Server-Verbindungen geben. Gemischte Client- und Server-Verbindungen werden bis zur maximalen Anzahl der von der CPU zugelassenen Verbindungen (Seite 633) unterstützt.

Jede MB_SERVER-Verbindung muss eine eindeutige Nummer für Instanz-DB und IP-Port verwenden. Je IP-Port wird nur eine Verbindung unterstützt. Für jede Verbindung muss MB_SERVER (mit eindeutigem Instanz-DB und IP-Port) einzeln ausgeführt werden.

Ein Modbus TCP-Client (Master) muss die Client/Server-Verbindung über den Parameter DISCONNECT steuern. Die grundlegenden Aktionen eines Modbus-Clients werden im Folgenden gezeigt.

1. Verbindung zu einem Server (Slave) mit bestimmter IP-Adresse und IP-Portnummer aufbauen
2. Client-Übertragung von Modbus-Meldungen auslösen und Antworten vom Server empfangen
3. Gegebenenfalls die Verbindungsunterbrechung zwischen Client und Server auslösen, um die Verbindung mit einem anderen Server zu ermöglichen

Modbus RTU-Anweisungen in Ihrem Programm

- Modbus_Comm_Load: Eine Ausführung von Modbus_Comm_Load ist erforderlich, um PtP-Portparameter wie Baudrate, Parität und Flusskontrolle einzurichten. Nachdem der CPU-Port für das Modbus RTU-Protokoll konfiguriert ist, kann er nur von der Anweisung Modbus_Master oder der Anweisung Modbus_Slave verwendet werden.
- Modbus_Master: Mit der Modbus_Master-Anweisung kann die CPU als Modbus-RTU-Mastergerät für die Kommunikation mit einem oder mehreren Modbus-Slavegeräten eingesetzt werden.
- Modbus_Slave: Mit der Modbus_Slave-Anweisung kann die CPU als Modbus-RTU-Slavegerät für die Kommunikation mit einem Modbus-Mastergerät eingesetzt werden.

Modbus TCP-Anweisungen in Ihrem Programm

- MB_CLIENT: Client/Server-TCP-Verbindung herstellen, Befehlsmeldung senden, Antwort empfangen und Trennen der Verbindung vom Server steuern
- MB_SERVER: Bei Anforderung Verbindung zu einem Modbus-TCP-Client aufbauen, Modbus-Meldung empfangen und Antwort senden

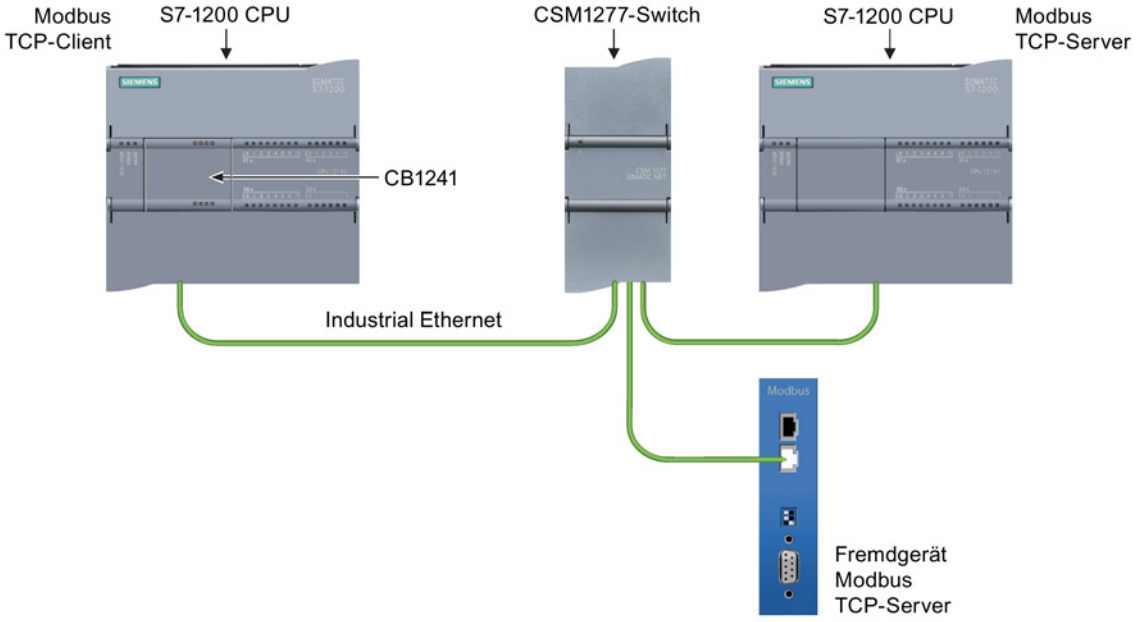
13.5.2 Modbus TCP

13.5.2.1 Übersicht

Version V4.1 der S7-1200 CPU in Verbindung mit STEP 7 V13 SP1 erweitert die Fähigkeit von Modbus TCP dahingehend, dass erweiterte T-Bausteinanweisungen verwendet werden können.

Die S7-1200 unterstützt daher zwei Sätze PtP-Anweisungen:

- Ältere Modbus TCP-Anweisungen (Seite 1055): Diese Modbus TCP-Anweisungen waren vor der Version V4.0 der S7-1200 vorhanden.
- Modbus TCP-Anweisungen (Seite 978): Diese Modbus TCP-Anweisungen stellen alle Funktionen der alten Anweisungen bereit.



13.5.2.2 Version der Modbus TCP-Anweisungen auswählen

Es gibt zwei Versionen der Modbus TCP-Anweisungen in STEP 7:

- Version 3.0 war anfangs in STEP 7 Basic/Professional V13 verfügbar.
- Version 3.1 ist in STEP 7 Basic/Professional V13, SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Verwenden Sie niemals Anweisungen der Version 3.0 und 3.1 zusammen in einem CPU-Programm. Die Modbus TCP-Anweisungen in Ihrem Programm müssen dieselbe Versionsnummer haben (**1**.x, **2**.y, or **V**.z). Die einzelnen Anweisungen in einer Versionsgruppe können dann verschiedene Unterversionen aufweisen (1.**x**).

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.

Um die Version einer Modbus TCP-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine Modbus TCP-Anweisung über das Anweisungsverzeichnis in Ihr Programm einfügen, wird in der Projektnavigation eine neue FB-Instanz angelegt. Die neue FB-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer Modbus TCP-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation die Instanz eines Modbus TCP-FBs aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der Modbus TCP-Anweisung anzuzeigen.

13.5.2.3 Modbus TCP-Anweisungen

Anweisung MB_CLIENT (Über PROFINET als Modbus TCP-Client kommunizieren)

Tabelle 13- 53 Anweisung MB_CLIENT

KOP/FUP	SCL	Beschreibung

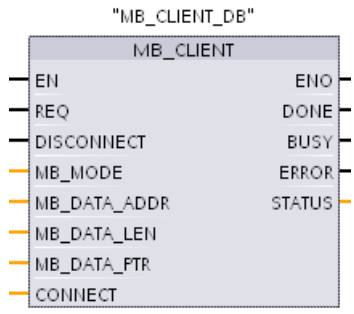
	`"MB_CLIENT_DB"(`
	`REQ:=_bool_in_,`
	`DISCONNECT:=_bool_in_,`
	`MB_MODE:=_usint_in_,`
	`MB_DATA_ADDR:=_udint_in_,`
	`MB_DATA_LEN:=_uint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_DATA_PTR:=_variant_inout_,`
	`CONNECT:=_variant_inout_);`
		MB_CLIENT kommuniziert als Modbus TCP-Client über den PROFINET-Port an der S7-1200 CPU. Es ist kein zusätzliches Hardwaremodul für die Kommunikation erforderlich.
		MB_CLIENT kann eine Client-Server-Verbindung herstellen, eine Modbus-Funktionsanforderung senden, eine Antwort empfangen und das Trennen der Verbindung von einem Modbus TCP-Server steuern.

Tabelle 13- 54 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	In	Bool	FALSCH = Keine Modbus-Kommunikationsanforderung WAHR = Anforderung für die Kommunikation mit einem Modbus TCP-Server
DISCONNECT	IN	Bool	Mit dem Parameter DISCONNECT kann Ihr Programm den Verbindungsaufbau und -abbau zu einem Modbus-Servergerät steuern. Ist DISCONNECT = 0 und keine Verbindung vorhanden, versucht MB_CLIENT eine Verbindung zur zugewiesenen IP-Adresse und Portnummer aufzubauen. Ist DISCONNECT = 1 und eine Verbindung vorhanden, wird versucht, die Verbindung zu trennen. Immer wenn dieser Eingang aktiviert ist, wird kein anderer Vorgang eingeleitet.
MB_MODE	IN	USInt	Modusauswahl: Weist die Art der Anforderung (Lesen, Schreiben oder Diagnose) zu. Weitere Informationen finden Sie in der Tabelle der Modbus-Funktionen unten.
MB_DATA_ADDR	IN	UDInt	Modbus-Anfangsadresse: Weist die Anfangsadresse der Daten zu, auf die MB_CLIENT zugreifen soll. In der folgenden Tabelle der Modbus-Funktionen finden Sie gültige Adressen.
MB_DATA_LEN	IN	UInt	Modbus-Datenlänge: Weist die Anzahl der Bits oder Wörter zu, auf die diese Anforderung zugreifen soll. In der folgenden Tabelle der Modbus-Funktionen finden Sie gültige Längen.
MB_DATA_PTR	IN_OUT	Variant	Pointer auf das Modbus-Datenregister: Das Register puffert Daten, die an einen Modbus-Server gesendet oder von dort empfangen werden. Der Pointer muss einen globalen Standard-DB oder eine Adresse im Speicherbereich der Merker zuweisen.
CONNECT	IN_OUT	Variant	Verweis auf eine Datenbausteinstruktur, die Verbindungsparameter im Systemdatentyp "TCON_IP_v4" enthält.

Parameter und Datentyp		Datentyp	Beschreibung
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
BUSY	OUT	Bool	• 0 - Keine Anweisung MB_CLIENT in Bearbeitung • 1 - Anweisung MB_CLIENT in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die Ausführung von MB_CLIENT mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung

Hinweis

Vorausgesetzte Firmwareversion der CPU

Für die in diesem Abschnitt des Handbuchs beschriebenen Modbus TCP-Anweisungen ist die Firmwareversion V4.1 oder höher erforderlich.

Parameter REQ

FALSCH = Keine Modbus-Kommunikationsanforderung
WAHR = Anforderung für die Kommunikation mit einem Modbus TCP-Server

Wenn keine Instanz von MB_CLIENT aktiv ist und der Parameter DISCONNECT = 0 ist, wird bei REQ = 1 eine neue Modbus-Anforderung gestartet. Ist die Verbindung noch nicht hergestellt, wird eine neue Verbindung aufgebaut.

Wird dieselbe Instanz von MB_CLIENT erneut mit DISCONNECT = 0 and REQ = 1 ausgeführt, bevor die aktuelle Anforderung abgearbeitet ist, werden keine nachfolgenden Modbus-Übertragungen durchgeführt. Sobald die aktuelle Anforderung jedoch abgearbeitet ist, kann eine neue Anforderung verarbeitet werden, sofern MB_CLIENT mit REQ = 1 ausgeführt wird.

Wenn die aktuelle MB_CLIENT-Kommunikationsanforderung abgearbeitet ist, ist das Bit DONE einen Zyklus lang WAHR. Das Bit DONE kann als Zeitfenster für die Sequenzierung mehrerer MB_CLIENT-Anforderungen verwendet werden.

Hinweis

Konsistenz der Eingangsdaten wähend der Verarbeitung von MB_CLIENT

Nachdem ein Modbus-Client eine Modbus-Operation initiiert, werden alle Eingangszustände intern gespeichert und dann bei jedem nachfolgenden Aufruf verglichen. Der Vergleich dient dazu, festzustellen, ob ein bestimmter Aufruf der ursprüngliche Auslöser der aktiven Client-Anforderung war. Mehrere Aufrufe von MB_CLIENT können mit Hilfe eines gemeinsamen Instanz-DBs durchgeführt werden.

Es ist wichtig, dass die Eingänge während des Zeitraums, in dem eine MB_CLIENT - Operation aktiv verarbeitet wird, nicht geändert werden. Wird diese Regel missachtet, kann MB_CLIENT die aktive Instanz nicht ermitteln.

Über die Parameter MB_MODE und MB_DATA_ADDR wählen Sie die Modbus-Kommunikationsfunktion aus.

Anweisung MB_CLIENT nutzt einen MB_MODE-Eingang statt eines Funktionscodeeingangs. MB_DATA_ADDR weist die Modbus-Anfangsadresse der dezentralen Daten zu.

Die Kombination aus MB_MODE und MB_DATA_ADDR legt den Funktionscode fest, der in der eigentlichen Modbus-Meldung verwendet wird. Die folgende Tabelle zeigt die Entsprechung zwischen dem Parameter MB_MODE, MB_DATA_ADDR und der Modbus-Funktion.

Tabelle 13- 55 Modbus-Funktionen

MB_MODE	MB_DATA_ADDR	Länge der Daten	Modbus-Funktionscode aktiviert	Operation und Daten
0	1 bis 9999	1 bis 2000	01	Ausgangsbits lesen: 1 bis 2000 Bit pro Anforderung
0	10001 bis 19999	1 bis 2000	02	Eingangsbits lesen: 1 bis 2000 Bit pro Anforderung
0	40001 bis 49999 oder 400001 bis 465535	1 bis 125	03	Halteregister lesen: 1 bis 125 Wörter pro Anforderung
0	30001 bis 39999	1 bis 125	04	Eingangswörter lesen: 1 bis 125 Wörter pro Anforderung
1	1 bis 9999	1	05	Ein Ausgangsbit schreiben: Ein Bit pro Anforderung
1	40001 bis 49999 oder 400001 bis 465535	1	06	Ein Halteregister schreiben: 1 Wort pro Anforderung
1	1 bis 9999	2 bis 1968	15	Mehrere Ausgangsbits schreiben: 2 bis 1968 Bits pro Anforderung
1	40001 bis 49999 oder 400001 bis 465535	2 bis 123	16	Mehrere Halteregister schreiben: 2 bis 123 Wörter pro Anforderung
2	1 bis 9999	1 bis 1968	15	Ein oder mehrere Ausgangsbits schreiben: 1 bis 1968 Bit pro Anforderung
2	40001 bis 49999 oder 400001 bis 465535	1 bis 123	16	Ein oder mehrere Halteregister schreiben: 1 bis 123 Wörter pro Anforderung
11		0	11	Statuswort und Ereigniszähler der Serverkommunikation lesen. Das Statuswort zeigt beschäftigt an (0 = nicht beschäftigt, 0xFFFF = beschäftigt). Der Ereigniszähler wird bei jeder erfolgreichen Abarbeitung einer Meldung inkrementiert. Sowohl Parameter MB_DATA_ADDR als auch Parameter MB_DATA_LEN von MB_CLIENT wird bei dieser Funktion ignoriert.

MB_MODE	MB_DATA_ADDR	Länge der Daten	Modbus-Funktions-code aktiviert	Operation und Daten
80		1	08	Serverzustand mit Diagnosecode 0x0000 prüfen (Prüfschleifentest, Server gibt ein Echo der Anforderung zurück) 1 Wort pro Anforderung
81		1	08	Server-Ereigniszähler mit Diagnosecode 0x000A zurücksetzen 1 Wort pro Anforderung
3 bis 10, 12 bis 79, 82 bis 255				Reserviert

Hinweis

MB_DATA_PTR weist einen Puffer zu, um aus einem Modbus TCP-Server gelesene bzw. in den Server geschriebene Daten zu speichern

Der Datenpuffer kann sich in einem globalen Standard-DB oder in einer Adresse im Speicherbereich der Merker befinden.

Verwenden Sie für einen Puffer im Speicherbereich der Merker das Format des Any-Pointers. Hierbei handelt es sich um das Format P#"Bitadresse" "Datentyp" "Länge". Ein Beispiel: P#M1000.0 WORD 500.

Zuweisung eines Kommunikationspuffers durch den Parameter MB_DATA_PTR

- Kommunikationsfunktionen von MB_CLIENT:
  - 1-Bit-Daten aus Modbus-Serveradressen lesen und schreiben (00001 bis 09999)
  - 1-Bit-Daten aus Modbus-Serveradressen lesen (10001 bis 19999)
  - 16-Bit-Wortdaten aus Modbus-Serveradressen lesen (30001 bis 39999) und (40001 bis 49999)
  - 16-Bit-Wortdaten in Modbus-Serveradressen schreiben (40001 bis 49999)
- Daten in Wort- oder Bitgröße werden in den oder aus dem von MB_DATA_PTR zugewiesenen Puffer im DB oder Merkerbereich übertragen.
- Wenn von MB_DATA_PTR ein DB als Puffer zugewiesen ist, müssen Sie allen DB-Datenelementen Datentypen zuweisen.
  - Der 1-Bit-Datentyp Bool stellt eine Modbus-Bitadresse dar.
  - Datentypen mit einzelnen 16-Bit-Wörtern wie WORD, UInt und Int stellen eine Modbus-Wortadresse dar.
  - Datentypen mit 32-Bit-Doppelwörtern wie DWORD, DInt und Real stellen zwei Modbus-Wortadressen dar.

- Komplexe DB-Elemente können von MB_DATA_PTR zugewiesen werden, z.B.
  - Arrays
  - Benannte Strukturen, in denen jedes Element eindeutig ist.
  - Benannte komplexe Strukturen, in denen jedes Element einen eindeutigen Namen und einen 16- oder 32-Bit-Datentyp hat.
- Es ist nicht erforderlich, dass die Datenbereiche für MB_DATA_PTR in demselben globalen Datenbaustein (oder Speicherbereich der Merker) liegen. Sie können einen Datenbaustein für Modbus-Lesevorgänge, einen anderen Datenbaustein für Modbus-Schreibvorgänge oder einen Datenbaustein für jeden MB_CLIENT anlegen.

Parameter CONNECT weist Daten zum Herstellen einer PROFINET-Verbindung zu

Sie müssen einen globalen Datenbaustein verwenden und die erforderlichen Verbindungsdaten speichern, bevor Sie diesen DB am Parameter CONNECT referenzieren können.

1. Um die CONNECT-Daten zu speichern, erstellen Sie einen neuen globalen DB oder verwenden einen vorhandenen globalen DB. In einem DB können Sie mehrere TCON_IP_v4-Datenstrukturen speichern. Jede Modbus TCP-Client- bzw. -Server-Verbindung verwendet eine TCON_IP_v4-Datenstruktur. Am Parameter CONNECT verweisen Sie auf die Verbindungsdaten.
2. Geben Sie dem DB und einer statischen Variable aussagekräftige Namen. Nennen Sie den Datenbaustein beispielsweise "Modbus-Verbindungen" und die statische Variable "TCPaktiv_1" (für die Modbus TCP-Clientverbindung 1).
3. Im DB-Editor weisen Sie der statischen Variable des Beispiels "TCPaktiv_1" den Systemdatentyp "TCON_IP_v4" in der Spalte "Datentyp" zu.
4. Erweitern Sie die Struktur TCON_IP_v4, damit Sie die Verbindungsparameter ändern können, wie in der folgenden Abbildung gezeigt.
5. Ändern Sie die Daten in der Struktur TCON_IP_v4 für eine MB_CLIENT-Verbindung.
6. Geben Sie den Verweis auf die DB-Struktur für den Parameter CONNECT von MB_CLIENT ein. In diesem Beispiel lautet der Verweis "Modbus-Verbindungen".TCPaktiv_1.

Modbus connections

	Name	Datentyp	Startwert	Kommentar
1	Static			
2	TCPactive_1	TCON_IP_v4		
3	InterfaceId	HW_ANY	64	HW-identifier of IE-interface submodule
4	ID	CONN_OUC	1	connection reference / identifier
5	ConnectionType	Byte	16#0B	type of connetion: 11=TCP/IP, 19=UDP (17=TC...
6	ActiveEstablished	Bool	True	active/passive connection establishment
7	RemoteAddress	IP_V4		remote IP address (IPv4)
8	ADDR	array [1..4] of Byte		IPv4 address
9	ADDR[1]	Byte	192	
10	ADDR[2]	Byte	168	
11	ADDR[3]	Byte	2	
12	ADDR[4]	Byte	241	
13	RemotePort	UInt	502	remote UDP/TCP port number
14	LocalPort	UInt	0	local UDP/TCP port number

Ändern der TCP_IP_v4-DB-Daten für jede MB_CLIENT-Verbindung

- **InterfaceID**: Klicken Sie im Fenster der Gerätekonfiguration auf das Bild des PROFINET-Anschlusses der CPU. Klicken Sie dann auf das Register "Allgemein" und verwenden Sie die dort angezeigte Hardwarekennung.
- **ID**: Geben Sie eine Nummer für die Verbindungskennung zwischen 1 und 4095 ein. Die Modbus TCP-Kommunikation erfolgt über die zugrunde liegenden Anweisungen TCON, TDISCON, TSEND und TRCV für die offene Benutzerkommunikation (OUC).
- **ConnectionType**: Verwenden Sie bei TCP/IP den Standard 16#0B (Dezimalzahl = 11).
- **ActiveEstablished**: Dieser Wert muss 1 oder WAHR sein. Die Verbindung ist deshalb aktiv, weil MB_CLIENT die Modbus-Kommunikation initiiert.
- **RemoteAddress**: Geben Sie die IP-Adresse des Modbus TCP-Zielservers in die vier ADDR-Array-Elemente ein. Geben Sie beispielsweise 192.168.2.241 wie im Bild oben gezeigt ein.
- **RemotePort**: Voreingestellt ist ein Wert von 502. Diese Nummer ist die IP-Portnummer des Modbus-Servers, mit dem MB_CLIENT eine Verbindung aufzubauen und zu kommunizieren versucht. Bei den Modbus-Servern einiger Dritthersteller ist es erforderlich, dass Sie eine andere Portnummer verwenden.
- **LocalPort**: Dieser Wert muss bei einer MB_CLIENT-Verbindung 0 sein.

Mehrere Clientverbindungen

Ein Modbus TCP-Client unterstützt gleichzeitige Verbindungen bis zur maximalen für das PLC-Gerät zulässigen Anzahl von Open User Communications-Verbindungen. Die Gesamtzahl der Verbindungen für ein PLC-Gerät, einschließlich Modbus TCP-Clients und -Server, darf die maximale Anzahl der unterstützten Open User Communications-Verbindungen nicht überschreiten (Seite 633).

Einzelne gleichzeitige Clientverbindungen müssen die folgenden Regeln einhalten:

- Jede MB_CLIENT-Verbindung muss einen eindeutigen Instanz-DB nutzen.
- Jede MB_CLIENT-Verbindung muss eine eindeutige IP-Adresse eines Servers zuweisen.
- Jede MB_CLIENT-Verbindung muss eine eindeutige Verbindungs-ID zuweisen.
- Eindeutige IP-Portnummern sind möglicherweise je nach Serverkonfiguration erforderlich.

Für jeden Instanz-DB muss eine andere Verbindungs-ID verwendet werden. Zusammengefasst heißt dies, dass Instanz-DB und Verbindungs-ID gepaart sind und für jede Verbindung eindeutig sein müssen.

Tabelle 13- 56 Instanz-Datenbaustein von MB_CLIENT: für den Benutzer zugängliche statische Variablen

Variable	Datentyp	Voreinstellung	Beschreibung
Blocked_Proc_Timeout	Real	3.0	Zeitdauer (in Sekunden), die auf eine blockierte Modbus-Client-Instanz gewartet werden soll, bevor diese Instanz als AKTIV entfernt wird. Dies kann beispielsweise vorkommen, wenn eine Clientanforderung ausgegeben wurde und die Anwendung dann aufhört, die Clientfunktion auszuführen, bevor die Anforderung vollständig abgearbeitet wurde. Maximaler Grenzwert bei der S7-1200 sind 55 Sekunden.
MB_Unit_ID	Wort	255	Modbus-Gerätekennung: Ein Modbus TCP-Server wird über seine IP-Adresse angesprochen. Deshalb wird der Parameter MB_UNIT_ID bei der Modbus TCP-Adressierung nicht verwendet. Der Parameter MB_UNIT_ID entspricht dem Feld der Slaveadresse beim Modbus RTU-Protokoll. Wenn ein Modbus TCP-Server als Gateway zu einem Modbus RTU-Protokoll verwendet wird, kann das Slavegerät im seriellen Netzwerk über MB_UNIT_ID identifiziert werden. Der Parameter MB_UNIT_ID würde in diesem Fall die Anforderung an die richtige Modbus RTU-Slaveadresse weiterleiten. Einige Modbus TCP-Geräte benötigen den Parameter MB_UNIT_ID möglicherweise, um innerhalb eines eingeschränkten Wertebereichs zu liegen.
RCV_TIMEOUT	Real	2.0	Zeit in Sekunden, die MB_CLIENT auf die Antwort eines Servers auf eine Anforderung wartet.
Verbunden	Bool	0	Gibt an, ob die Verbindung zum zugewiesenen Server verbunden oder nicht verbunden ist: 1 = verbunden, 0 = nicht verbunden

Tabelle 13- 57 MB_CLIENT-Protokollfehler

STATUS (W#16#)	Antwortcode an Modbus-Client (B#16#)	Modbus-Protokollfehler
8381	01	Funktionscode nicht unterstützt
8382	03	Fehler in der Datenlänge
8383	02	Datenadressfehler oder Zugriff außerhalb der Grenzen des Adressbereichs von MB_HOLD_REG
8384	03	Fehler im Datenwert
8385	03	Datendiagnosecode wird nicht unterstützt (Funktionscode 08)

Tabelle 13- 58 Bedingungscodes der Ausführung von MB_CLIENT [1]

STATUS (W#16#)	MB_CLIENT-Parameterfehler
7001	MB_CLIENT wartet auf die Antwort eines Modbus-Servers auf die Anforderung eines Verbindungsaufbaus oder -abbaus am zugewiesenen TCP-Port. Dieser Code wird nur für die erste Ausführung eines Verbindungsaufbaus oder -abbaus ausgegeben.
7002	MB_CLIENT wartet auf die Antwort eines Modbus-Servers auf die Anforderung eines Verbindungsaufbaus oder -abbaus am zugewiesenen TCP-Port. Dies wird für alle nachfolgenden Ausführungen ausgegeben, während auf die Fertigstellung eines Verbindungsaufbaus oder -abbaus gewartet wird.
7003	Ein Verbindungsabbau wurde erfolgreich durchgeführt (nur einen PLC-Zyklus lang gültig).
80C8	Der Server hat nicht während der festgelegten Zeit reagiert. MB_CLIENT muss innerhalb der zugewiesenen Zeit über die ursprünglich gesendete Transaktions-ID eine Antwort erhalten oder es wird dieser Fehler ausgegeben. Prüfen Sie die Verbindung zum Modbus-Servergerät. Dieser Fehler wird erst ausgegeben, nachdem Wiederholungen (sofern zutreffend) durchgeführt wurden.
8188	Ungültiger Modus
8189	Ungültige Datenadresse
818A	Ungültige Datenlänge
818B	Ungültiger Pointer auf den DATA_PTR-Bereich. Hierbei kann es sich um die Kombination aus MB_DATA_ADDRESS + MB_DATA_LEN handeln.
818C	Pointer DATA_PTR zeigt auf einen optimierten DB-Bereich (hierbei muss es sich um einen Standard-DB-Bereich oder um einen Bereich im Speicherbereich der Merker handeln)
8200	Der Port ist durch die Verarbeitung einer vorhandenen Modbus-Anforderung belegt.
8380	Der empfangene Modbus-Rahmen ist inkorrekt oder es wurden zu wenige Bytes empfangen.
8387	Der zugewiesene Parameter der Verbindungs-ID unterscheidet sich von der für vorherige Anforderungen verwendeten ID. Es darf in jedem Instanz-DB von MB_CLIENT nur eine einzige Verbindungs-ID verwendet werden. Dieser Code wird auch als interner Fehler ausgegeben, wenn die von einem Server empfangene Modbus TCP-Protokoll-ID nicht 0 ist.
8388	Ein Modbus-Server hat eine Datenmenge zurückgegeben, die sich von der angeforderten Menge unterscheidet. Dieser Code gilt nur für die Modbus-Funktionen 15 und 16.

[1] Neben den oben aufgeführten MB_CLIENT-Fehlern können auch von den zugrunde liegenden T-Bausteinanweisungen für die Kommunikation (TCON, TDISCON, TSEND und TRCV) Fehler gemeldet werden.

Anweisung MB_SERVER (Über PROFINET als Modbus TCP-Server kommunizieren)

Tabelle 13- 59 Anweisung MB_SERVER

KOP/FUP	SCL	Beschreibung

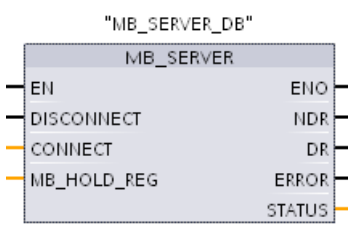
	`"MB_SERVER_DB"(`
	`DISCONNECT:=_bool_in_,`
	`CONNECT:=_variant_in_,`
	`NDR=>_bool_out_,`
	`DR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	
	`MB_HOLD_REG:=_variant_inout_);`
		MB_SERVER kommuniziert als Modbus TCP-Server über den PROFINET-Port an der S7-1200 CPU. Es ist kein zusätzliches Hardwaremodul für die Kommunikation erforderlich.
		MB_SERVER kann eine Anforderung für den Verbindungsaufbau mit einem Modbus TCP-Client annehmen, eine Modbus-Funktionsanforderung empfangen und eine Antwortmeldung senden.

Tabelle 13- 60 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
DISCONNECT	IN	Bool	MB_SERVER versucht, eine "passive" Verbindung mit einem Partnergerät aufzubauen. Das bedeutet, dass der Server passiv überwacht, ob es TCP-Verbindungsanforderungen von IP-Adressen gibt, die Anforderungen senden. Ist DISCONNECT = 0 und keine Verbindung vorhanden, kann eine passive Verbindung aufgebaut werden. Ist DISCONNECT = 1 und eine Verbindung vorhanden, wird versucht, die Verbindung zu trennen. Über diesen Parameter kann Ihr Programm steuern, wann eine Verbindung angenommen wird. Immer wenn dieser Eingang aktiviert ist, wird kein anderer Vorgang eingeleitet.
CONNECT	IN	Variant	Verweis auf eine Datenbausteinstruktur, die Verbindungsparameter im Systemdatentyp "TCON_IP_v4" enthält.
MB_HOLD_REG	IN_OUT	Variant	Pointer auf das Modbus-Halteregister von MB_SERVER: Bei dem Halteregister muss es sich um einen globalen Standard-DB oder um eine Adresse im Speicherbereich der Merker handeln. Dieser Speicherbereich dient dazu, die Daten zu speichern, auf die ein Modbus-Client mit den Modbus-Registerfunktionen 3 (Lesen), 6 (Schreiben) und 16 (Schreiben) zugreifen kann.
NDR	OUT	Bool	Neue Daten bereit: 0 = Keine neuen Daten, 1 = Gibt an, dass von einem Modbus-Client neue Daten geschrieben wurden
DR	OUT	Bool	Daten lesen: 0 = Daten nicht gelesen, 1 = Gibt an, dass die Daten von einem Modbus-Client gelesen wurden
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die Ausführung von MB_SERVER mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung

Hinweis

Vorausgesetzte Firmwareversion der CPU

Für die in diesem Abschnitt des Handbuchs beschriebenen Modbus TCP-Anweisungen ist die Firmwareversion V4.1 oder höher erforderlich.

Parameter CONNECT weist Daten zum Herstellen einer PROFINET-Verbindung zu

Sie müssen einen globalen Datenbaustein verwenden und die erforderlichen Verbindungsdaten speichern, bevor Sie diesen DB am Parameter CONNECT referenzieren können.

1. Um die CONNECT-Daten zu speichern, erstellen Sie einen neuen globalen DB oder verwenden einen vorhandenen globalen DB. In einem DB können Sie mehrere TCON_IP_v4-Datenstrukturen speichern. Jede Modbus TCP-Client- bzw. -Server-Verbindung verwendet eine TCON_IP_v4-Datenstruktur. Am Parameter CONNECT verweisen Sie auf die Verbindungsdaten.
2. Geben Sie dem DB und einer statischen Variable aussagekräftige Namen. Nennen Sie den Datenbaustein beispielsweise "Modbus-Verbindungen" und die statische Variable "TCPpassiv_1" (für die Modbus TCP-Serververbindung 1).
3. Im DB-Editor weisen Sie der statischen Variable des Beispiels "TCPaktiv_1" den Systemdatentyp "TCON_IP_v4" in der Spalte "Datentyp" zu.
4. Erweitern Sie die Struktur TCON_IP_v4, damit Sie die Verbindungsparameter ändern können, wie in der folgenden Abbildung gezeigt.
5. Ändern Sie die Daten in der Struktur TCON_IP_v4 für eine MB_SERVER-Verbindung.
6. Geben Sie den Verweis auf die DB-Struktur für den Parameter CONNECT von MB_SERVER ein. In diesem Beispiel lautet der Verweis "Modbus-Verbindungen".TCPpassiv_1.

	Name	Datentyp	Startwert	Kommentar
1	Static			
2	TCPpassive_1	TCON_IP_v4		
3	InterfaceId	HW_ANY	64	HW-identifier of IE-interface submodule
4	ID	CONN_OUC	1	connection reference / identifier
5	ConnectionType	Byte	16#0B	type of connetion: 11=TCP/IP, 19=UDP (17=TC...
6	ActiveEstablished	Bool	False	active/passive connection establishment
7	RemoteAddress	IP_V4		remote IP address (IPv4)
8	ADDR	array [1..4] of Byte		IPv4 address
9	ADDR[1]	Byte	192	
10	ADDR[2]	Byte	168	
11	ADDR[3]	Byte	2	
12	ADDR[4]	Byte	241	
13	RemotePort	UInt	0	remote UDP/TCP port number
14	LocalPort	UInt	502	local UDP/TCP port number

Ändern der TCP_IP_v4-DB-Daten für jede MB_SERVER-Verbindung

- **InterfaceID**: Klicken Sie im Fenster der Gerätekonfiguration auf das Bild des PROFINET-Anschlusses der CPU. Klicken Sie dann auf das Register "Allgemein" und verwenden Sie die dort angezeigte Hardwarekennung.
- **ID**: Geben Sie eine Zahl zwischen 1 und 4095 ein, die für diese Verbindung eindeutig ist. Die Modbus TCP-Kommunikation erfolgt über die zugrunde liegenden Anweisungen TCON, TDISCON, TSEND und TRCV für die offene Benutzerkommunikation. Bis zu acht Verbindungen der offenen Benutzerkommunikation sind gleichzeitig zulässig.
- **ConnectionType**: Verwenden Sie bei TCP/IP den Standard 16#0B (Dezimalwert = 11).
- **ActiveEstablished**: Dieser Wert muss 0 oder FALSCH sein. Die Verbindung ist deshalb passiv, weil MB_SERVER auf eine Kommunikationsanforderung von einem Modbus-Client wartet.
- **RemoteAddress**: Es gibt zwei Optionen.
  - Verwenden Sie 0.0.0.0, und MB_CLIENT reagiert auf eine Modbus-Anforderung von jedem TCP-Client.
  - Geben Sie die IP-Adresse eines Ziel-Modbus-TCP-Clients ein, und MB_CLIENT reagiert nur auf eine Anforderung von der IP-Adresse dieses Clients. Geben Sie beispielsweise 192.168.2.241 wie im Bild oben gezeigt ein.
- **RemotePort**: Dieser Wert muss bei einer MB_SERVER-Verbindung 0 sein.
- **LocalPort**: Voreingestellt ist ein Wert von 502. Diese Nummer ist die IP-Portnummer des Modbus-Clients, mit dem MB_SERVER eine Verbindung aufzubauen und zu kommunizieren versucht. Die Modbus-Clients einiger Dritthersteller benötigen eine andere Portnummer.

Modbus- und Prozessabbildadressen

Bei MB_SERVER können eingehende Modbus-Funktionscodes (1, 2, 4, 5 und 15) Bits und Wörter direkt im Prozessabbild der Eingänge und Ausgänge lesen und schreiben. Bei den Funktionscodes für die Datenübertragung (3, 6 und 16) muss der Parameter MB_HOLD_REG als Datentyp größer als ein Byte definiert sein. Die folgende Tabelle zeigt die Zuordnung der Modbus-Adressen zum Prozessabbild in der CPU.

Tabelle 13- 61 Zuordnung der Modbus-Adressen zum Prozessabbild

Modbus-Funktionen						**S7-1200**	
Codes	Funktion	Datenbereich	Adressbereich			Datenbereich	CPU-Adresse
01	Bits lesen	Ausgang	1	An	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7
02	Bits lesen	Eingang	10001	An	18192	Prozessabbild der Eingänge	E0.0 bis E1023.7
04	Wörter lesen	Eingang	30001	An	30512	Prozessabbild der Eingänge	EW0 bis EW1022
05	Bit schreiben	Ausgang	1	An	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7
15	Bits schreiben	Ausgang	1	An	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7

Funktionscodes eingehender Modbus-Meldungen (3, 6 und 16) lesen oder schreiben Wörter in einem Modbus-Halteregister, bei dem es sich um einen Merker oder um einen Datenbaustein handeln kann. Der Typ des Halteregisters wird vom Parameter MB_HOLD_REG angegeben.

Hinweis

Zuweisung des Parameters MB_HOLD_REG

Bei dem Modbus-Halteregister kann es sich um einen globalen Standard-DB oder um eine Adresse im Speicherbereich der Merker handeln.

Für ein Modbus-Halteregister im Speicherbereich der Merker verwenden Sie das Format des Any-Pointers. Hierbei handelt es sich um das Format P#"Bitadresse" "Datentyp" "Länge". Beispiel: P#M1000.0 WORD 500.

Die folgende Tabelle zeigt Beispiele für die Zuordnung von Modbus-Adressen zum Halteregister, das für die Modbus-Funktionscodes 03 (Wörter lesen), 06 (Wort schreiben) und 16 (Wörter schreiben) verwendet wird. Die tatsächliche obere Grenze der DB-Adressen wird vom maximalen Arbeitsspeicher und vom maximalen M-Speicher des jeweiligen CPU-Modells festgelegt.

Tabelle 13- 62 Beispiele für die Zuordnung von Modbus-Adressen zu Adressen im Speicher der CPU

Modbus-Adresse	Beispiele für Parameter von MB_HOLD_REG		
	P#M100.0 Word 5	P#DB10.DBx0.0 Word 5	"Rezept".Inhaltsstoff
40001	MW100	DB10.DBW0	"Rezept".Inhaltsstoff[1]
40002	MW102	DB10.DBW2	"Rezept".Inhaltsstoff[2]
40003	MW104	DB10.DBW4	"Rezept".Inhaltsstoff[3]
40004	MW106	DB10.DBW6	"Rezept".Inhaltsstoff[4]
40005	MW108	DB10.DBW8	"Rezept".Inhaltsstoff[5]

Mehrere Serververbindungen

Es können mehrere Serververbindungen hergestellt werden. Ein einziges PLC-Gerät kann gleichzeitig Verbindungen mit mehreren Modbus TCP-Clients herstellen.

Ein Modbus TCP-Server unterstützt gleichzeitige Verbindungen bis zur maximalen für das PLC-Gerät zulässigen Anzahl von Open User Communications-Verbindungen. Die Gesamtzahl der Verbindungen für ein PLC-Gerät, einschließlich Modbus TCP-Clients und -Server, darf die maximale Anzahl der unterstützten Open User Communications-Verbindungen nicht überschreiten (Seite 633). Die Modbus TCP-Verbindungen können von Client- und Serververbindungen gemeinsam genutzt werden.

Einzelne gleichzeitige Serververbindungen müssen die folgenden Regeln einhalten:

- Jede MB_SERVER-Verbindung muss einen eindeutigen Instanz-DB nutzen.
- Jede MB_SERVER-Verbindung muss eine eindeutige IP-Portnummer zuweisen. Je Port wird nur eine Verbindung unterstützt.
- Jede MB_SERVER-Verbindung muss eine eindeutige Verbindungs-ID zuweisen.
- MB_SERVER muss für jede Verbindung einzeln aufgerufen werden (mit dem entsprechenden Instanz-DB).

Die Verbindungs-ID muss für jede einzelne Verbindung eindeutig sein. Für jeden einzelnen Instanz-DB muss eine einzige Verbindungs-ID verwendet werden. Der Instanz-DB und die Verbindungs-ID sind gepaart und müssen für jede Verbindung eindeutig sein.

Tabelle 13- 63 Funktionscodes der Modbus-Diagnose

Modbus-Diagnosefunktionen von MB_SERVER		
Codes	Teilfunktion	Beschreibung
08	0x0000	Echotest Abfragedaten ausgeben: Die Anweisung MB_SERVER gibt einem Modbus-Client das Echo eines empfangenen Datenworts zurück.
08	0x000A	Kommunikationsereigniszähler löschen: Die Anweisung MB_SERVER löscht den Kommunikationsereigniszähler, der für Modbus-Funktion 11 verwendet wird.
11		Kommunikationsereigniszähler abrufen: Die Anweisung MB_SERVER nutzt einen internen Kommunikationsereigniszähler, um die Anzahl erfolgreicher Modbus-Lese- und Modbus-Schreibanforderungen, die an den Modbus-Server gesendet werden, zu erfassen. Der Zähler wird bei Funktion 8, Funktion 11 und allen Anforderungen, die zu einem Kommunikationsfehler führen, nicht hochgezählt. Die Broadcast-Funktion ist bei Modbus TCP nicht verfügbar, weil nur eine Client-Server-Verbindung zur Zeit vorhanden ist.

Variablen von MB_SERVER

Diese Tabelle zeigt die öffentlichen statischen Variablen im Instanz-Datenbaustein von MB_SERVER, die in Ihrem Programm verwendet werden können.

Tabelle 13- 64 Öffentliche statische Variablen von MB_SERVER

Variable	Datentyp	Voreinstellung	Beschreibung
HR_Start_Offset	Wort	0	Weist die Anfangsadresse des Modbus-Halteregisters zu
Request_Count	Wort	0	Die Anzahl aller von diesem Server empfangenen Anforderungen
Server_Message_Count	Wort	0	Die Anzahl der für diesen spezifischen Server empfangenen Anforderungen
Xmt_Rcv_Count	Wort	0	Die Anzahl der Sendungen oder Datenempfänge, bei denen ein Fehler aufgetreten ist. Wird auch inkrementiert, wenn eine Meldung empfangen wird, bei der es sich um eine ungültige Modbus-Meldung handelt.
Exception_Count	Wort	0	Modbus-spezifische Fehler, die eine zurückgegebene Ausnahme erfordern
Success_Count	Wort	0	Die Anzahl der für diesen spezifischen Server empfangenen Anforderungen ohne Protokollfehler
Verbunden	Bool	0	Gibt an, ob die Verbindung zum zugewiesenen Client verbunden oder nicht verbunden ist: 1 = verbunden, 0 = nicht verbunden

Ihr Programm kann in HR_Start_Offset Daten schreiben und die Modbus-Server-Operationen steuern. Die anderen Variablen können gelesen werden, um den Modbus-Zustand zu überwachen.

HR_Start_Offset

Die Adressen des Modbus-Halteregisters beginnen bei 40001. Diese Adressen entsprechen der Anfangsadresse des Halteregisters im Zielsystemspeicher. Sie können jedoch die Variable HR_Start_Offset verwenden, um eine andere Anfangsadresse als 40001 für das Modbus-Halteregister zu konfigurieren.

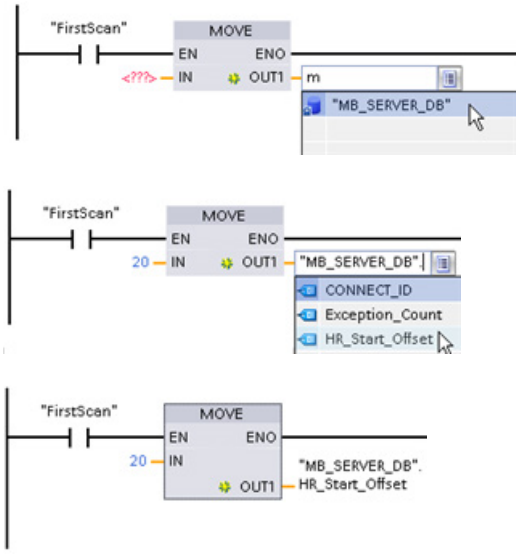
Sie können z. B. ein Halteregister mit Beginn an MW100 und einer Länge von 100 Wörtern konfigurieren. Mit einem Versatz von 20 geben Sie eine Anfangsadresse des Halteregisters von 40021 statt 40001 an. Jede Adresse unter 40021 und über 40119 führt zu einem Adressierungsfehler.

Tabelle 13- 65 Beispiel für die Adressierung des Modbus-Halteregisters

HR_Start_Offset	Adresse	Minimum	Maximum
0	Modbus-Adresse (Wort)	40001	40099
	S7-1200 Adresse	MW100	MW298
20	Modbus-Adresse (Wort)	40021	40119
	S7-1200 Adresse	MW100	MW298

HR_Start_Offset sind Wortdaten im Instanz-Datenbaustein von MB-SERVER, die die Anfangsadresse des Modbus-Halteregisters zuweisen. Sie können diese öffentliche statische Variable über die Parameter-Klappliste auswählen, nachdem Sie MB_SERVER in Ihr Programm eingefügt haben.

Wenn Sie beispielsweise MB_SERVER in ein KOP-Netzwerk eingefügt haben, können Sie in ein vorheriges Netzwerk gehen und HR_Start_Offset zuweisen. Die Anfangsadresse muss vor der Ausführung von MB_SERVER zugewiesen werden.



Eingabe einer Variable für den Modbus-Server

über den Namen des Standard-DB:

1. Positionieren Sie den Cursor im Parameterfeld und geben Sie das Zeichen m ein.
2. Wählen Sie in der Klappliste der DB-Namen "MB_SERVER_DB" aus.
3. Wählen Sie in der Klappliste der DB-Variablen "MB_SERVER_DB.HR_Start_Offset" aus.

Tabelle 13- 66 Bedingungscodes der Ausführung von MB_SERVER [1]

STATUS (W#16#)	Antwortcode an Modbus-Server(B#16#)	Modbus-Protokollfehler
7001		MB_SERVER wartet darauf, dass ein Modbus-Client die Verbindung zum zugewiesenen TCP-Port herstellt. Dieser Code wird bei der ersten Ausführung eines Verbindungsaufbaus oder -abbaus ausgegeben.
7002		MB_SERVER wartet darauf, dass ein Modbus-Client die Verbindung zum zugewiesenen TCP-Port herstellt. Dieser Code wird für alle nachfolgenden Ausführungen ausgegeben, während auf die Fertigstellung eines Verbindungsaufbaus oder -abbaus gewartet wird.
7003		Ein Verbindungsabbau wurde erfolgreich durchgeführt (nur einen PLC-Zyklus lang gültig).
8187		Ungültiger Pointer auf MB_HOLD_REG: Bereich ist zu klein
818C		Pointer MB_HOLD_REG zeigt auf einen optimierten DB-Bereich (hierbei muss es sich um einen globalen DB-Bereich oder um einen Bereich im Speicherbereich der Merker handeln) oder Timeout für gesperrten Prozess überschreitet den Grenzwert von 55 Sekunden. (spezifisch für die S7-1200)
8381	01	Funktionscode nicht unterstützt
8382	03	Fehler in der Datenlänge
8383	02	Datenadressfehler oder Zugriff außerhalb der Grenzen des Adressbereichs von MB_HOLD_REG
8384	03	Fehler im Datenwert
8385	03	Datendiagnosecode wird nicht unterstützt (Funktionscode 08)

[1] Neben den oben aufgeführten MB_SERVER-Fehlern können Fehler auch von den zugrunde liegenden T-Bausteinanweisungen für die Kommunikation (TCON, TDISCON, TSEND und TRCV) gemeldet werden.

13.5.2.4 Beispiele für Modbus TCP

Beispiel: MB_SERVER für mehrere TCP-Verbindungen

Sie können mehrere Verbindungen zum Modbus TCP-Server haben. Hierfür muss MB_SERVER für jede Verbindung unabhängig ausgeführt werden. Jede Verbindung muss einen unabhängigen Instanz-DB, eine Verbindungs-ID und einen IP-Port verwenden. Bei der S7-1200 ist nur je eine Verbindung pro IP-Port zulässig.

Um optimales Betriebsverhalten zu erzielen, muss MB_SERVER in jedem Programmzyklus für jede Verbindung ausgeführt werden.

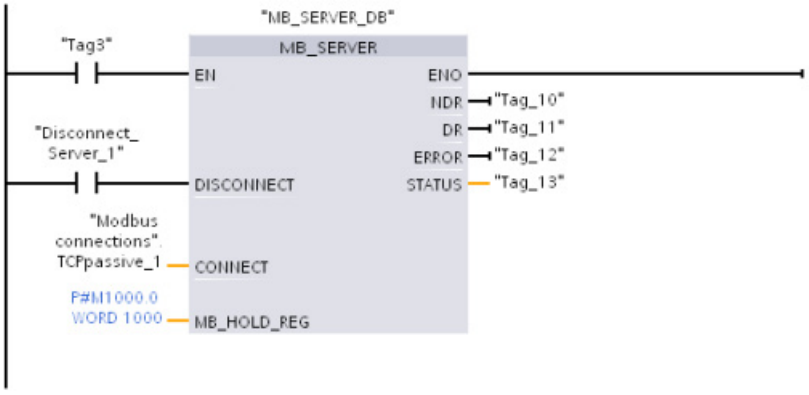
Der Parameter CONNECT verwendet den Systemdatentyp TCP_IP_v4. Im Beispiel befinden sich diese Datenstrukturen im DB "Modbus-Verbindungen". Der DB "Modbus-Verbindungen" enthält zwei TCP_IP_v4-Strukturen: "TCPpassiv_1" (für Verbindung 1) und "TCP_passiv_2" (für Verbindung 2). Bei der in den Netzwerkkommentaren beschriebenen ID der Verbindungseigenschaften und bei LocalPort handelt es sich um in der Datenstruktur CONNECT gespeicherte Datenelemente.

Die TCP_IP_v4 CONNECT-Daten enthalten auch eine IP-Adresse im ADDR-Array RemoteAddress. Die IP-Adresszuweisungen in TCPpassiv_1 und TCP_passiv_2 wirken sich nicht auf den Aufbau der TCP-Serververbindungen aus, bestimmen jedoch, welche Modbus TCP-Clients über die Verbindungen mit jedem MB_SERVER kommunizieren dürfen. MB_SERVER überwacht passiv auf eine Modbus-Client-Meldung und vergleicht die eingehende IP-Adresse der Meldung mit der im entsprechenden ADDR-Array RemoteAddress gespeicherten IP-Adresse.

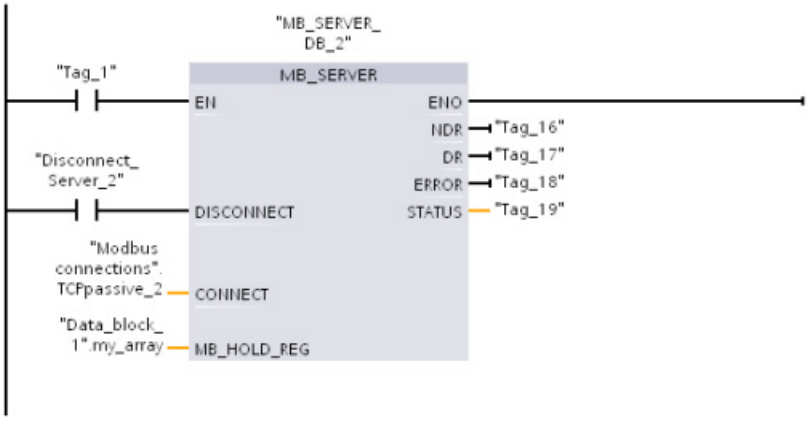
Für die beiden MB_SERVER-Anweisungen sind drei Varianten von IP-Adressen möglich:

- **IP-Adresse = 0.0.0.0**
  Jede MB_SERVER-Anweisung reagiert auf alle Modbus TCP-Clients mit beliebiger IP-Adresse.
- **IP-Adresse = Gleiche IP-Adresse in TCPpassiv_1 und TCPpassiv_2**
  Beide MB_SERVER-Verbindungen reagieren nur auf Modbus-Clients, die von dieser IP-Adresse stammen.
- **IP-Adresse = Unterschiedliche IP-Adresse in TCPpassiv_1 und TCPpassiv_2**
  Jede MB_SERVER-Anweisung reagiert nur auf Modbus-Clients, die von der IP-Adresse stammen, die in ihren jeweiligen TCP_IP_v4-Daten gespeichert ist.

Netzwerk 1: Verbindung 1, Instanz-DB = "MB_SERVER_DB", in "Modbus-Verbindungen.TCPpassiv_1" (ID = 1 und LocalPort = 502)



Netzwerk 2: Verbindung 2, Instanz-DB = "MB_SERVER_DB_1", in "Modbus-Verbindungen.TCPpassiv_2" (ID = 2 und LocalPort = 503)



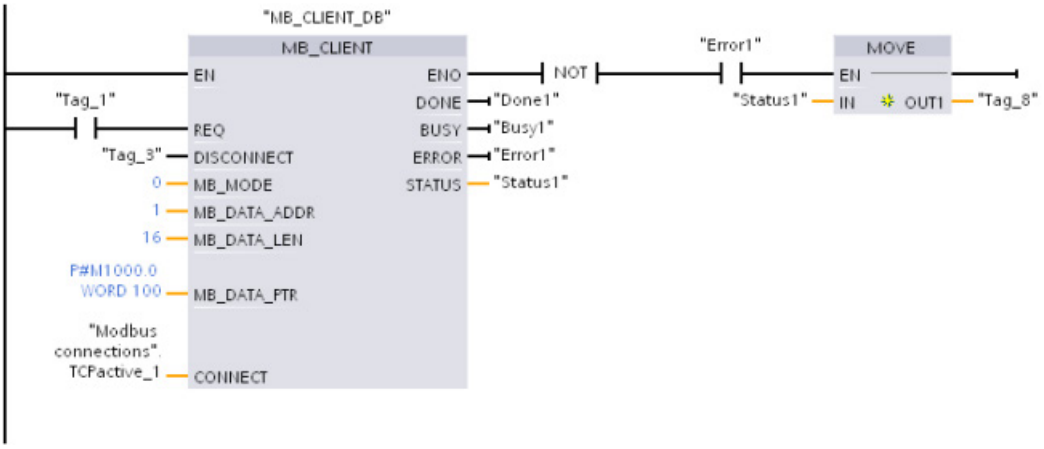
Beispiel: MB_CLIENT 1: Mehrere Anforderungen mit gemeinsamer TCP-Verbindung

Mehrere Modbus-Clientanforderungen können über die gleiche Verbindung gesendet werden. Hierfür verwenden Sie den gleichen Instanz-DB, die gleiche Verbindungs-ID und Portnummer.

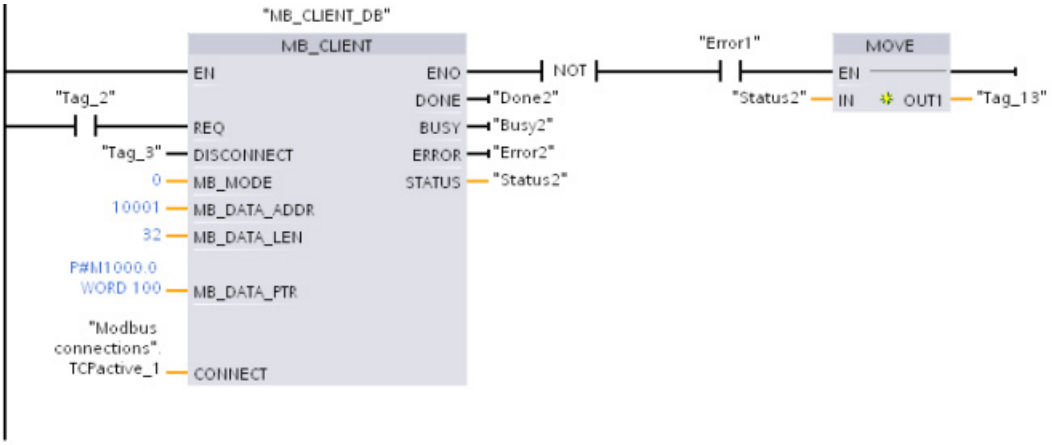
Weil beide MB_CLIENT-Boxen die gleiche TCON_IP_v4-Datenstruktur am Parameter CONNECT ("Modbus_Verbindungen".TCPaktiv_1) verwenden, sind die Verbindungs-ID, Portnummer und IP-Adresse identisch. Die IP-Adressdaten von CONNECT weisen die IP-Adresse des Ziel-Modbus-TCP-Servers zu.

Es kann jeweils nur eine Anweisung MB_CLIENT aktiv sein. Nachdem ein Client seine Ausführung beendet hat, kann der nächste Client mit der Ausführung beginnen. Ihre Programmlogik ist für die Reihenfolge der Ausführung verantwortlich. Das Beispiel zeigt beide Clients, die dezentrale Daten aus einem einzelnen Modbus-Client lesen und die Daten an die CPU des Modbus-Clients übertragen (Speicherbereich der Merker mit Beginn an M1000.0). Ein ausgegebener Fehler wird erfasst (optionale Funktion).

Netzwerk 1: Modbus-Funktion 1 - 16 Ausgangsbits von einem Modbus TCP-Server mit der in "Modbus-Verbindungen".TCPaktiv_1 zugewiesenen IP-Adresse lesen.



Netzwerk 2: Modbus-Funktion 2 - 32 Eingangsbits von einem Modbus TCP-Server mit der in "Modbus-Verbindungen".TCPaktiv_1 zugewiesenen IP-Adresse lesen.



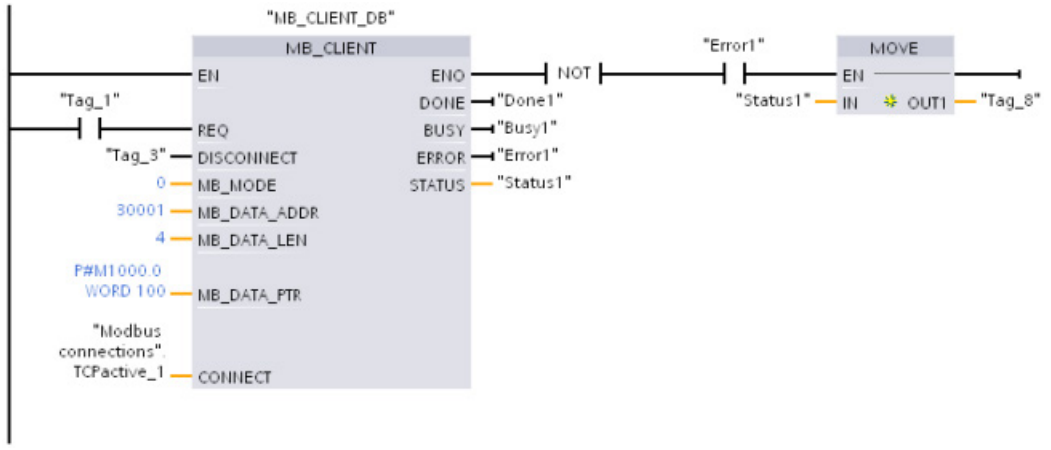
Beispiel: MB_CLIENT 2: Mehrere Anforderungen mit unterschiedlichen TCP-Verbindungen

Mehrere Modbus-TCP-Client-Anforderungen können über verschiedene Verbindungen gesendet werden. Hierfür müssen unterschiedliche Instanz-DBs und Verbindungs-IDs verwendet werden.

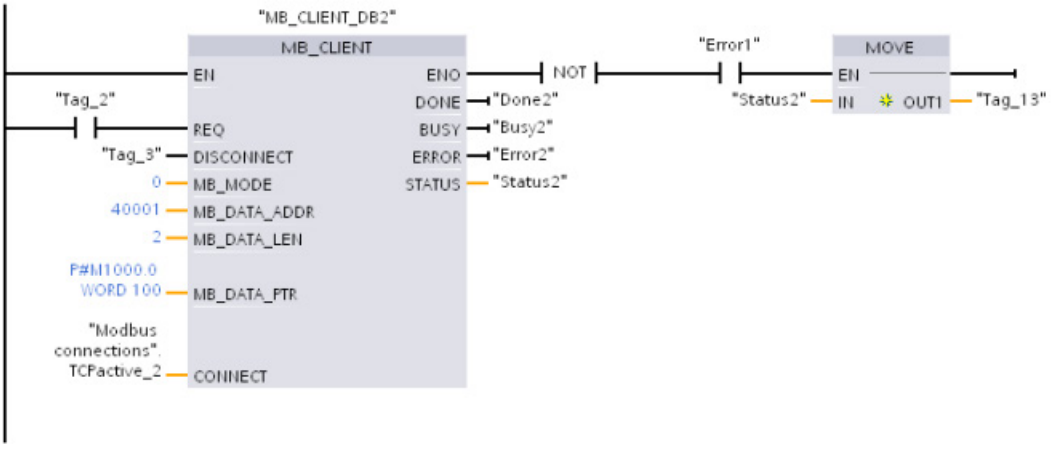
Die RemotePort-Nummer (IP-Port) muss unterschiedlich sein, wenn die Verbindungen zum selben Modbus-Server aufgebaut werden. Bei Verbindungen zu unterschiedlichen Servern gibt es keine Einschränkungen hinsichtlich der IP-Portnummer.

Das Beispiel zeigt zwei Modbus-TCP-Clients, die dezentrale Daten von zwei unterschiedlichen Modbus-TCP-Servern in denselben lokalen CPU-Speicherbereich mit Beginn an Adresse M1000.0 übertragen. Außerdem wird ein ausgegebener Fehler erfasst (optionale Funktion).

Netzwerk 1: Modbus-Funktion 4 - Wörter aus dem Prozessabbild der Eingänge von einem Modbus-TCP-Server lesen
Parameter CONNECT = "Modbus-Verbindungen".TCPaktiv_1: Verbindungs-**ID** = 1, **RemoteAddress** = 192.168.2.241, **RemotePort** = 502



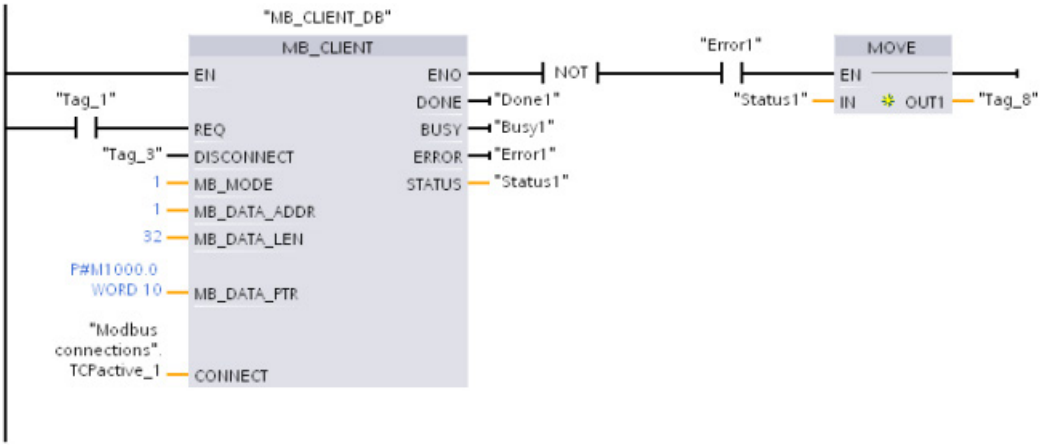
Netzwerk 2: Modbus-Funktion 3 - Wörter aus dem Halteregister von einem Modbus-TCP-Server lesen
Parameter CONNECT = "Modbus-Verbindungen".TCPaktiv_2: Verbindungs-**ID** = 2, **RemoteAddress** = 192.168.2.242, **RemotePort** = 502



Beispiel: MB_CLIENT 3: Schreibanforderung für das Prozessabbild der Ausgänge

Dieses Beispiel zeigt eine Modbus-Client-Anforderung, die Bitdaten aus dem lokalen CPU-Speicher (mit Beginn an M1000.0) zu einem dezentralen Modbus-TCP-Server überträgt.

Netzwerk 1: Modbus-Funktion 15 - Ausgangsbits in einen Modbus-Server schreiben

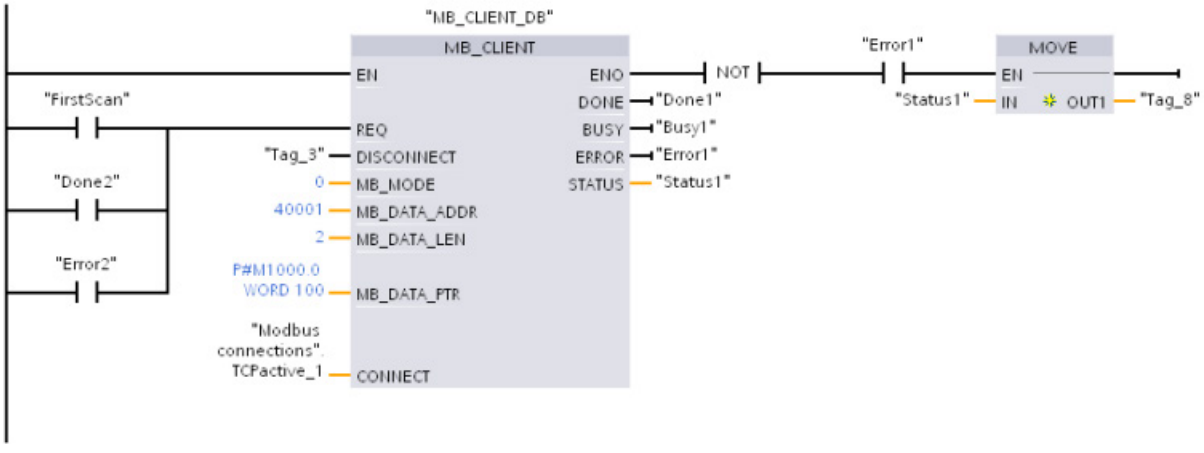


Beispiel: MB_CLIENT 4: Mehrere Anforderungen koordinieren

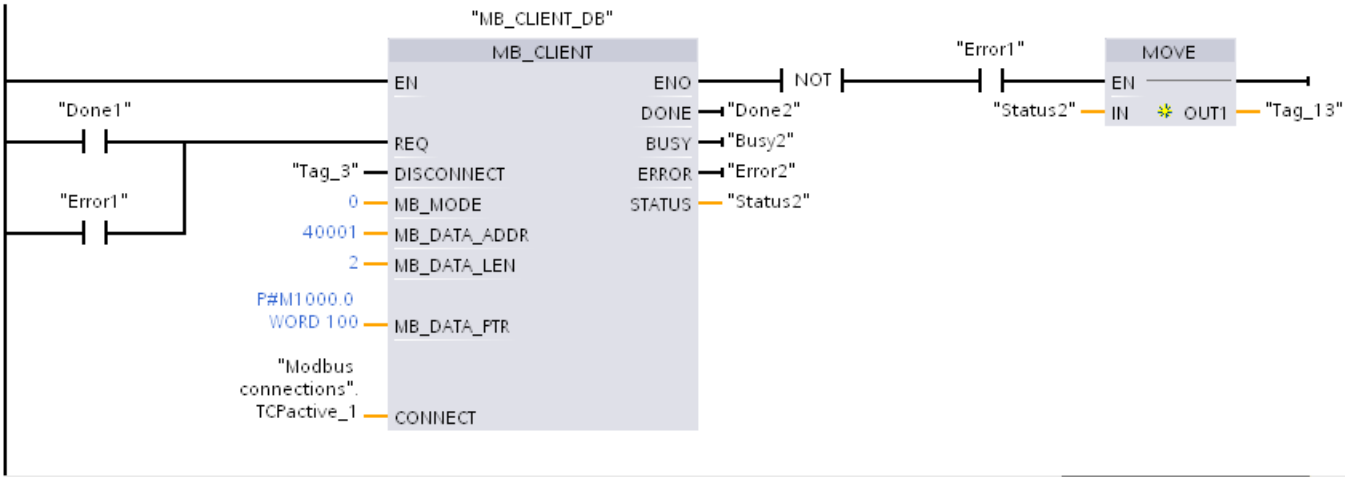
Sie müssen sicherstellen, dass die Ausführung jeder einzelnen Modbus TCP-Anforderung abgeschlossen wird. Die Ausführungsreihenfolge muss von Ihrer Programmlogik gesteuert werden. Das folgende Beispiel zeigt, wie die Ausführungsreihenfolge über die Ausgänge der ersten und der zweiten Clientanforderung gesteuert werden kann.

Das Beispiel zeigt beide Clients, die dieselben CONNECT-Verbindungsdaten verwenden (zu unterschiedlichen Zeiten). Die Clients übertragen Halteregisterdaten von demselben dezentralen Modbus-TCP-Server in dieselbe Adresse im Speicherbereich der Merker der lokalen CPU. Außerdem wird ein ausgegebener Fehler erfasst (optionale Funktion).

Netzwerk 1: Modbus-Funktion 3 - Wörter im Halteregister des Modbus-TCP-Servers lesen



Netzwerk 2: Modbus-Funktion 3 - Wörter im Halteregister des Modbus-TCP-Servers lesen



13.5.3 Modbus RTU

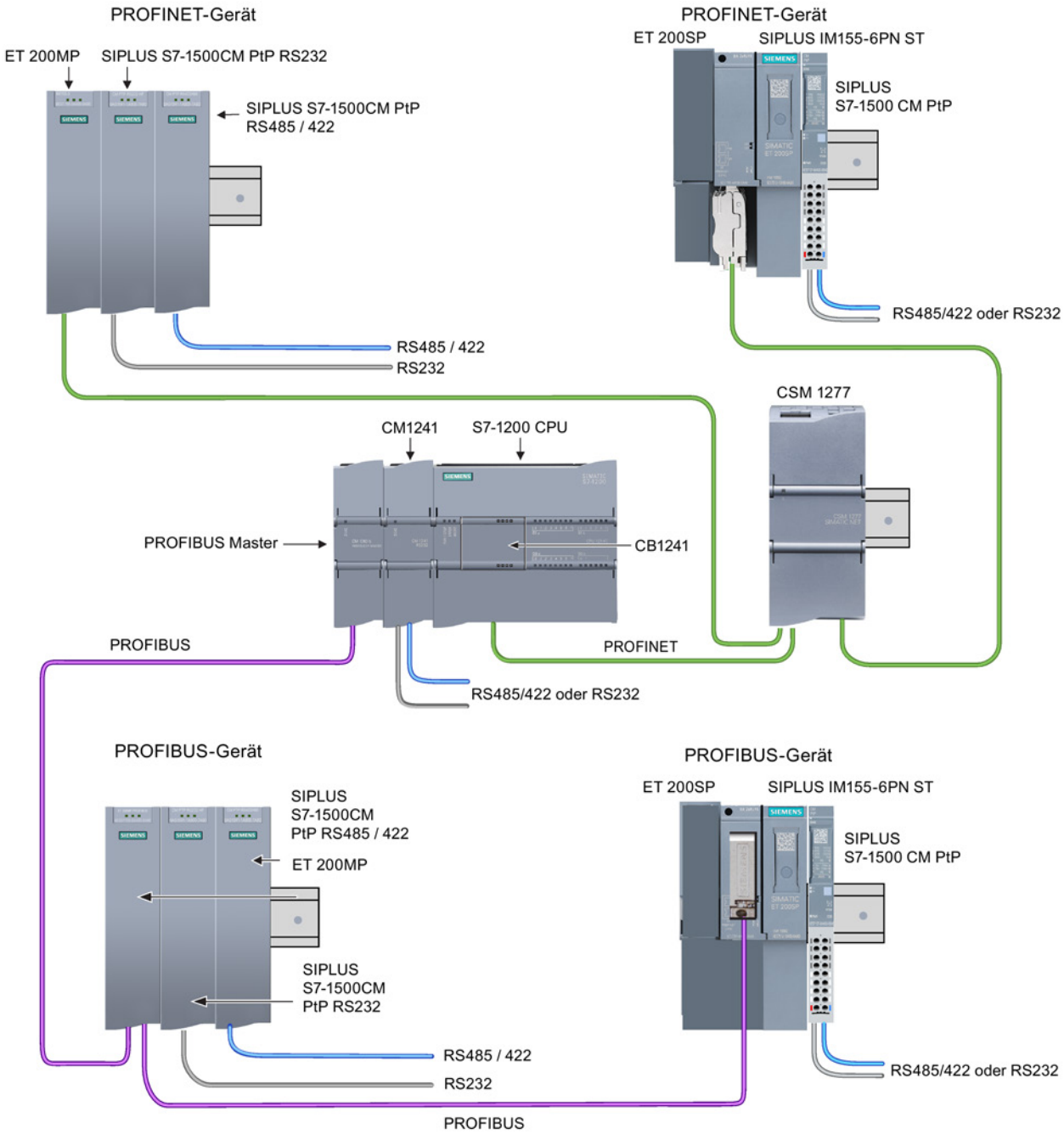
13.5.3.1 Übersicht

Version V4.1 der S7-1200 CPU in Verbindung mit STEP 7 V13 SP1 erweitert die Fähigkeit von Modbus RTU zur Nutzung eines dezentralen PROFINET- oder PROFIBUS-Peripheriebaugruppenträgers für die Kommunikation mit verschiedenen Geräten (RFID-Lesegeräte, GPS und andere):

- PROFINET (Seite 636): Die Ethernet-Schnittstelle der S7-1200 CPU wird mit einem PROFINET Schnittstellenmodul verbunden. Über PtP-Kommunikationsmodule im Baugruppenträger mit dem Schnittstellenmodul ist dann die serielle Kommunikation zu den PtP-Geräten möglich.
- PROFIBUS (Seite 782): Ein PROFIBUS-Kommunikationsmodul wird an der linken Seite des Baugruppenträgers mit der S7-1200 CPU eingesteckt. Das PROFIBUS-Kommunikationsmodul wird mit einem Baugruppenträger verbunden, der ein PROFIBUS-Schnittstellenmodul enthält. Über PtP-Kommunikationsmodule im Baugruppenträger mit dem Schnittstellenmodul ist dann die serielle Kommunikation zu den PtP-Geräten möglich.

Die S7-1200 unterstützt daher zwei Sätze PtP-Anweisungen:

- Ältere Modbus RTU-Anweisungen (Seite 1074): Diese Modbus RTU-Anweisungen wurden vor der Version V4.0 der S7-1200 benutzt und ermöglichen nur die serielle Kommunikation mit Hilfe eines Kommunikationsmoduls CM 1241 oder eines Kommunikationsboards CB 1241.
- Modbus RTU-Anweisungen (Seite 1001): Diese Modbus RTU-Anweisungen stellen alle Funktionen der alten Anweisungen bereit und zusätzlich die Möglichkeit zur Verbindung mit der dezentralen PROFINET- und PROFIBUS-Peripherie. Die PtP-Anweisungen ermöglichen das Konfigurieren der Kommunikation zwischen den PtP-Kommunikationsmodulen im Baugruppenträger mit der dezentralen Peripherie und den PtP-Geräten.



Hinweis

Ab der Version V4.1 der S7-1200 können PtP-Anweisungen für alle Arten der PtP-Kommunikation verwendet werden: seriell, seriell über PROFINET und seriell über PROFIBUS. STEP 7 stellt die alten PtP-Anweisungen nur zur Unterstützung vorhandener Programme bereit. Die alten Anweisungen funktionieren jedoch noch mit V4.1-CPUs sowie mit CPU-Versionen 4.0 und früher. Es ist nicht nötig, ältere Programme auf die neuen Anweisungen zu konvertieren.

13.5.3.2 Version der Modbus RTU-Anweisungen auswählen

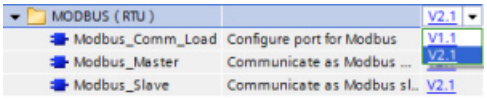
Es gibt zwei Versionen der Modbus RTU-Anweisungen in STEP 7:

- Version 1.1 war anfangs in STEP 7 Basic/Professional V13 verfügbar.
- Version 2.1 ist in STEP 7 Basic/Professional V13, SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Sie können nicht beide Versionen der Anweisungen bei demselben Modul verwenden, Sie können jedoch für zwei verschiedene Module unterschiedliche Versionen der Anwendungen einsetzen. Verwenden Sie niemals Anweisungen der Version 1.x und 2.y in einem CPU-Programm. Die Modbus RTU-Anweisungen in Ihrem Programm müssen dieselbe Versionsnummer haben (**1**.x, **2**.y oder **V**.z). Die einzelnen Anweisungen in einer Versionsgruppe können dann verschiedene Unterversionen aufweisen (1.**x**).

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.



Um die Version einer Modbus RTU-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine Modbus RTU-Anweisung über das Anweisungsverzeichnis in Ihr Programm einfügen, wird in der Projektnavigation eine neue FB-Instanz angelegt. Die neue FB-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer Modbus RTU-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation die Instanz eines Modbus RTU-FBs aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der Modbus RTU-Anweisung anzuzeigen.

13.5.3.3 Modbus RTU-Anweisungen

Anweisung Modbus_Comm_Load (SIPLUS I/O oder Port am PtP-Modul für Modbus RTU konfigurieren)

Tabelle 13- 67 Anweisung Modbus_Comm_Load

KOP/FUP	SCL	Beschreibung

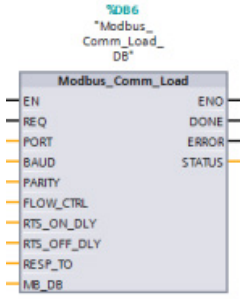
	`"Modbus_Comm_Load_DB"(`
	`REQ:=_bool_in,`
	`PORT:=_uint_in_,`
	`BAUD:=_udint_in_,`
	`PARITY:=_uint_in_,`
	`FLOW_CTRL:=_uint_in_,`
	`RTS_ON_DLY:=_uint_in_,`
	`RTS_OFF_DLY:=_uint_in_,`
	`RESP_TO:=_uint_in_,`
	`DONE=>_bool_out,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_DB:=_fbtref_inout_);`
		Die Anweisung Modbus_Comm_Load konfiguriert SIPLUS I/O oder einen PtP-Port für die Kommunikation über das Modbus-RTU-Protokoll.
		Hardwareoptionen des Modbus RTU-Ports: Einbau von max. drei CMs (RS485 oder RS232), plus eines CB (R4845).
		Optionen für Modbus RTU SIPLUS I/O: Installieren Sie die ET 200MP S7-1500CM PtP (RS485 / 422 oder RS232) oder ET 200SP S7-1500 CM PtP (RS485 / 422 oder RS232)
		Wenn Sie die Anweisung Modbus_Comm_Load in Ihr Programm einfügen, wird automatisch ein Instanz-Datenbaustein zugewiesen.

Tabelle 13- 68 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet. (Nur Version 2.0)
PORT	IN	Port	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen.
BAUD	IN	UDInt	Auswahl der Baudrate: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200. Alle anderen Werte sind ungültig.
PARITY	IN	UInt	Auswahl der Parität: • 0 – Keine • 1 – Ungerade • 2 – Gerade
FLOW_CTRL [1]	IN	UInt	Auswahl der Flusskontrolle: • 0 – (Standard) Keine Flusskontrolle • 1 – Hardware-Flusskontrolle mit RTS immer EIN (gilt nicht bei RS485-Ports) • 2 – Hardware-Flusskontrolle mit RTS geschaltet

Parameter und Datentyp		Datentyp	Beschreibung
RTS_ON_DLY [1]	IN	UInt	Auswahl RTS-Einschaltverzögerung: • 0 – (Standard) Keine Verzögerung von "RTS aktiv", bis das erste Zeichnung der Meldung gesendet wird. • 1 bis 65535 – Verzögerung in Millisekunden von "RTS aktiv", bis das erste Zeichen der Meldung gesendet wird (gilt nicht bei RS485-Ports). RTS-Verzögerungen müssen unabhängig von der Auswahl für FLOW_CTRL angewendet werden.
RTS_OFF_DLY [1]	IN	UInt	Auswahl RTS-Ausschaltverzögerung: • 0 – (Standard) Keine Verzögerung nach der Übertragung des letzten Zeichens bis "RTS inaktiv" • 1 bis 65535 – Verzögerung in Millisekunden von der Übertragung des letzten Zeichens bis "RTS inaktiv" (gilt nicht bei RS485-Ports). RTS-Verzögerungen müssen unabhängig von der Auswahl für FLOW_CTRL angewendet werden.
RESP_TO [1]	IN	UInt	Antwort-Timeout: Zeit in Millisekunden, die der Modbus_Master auf eine Antwort vom Slave wartet. Wenn der Slave während dieses Zeitraums nicht antwortet, wiederholt der Modbus_Master die Anforderung oder beendet die Anforderung mit einem Fehler, wenn die angegebene Anzahl von Wiederholungen gesendet wurde. 5 ms bis 65535 ms (Standardwert = 1000 ms).
MB_DB	IN	Variant	Ein Verweis auf den Instanz-Datenbaustein der Anweisungen Modbus_Master oder Modbus_Slave. Nachdem Sie Modbus_Master oder Modbus_Slave in Ihr Programm eingefügt haben, ist die DB-Kennung in der Parameter-Klappliste am Anschluss der MB_DB-Box verfügbar.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde. (Nur Version 2.0)
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung

[1] Optionale Parameter für Modbus_Comm_Load (V 2.x oder höher). Klicken Sie auf den Pfeil im unteren Bereich der KOP/FUP-Box, um die Box zu erweitern und diese Parameter aufzunehmen.

Modbus_Comm_Load wird ausgeführt, um einen Port für das Modbus-RTU-Protokoll zu konfigurieren. Nachdem ein Port für das Modbus-RTU-Protokoll konfiguriert ist, kann er nur von den Anweisungen Modbus_Master oder Modbus_Slave verwendet werden.

Für die Konfiguration jedes Kommunikationsports, der für die Modbus-Kommunikation eingesetzt wird, muss eine Ausführung von Modbus_Comm_Load genutzt werden. Sie müssen jedem Port, den Sie nutzen, einen eindeutigen Instanz-DB Modbus_Comm_Load zuweisen. Sie können bis zu drei Kommunikationsmodule (RS232 oder RS485) und ein Kommunikationsboard (RS485) in die CPU einbauen. Sie können Modbus_Comm_Load aus einem Anlauf-OB aufrufen und einmal ausführen, oder Sie können den Aufruf für die einmalige Ausführung mit Hilfe des Systemmerkers für den ersten Zyklus (Seite 106) initiieren. Führen Sie Modbus_Comm_Load nur dann erneut aus, wenn Kommunikationsparameter wie Baudrate oder Parität geändert werden müssen.

Wenn Sie die Modbus-Bibliothek mit einem Modul in einem dezentralen Baugruppenträger einsetzen, muss die Anweisung Modbus_Comm_Load in einem zyklischen Interruptprogramm ausgeführt werden (z. B. einmal pro Sekunde oder alle 10 Sekunden). Wenn die Spannungsversorgung des dezentralen Baugruppenträgers unterbrochen oder das Modul gezogen wird, wird bei Wiederkehr des Modulbetriebs nur der festgelegte HWConfig-Parameter an das PtP-Modul gesendet. Alle vom Modbus_Master-Timeout ausgelösten Anforderungen und der Modbus_Slave gehen in den stillen Modus (keine Reaktion auf Meldungen). Durch die zyklische Ausführung der Anweisung Modbus_Comm_Load werden diese Probleme behoben.

Wenn Sie Modbus_Master oder Modbus_Slave in Ihr Programm einfügen, wird der Anweisung ein Instanz-Datenbaustein zugewiesen. Dieser Instanz-Datenbaustein wird referenziert, wenn Sie in der Anweisung Modbus_Comm_Load den Parameter MB_DB angeben.

Modbus_Comm_Load Datenbausteinvariablen

Die folgende Tabelle zeigt die öffentlichen statischen Variablen im Instanz-DB von Modbus_Comm_Load, die in Ihrem Programm verwendet werden können.

Tabelle 13- 69 Statische Variablen im Instanz-DB

Variable	Datentyp	Beschreibung
ICHAR_GAP	UInt	Verzögerung für den Zeichenabstand zwischen den Zeichen. Dieser Parameter wird in Millisekunden angegeben und dient dazu, den erwarteten Zeitraum zwischen den empfangenen Zeichen zu erhöhen. Die entsprechende Anzahl Bitzeiten für diesen Parameter wird zum Modbus-Standardwert von 35 Bitzeiten (3,5 Zeichenzeiten) addiert.
RETRIES	UInt	Anzahl wiederholter Versuche, die der Master durchführt, bevor der Fehlercode "0x80C8" für "Keine Antwort" zurückgegeben wird.
STOP_BITS	USInt	Anzahl Stoppbits für jeden Frame eines Zeichens. Gültig sind die Werte 1 und 2.

Tabelle 13- 70 Bedingungscodes der Ausführung von Modbus_Comm_Load [1]

STATUS (W#16#)	Beschreibung
0000	Kein Fehler
8180	Ungültiger Wert der Port-ID (falscher Port/falsche Hardwarekennung des Kommunikationsmoduls)
8181	Ungültiger Wert für die Baudrate.
8182	Ungültiger Wert für die Parität.
8183	Ungültiger Wert für die Flusskontrolle.
8184	Ungültiger Wert für die Timeout-Zeit der Antwort (Antwort-Timeout ist kleiner als der Mindestwert von 5 ms)
8185	Der Parameter MB_DB ist kein Instanz-Datenbaustein einer Anweisung Modbus_Master oder Modbus_Slave.

[1] Neben den oben aufgeführten Fehlern der Anweisung Modbus_Comm_Load können auch die zugrunde liegenden PtP-Kommunikationsanweisungen Fehler zurückgeben.

Anweisung Modbus_Master (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Master kommunizieren)

Tabelle 13- 71 Anweisung Modbus_Master

KOP/FUP	SCL	Beschreibung

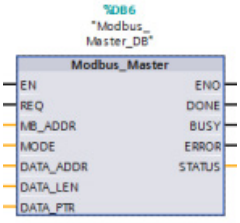
	`"Modbus_Master_DB"(`
	`REQ:=_bool_in_,`
	`MB_ADDR:=_uint_in_,`
	`MODE:=_usint_in_,`
	`DATA_ADDR:=_udint_in_,`
	`DATA_LEN:=_uint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`DATA_PTR:=_variant_inout_);`
		Die Anweisung Modbus_Master kommuniziert als Modbus-Master über einen Port, der von einer vorherigen Ausführung der Anweisung Modbus_Comm_Load konfiguriert wurde. Wenn Sie die Anweisung Modbus_Master in Ihr Programm einfügen, wird automatisch ein Instanz-Datenbaustein zugewiesen. Dieser Instanz-Datenbaustein Modbus_Master wird verwendet, wenn Sie in der Anweisung Modbus_Comm_Load den Parameter MB_DB angeben.

Tabelle 13- 72 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	0 = Keine Anforderung 1 = Anforderung, Daten an den Modbus-Slave zu senden
MB_ADDR	IN	V1.0: USInt V2.0: UInt	Modbus-RTU-Stationsadresse: Standard-Adressierungsbereich (1 bis 247) Erweiterter Adressierungsbereich (1 bis 65535) Der Wert 0 ist für den Broadcast einer Meldung an alle Modbus-Slaves reserviert. Für den Broadcast werden nur die Modbus-Funktionscodes 05, 06, 15 und 16 unterstützt.
MODE	IN	USInt	Auswahl Modus: Gibt die Art der Anforderung an (Lesen, Schreiben oder Diagnose). Weitere Informationen finden Sie in der Tabelle der Modbus-Funktionen unten.
DATA_ADDR	IN	UDInt	Anfangsadresse im Slave: Gibt die Anfangsadresse der Daten an, auf die im Modbus-Slave zugegriffen werden soll. Die gültigen Adressen finden Sie in der Tabelle der Modbus-Funktionen unten.
DATA_LEN	IN	UInt	Datenlänge: Gibt die Anzahl der Bits oder Wörter an, auf die diese Anforderung zugreifen soll. Die gültigen Längen finden Sie in der Tabelle der Modbus-Funktionen unten.
DATA_PTR	IN	Variant	Daten-Pointer: Zeigt auf die M- oder DB-Adresse (Standard-DB) für die zu schreibenden oder zu lesenden Daten.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
BUSY	OUT	Bool	• 0 - Kein Modbus_Master-Vorgang in Bearbeitung • 1 – Modbus_Master-Vorgang in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung

Regeln für die Modbus_Master-Kommunikation

- MB_COMM_LOAD muss ausgeführt werden, um einen Port zu konfigurieren, damit die Anweisung Modbus_Master mit diesem Port kommunizieren kann.
- Wenn ein Port zum Initiieren von Modbus-Masteranforderungen verwendet werden soll, darf dieser Port nicht von MB_SLAVE verwendet werden. Eine oder mehrere Instanzen von Modbus_Master können mit diesem Port verwendet werden, jedoch müssen alle Ausführungen von Modbus_Master den gleichen Instanz-DB Modbus_Master für den Port verwenden.
- Die Modbus-Anweisungen nutzen keine Kommunikations-Alarmereignisse zum Steuern des Kommunikationsprozesses. Ihr Programm muss die Anweisung Modbus_Master auf abgeschlossene Sende- und Empfangsvorgänge abfragen.
- Es wird empfohlen, die Ausführung von Modbus_Master für einen bestimmten Port immer aus einem Programmzyklus-OB aufzurufen. Modbus_Master-Anweisungen können nur in jeweils einem Programmzyklus oder in jeweils einer zyklischen/zeitverzögerten Ausführungsschicht ausgeführt werden. Sie dürfen nicht in beiden Prioritätsschichten der Ausführung bearbeitet werden. Die Vorrangunterbrechung einer Modbus_Master-Anweisung durch eine andere Modbus_Master-Anweisung in einer Ausführungsschicht mit höherer Priorität führt zu nicht ordnungsgemäßem Betrieb. Die Modbus_Master-Anweisungen dürfen nicht in Anlauf-, Diagnose- oder Zeitfehlerausführungsschichten bearbeitet werden.
- Nachdem eine Modbus_Master-Anweisung eine Übertragung ausgelöst hat, muss diese Instanz mit aktiviertem Eingang EN fortlaufend ausgeführt werden, bis der Zustand DONE = 1 oder ERROR = 1 ausgegeben wird. Eine bestimmte Modbus_Master-Instanz wird als aktiv betrachtet, bis eines dieser beiden Ereignisse auftritt. Während die ursprüngliche Instanz aktiv ist, führt jeder Aufruf einer weiteren Instanz mit aktiviertem Eingang REQ zu einem Fehler. Wenn die fortlaufende Ausführung der ursprünglichen Instanz stoppt, bleibt der Anforderungszustand für die von der statischen Variable "Blocked_Proc_Timeout" angegebene Zeitdauer aktiv. Nach Ablauf dieser Zeitdauer wird die nächste aufgerufene Modbus_Master-Anweisung mit aktiviertem Eingang REQ zur aktiven Instanz. Dies verhindert, dass eine einzige Modbus_Master-Anweisung ein Monopol innehat oder den Zugriff auf den Port sperrt. Wenn die ursprüngliche aktive Instanz nicht während des von der statischen Variable "Blocked_Proc_Timeout" angegebenen Zeitraums aktiviert wird, löscht die nächste Ausführung von dieser Instanz (ohne gesetzten Eingang REQ) den aktiven Zustand. Falls REQgesetzt ist, löst diese Ausführung eine neue Modbus_Master-Anforderung aus, so als wenn keine andere Instanz aktiv wäre.

Parameter REQ

0 = Keine Anforderung; 1 = Anforderung, Daten an den Modbus-Slave zu senden

Sie können diesen Eingang über einen pegel- oder flankengesteuerten Kontakt steuern. Immer wenn dieser Eingang aktiviert wird, wird eine Zustandsmaschine gestartet, um sicherzustellen, dass eine andere Anweisung Modbus_Master, die denselben Instanz-DB verwendet, erst dann eine Anforderung abgeben kann, wenn die aktuelle Anforderung abgearbeitet ist. Alle anderen Eingangszustände werden erfasst und intern für die aktuelle Anforderung gespeichert, bis die Antwort empfangen oder ein Fehler erkannt wird.

Wird dieselbe Instanz von Modbus_Master erneut mit Eingang REQ = 1 ausgeführt, bevor die aktuelle Anforderung abgearbeitet ist, werden keine nachfolgenden Übertragungen durchgeführt. Wenn die Anforderung jedoch abgearbeitet ist, wird zu dem Zeitpunkt, zu dem Modbus_Master erneut mit REQ-Eingang = 1 ausgeführt wird, eine neue Anforderung abgegeben.

Über die Parameter DATA_ADDR und MODE wählen Sie den Modbus-Funktionstyp aus.

DATA_ADDR (Modbus-Anfangsadresse im Slave): Gibt die Anfangsadresse der Daten an, auf die im Modbus-Slave zugegriffen werden soll.

Die Anweisung Modbus_Master nutzt statt eines Funktionscodeeingangs einen Eingang MODE. Die Kombination aus MODE und Modbus-Adresse legt den Funktionscode fest, der in der eigentlichen Modbus-Meldung verwendet wird. Die folgende Tabelle zeigt die Entsprechung zwischen dem Parameter MODE, dem Modbus-Funktionscode und dem Modbus-Adressbereich.

Tabelle 13- 73 Modbus-Funktionen

MODE	Modbus-Funktion	Länge der Daten	Operation und Daten	Modbus-Adresse
0	01	1 bis 2000 1 bis 1992 [1]	Ausgangsbits lesen: 1 bis (1992 oder 2000) Bit pro Anforderung	1 bis 9999
0	02	1 bis 2000 1 bis 1992 [1]	Eingangsbits lesen: 1 bis (1992 oder 2000) Bit pro Anforderung	10001 bis 19999
0	03	1 bis 125 1 bis 124 [1]	Halteregister lesen: 1 bis (124 oder 125) Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535
0	04	1 bis 125 1 bis 124 [1]	Eingangswörter lesen: 1 bis (124 oder 125) Bits pro Anforderung	30001 bis 39999
104	04	1 bis 125 1 bis 124 1	Eingangswörter lesen: 1 bis (124 oder 125) Bits pro Anforderung	00000 bis 65535
1	05	1	Ein Ausgangsbit schreiben: Ein Bit pro Anforderung	1 bis 9999
1	06	1	Ein Halteregister schreiben: 1 Wort pro Anforderung	40001 bis 49999 oder 400001 bis 465535
1	15	2 bis 1968 2 bis 1960 [1]	Mehrere Ausgangsbits schreiben: 2 bis (1960 oder 1968) Bit pro Anforderung	1 bis 9999
1	16	2 bis 123 2 bis 122 [1]	Mehrere Halteregister schreiben: 2 bis (122 oder 123) Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535
2	15	1 bis 1968 2 bis 1960 [1]	Ein oder mehrere Ausgangsbits schreiben: 1 bis (1960 oder 1968) Bit pro Anforderung	1 bis 9999

MODE	Modbus-Funktion	Länge der Daten	Operation und Daten	Modbus-Adresse
2	16	1 bis 123 1 bis 122 [1]	Ein oder mehrere Halteregister schreiben: 1 bis (122 oder 123) Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535
11	11	0	Statuswort und Ereigniszähler der Slave-Kommunikation lesen. Das Statuswort zeigt beschäftigt an (0 – nicht beschäftigt, 0xFFFF - beschäftigt). Der Ereigniszähler wird bei jeder erfolgreichen Abarbeitung einer Meldung inkrementiert. Beide Operanden DATA_ADDR und DATA_LEN von Modbus_Master werden bei dieser Funktion ignoriert.	
80	08	1	Slave-Zustand über Daten-Diagnosecode 0x0000 prüfen (Prüfschleifentest – Slave gibt ein Echo der Anforderung zurück) 1 Wort pro Anforderung	
81	08	1	Slave-Ereigniszähler über Daten-Diagnosecode 0x000A zurücksetzen 1 Wort pro Anforderung	
3 bis 10, 12 bis 79, 82 bis 255			Reserviert	

[1] Bei der erweiterten Adressierung ist die maximale Datenlänge je nach Datentyp der Funktion um 1 Byte oder 1 Wort geringer.

Parameter DATA_PTR

Der Parameter DATA_PTR zeigt auf die DB- oder M-Adresse, in die geschrieben oder aus der gelesen wird. Wenn Sie einen Datenbaustein verwenden, müssen Sie einen globalen Datenbaustein anlegen, der den Datenspeicher für Lese- und Schreibvorgänge auf Modbus-Slaves bereitstellt.

Hinweis

Der Datenbaustein DATA_PTR muss die direkte Adressierung zulassen

Der Datenbaustein muss die direkte (absolute) und die symbolische Adressierung zulassen. Wenn Sie den Datenbaustein anlegen, müssen Sie das Zugriffsattribut "Standard" auswählen.

Datenbausteinstrukturen für den Parameter DATA_PTR

- Diese Datentypen gelten für das **Lesen von Wörtern** der Modbus-Adressen 30001 bis 39999, 40001 bis 49999 und 400001 bis 465536 sowie für das **Schreiben von Wörtern** in die Modbus-Adressen 40001 bis 49999 und 400001 bis 465536.
  - Standard-Array der Datentypen WORD, UINT oder INT
  - Benannte Struktur vom Typ WORD, UINT oder INT, bei der jedes Element einen eindeutigen Namen und einen 16-Bit-Datentyp hat.
  - Benannte komplexe Struktur, bei der jedes Element einen eindeutigen Namen und einen 16- oder 32-Bit-Datentyp hat.
- Zum **Lesen** und Schreiben von Bits für die Modbus-Adressen 00001 bis 09999 und zum Lesen von Bits von 10001 bis 19999.
  - Standardfeld aus Booleschen Datentypen.
  - Benannte Boolesche Struktur aus eindeutig benannten Booleschen Variablen.
- Es ist zwar nicht erforderlich, doch empfehlenswert, dass jede Anweisung Modbus_Master einen eigenen getrennten Speicherbereich hat. Grund hierfür ist, dass die Möglichkeit der Datenzerstörung größer ist, wenn mehrere Anweisungen Modbus_Master in demselben Speicherbereich lesen und schreiben.
- Es ist nicht erforderlich, dass die Datenbereiche für DATA_PTR in demselben globalen Datenbaustein liegen. Sie können einen Datenbaustein mit mehreren Bereichen für Modbus-Lesevorgänge, einen Datenbaustein für Modbus-Schreibvorgänge oder einen Datenbaustein für jede Slave-Station anlegen.

Variablen im Datenbaustein des Modbus-Masters

Die folgende Tabelle zeigt die öffentlichen statischen Variablen im Instanz-DB des Modbus_Master, die Sie in Ihrem Programm verwenden können.

Tabelle 13- 74 Statische Variablen im Instanz-DB

Variable	Datentyp	Standard	Beschreibung
Blo-cked_Proc_Tim eout	Real	3.0	Zeitdauer (in Sekunden), die auf eine blockierte Modbus_Master-Instanz gewartet werden soll, bevor diese Instanz als AKTIV entfernt wird. Dies kann beispielsweise vorkommen, wenn eine Modbus_Master-Anforderung ausge-geben wurde und das Programm dann stoppt und die Modbus_Master-Funktion aufruft, bevor es die Anforderung vollständig beendet hat. Der Zeit-wert muss größer als 0 und kleiner als 55 Sekunden sein, ansonsten tritt ein Fehler auf.
Exten-ded_Addressing	Bool	FALSCH	Konfiguriert die Adressierung der Slave-Station als einzelnes oder doppeltes Byte: • FALSCH = 1-Byte-Adresse, 0 bis 247 • WAHR = 2-Byte-Adresse (entspricht der erweiterten Adressierung), 0 bis 65535
MD_DB	MB_BAS E	-	Der Parameter MB_DB der Anweisung Modbus_Comm_Load muss mit dem Parameter MB_DB der Anweisung Modnbus_Master verbunden werden.

Ihr Programm kann in die Variablen Blocked_Proc_Timeout und Extended_Addressing Werte schreiben, um die Modbus_Master-Operationen zu steuern. Im Abschnitt zur Anweisung MB_SLAVE finden Sie in der Beschreibung der Variablen HR_Start_Offset und Extended_Addressing ein Beispiel dafür, wie Sie diese Variablen im Programmiereditor nutzen, sowie weitere Einzelheiten zu erweiterten Modbus-Adressierung (Seite 1011).

Bedingungscodes

Tabelle 13- 75 Bedingungscodes für die Ausführung von Modbus_Master (Kommunikations- und Konfigurationsfehler) [1]

STATUS (W#16#)	Beschreibung
0000	Kein Fehler
80C8	Slave-Timeout. Der angegebene Slave hat nicht während der angegebenen Zeit geantwortet. Prüfen Sie die Baudrate, Parität und Verdrahtung des Slavegeräts. Dieser Fehler wird erst gemeldet, nachdem alle konfigurierten Wiederholungen durchgeführt wurden.
80C9	Die Zeit der Anweisung Modbus_Master ist aus einem der folgenden Gründe abgelaufen: • Die Anweisung wartet auf eine Antwort von dem Modul, das für die Kommunikation verwendet wird. • Der Wert von Blocked_Proc_Timeout ist zu klein festgelegt. Dieser Fehler wird gemeldet, wenn ein dezentrales PROFIBUS oder PROFINET E/A-Gerät aus einem der folgenden Zustände zurückkehrt: • Eine Unterbrechung der Spannung oder Kommunikation • Ein Ziehen/Stecken-Ereignis bei einem Kommunikationsmodul In diesen Fällen wird die Hardwarekonfiguration vom PLC neu geladen und die Anweisung Modbus_Comm_Load muss erneut ausgeführt werden, um das Kommunikationsmodul richtig zu konfigurieren.
80D1	Der Empfänger hat eine Flusskontrolle zur Unterbrechung einer laufenden Sendung angefordert, aber die Sendung innerhalb der Wartezeit nicht wieder aktiviert. Dieser Fehler wird auch bei der Hardware-Flusskontrolle erzeugt, wenn der Empfänger innerhalb der Wartezeit kein CTS erkennt.
80D2	Die Sendeanforderung wurde abgebrochen, weil vom DCE kein DSR-Signal empfangen wird.
80E0	Die Meldung wurde beendet, weil der Empfangspuffer voll ist.
80E1	Die Meldung wurde wegen eines Paritätsfehlers beendet.
80E2	Die Meldung wurde wegen eines Telegrammfehlers beendet.
80E3	Die Meldung wurde wegen eines Überlauffehlers beendet.
80E4	Die Meldung wurde abgebrochen, weil die angegebene Länge die Gesamtpuffergröße überschreitet.
8180	Ungültiger Wert für die Port-ID oder Fehler bei der Anweisung Modbus_Comm_Load
8186	Ungültige Modbus-Stationsadresse
8188	Ungültiger Modus für die Broadcast-Anforderung angegeben
8189	Ungültiger Wert für die Datenadresse.
818A	Ungültiger Wert für die Datenlänge.
818B	Ungültiger Pointer auf die lokale Datenquelle bzw. das lokale Datenziel: Größe nicht richtig

STATUS (W#16#)	Beschreibung
818C	Ungültiger Pointer für DATA_PTR oder Blocked_Proc_Timeout ungültig. Der Datenbereich muss einer der folgenden sein: • Klassischer DB • Array elementarer Datentypen in einem symbolischen oder remanenten DB • Merker
8200	Port ist durch Bearbeitung einer Sendeanforderung belegt.
8280	Negative Quittierung beim Lesen aus Modul. Prüfen Sie den Eingang am Parameter PORT. Dieser Fehler kann durch Verlust eines dezentralen PROFIBUS oder PROFINET E/A-Moduls entweder aufgrund eines Spannungs- oder Kommunikationsverlusts oder aufgrund des Ziehens eines Moduls verursacht werden.
8281	Negative Quittierung beim Schreiben ins Modul. Prüfen Sie den Eingang am Parameter PORT. Dieser Fehler kann durch Verlust eines dezentralen PROFIBUS oder PROFINET E/A-Moduls entweder aufgrund eines Spannungs- oder Kommunikationsverlusts oder aufgrund des Ziehens eines Moduls verursacht werden.

Tabelle 13- 76 Bedingungscodes für die Ausführung von Modbus_Master (Modbus-Protokollfehler) [1]

STATUS (W#16#)	Antwortcode vom Slave	Modbus-Protokollfehler
8380	-	CRC-Fehler
8381	01	Funktionscode nicht unterstützt
8382	03	Fehler in der Datenlänge
8383	02	Datenadressfehler oder Adresse außerhalb des gültigen Bereichs von DATA_PTR
8384	Größer als 03	Fehler im Datenwert
8385	03	Wert des Datendiagnosecodes wird nicht unterstützt (Funktionscode 08)
8386	-	Funktionscode in der Antwort entspricht nicht dem Code in der Anforderung.
8387	-	Der falsche Slave hat reagiert.
8388	-	Die Slave-Antwort auf eine Schreibanforderung ist inkorrekt. Die vom Slave ausgegebene Schreibanforderung entspricht nicht dem, was der Master gesendet hat.

[1] Neben den oben aufgeführten Fehlern der Anweisung Modbus_Master können auch die zugrunde liegenden PtP-Kommunikationsanweisungen Fehler zurückgeben.

Anweisung Modbus_Slave (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Slave kommunizieren)

Tabelle 13- 77 Anweisung Modbus_Slave

KOP/FUP	SCL	Beschreibung

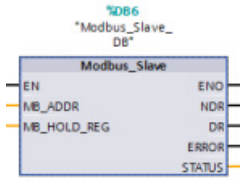
	`"Modbus_Slave_DB"(`
	`MB_ADDR:=_uint_in_,`
	`NDR=>_bool_out_,`
	`DR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_HOLD_REG:=_variant_inout_);`
		Mit der Anweisung Modbus_Slave kann Ihr Programm auf zweierlei Art kommunizieren:
		• Als Modbus RTU-Slave über einen PtP-Port des CM (RS485 oder RS232) und CB (RS485)
		• Als Modbus RTU-Slave über die Modbus RTU SIPLUS I/O-Optionen:
		– ET 200MP S7-1500CM PtP (RS485 / 422 oder RS232) installieren.
		– ET 200SP S7-1500 CM PtP (RS485 / 422 oder RS232) installieren.
		Wenn ein dezentraler Modbus-RTU-Master eine Anforderung ausgibt, reagiert Ihr Anwenderprogramm auf die Anforderung, indem es die Anweisung Modbus_Slave ausführt. STEP 7 erstellt automatisch einen Instanz-DB, wenn Sie die Anweisung einfügen. Sie geben mit diesem Modbus_Slave_DB-Namen den Parameter MB_DB für die Anweisung Modbus_Comm_Load an.

Tabelle 13- 78 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
MB_ADDR	IN	V1.0: USInt V2.0: UInt	Stationsadresse des Modbus-Slaves: Standard-Adressierungsbereich (1 bis 247) Erweiterter Adressierungsbereich (0 bis 65535)
MB_HOLD_REG	IN	Variant	Pointer auf den Modbus-Halteregister-DB: Bei dem Modbus-Halteregister kann es sich um den Speicherbereich der Merker oder um einen Datenbaustein handeln.
NDR	OUT	Bool	Neue Daten bereit: • 0 – Keine neuen Daten • 1 – Gibt an, dass neue Daten vom Modbus-Master geschrieben wurden
DR	OUT	Bool	Daten lesen: • 0 – Keine Daten gelesen • 1 – Gibt an, dass Daten vom Modbus-Master gelesen wurden

Parameter und Datentyp		Datentyp	Beschreibung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Wenn die Ausführung mit einem Fehler beendet wird, ist der Fehlercode im Parameter STATUS nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Fehlercode der Ausführung

Die Funktionscodes der Modbus-Kommunikation (1, 2, 4, 5 und 15) können Bits und Wörter direkt im Prozessabbild der Eingänge und im Prozessabbild der Ausgänge der CPU lesen und schreiben. Bei diesen Funktionscodes muss der Parameter MB_HOLD_REG als Datentyp größer als ein Byte definiert werden. Die folgende Tabelle zeigt die Beispielzuordnung der Modbus-Adressen zum Prozessabbild in der CPU.

Tabelle 13- 79 Zuordnung der Modbus-Adressen zum Prozessabbild

Modbus-Funktionen						S7-1200	
Codes	Funktion	Datenbereich	Adressbereich			Datenbereich	CPU-Adresse
01	Bits lesen	Ausgang	1	bis	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7
02	Bits lesen	Eingang	10001	bis	18192	Prozessabbild der Eingänge	E0.0 bis E1023.7
04	Wörter lesen	Eingang	30001	bis	30512	Prozessabbild der Eingänge	EW0 bis EW1022
05	Bit schreiben	Ausgang	1	bis	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7
15	Bits schreiben	Ausgang	1	bis	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7

Die Funktionscodes der Modbus-Kommunikation (3, 6, 16) nutzen ein Modbus-Halteregister, bei dem es sich um einen Adressbereich im Speicherbereich der Merker oder um einen Datenbaustein handeln kann. Der Typ des Halteregisters wird vom Parameter MB_HOLD_REG der Anweisung Modbus_Slave angegeben.

Hinweis

Typ des Datenbausteins MB_HOLD_REG

Ein Datenbaustein mit Modbus-Halteregister muss die direkte (absolute) und die symbolische Adressierung zulassen. Wenn Sie den Datenbaustein anlegen, müssen Sie das Zugriffsattribut "Standard" auswählen.

Die folgende Tabelle zeigt Beispiele für die Zuordnung von Modbus-Adressen zum Halteregister, das für die Modbus-Funktionscodes 03 (Wörter lesen), 06 (Wort schreiben) und 16 (Wörter schreiben) verwendet wird. Die tatsächliche obere Grenze der DB-Adressen wird vom maximalen Arbeitsspeicher und vom maximalen M-Speicher des jeweiligen CPU-Modells festgelegt.

Tabelle 13- 80 Zuordnung von Modbus-Adressen zum CPU-Speicher

Modbus-Master-Adresse	Beispiele für Parameter von MB_HOLD_REG				
	MW100	DB10.DBw0	MW120	DB10.DBW50	"Rezept".Inhaltsstoff
40001	MW100	DB10.DBW0	MW120	DB10.DBW50	"Rezept".Inhaltsstoff[1]
40002	MW102	DB10.DBW2	MW122	DB10.DBW52	"Rezept".Inhaltsstoff[2]
40003	MW104	DB10.DBW4	MW124	DB10.DBW54	"Rezept".Inhaltsstoff[3]
40004	MW106	DB10.DBW6	MW126	DB10.DBW56	"Rezept".Inhaltsstoff[4]
40005	MW108	DB10.DBW8	MW128	DB10.DBW58	"Rezept".Inhaltsstoff[5]

Tabelle 13- 81 Diagnosefunktionen

Modbus-Diagnosefunktionen von Modbus_Slave der S7-1200		
Codes	Teilfunktion	Beschreibung
08	0000H	Echotest Abfragedaten ausgeben: Die Anweisung Modbus_Slave gibt einem Modbus-Master das Echo eines empfangenen Datenworts zurück.
08	000AH	Kommunikationsereigniszähler löschen: Die Anweisung Modbus_Slave löscht den Kommunikationsereigniszähler, der für Modbus-Funktion 11 verwendet wird.
11		Kommunikationsereigniszähler abrufen: Die Anweisung Modbus_Slave nutzt einen internen Kommunikationsereigniszähler, um die Anzahl erfolgreicher Modbus-Lese- und -Schreibanforderungen, die an den Modbus_Slave gesendet werden, zu erfassen. Der Zähler wird bei Funktion 8, Funktion 11 und Broadcast-Anforderungen nicht hochgezählt. Außerdem wird bei Anforderungen, die zu Kommunikationsfehlern führen (z.B. Paritäts- oder CRC-Fehler), nicht hochgezählt.

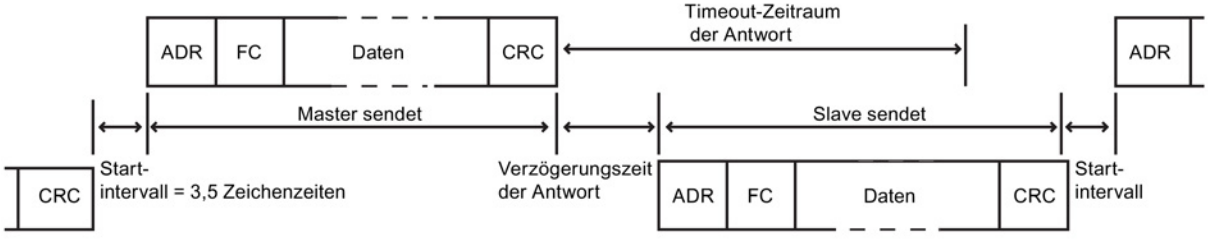
Die Anweisung Modbus_Slave unterstützt Broadcast-Schreibanforderungen von Modbus-Mastern, solange die Anforderungen den Zugriff auf gültige Adressen umfassen. Für Funktionscodes, die bei der Broadcast-Funktion nicht unterstützt werden, erzeugt Modbus_Slave den Fehlercode "0x8188".

Regeln für die Modbus_Slave-Kommunikation

- Modbus_Comm_Load muss ausgeführt werden, um einen Port zu konfigurieren, damit die Anweisung Modbus_Slave über diesen Port kommunizieren kann.
- Wenn ein Port als Slave auf einen Modbus_Master reagieren soll, darf dieser Port nicht mit der Anweisung Modbus_Master programmiert werden.
- Nur eine Instanz von Modbus_Slave kann mit einem bestimmten Port verwendet werden, andernfalls kann unerwartetes Verhalten auftreten.
- Die Modbus-Anweisungen nutzen keine Kommunikations-Alarmereignisse zum Steuern des Kommunikationsprozesses. Ihr Programm muss den Kommunikationsprozess durch Abfrage der Anweisung Modbus_Slave auf abgeschlossene Sende- und Empfangsvorgänge steuern.
- Die Anweisung Modbus_Slave muss regelmäßig mit einer Häufigkeit ausgeführt werden, die eine zeitnahe Antwort auf eingehende Anforderungen eines Modbus_Masters ermöglicht. Es wird empfohlen, Modbus_Slave in jedem Zyklus aus einem Programmzyklus-OB auszuführen. Modbus_Slave kann aus einem Weckalarm-OB ausgeführt werden, doch ist dies nicht empfehlenswert, weil übermäßige Zeitverzögerungen im Interruptprogramm die Ausführung anderer Interruptprogramme temporär blockieren können.

Zeitsteuerung des Modbus-Signals

Modbus_Slave muss regelmäßig ausgeführt werden, um jede Anforderung des Modbus_Masters zu empfangen und entsprechend zu antworten. Die Häufigkeit der Ausführung von Modbus_Slave richtet sich nach dem vom Modbus_Master vorgegebenen Timeout-Wert für die Antwort. Dies ist in der nachstehenden Abbildung dargestellt.



Der Timeout-Zeitraum für die Antwort RESP_TO ist die Zeitdauer, die ein Modbus_Master auf den Beginn einer Antwort von einem Modbus_Slave wartet. Dieser Zeitraum wird nicht vom Modbus-Protokoll definiert, sondern von einem Parameter des jeweiligen Modbus_Masters. Die Häufigkeit der Ausführung (die Zeit zwischen einer Ausführung und der nächsten Ausführung) von Modbus_Slave muss sich nach den jeweiligen Parametern des Modbus_Masters richten. Sie sollten Modbus_Slave mindestens zweimal während des Timeout-Zeitraums für die Antwort des Modbus_Masters ausführen.

Modbus_Slave-Variablen

Diese Tabelle zeigt die öffentlichen statischen Variablen im Instanz-Datenbaustein von Modbus_Slave, die in Ihrem Programm verwendet werden können.

Tabelle 13- 82 Modbus_Slave-Variablen

Variable	Datentyp	Beschreibung
Request_Count	Word	Die Anzahl aller von diesem Slave empfangenen Anforderungen
Slave_Message_Count	Word	Die Anzahl der für diesen spezifischen Slave empfangenen Anforderungen
Bad_CRC_Count	Word	Die Anzahl der empfangenen Anforderungen, die einen CRC-Fehler aufweisen
Broadcast_Count	Word	Die Anzahl der empfangenen Broadcast-Anforderungen
Exception_Count	Word	Modbus-spezifische Fehler, die eine zurückgegebene Ausnahme erfordern
Success_Count	Word	Die Anzahl der für diesen spezifischen Slave empfangenen Anforderungen ohne Protokollfehler
HR_Start_Offset	Word	Gibt die Anfangsadresse des Modbus-Halteregisters an (Standard = 0)
Extended_Addressing	Bool	Konfiguriert die Slave-Adressierung als einzelnes oder doppeltes Byte (0 = einzelne Byte-Adresse, 1 = doppelte Byte-Adresse, Standard = 0)

Ihr Programm kann in die Variablen HR_Start_Offset und Extended_Addressing Werte schreiben und die Modbus-Slave-Operationen steuern. Die anderen Variablen können gelesen werden, um den Modbus-Zustand zu überwachen.

HR_Start_Offset

Die Adressen des Modbus-Halteregisters beginnen bei 40001 oder 400001. Diese Adressen entsprechen der Anfangsadresse des Halteregisters im Zielsystemspeicher. Sie können jedoch die Variable HR_Start_Offset konfigurieren, um eine andere Anfangsadresse als 40001 oder 400001 für das Modbus-Halteregister zu konfigurieren.

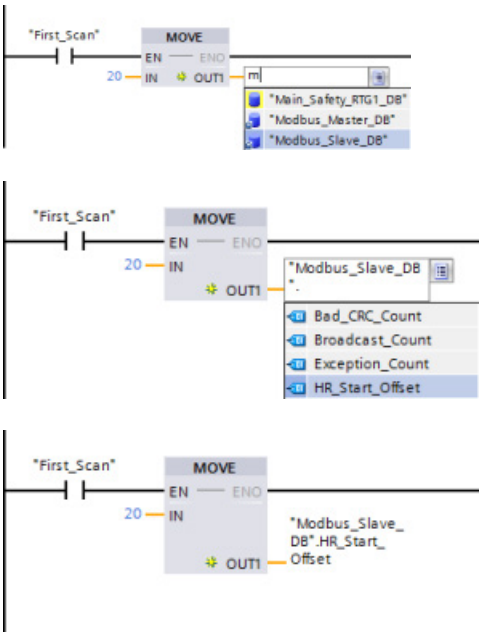
Sie können z. B. ein Halteregister mit Beginn an MW100 und einer Länge von 100 Wörtern konfigurieren. Mit einem Versatz von 20 geben Sie eine Anfangsadresse des Halteregisters von 40021 statt 40001 an. Jede Adresse unter 40021 und über 400119 führt zu einem Adressierungsfehler.

Tabelle 13- 83 Beispiel für die Adressierung des Modbus-Halteregisters

HR_Start_Offset	Adresse	Minimum	Maximum
0	Modbus-Adresse (Wort)	40001	40099
	S7-1200 Adresse	MW100	MW298
20	Modbus-Adresse (Wort)	40021	40119
	S7-1200 Adresse	MW100	MW298

HR_Start_Offset ist ein Wortwert, der die Anfangsadresse des Modbus-Halteregisters angibt und im Instanz-Datenbaustein Modbus_Slave gespeichert ist. Sie können diese öffentliche statische Variable über die Parameter-Klappliste auswählen, nachdem Sie Modbus_Slave in Ihr Programm eingefügt haben.

Wenn Sie beispielsweise Modbus_Slave in ein KOP-Netzwerk eingefügt haben, können Sie in ein vorheriges Netzwerk gehen und den Wert HR_Start_Offset zuweisen. Der Wert muss vor der Ausführung von Modbus_Slave zugewiesen werden.



Modbus-Slave-Variable über den Standard-DB-Namen eingeben:

1. Positionieren Sie den Cursor im Parameterfeld und geben Sie das Zeichen m ein.
2. Wählen Sie in der Klappliste "Modbus_Slave_DB" aus.
3. Positionieren Sie den Cursor rechts vom DB-Namen (nach den Anführungszeichen) und geben Sie einen Punkt ein.
4. Wählen Sie in der Klappliste "Modbus_Slave_DB.HR_Start_Offset" aus.

Extended_Addressing

Auf die Extended_Addressing-Variable wird in ähnlicher Weise zugegriffen wie auf die oben beschriebene HR_Start_Offset-Referenz, außer dass es sich bei der Extended_Addressing-Variable um einen Booleschen Wert handelt. Der Boolesche Wert muss von einer Ausgangsspule und kann nicht von einer Box "Verschieben" geschrieben werden.

Für die Modbus-Slave-Adressierung kann ein einzelnes Byte (dies ist der Modbus-Standard) oder ein doppeltes Byte konfiguriert werden. Die erweiterte Adressierung dient zur Adressierung von mehr als 247 Geräten in einem einzigen Netzwerk. Wenn Sie sich für die erweiterte Adressierung entscheiden, können Sie maximal 64.000 Adressen ansprechen. Im Folgenden wird ein Telegramm der Modbus-Funktion 1 als Beispiel gezeigt.

Tabelle 13- 84 Slave-Adresse mit einem Byte (Byte 0)

Funktion 1	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	
Anforderung	Slave-Adresse	F-Code	Anfangsadresse		Länge der Spulen		
Gültige Antwort	Slave-Adresse	F-Code	Länge	Spulendaten			
Fehlerantwort	Slave-Adresse	0x81	E-Code				

Tabelle 13- 85 Slave-Adresse mit zwei Bytes (Byte 0 und Byte 1)

	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
Anforderung	Slave-Adresse		F-Code	Anfangsadresse		Länge der Spulen	
Gültige Antwort	Slave-Adresse		F-Code	Länge	Spulendaten		
Fehlerantwort	Slave-Adresse		0x81	E-Code			

Bedingungscodes

Tabelle 13- 86 Bedingungscodes für die Ausführung von Modbus_Slave (Kommunikations- und Konfigurationsfehler) [1]

STATUS (W#16#)	Beschreibung
80D1	Der Empfänger hat eine Flusskontrolle zur Unterbrechung einer laufenden Sendung angefordert, aber die Sendung innerhalb der Wartezeit nicht wieder aktiviert. Dieser Fehler wird auch bei der Hardware-Flusskontrolle erzeugt, wenn der Empfänger innerhalb der Wartezeit kein CTS erkennt.
80D2	Die Sendeanforderung wurde abgebrochen, weil vom DCE kein DSR-Signal empfangen wird.
80E0	Die Meldung wurde beendet, weil der Empfangspuffer voll ist.
80E1	Die Meldung wurde wegen eines Paritätsfehlers beendet.
80E2	Die Meldung wurde wegen eines Telegrammfehlers beendet.
80E3	Die Meldung wurde wegen eines Überlauffehlers beendet.
80E4	Die Meldung wurde abgebrochen, weil die angegebene Länge die Gesamtpuffergröße überschreitet.
8180	Ungültiger Wert für die Port-ID oder Fehler bei der Anweisung Modbus_Comm_Load
8186	Ungültige Modbus-Stationsadresse
8187	Ungültiger Pointer auf MB_HOLD_REG-DB: Bereich ist zu klein
818C	Ungültiger Pointer MB_HOLD_REG. Der Datenbereich muss einer der folgenden sein: • Klassischer DB • Array elementarer Datentypen in einem symbolischen oder remanenten DB • Merker

Tabelle 13- 87 Bedingungscodes für die Ausführung von Modbus_Slave (Modbus-Protokollfehler) [1]

STATUS (W#16#)	Antwortcode vom Slave	Modbus-Protokollfehler
8380	Keine Antwort	CRC-Fehler
8381	01	Funktioncode nicht unterstützt oder nicht unterstützt in Broadcasts
8382	03	Fehler in der Datenlänge
8383	02	Datenadressfehler oder Adresse außerhalb des gültigen Bereichs von DATA_PTR

STATUS (W#16#)	Antwortcode vom Slave	Modbus-Protokollfehler
8384	03	Fehler im Datenwert
8385	03	Wert des Datendiagnosecodes wird nicht unterstützt (Funktionscode 08)

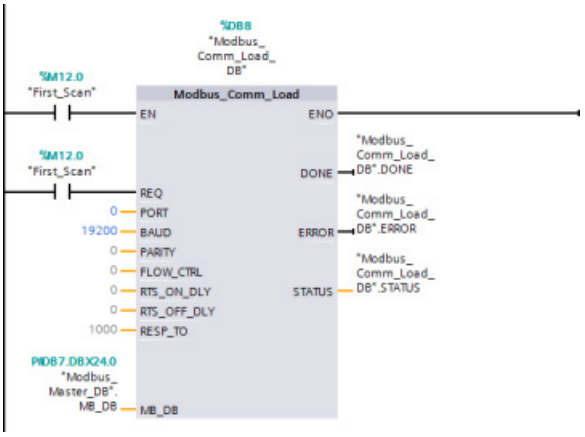
[1] Neben den oben aufgeführten Fehlern der Anweisung Modbus_Slave können auch die zugrunde liegenden PtP-Kommunikationsanweisungen Fehler zurückgeben.

13.5.3.4 Modbus RTU Beispiele

Beispiel: Beispielprogramm für einen Modbus RTU-Master

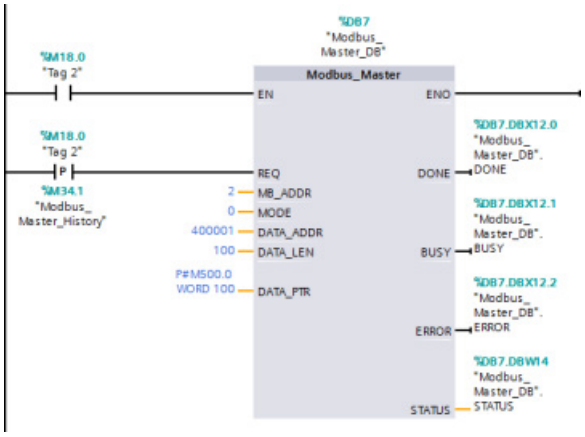
Modbus_Comm_Load wird während des Anlaufs über den Merker des ersten Zyklus initialisiert. Die Ausführung von Modbus_Comm_Load auf diese Weise darf nur durchgeführt werden, wenn sich die Konfiguration des seriellen Ports während der Laufzeit nicht ändert.

Netzwerk 1: Parameter des RS-485-Moduls nur einmal während des ersten Zyklus initialisieren.

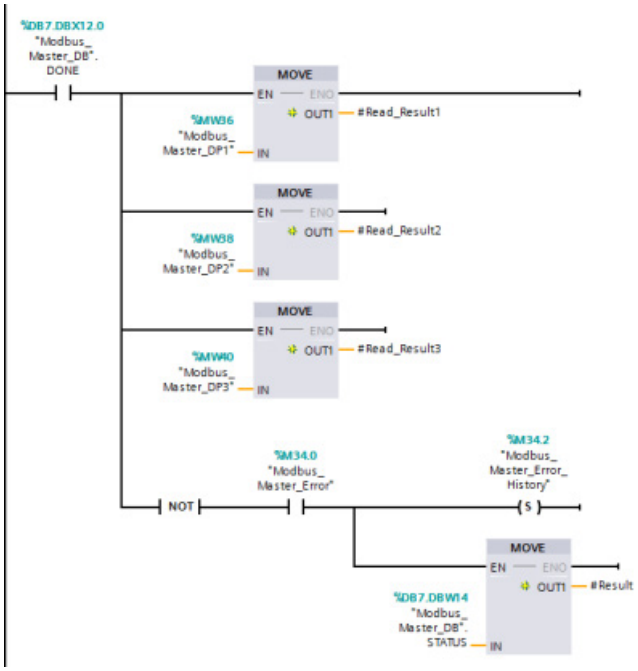


Eine Anweisung Modbus_Master dient im Programmzyklus-OB zur Kommunikation mit einem Slave. Weitere Anweisungen Modbus_Master können im Programmzyklus-OB für die Kommunikation mit anderen Slaves genutzt werden, oder es kann ein Modbus_Master-FB wiederverwendet werden, um mit weiteren Slaves zu kommunizieren.

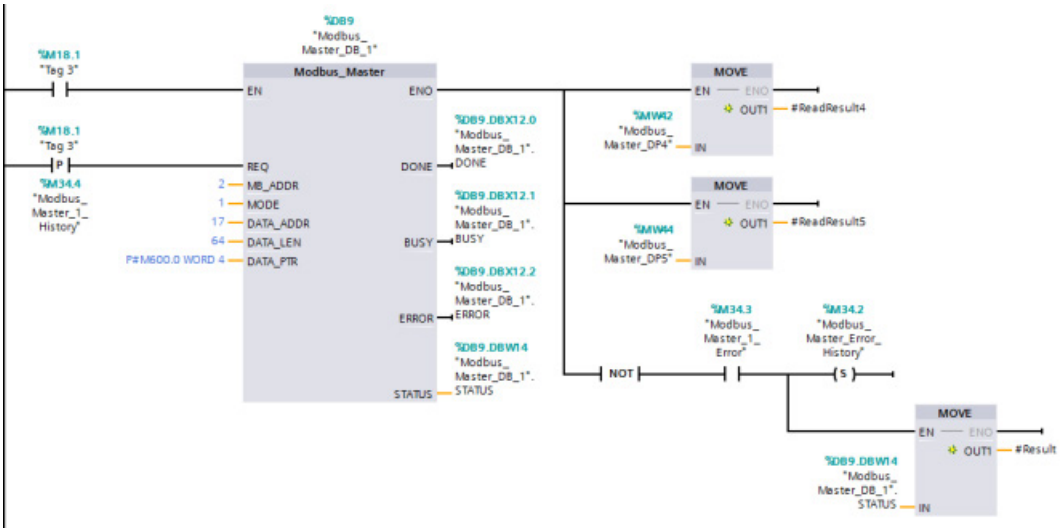
Netzwerk 2: 100 Wörter aus dem Halteregister des Slave lesen.



Netzwerk 3: Dies ist ein optionales Netzwerk, das die Werte der ersten 3 Wörter anzeigt, sobald die Leseoperation durchgeführt wurde.



Netzwerk 4: 64 Bits in das Prozessabbild der Ausgänge schreiben, Beginn an Slave-Adresse A2.0.

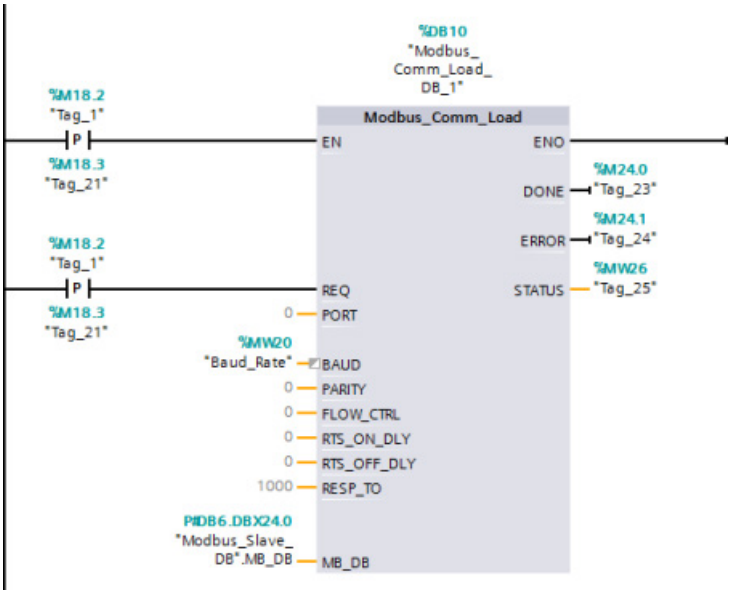


Beispiel: Beispielprogramm für einen Modbus RTU-Slave

Die im Folgenden gezeigte Anweisung MB_COMM_LOAD wird bei jeder Aktivierung von "Variable_1" initialisiert.

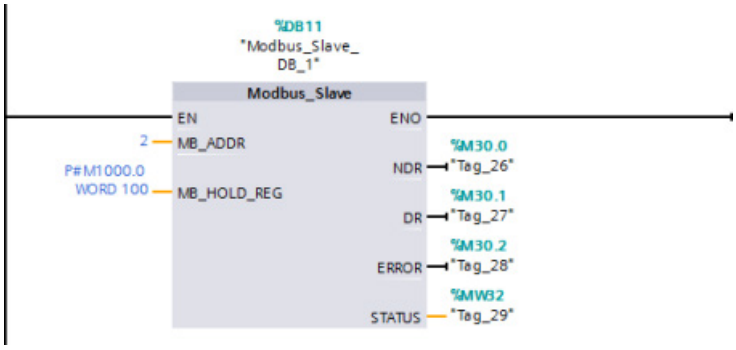
Die Ausführung von MB_COMM_LOAD auf diese Weise darf nur durchgeführt werden, wenn sich die Konfiguration des seriellen Ports als Folge der HMI-Konfiguration während der Laufzeit ändert.

Netzwerk 1: RS485-Modulparameter bei jeder Änderung durch ein HMI-Gerät initialisieren.



Die im Folgenden gezeigte Anweisung MB_SLAVE befindet sich in einem zyklischen OB, der alle 10 ms ausgeführt wird. Dies ermöglicht zwar nicht die absolut schnellste Reaktion durch den Slave, doch es bietet ein gutes Betriebsverhalten bei 9600 Baud für kurze Meldungen (bis zu 20 Byte in der Anforderung).

Netzwerk 2: In jedem Zyklus auf Anforderungen des Modbus-Masters prüfen. Für das Modbus-Halteregister sind 100 Wörter mit Beginn an MW1000 konfiguriert.



13.6 PtP-Kommunikation in älteren Systemen (nur CM/CB 1241)

Vor STEP 7 V13 SP1 und den S7-1200 V4.1-CPUs hatten die PtP-Kommunikationsanweisungen andere Namen und teilweise auch etwas andere Schnittstellen. Die allgemeinen Konzepte für die Punkt-zu-Punkt-Kommunikation (Seite 900) sowie die Port- (Seite 902) und Parameterkonfiguration (Seite 915) gelten für beide Anweisungsarten. Informationen zur Programmierung finden sich in den Beschreibungen zu den PtP-Anweisungen der älteren Versionen.

Tabelle 13- 88 Gemeinsame Fehlerklassen

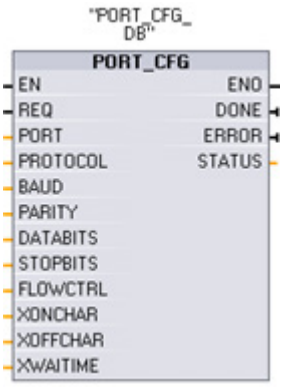
Beschreibung der Klasse	Fehlerklassen	Beschreibung
Portkonfiguration	80Ax	Für die Beschreibung häufiger Fehler in der Schnittstellenkonfiguration
Sendekonfiguration	80Bx	Für die Beschreibung häufiger Fehler in der Sendekonfiguration
Empfangskonfiguration	80Cx	Für die Beschreibung häufiger Fehler in der Empfangskonfiguration
Sende-Laufzeit	80Dx	Für die Beschreibung häufiger Laufzeitfehler beim Senden
Empfangs-Laufzeit	80Ex	Für die Beschreibung häufiger Laufzeitfehler beim Empfangen
Signalverarbeitung	80Fx	Für die Beschreibung häufiger Fehler in Verbindung mit der Signalverarbeitung

13.6.1 Ältere Punkt-zu-Punkt-Anweisungen

13.6.1.1 Anweisung PORT_CFG (Kommunikationsparameter dynamisch konfigurieren)

Tabelle 13- 89 Anweisung PORT_CFG (Port-Konfiguration)

KOP/FUP	SCL	Beschreibung

	`"PORT_CFG_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`PROTOCOL:=_uint_in_,`
	`BAUD:=_uint_in_,`
	`PARITY:=_uint_in_,`
	`DATABITS:=_uint_in_,`
	`STOPBITS:=_uint_in_,`
	`FLOWCTRL:=_uint_in_,`
	`XONCHAR:=_char_in_,`
	`XOFFCHAR:=_char_in_,`
	`WAITTIME:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		Mit PORT_CFG können Sie Portparameter wie die Baudrate über Ihr Programm ändern.
		Sie können die statische Anfangskonfiguration des Ports in den Eigenschaften der Gerätekonfiguration einrichten oder einfach die Standardwerte nutzen. Dann können Sie mit PORT_CFG in Ihrem Programm die Konfiguration ändern.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Die Konfigurationsänderungen von PORT_CFG werden in der CPU nicht dauerhaft gespeichert. Die in der Gerätekonfiguration gespeicherten Parameter werden nicht wiederhergestellt, wenn die CPU von RUN in STOP wechselt oder neu eingeschaltet wird. Weitere Informationen finden Sie unter Kommunikationsports konfigurieren (Seite 902) und Flusskontrolle verwalten (Seite 904).

Tabelle 13- 90 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert die Konfigurationsänderung bei steigender Flanke dieses Eingangs. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
PROTOCOL	IN	UInt	0 - Protokoll der Punkt-zu-Punkt-Kommunikation (Standardwert) 1..n - künftige Definition spezifischer Protokolle
BAUD	IN	UInt	Baudrate des Ports (Standardwert: 6): 1 = 300 Baud, 2 = 600 Baud, 3 = 1200 Baud, 4 = 2400 Baud, 5 = 4800 Baud, 6 = 9600 Baud, 7 = 19200 Baud, 8 = 38400 Baud, 9 = 57600 Baud, 10 = 76800 Baud, 11 = 115200 Baud
PARITY	IN	UInt	Parität des Ports (Standardwert: 1): 1 = Keine Parität, 2 = Gerade Parität, 3 = Ungerade Parität, 4 = Mark-Parität, 5 = Space-Parität
DATABITS	IN	UInt	Bits pro Zeichen (Standardwert):1): 1 = 8 Datenbits, 2 = 7 Datenbits
STOPBITS	IN	UInt	Stoppbits (Standardwert: 1): 1 = 1 Stoppbit, 2 = 2 Stoppbits
FLOWCTRL	IN	UInt	Flusskontrolle (Standardwert: 1): 1 = Keine Flusskontrolle, 2 = XON/XOFF, 3 = Hardware-RTS immer EIN, 4 = Hardware-RTS geschaltet
XONCHAR	IN	Char	Gibt das Zeichen an, das als XON-Zeichen dient. Dies ist typischerweise ein DC1-Zeichen (16#11). Dieser Parameter wird nur ausgewertet, wenn die Flusskontrolle aktiviert ist. (Standardwert: 16#11)
XOFFCHAR	IN	Char	Gibt das Zeichen an, das als XOFF-Zeichen dient. Dies ist typischerweise ein DC3-Zeichen (116#3). Dieser Parameter wird nur ausgewertet, wenn die Flusskontrolle aktiviert ist. (Standardwert: 16#13)
XWAITIME	IN	UInt	Gibt an, wie lange nach dem Empfang eines XON-Zeichens auf ein XOFF-Zeichen gewartet werden soll bzw. wie lange nach Aktivierung von RTS auf das CTS-Signal gewartet werden soll (0 bis 65535 ms). Dieser Parameter wird nur ausgewertet, wenn die Flusskontrolle aktiviert ist. (Standardwert: 2000)
DONE	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Tabelle 13- 91 Bedingungscodes

STATUS (W#16#....)	Beschreibung
80A0	Angegebenes Protokoll ist nicht vorhanden.
80A1	Angegebene Baudrate ist nicht vorhanden.
80A2	Angegebene Paritätsoption ist nicht vorhanden.
80A3	Angegebene Anzahl Datenbits ist nicht vorhanden.
80A4	Angegebene Anzahl Stoppbits ist nicht vorhanden.
80A5	Angegebene Art der Flusskontrolle ist nicht vorhanden.
80A6	Wartezeit ist 0 und Flusskontrolle ist aktiviert
80A7	XON und XOFF sind unzulässige Werte (z. B. der gleiche Wert)

13.6.1.2 Anweisung SEND_CFG (Serielle Übertragungsparameter dynamisch konfigurieren)

Tabelle 13- 92 Anweisung SEND_CFG (Sendekonfiguration)

KOP/FUP	SCL	Beschreibung

	`"SEND_CFG_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`RTSONDLY:=_uint_in_,`
	`RTSOFFDLY:=_uint_in_,`
	`BREAK:=_uint_in_,`
	`IDLELINE:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		SEND_CFG ermöglicht das dynamische Konfigurieren serieller Übertragungsparameter für eine PtP-Kommunikationsschnittstelle. In einem CM oder CB anstehende Meldungen werden gelöscht, wenn SEND_CFG ausgeführt wird.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Sie können die statische Anfangskonfiguration des Ports in den Eigenschaften der Gerätekonfiguration einrichten oder einfach die Standardwerte nutzen. Dann können Sie mit SEND_CFG in Ihrem Programm die Konfiguration ändern.

Die Konfigurationsänderungen von SEND_CFG werden in der CPU nicht dauerhaft gespeichert. Die in der Gerätekonfiguration gespeicherten Parameter werden nicht wiederhergestellt, wenn die CPU von RUN in STOP wechselt oder neu eingeschaltet wird. Siehe Sendeparameter konfigurieren (Seite 906).

Tabelle 13- 93 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert die Konfigurationsänderung bei steigender Flanke an diesem Eingang. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie ein CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)

Parameter und Datentyp		Datentyp	Beschreibung
RTSONDLY	IN	UInt	Anzahl der Millisekunden, die nach Aktivierung von RTS gewartet werden soll, bevor eine Übertragung von Tx-Daten durchgeführt wird. Dieser Parameter ist nur gültig, wenn die Hardwareflusskontrolle aktiviert ist. Der gültige Bereich ist 0 bis 65535 ms. Der Wert 0 deaktiviert die Funktion. (Standardwert: 0)
RTSOFFDLY	IN	UInt	Anzahl der Millisekunden, die nach der Übertragung von Tx-Daten gewartet werden soll, bevor RTS deaktiviert wird: Dieser Parameter ist nur gültig, wenn die Hardwareflusskontrolle aktiviert ist. Der gültige Bereich ist 0 bis 65535 ms. Der Wert 0 deaktiviert die Funktion. (Standardwert: 0)
BREAK	IN	UInt	Dieser Parameter gibt an, dass beim Start jeder Meldung für die angegebene Anzahl von Bitzeiten eine Pause gesendet wird. Maximal sind 65535 Bitzeiten bis maximal 8 Sekunden möglich. Der Wert 0 deaktiviert die Funktion. (Standardwert: 12)
IDLELINE	IN	UInt	Dieser Parameter gibt an, dass vor dem Start jeder Meldung für die angegebene Anzahl von Bitzeiten die Leitung im Leerlauf bleibt. Maximal sind 65535 Bitzeiten bis maximal 8 Sekunden möglich. Der Wert 0 deaktiviert die Funktion. (Standardwert: 12)
DONE	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

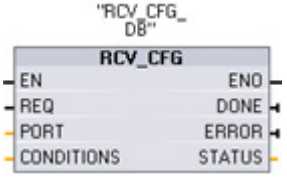
Tabelle 13- 94 Bedingungscodes

STATUS (W#16#....)	Beschreibung
80B0	Konfiguration eines Sendealarms ist nicht zulässig.
80B1	Pausenzeit ist größer als der maximal zulässige Wert.
80B2	Leerlaufzeit ist größer als der maximal zulässige Wert.

13.6.1.3 RCV_CFG (Serielle Empfangsparameter dynamisch konfigurieren)

Tabelle 13- 95 Anweisung RCV_CFG (Empfangskonfiguration)

KOP/FUP	SCL	Beschreibung

	`"RCV_CFG_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`CONDITIONS:=_struct_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		RCV_CFG führt die dynamische Konfiguration serieller Empfangsparameter für eine PtP-Kommunikationsschnittstelle durch. Diese Anweisung konfiguriert die Bedingungen, die den Beginn und das Ende einer empfangenen Meldung kennzeichnen. In einem CM oder CB anstehende Meldungen werden gelöscht, wenn RCV_CFG ausgeführt wird.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Sie können die statische Anfangskonfiguration des Kommunikationsanschlusses in den Eigenschaften der Gerätekonfiguration einrichten oder einfach die Standardwerte nutzen. Dann können Sie mit RCV_CFG in Ihrem Programm die Konfiguration ändern.

Die Konfigurationsänderungen von RCV_CFG werden in der CPU nicht dauerhaft gespeichert. Die in der Gerätekonfiguration gespeicherten Parameter werden nicht wiederhergestellt, wenn die CPU von RUN in STOP wechselt oder neu eingeschaltet wird. Weitere Informationen finden Sie unter Empfangsparameter konfigurieren (Seite 907).

Tabelle 13- 96 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert die Konfigurationsänderung bei steigender Flanke an diesem Eingang. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
CONDITIONS	IN	CONDITIONS	Die Datenstruktur von CONDITIONS gibt die Anfangs- und Endbedingungen der Meldung wie im Folgenden beschrieben an.
DONE	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

Anfangsbedingungen für die Anweisung RCV_PTP

Die Anweisung RCV_PTP nutzt die von der Anweisung RCV_CFG angegebene Konfiguration, um Anfang und Ende von Meldungen der Punkt-zu-Punkt-Kommunikation zu ermitteln. Der Meldungsbeginn wird von den Startbedingungen festgelegt. Der Meldungsbeginn kann anhand von einer oder mehreren Startbedingungen ermittelt werden. Sind mehrere Startbedingungen angegeben, müssen alle Bedingungen erfüllt sein, damit die Meldung gestartet wird.

Eine Beschreibung der Bedingungen für den Meldungsbeginn finden Sie unter "Empfangsparameter konfigurieren (Seite 907)".

Datentypstruktur des Parameters CONDITIONS, Teil 1 (Startbedingungen)

Tabelle 13- 97 Struktur von CONDITIONS für START-Bedingungen

Parameter und Datentyp		Datentyp	Beschreibung
STARTCOND	IN	UInt	Angabe der Startbedingung (Standardwert: 1) • 01H - Startzeichen • 02H - Beliebiges Zeichen • 04H - Leitungspause • 08H - Leitung im Leerlauf • 10H - Zeichenfolge 1 • 20H - Zeichenfolge 2 • 40H - Zeichenfolge 3 • 80H - Zeichenfolge 4
IDLETIME	IN	UInt	Die Anzahl der erforderlichen Bitzeiten für Timeout des Leitungsleerlaufs. (Standardwert: 40). Nur in Verbindung mit der Bedingung "Leitung im Leerlauf". 0 bis 65535
STARTCHAR	IN	Byte	Das Startzeichen für die Bedingung "Startzeichen". (Standardwert: B#16#2)
SEQ[1].CTL	IN	Byte	Zeichenfolge 1, Steuerung für jedes Zeichen ignorieren/vergleichen: (Standardwert: B#16#0) Dies sind die Aktivierungsbits für jedes Zeichen der Startzeichenfolge. • 01H - Zeichen 1 • 02H - Zeichen 2 • 04H - Zeichen 3 • 08H - Zeichen 4 • 10H - Zeichen 5 Wird ein Bit für ein bestimmtes Zeichen deaktiviert, bedeutet dies, dass an dieser Position der Zeichenfolge jedes Zeichen eine Übereinstimmung darstellt.
SEQ[1].STR	IN	Char[5]	Zeichenfolge 1, Startzeichen (5 Zeichen). Standardwert: 0
SEQ[2].CTL	IN	Byte	Zeichenfolge 2, Steuerung für jedes Zeichen ignorieren/vergleichen. Standardwert: B#16#0)
SEQ[2].STR	IN	Char[5]	Zeichenfolge 2, Startzeichen (5 Zeichen). Standardwert: 0
SEQ[3].CTL	IN	Byte	Zeichenfolge 3, Steuerung für jedes Zeichen ignorieren/vergleichen. Standardwert: B#16#0
SEQ[3].STR	IN	Char[5]	Zeichenfolge 3, Startzeichen (5 Zeichen). Standardwert: 0
SEQ[4].CTL	IN	Byte	Zeichenfolge 4, Steuerung für jedes Zeichen ignorieren/vergleichen. Standardwert: B#16#0
SEQ[4].STR	IN	Char[5]	Zeichenfolge 4, Startzeichen (5 Zeichen), Standardwert: 0

Beispiel

Sehen Sie sich die folgende empfangene hexadezimal-codierte Meldung an: "**68** 10 aa **68** bb 10 aa 16". Die konfigurierten Startzeichenfolgen finden Sie in der folgenden Tabelle. Startzeichenfolgen werden ausgewertet, nachdem das erste Zeichen 68H erfolgreich empfangen wurde. Nach erfolgreichem Empfang des vierten Zeichens (das zweite 68H) ist die Startbedingung 1 erfüllt. Wenn die Startbedingungen erfüllt sind, beginnt die Auswertung der Endebedingungen.

Die Verarbeitung der Startzeichenfolge kann aufgrund verschiedener Fehler bei Parität, Framing oder Zeitabständen zwischen den Zeichen beendet werden. Diese Fehler führen dazu, dass die Meldung nicht empfangen wird, weil die Startbedingung nicht erfüllt wurde.

Tabelle 13- 98 Startbedingungen

Startbedingung	Erstes Zeichen	Erstes Zeichen +1	Erstes Zeichen +2	Erstes Zeichen +3	Erstes Zeichen +4
1	**68**H	xx	xx	**68**H	xx
2	10H	aaH	xx	xx	xx
3	dcH	aaH	xx	xx	xx
4	e5H	xx	xx	xx	xx

Endebedingungen für die Anweisung RCV_PTP

Das Ende einer Meldung wird von den angegebenen Endebedingungen festgelegt. Das Ende einer Meldung wird durch das erste Auftreten einer oder mehrerer konfigurierter Endebedingungen festgelegt. Im Abschnitt "Bedingungen für den Meldungsbeginn" unter "Empfangsparameter konfigurieren (Seite 907)" werden die Endebedingungen beschrieben, die Sie für die Anweisung RCV_CFG konfigurieren können.

Sie können die Endebedingungen entweder in den Eigenschaften der Kommunikationsschnittstelle in der Gerätekonfiguration oder über die Anweisung RCV_CFG konfigurieren. Immer wenn die CPU von STOP in RUN wechselt, werden die Empfangsparameter (Start- und Endebedingungen) wieder auf die Einstellungen der Gerätekonfiguration gesetzt. Wenn das STEP 7 Anwenderprogramm RCV_CFG ausführt, werden die Einstellungen auf die Bedingungen von RCV_CFG gesetzt.

Datentypstruktur des Parameters CONDITIONS, Teil 2 (Endebedingungen)

Tabelle 13- 99 Struktur von CONDITIONS für END-Bedingungen

Parameter	Parametertyp	Datentyp	Beschreibung
ENDCOND	IN	UInt 0	Dieser Parameter gibt die Bedingung für das Meldungsende an: • 01H - Antwortzeit • 02H - Meldungszeit • 04H - Zeichenabstand • 08H - Maximale Länge • 10H - N + LEN + M • 20H - Zeichenfolge
MAXLEN	IN	UInt 1	Maximale Meldungslänge: Wird nur verwendet, wenn die Endebedingung "Maximale Länge" ausgewählt ist. 1 bis 1024 Bytes
N	IN	UInt 0	Byteposition des Längenfelds in der Meldung. Wird nur bei der Endebedingung N + LEN + M verwendet. 1 bis 1022 Bytes
LENGTHSIZE	IN	UInt 0	Größe des Längenfelds (1, 2 oder 4 Byte). Wird nur bei der Endebedingung N + LEN + M verwendet.
LENGTHM	IN	UInt 0	Geben Sie die Anzahl der Zeichen nach dem Längenfeld an, die nicht im Wert des Längenfelds enthalten sind. Diese Angabe wird nur bei der Endebedingung N + LEN + M verwendet. 0 bis 255 Bytes
RCVTIME	IN	UInt 200	Geben Sie an, wie lange auf das erste empfangene Zeichen gewartet werden soll. Die Empfangsanweisung wird mit einem Fehler beendet, wenn nicht innerhalb der angegebenen Zeit ein Zeichen erfolgreich empfangen wird. Diese Angabe wird nur bei der Bedingung "Antwortzeit" verwendet. (0 bis 65535 Bitzeiten, maximal 8 Sekunden) Bei diesem Parameter handelt es sich nicht um eine Endebedingung, weil die Auswertung beendet wird, wenn das erste Zeichen einer Antwort empfangen wird. Es handelt sich nur in dem Sinn um eine Endebedingung, als dass eine Empfangsanweisung beendet wird, weil bei Erwartung einer Antwort keine Antwort empfangen wird. Sie müssen eine getrennte Endebedingung definieren.
MSGTIME	IN	UInt 200	Geben Sie an, wie lange nach dem Empfang des ersten Zeichens auf den vollständigen Empfang der gesamten Meldung gewartet werden soll. Dieser Parameter wird nur verwendet, wenn die Bedingung "Timeout der Meldung" ausgewählt ist. (0 bis 65535 ms)
CHARGAP	IN	UInt 12	Geben Sie die Anzahl der Bitzeiten zwischen den Zeichen an. Wenn die Anzahl der Bitzeiten zwischen den Zeichen den angegebenen Wert überschreitet, ist die Endebedingung erfüllt. Diese Angabe wird nur bei der Bedingung "Zeichenabstand" verwendet. (0 bis 65535 Bitzeiten, maximal 8 Sekunden)

Parameter	Parametertyp	Datentyp	Beschreibung
SEQ.CTL	IN	Byte B#16#0	Zeichenfolge 1, Steuerung für jedes Zeichen ignorieren/vergleichen: Dies sind die Aktivierungsbits für jedes Zeichen der Endezeichenfolge. Zeichen 1 ist Bit 0, Zeichen 2 ist Bit 1, ..., Zeichen 5 ist Bit 4. Wird ein Bit für ein bestimmtes Zeichen deaktiviert, bedeutet dies, dass an dieser Position der Zeichenfolge jedes Zeichen eine Übereinstimmung darstellt.
SEQ.STR	IN	Char[5] 0	Zeichenfolge 1, Startzeichen (5 Zeichen)

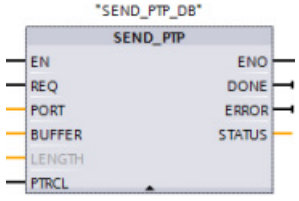
Tabelle 13- 100 Bedingungscodes

STATUS (W#16#....)	Beschreibung
80C0	Unzulässige Startbedingung ausgewählt
80C1	Unzulässige Endebedingung ausgewählt, keine Endebedingung ausgewählt
80C2	Empfangsalarm aktiviert und dies ist nicht möglich.
80C3	Endebedingung "Maximale Länge" ist aktiviert und die maximale Länge ist 0 oder > 1024.
80C4	Berechnete Länge ist aktiviert und N ist >= 1023.
80C5	Berechnete Länge ist aktiviert und Länge ist nicht 1, 2 oder 4.
80C6	Berechnete Länge ist aktiviert und der Wert von M ist > 255.
80C7	Berechnete Länge ist aktiviert und die berechnete Länge ist > 1024.
80C8	Timeout der Antwort ist aktiviert und das Antwort-Timeout ist null.
80C9	Timeout für den Zeichenabstand ist aktiviert und das Timeout ist null.
80CA	Timeout für den Leitungsleerlauf ist aktiviert und das Timeout ist null.
80CB	Endezeichenfolge ist aktiviert, doch alle Zeichen sind "nicht relevant".
80CC	Startzeichenfolge (eine von 4) ist aktiviert, doch alle Zeichen sind "nicht relevant".

13.6.1.4 Anweisung SEND_PTP (Sendepufferdaten übertragen)

Tabelle 13- 101 Anweisung SEND_PTP (Punkt-zu-Punkt-Daten senden)

KOP/FUP	SCL	Beschreibung

	`"SEND_PTP_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	
	`BUFFER:=_variant_in_,`
	`LENGTH:=_uint_in_,`
	`PTRCL:=_bool_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		SEND_PTP startet die Übertragung der Daten und überträgt den zugewiesenen Puffer zur Kommunikationsschnittstelle. Das Programm der CPU wird weiterhin ausgeführt, während das CM oder CB die Daten mit der zugewiesenen Baudrate sendet. Es darf zu jeder Zeit nur eine Sendeanweisung anstehen. Das CM oder CB gibt einen Fehler aus, wenn eine zweite Anweisung SEND_PTP ausgeführt wird, während das CM oder CB bereits eine Meldung sendet.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 13- 102 Datentypen für die Parameter

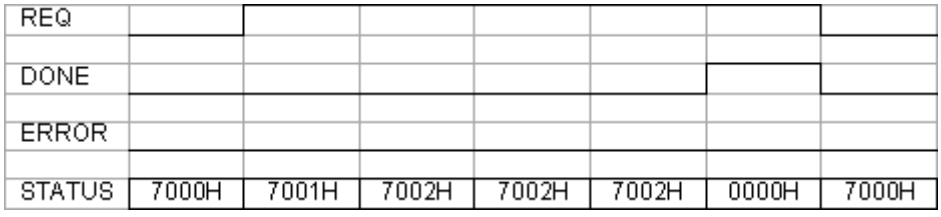
Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Freigabe der angeforderten Übertragung bei steigender Flanke dieses Freigabeeingangs. Dadurch wird der Inhalt des Puffers zur Punkt-zu-Punkt-Kommunikationsschnittstelle übertragen. (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
BUFFER	IN	Variante	Dieser Parameter zeigt auf die Anfangsadresse des Sendepuffers. (Standardwert: 0) **Hinweis:** Boolesche Daten oder Boolesche Felder werden nicht unterstützt.
LENGTH [1]	IN	UInt	Übertragene Rahmenlänge in Byte (Standardwert: 0) Verwenden Sie beim Senden einer komplexen Struktur immer die Länge 0.
PTRCL	IN	Bool	Reserviert für künftige Verwendung
DONE	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

[1] Optionaler Parameter: Klicken Sie auf den Pfeil im unteren Bereich der KOP/FUP-Box, um die Box zu erweitern und diesen Parameter aufzunehmen.

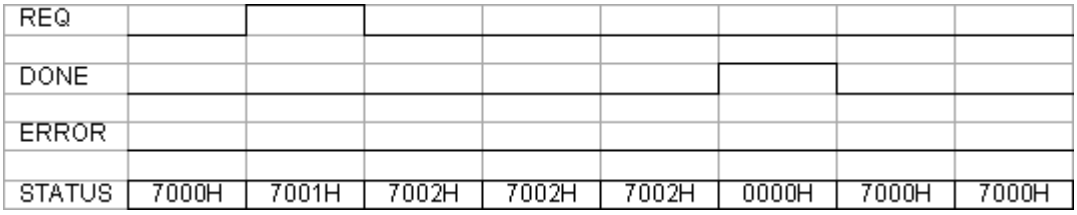
Wenn eine Sendeanweisung in Bearbeitung ist, sind die Ausgänge DONE und ERROR im Zustand FALSCH. Nach dem Ende einer Sendeanweisung wird einer der Ausgänge DONE oder ERROR auf WAHR gesetzt, um den Zustand der Sendeanweisung zu melden. Während DONE oder ERROR im Zustand WAHR ist, ist der Ausgang STATUS gültig.

Die Anweisung gibt den Status 16#7001 aus, wenn die Kommunikationsschnittstelle die Sendedaten annimmt. Nachfolgende Ausführungen von SEND_PTP geben den Wert 16#7002 aus, wenn das CM oder CB immer noch sendet. Nach dem Ende der Sendeanweisung gibt das CM oder CB den Status 16#0000 für die Sendeanweisung aus (sofern kein Fehler aufgetreten ist). Nachfolgende Ausführungen von SEND_PTP mit REQ = 0 geben den Status 16#7000 (frei) aus.

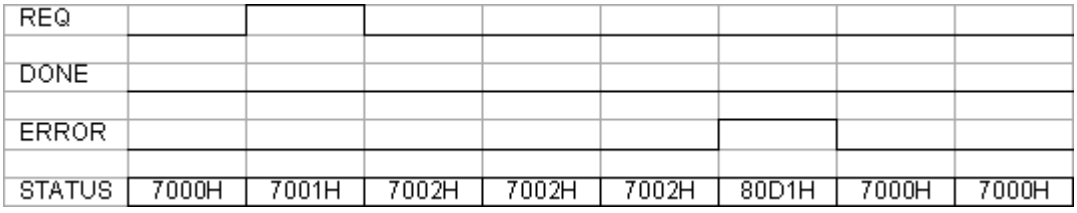
Das folgende Diagramm zeigt die Beziehung zwischen den Ausgangswerten und REQ. Hierbei wird vorausgesetzt, dass die Anweisung regelmäßig aufgerufen wird, um den Status des Sendevorgangs zu prüfen. In der folgenden Abbildung wird davon ausgegangen, dass die Anweisung in jedem Zyklus aufgerufen wird (dargestellt durch die STATUS-Werte).



Die folgende Abbildung zeigt, wie die Parameter DONE und STATUS nur einen Zyklus lang gültig sind, wenn an der REQ-Leitung (einen Zyklus lang) ein Impuls anliegt, um die Sendeanweisung anzustoßen.



Die folgende Abbildung zeigt die Beziehung der Parameter DONE, ERROR und STATUS im Fehlerfall.



Die Werte DONE, ERROR und STATUS sind nur solange gültig, bis SEND_PTP erneut mit dem gleichen Instanz-DB ausgeführt wird.

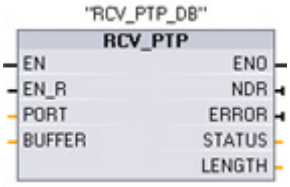
Tabelle 13- 103 Bedingungscodes

STATUS (W#16#....)	Beschreibung
80D0	Neue Anforderung bei aktivem Sender
80D1	Sendung abgebrochen, weil kein CTS innerhalb der Wartezeit
80D2	Sendung abgebrochen, weil kein DSR vom DCE-Gerät
80D3	Sendung wegen Überlauf der Warteschlange abgebrochen (mehr als 1024 Bytes senden)
80D5	Umgekehrtes Biassignal (Drahtbruch)
833A	Der DB für den Parameter BUFFER ist nicht vorhanden.

13.6.1.5 Anweisung RCV_PTP (Empfangsmeldungen aktivieren)

Tabelle 13- 104 Anweisung RCV_PTP (Punkt-zu-Punkt-Daten empfangen)

KOP/FUP	SCL	Beschreibung

	`"RCV_PTP_DB"(`
	`EN_R:=_bool_in_,`
	`PORT:=_uint_in_,`
	`BUFFER:=_variant_in_,`
	`NDR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`LENGTH=>_uint_out_);`
		RCV_PTP prüft die Meldungen, die im CM oder CB empfangen wurden. Wenn eine Meldung verfügbar ist, wird sie vom CM oder CB zur CPU übertragen. Ein Fehler gibt den entsprechenden STATUS-Wert aus.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 13- 105 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
EN_R	IN	Bool	Wenn dieser Eingang WAHR ist und eine Meldung verfügbar ist, wird die Meldung vom CM oder DB in BUFFER übertragen. Wenn EN_R = FALSCH ist, wird das CM oder CB auf Meldungen geprüft und die Ausgänge NDR, ERROR und STATUS werden aktualisiert, doch die Meldung wird nicht in BUFFER übertragen. (Standardwert: 0)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
BUFFER	IN	Variante	Dieser Parameter zeigt auf die Anfangsadresse des Empfangspuffers. Dieser Puffer muss groß genug sein, um die maximale Meldungslänge zu empfangen. Boolesche Daten oder Boolesche Felder werden nicht unterstützt. (Standardwert: 0)
NDR	OUT	Bool	Eine Ausführung lang WAHR, wenn neue Daten bereit sind und die Anweisung fehlerfrei beendet wurde.
ERROR	OUT	Bool	Eine Ausführung lang WAHR, nachdem die Anweisung mit Fehler beendet wurde.
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)
LENGTH	OUT	UInt	Länge der ausgegebenen Meldung in Byte (Standardwert: 0)

Beachten Sie den folgenden Zusammenhang zwischen dem Eingang EN_R und dem Meldungspuffer der Anweisung RCV_PTP:

Der Eingang EN_R steuert die Kopie einer empfangenen Meldung im BUFFER.

Wenn der Eingang EN_R gleich TRUE ist und eine Meldung verfügbar ist, überträgt die CPU die Meldung vom CM oder CB zum BUFFER und aktualisiert die Ausgänge NDR, ERROR, STATUS, und LENGTH.

Wenn EN_R gleich FALSE ist, prüft die CPU das CM oder CB auf Meldungen und aktualisiert die Ausgänge NDR, ERROR, und STATUS, überträgt die Meldung jedoch nicht zum BUFFER. (Beachten Sie, dass der Standardwert von EN_R gleich FALSE ist.)

Es wird empfohlen, EN_R auf TRUE zu setzen und die Ausführung der Anweisung RCV_PTP mit Eingang EN zu steuern.

Der STATUS-Wert ist gültig, wenn NDR oder ERROR im Zustand WAHR ist. Der STATUS-Wert liefert den Grund für die Beendigung der Empfangsoperation im CM oder CB. Dies ist typischerweise ein positiver Wert, der angibt, dass die Empfangsoperation erfolgreich war und dass der Empfangsvorgang normal beendet wurde. Ist der STATUS-Wert negativ (das höchstwertige Bit des Hexadezimalwerts ist gesetzt), wurde die Empfangsoperation wegen einer Fehlerbedingung wie Paritäts-, Framing- oder Überlauffehler beendet.

Jede PtP-Kommunikationsschnittstelle kann maximal 1024 Bytes puffern. Hierbei kann es sich um eine große oder mehrere kleinere Meldungen handeln. Sind mehrere Meldungen im CM oder CB verfügbar, gibt die Anweisung RCV_PTP die älteste verfügbare Meldung aus. Eine nachfolgend ausgeführte Anweisung RCV_PTP gibt die zweitälteste Meldung aus.

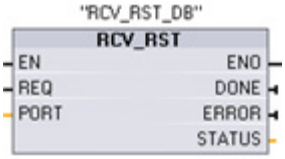
Tabelle 13- 106 Bedingungscodes

STATUS (W#16#...)	Beschreibung
0000	Kein Puffer vorhanden
0094	Meldung wurde beendet, weil die maximale Zeichenlänge empfangen wurde
0095	Meldung wurde wegen Meldungs-Timeout beendet
0096	Meldung wurde wegen Zeichenabstands-Timeout beendet
0097	Meldung wurde wegen Antwort-Timeout beendet
0098	Meldung wurde beendet, weil die Längenbedingung "N+LEN+M" erfüllt war
0099	Meldung wurde beendet, weil die Endezeichenfolge erfüllt war
80E0	Meldung wurde beendet, weil der Empfangspuffer voll ist
80E1	Meldung wurde wegen Paritätsfehler beendet
80E2	Meldung wurde wegen Framingfehler beendet
80E3	Meldung wurde wegen Überlauffehler beendet
80E4	Meldung wurde beendet, weil die berechnete Länge die Puffergröße überschreitet
80E5	Umgekehrtes Biassignal (Drahtbruch)
833A	Der DB für den Parameter BUFFER ist nicht vorhanden.

13.6.1.6 Anweisung RCV_RST (Empfangspuffer löschen)

Tabelle 13- 107 Anweisung RCV_RST (Empfänger zurücksetzen)

KOP/FUP	SCL	Beschreibung

	`"RCV_RST_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		RCV_RST löscht den Empfangspuffer im CM oder CB.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 13- 108 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Aktiviert das Löschen des Empfangspuffers bei steigender Flanke dieses Freigabeeingangs (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
DONE	OUT	Bool	Einen Zyklus lang WAHR bedeutet, dass die letzte Anforderung fehlerfrei ausgeführt wurde.
ERROR	OUT	Bool	WAHR bedeutet, dass die letzte Anforderung mit Fehlern ausgeführt wurde. Ist dieser Ausgang WAHR, enthält Ausgang STATUS die zugehörigen Fehlercodes.
STATUS	OUT	Word	Fehlercode (Standardwert: 0) Siehe Gemeinsame Parameter für Punkt-zu-Punkt-Operationen (Seite 915) zu den Codes für den Kommunikationsstatus.

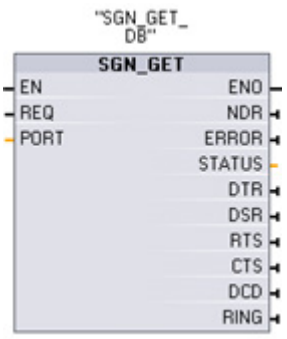
Hinweis

Mit der Anweisung RCV_RST können Sie sicherstellen, dass die Meldungspuffer nach einem Kommunikationsfehler oder nach dem Ändern eines Kommunikationsparameters wie z. B. der Baudrate gelöscht werden. Die Ausführung von RCV_RST bewirkt die Löschung aller internen Meldungspuffer des Moduls. Nach dem Löschen der Meldungspuffer können Sie sicher sein, dass Ihr Programm bei der Ausführung der nächsten Empfangsanweisung nur neue Meldungen zurückgibt und keine alten Meldungen aus dem Zeitraum vor dem Aufruf von RCV_RST.

13.6.1.7 Anweisung SGN_GET (RS232-Signale abfragen)

Tabelle 13- 109 Anweisung SGN_GET (RS232-Signale abrufen)

KOP/FUP	SCL	Beschreibung

	`"SGN_GET_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`NDR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`DTR=>_bool_out_,`
	`DSR=>_bool_out_,`
	`RTS=>_bool_out_,`
	`CTS=>_bool_out_,`
	`DCD=>_bool_out_,`
	`RING=>_bool_out_);`
		SGN_GET liest die aktuellen Zustände der RS232-Kommunikationssignale.
		Diese Funktion gilt nur beim RS232-CM.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 13- 110 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	RS232-Signalzustände werden bei der steigenden Flanke an diesem Eingang abgerufen (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen.
NDR	OUT	Bool	Einen Zyklus lang WAHR, wenn neue Daten bereit sind und die Anweisung fehlerfrei beendet wurde
ERROR	OUT	Bool	Einen Zyklus lang WAHR, nachdem die Anweisung mit Fehler beendet wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)
DTR	OUT	Bool	Datenterminal bereit, Modul bereit (Ausgang). Standardwert: False
DSR	OUT	Bool	Datensatz bereit, Kommunikationsteilnehmer bereit (Eingang). Standardwert: False
RTS	OUT	Bool	Sendeanforderung, Modul sendebereit (Ausgang). Standardwert: False
CTS	OUT	Bool	Sendebereit, Kommunikationsteilnehmer kann Daten empfangen (Eingang). Standardwert: False
DCD	OUT	Bool	Datenträger erkannt, Signalpegel empfangen (immer False, nicht unterstützt)
RING	OUT	Bool	Rufanzeige, Meldung eines eingehenden Rufs (immer False, nicht unterstützt)

Tabelle 13- 111 Bedingungscodes

STATUS (W#16#....)	Beschreibung
80F0	CM oder CB ist RS485 und es sind keine Signale verfügbar

13.6.1.8 Anweisung SGN_SET (RS232-Signale festlegen)

Tabelle 13- 112 Anweisung SGN_SET (RS232-Signale setzen)

KOP/FUP	SCL	Beschreibung

	`"SGN_SET_DB"(`
	`REQ:=_bool_in_,`
	`PORT:=_uint_in_,`
	`SIGNAL:=_byte_in_,`
	`RTS:=_bool_in_,`
	`DTR:=_bool_in_,`
	`DSR:=_bool_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_);`
		SGN_SET setzt die Zustände der RS232-Kommunikationssignale.
		Diese Funktion gilt nur beim RS232-CM.

[1] STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

Tabelle 13- 113 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung zum Setzen der RS232-Signale wird bei der steigenden Flanke an diesem Eingang gestartet (Standardwert: Falsch)
PORT	IN	PORT	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen. (Standardwert: 0)
SIGNAL	IN	Byte	Auswahl des zu setzenden Signals: (mehrere zulässig). Standardwert: 0 • 01H = RTS • 02H = DTR • 04H = DSR
RTS	IN	Bool	Sendeanforderung, Modul bereit, zu setzenden Wert zu senden (wahr oder falsch), Standardwert: Falsch
DTR	IN	Bool	Datenterminal bereit, Modul bereit, zu setzenden Wert zu senden (wahr oder falsch). Standardwert: Falsch
DSR	IN	Bool	Datensatz bereit (gilt nur für Schnittstellentyp DCE), nicht verwendet.
DONE	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde
ERROR	OUT	Bool	Eine Ausführung lang WAHR, nachdem die letzte Anforderung mit Fehler ausgeführt wurde
STATUS	OUT	Word	Ausführungsbedingung (Standardwert: 0)

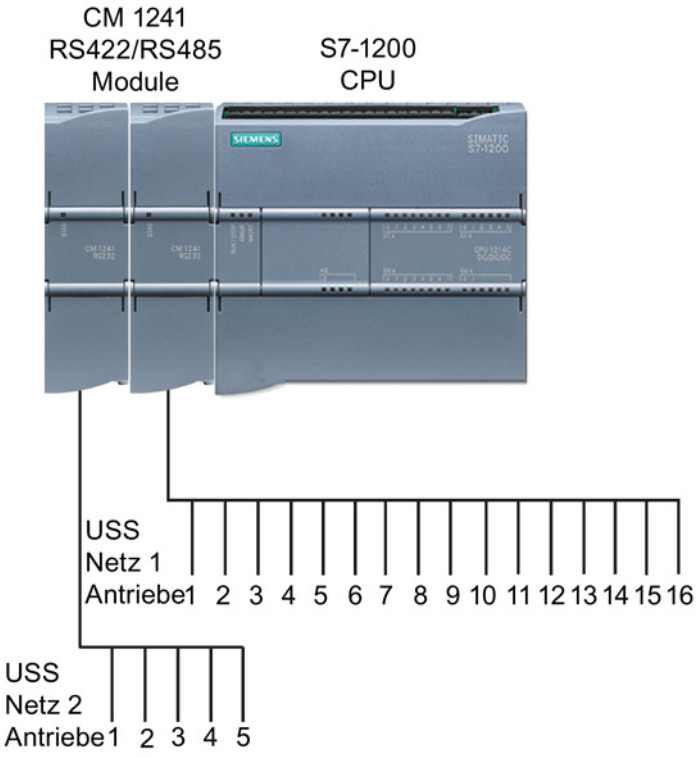
Tabelle 13- 114 Bedingungscodes

STATUS (W#16#....)	Beschreibung
80F0	CM oder CB ist RS485 und es können keine Signale gesetzt werden
80F1	Signale können wegen Hardwareflusskontrolle nicht gesetzt werden
80F2	DSR kann nicht gesetzt werden, weil das Modul ein DTE-Gerät ist
80F3	DTR kann nicht gesetzt werden, weil das Modul ein DCE-Gerät ist

13.7 USS-Kommunikation in älteren Systemen (nur CM/CB 1241)

Die USS-Anweisungen steuern den Betrieb von Motorantrieben, die das Protokoll der universellen seriellen Schnittstelle (USS) unterstützen. Mit den USS-Anweisungen können Sie über RS485-Verbindungen mit mehreren Antrieben mit CM 1241 RS485-Kommunikationsmodulen oder einem CB 1241 RS485-Kommunikationsboard kommunizieren. In einer S7-1200 CPU können bis zu drei CM 1241 RS422/RS485-Module und ein CB 1241 RS485-Board eingebaut werden. Jeder RS485-Port kann bis zu sechzehn Antriebe betreiben.

Das USS-Protokoll nutzt ein Master/Slave-Netzwerk für die Kommunikation über einen seriellen Bus. Der Master verwendet einen Adressparameter, um eine Meldung an einen ausgewählten Slave zu senden. Ein Slave selbst kann niemals senden, ohne dafür zuvor eine Anforderung zu erhalten. Die direkte Meldungsübertragung zwischen den einzelnen Slaves ist nicht möglich. Die USS-Kommunikation funktioniert im Halbduplex-Betrieb. Die folgende USS-Abbildung zeigt ein Netzwerkdiagramm für eine Beispielanwendung eines Antriebs.



Vor STEP 7 V13 SP1 und den S7-1200 V4.1-CPUs hatten die USS-Kommunikationsanweisungen andere Namen und teilweise auch etwas andere Schnittstellen. Für beide Anweisungsarten gelten die allgemeinen Konzepte. Informationen zur Programmierung finden sich in den Beschreibungen zu den einzelnen USS-Anweisungen.

13.7.1 Version der USS-Anweisungen auswählen

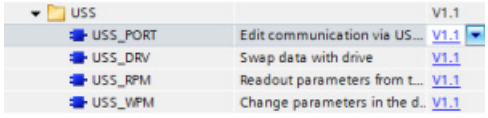
Es gibt zwei Versionen der USS-Anweisungen in STEP 7:

- Version 2.0 war anfangs in STEP 7 Basic/Professional V13 verfügbar.
- Version 2.1 ist in STEP 7 Basic/Professional V13, SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Sie können nicht beide Versionen der Anweisungen bei demselben Modul verwenden, Sie können jedoch für zwei verschiedene Module unterschiedliche Versionen der Anwendungen einsetzen.

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.



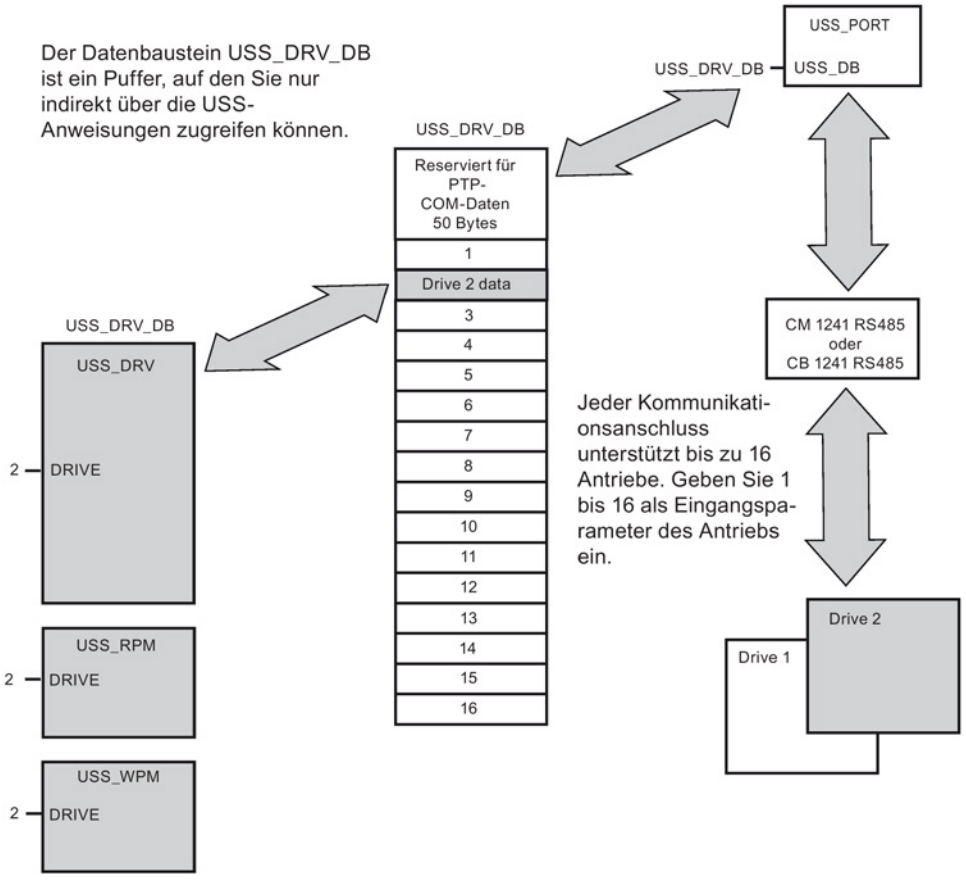
Um die Version einer USS-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine USS-Anweisung über das Anweisungsverzeichnis in Ihr Programm platzieren, wird je nach der ausgewählten USS-Anweisung eine neue FB- oder FC-Instanz in der Projektnavigation angelegt. Die neue FB- oder FC-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer USS-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation die Instanz eines USS-FBs oder FCs aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der USS-Anweisung anzuzeigen.

13.7.2 Voraussetzungen für den Einsatz des USS-Protokolls

Die vier USS-Anweisungen verwenden 1 FB und 3 FCs zur Unterstützung des USS-Protokolls. Für jedes USS-Netzwerk wird ein Instanz-Datenbaustein (DB) USS_PORT verwendet. Der Instanz-Datenbaustein USS_PORT enthält temporäre Speicher und Puffer für alle Antriebe in dem USS-Netzwerk. Die USS-Anweisungen nutzen die Informationen in diesem Datenbaustein gemeinsam.



Alle Antriebe (max. 16), die an einen RS485-Port angeschlossen sind, sind Teil desselben USS-Netzwerks. Alle Antriebe, die an einen anderen RS485-Port angeschlossen sind, sind Teil eines anderen USS-Netzwerks. Jedes USS-Netzwerk wird mithilfe eines eindeutigen Datenbausteins verwaltet. Alle Anweisungen, die zu einem USS-Netzwerk gehören, müssen diesen Datenbaustein gemeinsam nutzen. Dies umfasst alle Anweisungen USS_DRV, USS_PORT, USS_RPM und USS_WPM für die Steuerung aller Antriebe in einem USS-Netzwerk.

Die Anweisung USS_DRV ist ein Funktionsbaustein (FB). Wenn Sie die Anweisung USS_DRV in den Programmiereditor einfügen, werden Sie im Dialog "Aufrufoptionen" aufgefordert, einen DB für diesen FB zuzuweisen. Wenn es sich um die erste Anweisung USS_DRV in diesem Programm für dieses USS-Netzwerk handelt, können Sie die DB-Standardzuweisung übernehmen (oder ggf. den Namen ändern), und der neue DB wird für Sie erstellt. Wenn es sich jedoch nicht um die erste Anweisung USS_DRV für diesen Kanal handelt, müssen Sie im Dialog "Aufrufoptionen" in der Klappliste den DB auswählen, der diesem USS-Netzwerk bereits zuvor zugewiesen wurde.

Bei allen Anweisungen USS_PORT, USS_RPM und USS_WPM handelt es sich um Funktionen (FCs). Wenn Sie diese FCs im Editor einfügen, wird kein DB zugewiesen. Stattdessen müssen Sie dem Eingang USS_DB dieser Anweisungen den jeweiligen DB zuweisen. Doppelklicken Sie auf das Parameterfeld und klicken Sie dann auf das Symbol, um die verfügbaren DBs anzuzeigen.

Die Funktion USS_PORT steuert die Kommunikation zwischen der CPU und den Antrieben über den Punkt-zu-Punkt(PtP)-RS485-Kommunikationsport. Bei jedem Aufruf dieser Funktion wird eine Kommunikation mit einem Antrieb bearbeitet. Ihr Programm muss diese Funktion schnell genug aufrufen, so dass die Antriebe keine Zeitüberschreitung melden. Diese Funktion kann aus dem Zyklus-OB des Hauptprogramms oder aus einem beliebigen Alarm-OB aufgerufen werden.

In der Regel wird die Funktion USS_PORT aus einem Weckalarm-OB aufgerufen. Stellen Sie die Zykluszeit des Weckalarm-OBs etwa auf die Hälfte eines Mindestaufrufintervalls ein (beispielsweise sollte für die Kommunikation mit 1200 Baud eine Zykluszeit von maximal 350 ms verwendet werden).

Der Funktionsbaustein USS_DRV gibt Ihrem Programm Zugriff auf einen angegebenen Antrieb im USS-Netzwerk. Seine Ein- und Ausgänge entsprechen den Zuständen und den Bedienfunktionen des Antriebs. Sind 16 Antriebe im Netzwerk vorhanden, so muss USS_DRV in Ihrem Programm mindestens 16mal aufgerufen werden, also jeweils einmal für jeden Antrieb. Wie schnell diese Bausteine aufgerufen werden, hängt von der erforderlichen Geschwindigkeit für die Steuerung des Antriebsbetriebs ab.

Sie können den Funktionsbaustein USS_DRV nur aus dem Zyklus-OB eines Hauptprogramms aufrufen.

⚠ VORSICHT

Beim Aufruf von USS-Anweisungen aus OBs zu beachten

Rufen Sie USS_DRV, USS_RPM und USS_WPM nur aus einem Zyklus-OB des Hauptprogramms auf. Die Funktion USS_PORT kann aus einem beliebigen OB aufgerufen werden, üblicherweise wird sie aus einem Weckalarm-OB aufgerufen.

Verwenden Sie die Anweisungen USS_DRV, USS_RPM und USS_WPM nicht in einem OB mit einer höheren Priorität als die entsprechende Anweisung USS_PORT. Fügen Sie beispielsweise USS_PORT nicht in das Hauptprogramm und USS_RPM nicht in einen Weckalarm-OB ein. Wird die Unterbrechung der Ausführung von USS_PORT nicht verhindert, kann es zu unerwarteten Fehlern kommen, die zu Verletzungen führen können.

Mit den Funktionen USS_RPM und USS_WPM werden die Betriebsparameter des entfernten Antriebs gelesen und geschrieben. Diese Parameter steuern die interne Funktionsweise des Antriebs. Eine Definition dieser Parameter finden Sie im Handbuch des Antriebs. Ihr Programm kann eine beliebige Anzahl dieser Funktionen enthalten, es kann jedoch immer nur eine Lese- oder Schreibanforderung für einen Antrieb aktiv sein. Sie dürfen die Funktionen USS_RPM und USS_WPM nur aus dem Zyklus-OB eines Hauptprogramms aufrufen.

Zeit für die Kommunikation mit dem Antrieb berechnen

Die Kommunikation mit dem Antrieb läuft asynchron zum Zyklus der S71200 ab. Die S7-1200 durchläuft üblicherweise mehrere Zyklen, bevor die Kommunikation mit einem Antrieb beendet ist.

Das Intervall USS_PORT ist die Zeit, die für eine Transaktion des Antriebs erforderlich ist. Die folgende Tabelle zeigt die Mindestintervalle für USS_PORT für jede Baudrate der Kommunikation. Wenn Sie die Funktion USS_PORT häufiger aufrufen, als es das USS_PORT-Intervall vorgibt, wird die Anzahl der Transaktionen nicht erhöht. Das Timeout-Intervall des Antriebs ist die Zeitdauer, die für eine Transaktion zur Verfügung steht, wenn zur Fertigstellung der Transaktion aufgrund von Kommunikationsfehlern 3 Versuche nötig sind. Standardmäßig führt die Bibliothek für das USS-Protokoll bei jeder Transaktion bis zu 2 Wiederholungen durch.

Tabelle 13- 115 Zeitbedarf berechnen

Baudrate	Berechnetes Mindestintervall für Aufruf von USS_PORT (ms)	Intervall-Timeout für Antriebsmeldung pro Antrieb (ms)
1200	790	2370
2400	405	1215
4800	212.5	638
9600	116.3	349
19200	68.2	205
38400	44.1	133
57600	36.1	109
115200	28.1	85

13.7.3 Ältere USS-Anweisungen

13.7.3.1 Anweisung USS_PORT (Kommunikation über USS-Netzwerk bearbeiten)

Tabelle 13- 116 Anweisung USS_PORT

KOP/FUP	SCL	Beschreibung

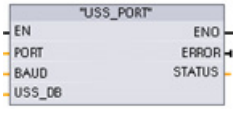
	`USS_PORT(`
	`  PORT:=_uint_in_,`
	`  BAUD:=_dint_in_,`
	`  ERROR=>_bool_out_,`
	`  STATUS=>_word_out_,`
	`  USS_DB:=_fbtref_inout_);`
		Die Anweisung USS_PORT bearbeitet die Kommunikation über ein USS-Netzwerk.

Tabelle 13- 117 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
PORT	IN	Port	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen.
BAUD	IN	DInt	Die Baudrate für die USS-Kommunikation.
USS_DB	INOUT	USS_BASE	Der Name des Instanz-DBs, der erstellt und initialisiert wird, wenn eine Anweisung USS_DRV in Ihr Programm eingefügt wird.
ERROR	OUT	Bool	Wenn WAHR, weist dieser Ausgang darauf hin, dass ein Fehler aufgetreten und Ausgang STATUS gültig ist.
STATUS	OUT	Word	Der Zustandswert der Anforderung zeigt des Ergebnis des Zyklus oder der Initialisierung an. Weitere Informationen stehen für einige Statuscodes in der Variablen "USS_Extended_Error" zur Verfügung.

Normalerweise ist nur eine Anweisung USS_PORT pro PtP-Kommunikationsport im Programm vorhanden und jeder Aufruf dieser Anweisung steuert eine Übertragung zu oder von einem einzigen Antrieb. Alle USS-Funktionen, die einem USS-Netzwerk und einem PtP-Kommunikationsport zugewiesen sind, müssen den gleichen Instanz-DB nutzen.

Ihr Programm muss die Anweisung USS_PORT so oft ausführen, dass kein Timeout im Antrieb auftritt. Die Anweisung USS_PORT wird üblicherweise aus einem Weckalarm-OB aufgerufen, um Antriebs-Timeouts zu verhindern und die letzten USS-Datenaktualisierungen für Aufrufe von USS_DRV verfügbar zu haben.

13.7.3.2 Anweisung USS_DRV (Daten mit Antrieb tauschen)

Tabelle 13- 118 Anweisung USS_DRV

KOP/FUP	SCL	Beschreibung

Standardansicht
 	
Erweiterte Ansicht
 	
	`"USS_DRV_DB"(`
	`RUN:=_bool_in_,`
	`OFF2:=_bool_in_,`
	`OFF3:=_bool_in_,`
	`F_ACK:=_bool_in_,`
	`DIR:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PZD_LEN:=_usint_in_,`
	`SPEED_SP:=_real_in_,`
	`CTRL3:=_word_in_,`
	`CTRL4:=_word_in_,`
	`CTRL5:=_word_in_,`
	`CTRL6:=_word_in_,`
	`CTRL7:=_word_in_,`
	`CTRL8:=_word_in_,`
	`NDR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`RUN_EN=>_bool_out_,`
	`D_DIR=>_bool_out_,`
	`INHIBIT=>_bool_out_,`
	`FAULT=>_bool_out_,`
	`SPEED=>_real_out_,`
	`STATUS1=>_word_out_,`
	`STATUS3=>_word_out_,`
	`STATUS4=>_word_out_,`
	`STATUS5=>_word_out_,`
	`STATUS6=>_word_out_,`
	`STATUS7=>_word_out_,`
	`STATUS8=>_word_out_);`
		Die Anweisung USS_DRV tauscht Daten mit einem Antrieb aus, indem Anfragemeldungen erzeugt und die Antwortmeldungen des Antriebs ausgewertet werden. Es sollte für jeden Antrieb ein eigener Funktionsbaustein verwendet werden, jedoch müssen alle USS-Funktionen, die einem USS-Netzwerk und einem PtP-Kommunikationsport zugewiesen sind, den gleichen Instanz-Datenbaustein verwenden. Sie müssen den DB-Namen eingeben, wenn Sie die erste Anweisung USS_DRV einfügen. Dann verweisen Sie auf diesen DB, der beim Einfügen der ersten Anweisung angelegt wurde.
		STEP 7 erstellt automatisch den DB, wenn Sie die Anweisung einfügen.

[1] KOP und FUP: Erweitern Sie die Box, um alle Parameter anzuzeigen. Klicken Sie dazu auf den unteren Bereich der Box. Die Parameteranschlüsse, die grau dargestellt sind, sind optional, eine Parametrierung ist nicht erforderlich.

Tabelle 13- 119 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
RUN	IN	Bool	Startbit des Antriebs: Ist dieser Parameter WAHR, so ermöglicht dieser Eingang den Betrieb des Antriebs mit der voreingestellten Drehzahl. Wenn RUN im Betrieb des Antriebs nach Falsch wechselt, läuft der Motor bis zum Stillstand aus. Dieses Verhalten unterscheidet sich von der Abschaltung der Spannungsversorgung (OFF2) und vom Bremsen des Motors (OFF3).
OFF2	IN	Bool	Bit "Zum Stillstand auslaufen": Ist dieser Parameter FALSCH, so veranlasst dieses Bit das Auslaufen des Antriebs, ohne zu bremsen.
OFF3	IN	Bool	Schnelles Stoppbit: Ist dieser Parameter FALSCH, so verursacht dieses Bit einen schnellen Halt durch Abbremsen des Antriebs.
F_ACK	IN	Bool	Fehlerquittierungsbit: Mit diesem Bit wird das Fehlerbit eines Antriebs zurückgesetzt. Das Bit wird nach dem Löschen des Fehlers gesetzt und der Antrieb erkennt damit, dass der vorherige Fehler nicht mehr gemeldet werden muss.
DIR	IN	Bool	Richtungssteuerung des Antriebs: Dieses Bit wird gesetzt, wenn der Antrieb in Vorwärtsrichtung laufen soll (wenn SPEED_SP positiv ist).
DRIVE	IN	USInt	Adresse des Antriebs: Dieser Eingang ist die Adresse des USS-Antriebs. Der gültige Bereich liegt zwischen Antrieb 1 und Antrieb 16.
PZD_LEN	IN	USInt	Wortlänge: Dies ist die Anzahl der PZD-Datenwörter. Gültige Werte sind 2, 4, 6 oder 8 Wörter. Der Standardwert ist 2.
SPEED_SP	IN	Real	Drehzahlsollwert: Dies ist die Drehzahl des Antriebs prozentual zur konfigurierten Frequenz. Ein positiver Wert bedeutet, dass der Antrieb vorwärts läuft (wenn DIR wahr ist). Gültig ist der Bereich von 200,00 bis -200,00.
CTRL3	IN	Word	Steuerwort 3: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL4	IN	Word	Steuerwort 4: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL5	IN	Word	Steuerwort 5: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL6	IN	Word	Steuerwort 6: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL7	IN	Word	Steuerwort 7: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
CTRL8	IN	Word	Steuerwort 8: Wert, der in einen benutzerkonfigurierbaren Parameter des Antriebs geschrieben wird. Sie müssen dies im Antrieb konfigurieren (optionaler Parameter).
NDR	OUT	Bool	Neue Daten bereit: Ist dieser Parameter wahr, so meldet das Bit, dass am Ausgang Daten einer neuen Kommunikationsanforderung bereitstehen.

Parameter und Datentyp		Datentyp	Beschreibung
ERROR	OUT	Bool	Fehler aufgetreten: Wenn WAHR, weist dies darauf hin, dass ein Fehler aufgetreten und Ausgang STATUS gültig ist. Alle anderen Ausgänge werden bei einem Fehler auf Null gesetzt. Kommunikationsfehler werden nur an den Ausgängen ERROR und STATUS der Anweisung USS_PORT gemeldet.
STATUS	OUT	Word	Der Zustandswert der Anforderung zeigt des Ergebnis des Zyklus an. Dies ist kein vom Antrieb ausgegebenes Zustandswort.
RUN_EN	OUT	Bool	Betrieb freigegeben: Dieses Bit meldet, ob der Antrieb läuft.
D_DIR	OUT	Bool	Antriebsrichtung: Dieses Bit meldet, ob der Antrieb vorwärts läuft.
INHIBIT	OUT	Bool	Antrieb gesperrt: Dieses Bit meldet den Zustand des Sperrbits für den Antrieb.
FAULT	OUT	Bool	Antriebsfehler: Dieses Bit meldet, dass im Antrieb ein Fehler aufgetreten ist. Sie müssen die Störung beheben und Bit F_ACK setzen, um dieses Bit zu löschen.
SPEED	OUT	Real	Istwert Antriebsdrehzahl (skalierter Wert von Zustandswort 2 des Antriebs): Dies ist die Drehzahl des Antriebs prozentual zur konfigurierten Drehzahl.
STATUS1	OUT	Word	Zustandswort 1 des Antriebs: Dieser Wert enthält feste Zustandsbits eines Antriebs.
STATUS3	OUT	Word	Zustandswort 3 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS4	OUT	Word	Zustandswort 4 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS5	OUT	Word	Zustandswort 5 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS6	OUT	Word	Zustandswort 6 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS7	OUT	Word	Zustandswort 7 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.
STATUS8	OUT	Word	Zustandswort 8 des Antriebs: Dieser Wert enthält ein benutzerkonfigurierbares Zustandswort des Antriebs.

Wenn die erste Ausführung von USS_DRV erfolgt, wird der von der USS-Adresse (Parameter DRIVE) angegebene Antrieb im Instanz-DB initialisiert. Nach dieser Initialisierung können nachfolgende Anweisungen USS_PORT die Kommunikation mit dem Antrieb an dieser Antriebsnummer beginnen.

Wenn Sie die Antriebsnummer ändern, muss die CPU zunächst in STOP und dann wieder in RUN versetzt werden, damit der Instanz-DB initialisiert wird. Die Eingangsparameter werden im USS-Sendepuffer konfiguriert und die Ausgänge werden, sofern vorhanden, aus einem "vorherigen" gültigen Antwortpuffer gelesen. Während der Ausführung der Anweisung USS_DRV findet keine Datenübertragung statt. Nach der Ausführung von USS_PORT kommunizieren die Antriebe. USS_DRV konfiguriert nur die zu sendenden Meldungen und wertet Daten aus, die möglicherweise in einer vorherigen Anforderung empfangen wurden.

Sie können die Drehrichtung des Antriebs entweder über den Eingang DIR (Bool) oder über das Vorzeichen (positiv oder negativ) am Eingang SPEED_SP (Real) steuern. Die folgende Tabelle erläutert, wie diese Eingänge zusammen funktionieren, um die Drehrichtung des Antriebs zu bestimmen, vorausgesetzt der Motor dreht vorwärts.

Tabelle 13- 120 Interaktion der Parameter SPEED_SP und DIR

SPEED_SP	DIR	Drehrichtung des Antriebs
Wert > 0	0	Rückwärts
Wert > 0	1	Vorwärts
Wert < 0	0	Vorwärts
Wert < 0	1	Rückwärts

13.7.3.3 Anweisung USS_RPM (Parameter aus dem Antrieb auslesen)

Tabelle 13- 121 Anweisung USS_RPM

KOP/FUP	SCL	Beschreibung

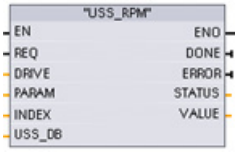
	`USS_RPM(REQ:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PARAM:=_uint_in_,`
	`INDEX:=_uint_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`VALUE=>_variant_out_,`
	`USS_DB:=_fbtref_inout_);`
		Die Anweisung USS_RPM liest einen Parameter aus einem Antrieb. Alle USS-Funktionen, die einem USS-Netzwerk und einem PtP-Kommunikationsport zugewiesen sind, müssen den gleichen Datenbaustein verwenden. USS_RPM muss aus einem Zyklus-OB des Hauptprogramms aufgerufen werden.

Tabelle 13- 122 Datentypen für die Parameter

Parametertyp		Datentyp	Beschreibung
REQ	IN	Bool	Sendeanforderung: Ist REQ WAHR, so wird eine neue Leseanforderung benötigt. Dies wird ignoriert, wenn die Anforderung für diesen Parameter bereits ansteht.
DRIVE	IN	USInt	Adresse des Antriebs: DRIVE ist die Adresse des USS-Antriebs. Der gültige Bereich liegt zwischen Antrieb 1 und Antrieb 16.
PARAM	IN	UInt	Parameternummer: PARAM gibt an, welcher Antriebsparameter geschrieben wird. Der Bereich für diesen Parameter liegt zwischen 0 und 2047. Bei einigen Antrieben kann das höchstwertige Byte auf PARAM-Werte größer als 2047 zugreifen. Weitere Informationen für den Zugriff auf einen erweiterten Bereich finden Sie im Handbuch zu Ihrem Antrieb.
INDEX	IN	UInt	Parameterindex: INDEX gibt an, in welchen Antriebsparameterindex geschrieben werden soll. Es handelt sich um einen 16-Bit-Wert, bei dem das niederwertigste Byte der tatsächliche Indexwert ist, mit einem Bereich von (0 bis 255). Das höchstwertige Byte kann ebenfalls von dem Antrieb verwendet werden und ist antriebsspezifisch. Weitere Informationen finden Sie im Handbuch Ihres Antriebs.

Parametertyp		Datentyp	Beschreibung
USS_DB	INOUT	USS_BASE	Der Name des Instanz-DBs, der erstellt und initialisiert wird, wenn eine Anweisung USS_DRV in Ihr Programm eingefügt wird.
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	Dies ist der Wert des Parameters, der gelesen wurde und er ist nur gültig, wenn das Bit DONE wahr ist.
DONE[1]	OUT	Bool	Ist dieser Parameter WAHR, so steht am Ausgang VALUE der zuvor angeforderte Wert des Leseparameters an. Dieses Bit wird gesetzt, wenn die Anweisung USS_DRV die Leseantwort des Antriebs erkennt. Dieses Bit wird zurückgesetzt, wenn: Sie die Antwortdaten über eine andere Abfrage USS_RPM anfordern oder beim zweiten der nächsten beiden Aufrufe von USS_DRV
ERROR	OUT	Bool	Fehler aufgetreten: Wenn WAHR, weist ERROR darauf hin, dass ein Fehler aufgetreten und Ausgang STATUS gültig ist. Alle anderen Ausgänge werden bei einem Fehler auf Null gesetzt. Kommunikationsfehler werden nur an den Ausgängen ERROR und STATUS der Anweisung USS_PORT gemeldet.
STATUS	OUT	Word	STATUS gibt das Ergebnis der Leseanforderung an. Weitere Informationen stehen für einige Statuscodes in der Variablen "USS_Extended_Error" zur Verfügung.

[1] Das Bit DONE weist darauf hin, dass gültige Daten aus dem referenzierten Motorantrieb ausgelesen und an die CPU geliefert wurden. Es weist nicht darauf hin, dass die USS-Bibliothek in der Lage ist, sofort einen weiteren Parameter auszulesen. Eine leere PKW-Anforderung muss an den Motorantrieb gesendet und auch von der Anweisung quittiert werden, bevor der Parameterkanal zur Verwendung durch den jeweiligen Antrieb frei wird. Der sofortige Aufruf von USS_RPM oder USS_WPM FC für den spezifischen Motorantrieb führt zu dem Fehler 0x818A.

13.7.3.4 Anweisung USS_WPM (Parameter im Antrieb ändern)

Hinweis

EEPROM-Schreibanweisungen (für den EEPROM in einem USS-Antrieb)

Übertreiben Sie die Verwendung der EEPROM-Schreiboperation nicht. Halten Sie die Anzahl der EEPROM-Schreiboperationen möglichst gering, um die Lebensdauer des EEPROM zu verlängern.

Tabelle 13- 123 Anweisung USS_WPM

KOP/FUP	SCL	Beschreibung

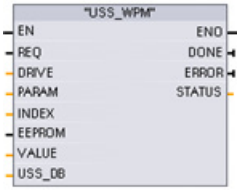
	`USS_WPM(REQ:=_bool_in_,`
	`DRIVE:=_usint_in_,`
	`PARAM:=_uint_in_,`
	`INDEX:=_uint_in_,`
	`EEPROM:=_bool_in_,`
	`VALUE:=_variant_in_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`USS_DB:=_fbtref_inout_);`
		Die Anweisung USS_WPM ändert einen Parameter im Antrieb. Alle USS-Funktionen, die einem USS-Netzwerk und einem PtP-Kommunikationsport zugewiesen sind, müssen den gleichen Datenbaustein nutzen.
		USS_WPM muss aus dem Zyklus-OB eines Hauptprogramms aufgerufen werden.

Tabelle 13- 124 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Sendeanforderung: Ist REQ WAHR, so wird eine neue Schreibanforderung benötigt. Dies wird ignoriert, wenn die Anforderung für diesen Parameter bereits ansteht.
DRIVE	IN	USInt	Adresse des Antriebs: DRIVE ist die Adresse des USS-Antriebs. Der gültige Bereich liegt zwischen Antrieb 1 und Antrieb 16.
PARAM	IN	UInt	Parameternummer: PARAM gibt an, welcher Antriebsparameter geschrieben wird. Der Bereich für diesen Parameter liegt zwischen 0 und 2047. Bei einigen Antrieben kann das höchstwertige Byte auf PARAM-Werte größer als 2047 zugreifen. Weitere Informationen für den Zugriff auf einen erweiterten Bereich finden Sie im Handbuch zu Ihrem Antrieb.
INDEX	IN	UInt	Parameterindex: INDEX gibt an, in welchen Antriebsparameterindex geschrieben werden soll. Es handelt sich um einen 16-Bit-Wert, bei dem das niederwertigste Byte der tatsächliche Indexwert ist, mit einem Bereich von (0 bis 255). Das höchstwertige Byte kann ebenfalls von dem Antrieb verwendet werden und ist antriebsspezifisch. Weitere Informationen finden Sie im Handbuch Ihres Antriebs.
EEPROM	IN	Bool	Im EEPROM des Antriebs speichern: Wenn WAHR, wird die Transaktion eines Parameters zum Schreiben in den Antrieb im EEPROM des Antriebs gespeichert. Wenn FALSCH, so wird der geschriebene Wert nur temporär gespeichert und geht beim nächsten Einschalten des Antriebs verloren.

Parameter und Datentyp		Datentyp	Beschreibung
VALUE	IN	Word, Int, UInt, DWord, DInt, UDInt, Real	Wert des Parameters, in den geschrieben werden soll. Er muss beim Zustandswechsel von REQ gültig sein.
USS_DB	INOUT	USS_BASE	Der Name des Instanz-DBs, der erstellt und initialisiert wird, wenn eine Anweisung USS_DRV in Ihr Programm eingefügt wird.
DONE[1]	OUT	Bool	Wenn WAHR, weist DONE darauf hin, dass der Eingang VALUE in den Antrieb geschrieben wurde. Dieses Bit wird gesetzt, wenn die Anweisung USS_DRV die Schreibantwort des Antriebs erkennt. Dieses Bit wird zurückgesetzt, entweder wenn Sie die Antwortdaten über eine weitere Abfrage USS_WPM anfordern, oder beim zweiten der nächsten beiden Aufrufe von USS_DRV.
ERROR	OUT	Bool	Wenn WAHR, weist ERROR darauf hin, dass ein Fehler aufgetreten und Ausgang STATUS gültig ist. Alle anderen Ausgänge werden bei einem Fehler auf Null gesetzt. Kommunikationsfehler werden nur an den Ausgängen ERROR und STATUS der Anweisung USS_PORT gemeldet.
STATUS	OUT	Word	STATUS gibt das Ergebnis der Schreibanforderung an. Weitere Informationen stehen für einige Statuscodes in der Variablen "USS_Extended_Error" zur Verfügung.

[1] Das Bit DONE weist darauf hin, dass gültige Daten aus dem referenzierten Motorantrieb ausgelesen und an die CPU geliefert wurden. Es weist nicht darauf hin, dass die USS-Bibliothek in der Lage ist, sofort einen weiteren Parameter auszulesen. Eine leere PKW-Anforderung muss an den Motorantrieb gesendet und auch von der Anweisung quittiert werden, bevor der Parameterkanal zur Verwendung durch den jeweiligen Antrieb frei wird. Der sofortige Aufruf von USS_RPM oder USS_WPM FC für den spezifischen Motorantrieb führt zu dem Fehler 0x818A.

13.7.4 Alte USS-Statuscodes

Statuscodes der USS-Anweisung werden im Ausgang STATUS der USS-Funktionen ausgegeben.

Tabelle 13- 125 STATUS-Codes [1]

STATUS (W#16#....)	Beschreibung
0000	Kein Fehler
8180	Die Länge der Antwort des Antriebs entsprach nicht den vom Antrieb empfangenen Zeichen. Die Nummer des Antriebs, in dem der Fehler aufgetreten ist, wird in der Variablen "USS_Extended_Error" ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
8181	Der Parameter VALUE gehört nicht zum Datentyp Wort, Real oder Doppelwort.
8182	Der Anwender hat einen Parameterwert vom Typ Wort eingegeben und die Antwort vom Antrieb im Format Doppelwort oder Real empfangen.
8183	Der Anwender hat einen Parameterwert vom Typ Doppelwort oder Real eingegeben und die Antwort vom Antrieb im Format Wort empfangen.
8184	Das Antworttelegramm des Antriebs hatte eine falsche Prüfsumme. Die Nummer des Antriebs, in dem der Fehler aufgetreten ist, wird in der Variablen "USS_Extended_Error" ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
8185	Unzulässige Antriebsadresse (gültiger Adressbereich für Antriebe: 1 bis16)

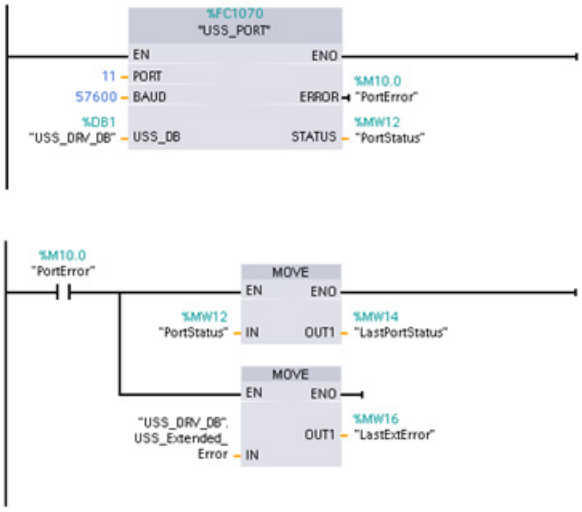
STATUS (W#16#....)	Beschreibung
8186	Der Drehzahlsollwert liegt außerhalb des gültigen Bereichs (gültiger Sollwertbereich für die Drehzahl: -200 % bis 200 %).
8187	Die falsche Antriebsnummer hat auf die gesendete Anforderung geantwortet. Die Nummer des Antriebs, in dem der Fehler aufgetreten ist, wird in der Variablen "USS_Extended_Error" ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
8188	Unzulässige PZD-Wortlänge angegeben (gültiger Bereich = 2, 4, 6 oder 8 Wörter)
8189	Unzulässige Baudrate angegeben.
818A	Der Anforderungskanal für Parameter wird von einer anderen Anforderung für diesen Antrieb verwendet.
818B	Der Antrieb hat nicht auf Anforderungen und Wiederholungen reagiert. Die Nummer des Antriebs, in dem der Fehler aufgetreten ist, wird in der Variablen "USS_Extended_Error" ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
818C	Der Antrieb hat einen erweiterten Fehler zu einer Parameteranforderung ausgegeben. Die Beschreibung der erweiterten Fehler finden Sie unterhalb dieser Tabelle.
818D	Der Antrieb hat einen Fehler "Unzulässiger Zugriff" zu einer Parameteranforderung ausgegeben. Weitere Informationen dazu, weshalb der Parameterzugriff begrenzt sein kann, finden Sie im Handbuch zu Ihrem Antrieb.
818E	Der Antrieb wurde nicht initialisiert. Dieser Fehlercode wird an USS_RPM oder USS_WPM ausgegeben, wenn die Anweisung USS_DRV nicht mindestens einmal für diesen Antrieb aufgerufen wurde. Dadurch wird verhindert, dass die Initialisierung im ersten Zyklus von USS_DRV eine anstehende Anforderung zum Lesen oder Schreiben von Parametern überschreibt, weil dabei der Antrieb als neuer Eintrag initialisiert wird. Um diesen Fehler zu beheben, rufen Sie die Anweisung USS_DRV für diesen Antrieb auf.
80Ax-80Fx	Spezifische Fehler, die von den von der USS-Bibliothek aufgerufenen FBs für die PtP-Kommunikation zurückgegeben werden - Diese Fehlercodes werden von der USS-Bibliothek nicht verändert und sind in den Beschreibungen der PtP-Anweisung definiert.

[1] Neben den oben aufgeführten Fehlern der USS-Anweisungen können auch die zugrunde liegenden PtP-Kommunikationsanweisungen Fehler zurückgeben.

Für verschiedene STATUS-Codes werden weitere Informationen in der Variable "USS_Extended_Error" des Instanz-DB USS_DRV zur Verfügung gestellt. Für die STATUS-Codes hexadezimal 8180, 8184, 8187 und 818B, enthält USS_Extended_Error die Antriebsnummer des Antriebs, bei dem der Kommunikationsfehler auftrat. Für die STATUS-Codes hexadezimal 818C enthält USS_Extended_Error einen Antriebsfehlercode, der bei der Verwendung einer Anweisung USS_RPM oder USS_WPM vom Antrieb ausgegeben wird.

Beispiel: Meldung von Kommunikationsfehlern

Kommunikationsfehler (STATUS = 16#818B) werden nur bei der Anweisung USS_PORT und nicht bei der Anweisung USS_DRV gemeldet. Beispiel: Wenn das Netzwerk nicht ordnungsgemäß beendet wird, kann ein Antrieb in RUN wechseln, doch die Anweisung USS_DRV zeigt an allen Ausgangsparametern 0 an. In diesem Fall können Sie den Kommunikationsfehler nur über die Anweisung USS_PORT erkennen. Weil dieser Fehler nur einen Zyklus lang sichtbar ist, müssen Sie entsprechende Erfassunglogik einfügen. Dies wird in dem folgenden Beispiel dargestellt. In diesem Beispiel werden, wenn das Fehlerbit der Anweisung USS_PORT WAHR ist, die Werte STATUS und USS_Extended_Error im Speicherbereich der Merker abgelegt. Die Antriebsnummer wird in der Variable USS_Extended_Error abgelegt, wenn der Wert des STATUS-Codes hexadezimal 8180, 8184, 8187 oder 818B ist.



Netzwerk 1 Der Portzustand "PortStatus" und die erweiterten Fehlercodewerte "USS_DRV_DB".USS_Extended_Error sind nur einen Programmzyklus lang gültig. Die Werte müssen zur späteren Bearbeitung erfasst werden.

Netzwerk 2 Der "PortError"-Kontakt löst die Speicherung des "PortStatus"-Werts in "LastPortStatus" und des "USS_DRV_DB".USS_Extended_Error -Werts in "LastExtError" aus.

Lese- und Schreibzugriff auf die internen Parameter eines Antriebs

USS-Antriebe unterstützen Lese- und Schreibzugriff auf die internen Parameter eines Antriebs. Diese Funktion ermöglicht die dezentrale Steuerung und Konfiguration des Antriebs. Zugriffe der Antriebsparameter können aufgrund von Fehlern wie Wert außerhalb des Bereichs oder unzulässige Anforderungen in der aktuellen Betriebsart des Antriebs fehlschlagen. Der Antrieb erzeugt einen Fehlercde, der in der Variablen "USS_Extended_Error" ausgegeben wird. Dieser Fehlercode gilt nur für die letzte Ausführung der Anweisung USS_RPM oder USS_WPM. Der Fehlercode des Antriebs wird in der Variablen "USS_Extended_Error" abgelegt, wenn der Wert von STATUS code hexadezimal 818C ist. Der Fehlercode von "USS_Extended_Error" richtet sich nach der Variante des Antriebs. Eine Beschreibung der erweiterten Fehlercodes von Lese- und Schreibfunktionen für Parameter finden Sie im Handbuch des Antriebs.

13.7.5 Allgemeine Voraussetzungen für die Antriebseinrichtung mit der alten Anweisung USS

Für die Antriebseinrichtung mit der alten Anweisung USS gelten die folgenden allgemeinen Voraussetzungen:

- Für die Antriebe muss die Verwendung von 4 PKW-Wörtern eingerichtet werden.
- Die Antriebe können für 2, 4, 6 oder 8 PZD-Wörter konfiguriert werden.
- Die Anzahl der PZD-Wörter im Antrieb muss dem Eingang PZD_LEN der Anweisung USS_DRV des Antriebs entsprechen.
- Die Baudrate aller Antriebe muss dem Eingang BAUD der Anweisung USS_PORT entsprechen.
- Der Antrieb muss für die Fernsteuerung eingerichtet sein.
- Für den Frequenzsollwert an der COM-Verbindung des Antriebs muss USS festgelegt sein.
- Für die Antriebsadresse muss 1 bis 16 festgelegt sein. Diese Adresse muss dem Eingang DRIVE am Baustein USS_DRV des Antriebs entsprechen.
- Für die Richtungssteuerung des Antriebs muss die Verwendung der Polarität des Antriebssollwerts eingerichtet werden.
- Das RS485-Netzwerk muss ordnungsgemäß abgeschlossen sein.

Die allgemeine USS-Antriebsverbindung und -einrichtung ist bei den USS-Anweisungen (V4.1) und bei den alten USS-Anweisungen (bis V4.0) identisch. Weitere Informationen finden Sie unter Beispiel: Allgemeine USS-Antriebsverbindung und -einrichtung (Seite 969).

13.8 Modbus TCP-Kommunikation in älteren Systemen

13.8.1 Übersicht

Vor STEP 7 V13 SP1 und den S7-1200 V4.1-CPUs hatten die Modbus TCP-Kommunikationsanweisungen andere Namen und teilweise auch etwas andere Schnittstellen. Für beide Anweisungsarten gelten die allgemeinen Konzepte. Informationen zur Programmierung finden sich in den Beschreibungen zu den einzelnen Modbus TCP-Anweisungen.

13.8.2 Version der Modbus TCP-Anweisungen auswählen

Es gibt zwei Versionen der Modbus TCP-Anweisungen in STEP 7:

- Version 3.0 war anfangs in STEP 7 Basic/Professional V13 verfügbar.
- Version 3.1 ist in STEP 7 Basic/Professional V13, SP1 verfügbar.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Verwenden Sie niemals Anweisungen der Version 3.0 und 3.1 zusammen in einem CPU-Programm. Die Modbus TCP-Anweisungen in Ihrem Programm müssen dieselbe Versionsnummer haben (**1**.x, **2**.y, or **V**.z). Die einzelnen Anweisungen in einer Versionsgruppe können dann verschiedene Unterversionen aufweisen (1.**x**).

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.

Um die Version einer Modbus TCP-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine Modbus TCP-Anweisung über das Anweisungsverzeichnis in Ihr Programm einfügen, wird in der Projektnavigation eine neue FB-Instanz angelegt. Die neue FB-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer Modbus TCP-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation die Instanz eines Modbus TCP-FBs aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der Modbus TCP-Anweisung anzuzeigen.

13.8.3 Ältere Modbus TCP-Anweisungen

13.8.3.1 Anweisung MB_CLIENT (Über PROFINET als Modbus TCP-Client kommunizieren)

Tabelle 13- 126 Anweisung MB_CLIENT

KOP/FUP	SCL	Beschreibung

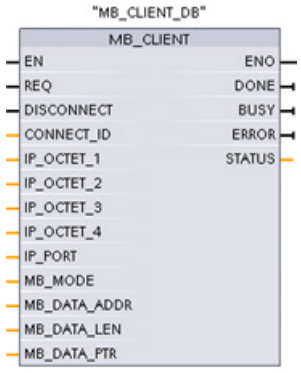
	`"MB_CLIENT_DB"(`
	`REQ:=_bool_in_,`
	`DISCONNECT:=_bool_in_,`
	`CONNECT_ID=_uint_in_,`
	`IP_OCTET_1:=_byte_in_,`
	`IP_OCTET_2:=_byte_in_,`
	`IP_OCTET_3:=_byte_in_,`
	`IP_OCTET_4:=_byte_in_,`
	`IP_PORT:=_uint_in_,`
	`MB_MODE:=_usint_in_,`
	`MB_DATA_ADDR:=_udint_in_,`
	`MB_DATA_LEN:=_uint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_DATA_PTR:=_variant_inout_);`
		MB_CLIENT kommuniziert als Modbus TCP-Client über den PROFINET-Anschluss an der S7-1200 CPU. Es ist kein zusätzliches Hardwaremodul für die Kommunikation erforderlich.
		MB_CLIENT kann eine Client-Server-Verbindung herstellen, eine Modbus-Funktionsanforderung senden, eine Antwort empfangen und das Trennen der Verbindung von einem Modbus TCP-Server steuern.

Tabelle 13- 127 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	In	Bool	FALSCH = Keine Modbus-Kommunikationsanforderung WAHR = Anforderung für die Kommunikation mit einem Modbus TCP-Server
DISCONNECT	IN	Bool	Mit dem Parameter DISCONNECT kann Ihr Programm den Verbindungsaufbau und -abbau zu einem Modbus-Servergerät steuern. Ist DISCONNECT = 0 und keine Verbindung vorhanden, versucht MB_CLIENT eine Verbindung zur zugewiesenen IP-Adresse und Portnummer aufzubauen. Ist DISCONNECT = 1 und eine Verbindung vorhanden, wird versucht, die Verbindung zu trennen. Immer wenn dieser Eingang aktiviert ist, wird kein anderer Vorgang eingeleitet.
CONNECT_ID	IN	UInt	Der Parameter CONNECT_ID muss jede Verbindung im PLC-Gerät eindeutig identifizieren. Jede eindeutige Instanz der Anweisung MB_CLIENT oder MB_SERVER muss einen eindeutigen Parameter CONNECT_ID enthalten.
IP_OCTET_1	IN	USInt	IP-Adresse des Modbus TCP-Servers: Oktett 1 8-Bit-Anteil der 32-Bit-IP-Adresse (IPv4) des Modbus TCP-Servers, mit dem der Client über das Modbus TCP-Protokoll eine Verbindung herstellt und kommuniziert.
IP_OCTET_2	IN	USInt	IP-Adresse des Modbus TCP-Servers: Oktett 2
IP_OCTET_3	IN	USInt	IP-Adresse des Modbus TCP-Servers: Oktett 3

Parameter und Datentyp		Datentyp	Beschreibung
IP_OCTET_4	IN	USInt	IP-Adresse des Modbus TCP-Servers: Oktett 4
IP_PORT	IN	UInt	Standardwert = 502: Die IP-Portnummer des Servers, mit dem der Client versucht, über das TCP/IP-Protokoll eine Verbindung herzustellen und anschließend zu kommunizieren.
MB_MODE	IN	USInt	Auswahl Modus: Weist die Art der Anforderung (Lesen, Schreiben oder Diagnose) zu. Weitere Informationen finden Sie in der Tabelle der Modbus-Funktionen unten.
MB_DATA_ADDR	IN	UDInt	Modbus-Anfangsadresse: Weist die Anfangsadresse der Daten zu, auf die MB_CLIENT zugreifen soll. Die gültigen Adressen finden Sie in der Tabelle der Modbus-Funktionen unten.
MB_DATA_LEN	IN	UInt	Modbus-Datenlänge: Weist die Anzahl der Bits oder Wörter zu, auf die diese Anforderung zugreifen soll. Die gültigen Längen finden Sie in der Tabelle der Modbus-Funktionen unten.
MB_DATA_PTR	IN_OUT	Variant	Pointer auf das Modbus-Datenregister: Das Register puffert Daten, die an einen Modbus-Server gesendet oder von dort empfangen werden. Der Pointer muss einen globalen Standard-DB oder eine Adresse im Speicherbereich der Merker zuweisen.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
BUSY	OUT	Bool	• 0 - Keine Anweisung MB_CLIENT in Bearbeitung • 1 - Anweisung MB_CLIENT in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die Ausführung von MB_CLIENT mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung

Parameter REQ

FALSCH = Keine Modbus-Kommunikationsanforderung
WAHR = Anforderung für die Kommunikation mit einem Modbus TCP-Server

Wenn keine Instanz von MB_CLIENT aktiv ist und der Parameter DISCONNECT = 0 ist, wird bei REQ = 1 eine neue Modbus-Anforderung gestartet. Ist die Verbindung noch nicht hergestellt, so wird eine neue Verbindung aufgebaut.

Wird dieselbe Instanz von MB_CLIENT erneut mit DISCONNECT = 0 and REQ = 1 ausgeführt, bevor die aktuelle Anforderung abgearbeitet ist, werden keine nachfolgenden Modbus-Übertragungen durchgeführt. Sobald die aktuelle Anforderung jedoch abgearbeitet ist, kann eine neue Anforderung verarbeitet werden, sofern MB_CLIENT mit REQ = 1 ausgeführt wird.

Wenn die aktuelle MB_CLIENT-Kommunikationsanforderung abgearbeitet ist, ist das Bit DONE einen Zyklus lang WAHR. Das Bit DONE kann als Zeitfenster für die Sequenzierung mehrerer MB_CLIENT-Anforderungen verwendet werden.

Hinweis

Konsistenz der Eingangsdaten wähend der Verarbeitung von MB_CLIENT

Nachdem ein Modbus-Client eine Modbus-Operation initiiert, werden alle Eingangszustände intern gespeichert und dann bei jedem nachfolgenden Aufruf verglichen. Der Vergleich dient dazu, festzustellen, ob ein bestimmter Aufruf der ursprüngliche Auslöser der aktiven Client-Anforderung war. Mehrere Aufrufe von MB_CLIENT können mit Hilfe eines gemeinsamen Instanz-DBs durchgeführt werden.

Deshalb ist es wichtig, dass die Eingänge während des Zeitraums, in dem eine MB_CLIENT -Operation aktiv verarbeitet wird, nicht geändert werden. Wird diese Regel missachtet, kann MB_CLIENT nicht feststellen, ob es die aktive Instanz ist.

Über die Parameter MB_MODE und MB_DATA_ADDR wählen Sie die Modbus-Kommunikationsfunktion aus.

MB_DATA_ADDR weist die Modbus-Anfangsadresse der Daten zu, auf die zugegriffen werden soll. Die Anweisung MB_CLIENT nutzt statt eines Funktionscodeeingangs einen Eingang MB_MODE.

Die Kombination aus MB_MODE- und MB_DATA_ADDR -Werten legt den Funktionscode fest, der in der eigentlichen Modbus-Meldung verwendet wird. Die folgende Tabelle zeigt die Entsprechung zwischen dem Parameter MB_MODE, der Modbus-Funktion und dem Modbus-Adressbereich.

Tabelle 13- 128 Modbus-Funktionen

MB_MODE	Modbus-Funktion	Länge der Daten	Operation und Daten	MB_DATA_ADDR
0	01	1 bis 2000	Ausgangsbits lesen: 1 bis 2000 Bit pro Anforderung	1 bis 9999
0	02	1 bis 2000	Eingangsbits lesen: 1 bis 2000 Bit pro Anforderung	10001 bis 19999
0	03	1 bis 125	Halteregister lesen: 1 bis 125 Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535
0	04	1 bis 125	Eingangswörter lesen: 1 bis 125 Wörter pro Anforderung	30001 bis 39999
1	05	1	Ein Ausgangsbit schreiben: Ein Bit pro Anforderung	1 bis 9999
1	06	1	Ein Halteregister schreiben: 1 Wort pro Anforderung	40001 bis 49999 oder 400001 bis 465535
1	15	2 bis 1968	Mehrere Ausgangsbits schreiben: 2 bis 1968 Bits pro Anforderung	1 bis 9999
1	16	2 bis 123	Mehrere Halteregister schreiben: 2 bis 123 Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535

MB_MODE	Modbus-Funktion	Länge der Daten	Operation und Daten	MB_DATA_ADDR
2	15	1 bis 1968	Ein oder mehrere Ausgangsbits schreiben: 1 bis 1968 Bit pro Anforderung	1 bis 9999
2	16	1 bis 123	Ein oder mehrere Halteregister schreiben: 1 bis 123 Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535
11	11	0	Statuswort und Ereigniszähler der Serverkommunikation lesen. Das Statuswort zeigt beschäftigt an (0 – nicht beschäftigt, 0xFFFF - beschäftigt). Der Ereigniszähler wird bei jeder erfolgreichen Abarbeitung einer Meldung inkrementiert. Sowohl Parameter MB_DATA_ADDR als auch Parameter MB_DATA_LEN von MB_CLIENT wird bei dieser Funktion ignoriert.	
80	08	1	Serverzustand über Datendiagnosecode 0x0000 prüfen (Prüfschleifentest – Server gibt ein Echo der Anforderung zurück) 1 Wort pro Anforderung	
81	08	1	Server-Ereigniszähler über Datendiagnosecode 0x000A zurücksetzen 1 Wort pro Anforderung	
3 bis 10, 12 bis 79, 82 bis 255			Reserviert	

Hinweis

MB_DATA_PTR weist einen Puffer zu, um aus einem Modbus TCP-Server gelesene bzw. in den Server geschriebene Daten zu speichern

Der Datenpuffer kann sich in einem globalen Standard-DB oder in einer Adresse im Speicherbereich der Merker befinden.

Verwenden Sie für einen Puffer im Speicherbereich der Merker das Standardformat des Any-Pointer. Hierbei handelt es sich um das Format P#"Bitadresse" "Datentyp" "Länge". Ein Beispiel: P#M1000.0 WORD 500.

Zuweisung eines Kommunikationspuffers durch MB_DATA_PTR

- Kommunikationsfunktionen von MB_CLIENT:
  - 1-Bit-Daten aus Modbus-Serveradressen lesen und schreiben (00001 bis 09999)
  - 1-Bit-Daten aus Modbus-Serveradressen lesen (10001 bis 19999)
  - 16-Bit-Wortdaten aus Modbus-Serveradressen lesen (30001 bis 39999) und (40001 bis 49999)
  - 16-Bit-Wortdaten in Modbus-Serveradressen schreiben (40001 bis 49999)
- Daten in Wort- oder Bitgröße werden in den oder aus dem von MB_DATA_PTR zugewiesenen Puffer im DB oder Merkerbereich übertragen.
- Wenn von MB_DATA_PTR ein DB als Puffer zugewiesen ist, müssen Sie allen DB-Datenelementen Datentypen zuweisen.
  - Der 1-Bit-Datentyp Bool stellt eine Modbus-Bitadresse dar.
  - Datentypen mit einzelnen 16-Bit-Wörtern wie WORD, UInt und Int stellen eine Modbus-Wortadresse dar.
  - Datentypen mit 32-Bit-Doppelwörtern wie DWORD, DInt und Real stellen zwei Modbus-Wortadressen dar.
- Komplexe DB-Elemente können von MB_DATA_PTR zugewiesen werden, z.B.
  - Standard-Arrays
  - Benannte Strukturen, in denen jedes Element eindeutig ist.
  - Benannte komplexe Strukturen, in denen jedes Element einen eindeutigen Namen und einen 16- oder 32-Bit-Datentyp hat.
- Es ist nicht erforderlich, dass die Datenbereiche für MB_DATA_PTR in demselben globalen Datenbaustein (oder Speicherbereich der Merker) liegen. Sie können einen Datenbaustein für Modbus-Lesevorgänge, einen anderen Datenbaustein für Modbus-Schreibvorgänge oder einen Datenbaustein für jede MB_CLIENT-Station anlegen.

Mehrere Clientverbindungen

Ein Modbus TCP-Client unterstützt gleichzeitige Verbindungen bis zur maximalen für das PLC-Gerät zulässigen Anzahl von Open User Communications-Verbindungen. Die Gesamtzahl der Verbindungen für ein PLC-Gerät, einschließlich Modbus TCP-Clients und -Server, darf die maximale Anzahl der unterstützten Open User Communications-Verbindungen nicht überschreiten (Seite 633). Die Modbus TCP-Verbindungen können von Client- und/oder Serververbindungen gemeinsam genutzt werden.

Einzelne Clientverbindungen müssen die folgenden Regeln einhalten:

- Jede MB_CLIENT-Verbindung muss einen unterschiedlichen Instanz-DB nutzen.
- Jede MB_CLIENT-Verbindung muss eine eindeutige IP-Adresse eines Servers angeben.
- Jede MB_CLIENT-Verbindung muss eine eindeutige Verbindungs-ID angeben.
- Eindeutige IP-Portnummern sind möglicherweise je nach Serverkonfiguration erforderlich.

Die Verbindungs-ID muss für jede einzelne Verbindung eindeutig sein. Das bedeutet, dass für die einzelnen Instanz-DBs nur jeweils eine einzelne, eindeutige Verbindungs-ID verwendet werden darf. Zusammengefasst heißt dies, dass Instanz-DB und Verbindungs-ID gepaart sind und für jede Verbindung eindeutig sein müssen.

Tabelle 13- 129 Für den Benutzer zugänglich statische Variablen des Instanz-Datenbausteins MB_CLIENT

Variable	Datentyp	Voreinstellung	Beschreibung
Blocked_Proc_Timeout	Real	3.0	Zeitdauer (in Sekunden), die auf eine blockierte Modbus-Client-Instanz gewartet werden soll, bevor diese Instanz als AKTIV entfernt wird. Dies kann beispielsweise vorkommen, wenn eine Clientanforderung ausgegeben wurde und die Anwendung dann aufhört, die Clientfunktion auszuführen, bevor die Anforderung vollständig abgearbeitet wurde. Maximaler Grenzwert bei der S7-1200 sind 55 Sekunden.
MB_Unit_ID	Wort	255	Modbus-Gerätekennung: Ein Modbus TCP-Server wird über seine IP-Adresse angesprochen. Deshalb wird der Parameter MB_UNIT_ID bei der Modbus TCP-Adressierung nicht verwendet. Der Parameter MB_UNIT_ID entspricht dem Feld der Slaveadresse beim Modbus RTU-Protokoll. Wenn ein Modbus TCP-Server als Gateway zu einem Modbus RTU-Protokoll verwendet wird, kann das Slavegerät im seriellen Netzwerk über MB_UNIT_ID identifiziert werden. Der Parameter MB_UNIT_ID würde in diesem Fall die Anforderung an die richtige Modbus RTU-Slaveadresse weiterleiten. Einige Modbus TCP-Geräte benötigen den Parameter MB_UNIT_ID möglicherweise für die Initialisierung innerhalb eines eingeschränkten Wertebereichs.
RCV_TIMEOUT	Real	2.0	Zeit in Sekunden, die MB_CLIENT auf die Antwort eines Servers auf eine Anforderung wartet.
Verbunden	Bool	0	Gibt an, ob die Verbindung zum zugewiesenen Server verbunden oder nicht verbunden ist: 1 = verbunden, 0 = nicht verbunden

Tabelle 13- 130 MB_CLIENT-Protokollfehler

STATUS (W#16#)	Antwortcode an Modbus-Client (B#16#)	Modbus-Protokollfehler
8381	01	Funktionscode nicht unterstützt
8382	03	Fehler in der Datenlänge
8383	02	Datenadressfehler oder Zugriff außerhalb der Grenzen des Adressbereichs von MB_HOLD_REG
8384	03	Fehler im Datenwert
8385	03	Wert des Datendiagnosecodes wird nicht unterstützt (Funktionscode 08)

Tabelle 13- 131 Bedingungscodes der Ausführung von MB_CLIENT [1]

STATUS (W#16#)	MB_CLIENT-Parameterfehler
7001	MB_CLIENT wartet auf die Antwort eines Modbus-Servers auf die Anforderung eines Verbindungsaufbaus oder -abbaus am zugewiesenen TCP-Port. Dies wird nur für die erste Ausführung eines Verbindungsaufbaus oder -abbaus gemeldet.
7002	MB_CLIENT wartet auf die Antwort eines Modbus-Servers auf die Anforderung eines Verbindungsaufbaus oder -abbaus am zugewiesenen TCP-Port. Dies wird für alle nachfolgenden Ausführungen gemeldet, während auf die Fertigstellung eines Verbindungsausbaus oder -abbaus gewartet wird.
7003	Ein Verbindungsabbau wurde erfolgreich durchgeführt (nur einen PLC-Zyklus lang gültig).
80C8	Der Server hat nicht während der festgelegten Zeit reagiert. MB_CLIENT muss innerhalb der zugewiesenen Zeit über die ursprünglich gesendete Transaktions-ID eine Antwort erhalten oder es wird dieser Fehler ausgegeben. Prüfen Sie die Verbindung zum Modbus-Servergerät. Dieser Fehler wird erst gemeldet, nachdem konfigurierte Wiederholungen (sofern zutreffend) durchgeführt wurden.
8188	Moduswert ungültig
8189	Ungültiger Wert für die Datenadresse
818A	Ungültiger Wert für die Datenlänge
818B	Ungültiger Pointer auf den DATA_PTR-Bereich. Hierbei kann es sich um die Kombination aus MB_DATA_ADDRESS + MB_DATA_LEN handeln.
818C	Pointer auf einen optimierten DATA_PTR-Bereich (hierbei muss es sich um einen Standard-DB-Bereich oder um einen Bereich im Speicherbereich der Merker handeln)
8200	Der Port ist durch die Verarbeitung einer vorhandenen Modbus-Anforderung belegt.
8380	Der empfangene Modbus-Rahmen hat ein fehlerhaftes Format oder es wurden zu wenige Bytes empfangen.
8387	Der zugewiesene Parameter der Verbindungs-ID unterscheidet sich von der für vorherige Anforderungen verwendeten ID. Es darf in jedem Instanz-DB von MB_CLIENT nur eine einzige Verbindungs-ID verwendet werden. Dies ist auch ein interner Fehler, wenn die von einem Server empfangene Modbus TCP-Protokoll-ID nicht 0 ist.
8388	Ein Modbus-Server hat eine Datenmenge zurückgegeben, die sich von der angeforderten Menge unterscheidet. Dies gilt nur für die Modbus-Funktionen 15 und 16.

[1] Neben den oben aufgeführten MB_CLIENT-Fehlern können auch von den zugrunde liegenden T-Bausteinanweisungen für die Kommunikation (TCON, TDISCON, TSEND und TRCV (Seite 688)) Fehler gemeldet werden.

13.8.3.2 Anweisung MB_SERVER (Über PROFINET als Modbus TCP-Server kommunizieren)

MB_SERVER kommuniziert als Modbus TCP-Server über den PROFINET-Anschluss an der S7-1200 CPU. MB_SERVER verarbeitet Verbindungsanforderungen eines Modbus TCP-Clients, empfängt und verarbeitet Modbus-Anforderungen und sendet Antworten.

Für die Verwendung der Anweisung wird kein zusätzliches Hardwaremodul benötigt.

ACHTUNG
Sicherheitshinweise Jeder Client im Netzwerk hat Lese- und Schreibzugriff auf die Ein- und Ausgänge im Prozessbild und auf den Datenbaustein- oder Bitspeicherbereich, die im Modbus-Halteregister festgelegt sind. Diese Option soll den Zugriff auf eine IP-Adresse einschränken und unbefugtes Lesen und Schreiben verhindern. Zu beachten ist jedoch, dass die geteilte Adresse auch für den unbefugten Zugriff verwendet werden kann.

Tabelle 13- 132 Anweisung MB_SERVER

KOP/FUP	SCL	Beschreibung

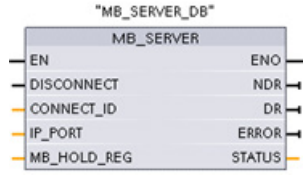
	`"MB_SERVER_DB"(`
	`DISCONNECT:=_bool_in_,`
	`CONNECT_ID:=_uint_in_,`
	`IP_PORT:=_uint_in_,`
	`NDR=>_bool_out_,`
	`DR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_HOLD_REG:=_variant_inout_);`
		MB_SERVER kommuniziert als Modbus TCP-Server über den PROFINET-Anschluss an der S7-1200 CPU. Es ist kein zusätzliches Hardwaremodul für die Kommunikation erforderlich.
		MB_SERVER kann eine Anforderung für den Verbindungsaufbau mit einem Modbus TCP-Client annehmen, eine Modbus-Funktionsanforderung empfangen und eine Antwortmeldung senden.

Tabelle 13- 133 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
DISCONNECT	IN	Bool	MB_SERVER versucht, eine "passive" Verbindung mit einem Partnergerät aufzubauen. Das bedeutet, dass der Server passiv überwacht, ob es TCP-Verbindungsanforderungen von IP-Adressen gibt, die Anforderungen senden. Ist DISCONNECT = 0 und keine Verbindung vorhanden, kann eine passive Verbindung aufgebaut werden. Ist DISCONNECT = 1 und eine Verbindung vorhanden, wird versucht, die Verbindung zu trennen. Auf diese Weise kann Ihr Programm steuern, wann eine Verbindung angenommen wird. Immer wenn dieser Eingang aktiviert ist, wird kein anderer Vorgang eingeleitet.
CONNECT_ID	IN	UInt	CONNECT_ID identifiziert jede Verbindung im PLC-Gerät eindeutig. Jede eindeutige Instanz der Anweisung MB_CLIENT oder MB_SERVER muss einen eindeutigen Parameter CONNECT_ID enthalten.
IP_PORT	IN	UInt	Standardwert = 502: Die IP-Portnummer, die den Port identifiziert, der von einem Modbus-Client auf eine Verbindungsanforderung hin überwacht wird. Die folgenden TCP-Portnummern sind für eine passive MB_SERVER-Verbindung nicht zulässig: 20, 21, 25, 80, 102, 123, 5001, 34962, 34963 und 34964.
MB_HOLD_REG	IN_OUT	Variant	Pointer auf das Modbus-Halteregister von MB_SERVER: Bei dem Halteregister muss es sich um einen globalen Standard-DB oder um eine Adresse im Speicherbereich der Merker handeln. Dieser Speicherbereich dient dazu, die Werte zu speichern, auf die ein Modbus-Client mit den Modbus-Registerfunktionen 3 (Lesen), 6 (Schreiben) und 16 (Schreiben) zugreifen kann.
NDR	OUT	Bool	Neue Daten bereit: 0 = Keine neuen Daten, 1 = Gibt an, dass von einem Modbus-Client neue Daten geschrieben wurden
DR	OUT	Bool	Daten lesen: 0 = Daten nicht gelesen, 1 = Gibt an, dass die Daten von einem Modbus-Client gelesen wurden
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die Ausführung von MB_SERVER mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung

Bei MB_SERVER können eingehende Modbus-Funktionscodes (1, 2, 4, 5 und 15) Bits und Wörter direkt in den Prozessabbildern der Eingänge und Ausgänge der S7-1200 CPU lesen und schreiben. Bei den Funktionscodes für die Datenübertragung (3, 6 und 16) muss der Parameter MB_HOLD_REG als Datentyp größer als ein Byte definiert sein. Die folgende Tabelle zeigt die Zuordnung der Modbus-Adressen zum Prozessabbild in der CPU.

Tabelle 13- 134 Zuordnung der Modbus-Adressen zum Prozessabbild

Modbus-Funktionen						S7-1200	
Codes	Funktion	Datenbereich	Adressbereich			Datenbereich	CPU-Adresse
01	Bits lesen	Ausgang	1	An	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7
02	Bits lesen	Eingang	10001	An	18192	Prozessabbild der Eingänge	E0.0 bis E1023.7
04	Wörter lesen	Eingang	30001	An	30512	Prozessabbild der Eingänge	EW0 bis EW1022
05	Bit schreiben	Ausgang	1	An	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7
15	Bits schreiben	Ausgang	1	An	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7

Funktionscodes eingehender Modbus-Meldungen (3, 6 und 16) lesen oder schreiben Wörter in einem Modbus-Halteregister, bei dem es sich um einen Adressbereich im Speicherbereich der Merker oder um einen Datenbaustein handeln kann. Der Typ des Halteregisters wird vom Parameter MB_HOLD_REG angegeben.

Hinweis

Zuweisung des Parameters MB_HOLD_REG

Bei dem Modbus-Halteregister kann es sich um einen globalen Standard-DB oder um eine Adresse im Speicherbereich der Merker handeln.

Für ein Modbus-Halteregister im Speicherbereich der Merker verwenden Sie das Standardformat des Any-Pointer. Hierbei handelt es sich um das Format P#"Bitadresse" "Datentyp" "Länge". Beispiel: P#M1000.0 WORD 500.

Die folgende Tabelle zeigt Beispiele für die Zuordnung von Modbus-Adressen zum Halteregister, das für die Modbus-Funktionscodes 03 (Wörter lesen), 06 (Wort schreiben) und 16 (Wörter schreiben) verwendet wird. Die tatsächliche obere Grenze der DB-Adressen wird vom maximalen Arbeitsspeicher und vom maximalen M-Speicher des jeweiligen CPU-Modells festgelegt.

Tabelle 13- 135 Beispiele für die Zuordnung von Modbus-Adressen zu Adressen im Speicher der CPU

Modbus-Adresse	Beispiele für Parameter von MB_HOLD_REG		
	P#M100.0 Word 5	P#DB10.DBx0.0 Word 5	"Rezept".Inhaltsstoff
40001	MW100	DB10.DBW0	"Rezept".Inhaltsstoff[1]
40002	MW102	DB10.DBW2	"Rezept".Inhaltsstoff[2]
40003	MW104	DB10.DBW4	"Rezept".Inhaltsstoff[3]
40004	MW106	DB10.DBW6	"Rezept".Inhaltsstoff[4]
40005	MW108	DB10.DBW8	"Rezept".Inhaltsstoff[5]

Mehrere Serververbindungen

Es können mehrere Serververbindungen hergestellt werden. Dadurch ist es möglich, dass ein einziges PLC-Gerät gleichzeitig Verbindungen mit mehreren Modbus TCP-Clients herstellt.

Ein Modbus TCP-Server unterstützt gleichzeitige Verbindungen bis zur maximalen für das PLC-Gerät zulässigen Anzahl von Open User Communications-Verbindungen. Die Gesamtzahl der Verbindungen für ein PLC-Gerät, einschließlich Modbus TCP-Clients und -Server, darf die maximale Anzahl der unterstützten Open User Communications-Verbindungen nicht überschreiten (Seite 633). Die Modbus TCP-Verbindungen können von Client- und/oder Serververbindungen gemeinsam genutzt werden.

Einzelne Serververbindungen müssen die folgenden Regeln einhalten:

- Jede MB_SERVER-Verbindung muss einen unterschiedlichen Instanz-DB nutzen.
- Jede MB_SERVER-Verbindung muss mit einer eindeutigen IP-Portnummer hergestellt werden. Je Port wird nur eine Verbindung unterstützt.
- Jede MB_SERVER-Verbindung muss eine eindeutige Verbindungs-ID verwenden.
- MB_SERVER muss für jede Verbindung einzeln aufgerufen werden (mit dem entsprechenden Instanz-DB).

Die Verbindungs-ID muss für jede einzelne Verbindung eindeutig sein. Das bedeutet, dass nur mit jedem einzelnen Instanz-DB eine einzelne, eindeutige Verbindungs-ID verwendet werden darf. Zusammengefasst heißt dies, dass Instanz-DB und Verbindungs-ID gepaart sind und für jede Verbindung eindeutig sein müssen.

Tabelle 13- 136 Funktionscodes der Modbus-Diagnose

Modbus-Diagnosefunktionen von MB_SERVER		
Codes	Teilfunktion	Beschreibung
08	0x0000	Echotest Abfragedaten ausgeben: Die Anweisung MB_SERVER gibt einem Modbus-Client das Echo eines empfangenen Datenworts zurück.
08	0x000A	Kommunikationsereigniszähler löschen: Die Anweisung MB_SERVER löscht den Kommunikationsereigniszähler, der für Modbus-Funktion 11 verwendet wird.
11		Kommunikationsereigniszähler abrufen: Die Anweisung MB_SERVER nutzt einen internen Kommunikationsereigniszähler, um die Anzahl erfolgreicher Modbus-Lese- und Modbus-Schreibanforderungen, die an den Modbus-Server gesendet werden, zu erfassen. Der Zähler wird bei Funktion 8 oder Funktion 11 nicht hochgezählt. Auch bei Anforderungen, die zu einem Kommunikationsfehler führen, wird der Zähler nicht inkrementiert. Die Broadcast-Funktion ist bei Modbus TCP nicht verfügbar, weil nur eine Client-Server-Verbindung zur Zeit vorhanden ist.

Variablen von MB_SERVER

Diese Tabelle zeigt die öffentlichen statischen Variablen im Instanz-Datenbaustein von MB_SERVER, die in Ihrem Programm verwendet werden können.

Tabelle 13- 137 Öffentliche statische Variablen von MB_SERVER

Variable	Datentyp	Standardwert	Beschreibung
HR_Start_Offset	Wort	0	Weist die Anfangsadresse des Modbus-Halteregisters zu
Request_Count	Wort	0	Die Anzahl aller von diesem Server empfangenen Anforderungen
Server_Message_Count	Wort	0	Die Anzahl der für diesen spezifischen Server empfangenen Anforderungen
Xmt_Rcv_Count	Wort	0	Die Anzahl der Sendungen oder Datenempfänge, bei denen ein Fehler aufgetreten ist. Wird auch inkrementiert, wenn eine Meldung empfangen wird, bei der es sich um eine ungültige Modbus-Meldung handelt.
Exception_Count	Wort	0	Modbus-spezifische Fehler, die eine zurückgegebene Ausnahme erfordern
Success_Count	Wort	0	Die Anzahl der für diesen spezifischen Server empfangenen Anforderungen ohne Protokollfehler.
Verbunden	Bool	0	Gibt an, ob die Verbindung zum zugewiesenen Client verbunden oder nicht verbunden ist: 1 = verbunden, 0 = nicht verbunden

Ihr Programm kann in HR_Start_Offset Werte schreiben und die Modbus-Server-Operationen steuern. Die anderen Variablen können gelesen werden, um den Modbus-Zustand zu überwachen.

HR_Start_Offset

Die Adressen des Modbus-Halteregisters beginnen bei 40001. Diese Adressen entsprechen der Anfangsadresse des Halteregisters im Zielsystemspeicher. Sie können jedoch die Variable HR_Start_Offset konfigurieren, um eine andere Anfangsadresse als 40001 für das Modbus-Halteregister zu konfigurieren.

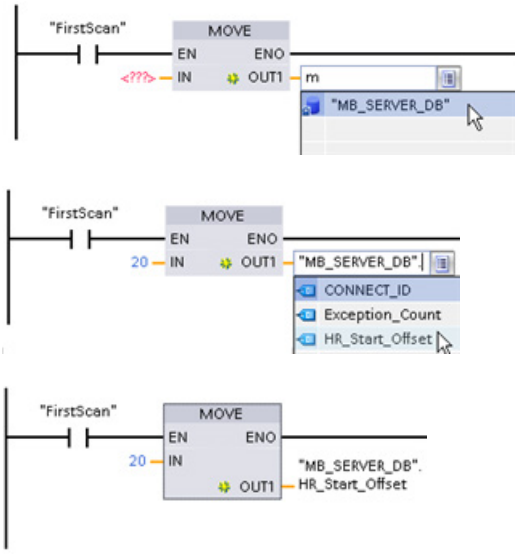
Sie können z. B. ein Halteregister mit Beginn an MW100 und einer Länge von 100 Wörtern konfigurieren. Mit einem Versatz von 20 geben Sie eine Anfangsadresse des Halteregisters von 40021 statt 40001 an. Jede Adresse unter 40021 und über 40119 führt zu einem Adressierungsfehler.

Tabelle 13- 138 Beispiel für die Adressierung des Modbus-Halteregisters

HR_Start_Offset	Adresse	Minimum	Maximum
0	Modbus-Adresse (Wort)	40001	40099
	S7-1200 Adresse	MW100	MW298
20	Modbus-Adresse (Wort)	40021	40119
	S7-1200 Adresse	MW100	MW298

HR_Start_Offset ist ein Wortwert, der die Anfangsadresse des Modbus-Halteregisters angibt und im Instanz-Datenbaustein MB_SERVER gespeichert ist. Sie können diese öffentliche statische Variable über die Parameter-Klappliste auswählen, nachdem Sie MB_SERVER in Ihr Programm eingefügt haben.

Wenn Sie beispielsweise MB_SERVER in ein KOP-Netzwerk eingefügt haben, können Sie in ein vorheriges Netzwerk gehen und den Wert HR_Start_Offset zuweisen. Der Wert muss vor der Ausführung von MB_SERVER zugewiesen werden.



Eingabe einer Variable für den Modbus-Server

über den Namen des Standard-DB:

1. Positionieren Sie den Cursor im Parameterfeld und geben Sie das Zeichen m ein.
2. Wählen Sie in der Klappliste der DB-Namen "MB_SERVER_DB" aus.
3. Wählen Sie in der Klappliste der DB-Variablen "MB_SERVER_DB.HR_Start_Offset" aus.

Tabelle 13- 139 Bedingungscodes der Ausführung von MB_SERVER [1]

STATUS (W#16#)	Antwortcode an Modbus-Server (B#16#)	Modbus-Protokollfehler
7001		MB_SERVER wartet darauf, dass ein Modbus-Client die Verbindung zum zugewiesenen TCP-Port herstellt. Dieser Code wird bei der ersten Ausführung eines Verbindungsaufbaus oder -abbaus gemeldet.
7002		MB_SERVER wartet darauf, dass ein Modbus-Client die Verbindung zum zugewiesenen TCP-Port herstellt. Dieser Code wird für alle nachfolgenden Ausführungen gemeldet, während auf die Fertigstellung eines Verbindungsaufbaus oder -abbaus gewartet wird.
7003		Ein Verbindungsabbau wurde erfolgreich durchgeführt (nur einen PLC-Zyklus lang gültig).
8187		Ungültiger Pointer auf MB_HOLD_REG: Bereich ist zu klein
818C		Pointer auf einen optimierten MB_HOLD_REG-Bereich (hierbei muss es sich um einen Standard-DB-Bereich oder um einen Bereich im Speicherbereich der Merker handeln) oder Timeout für gesperrten Prozess überschreitet den Grenzwert von 55 Sekunden. (spezifisch für die S7-1200)
8381	01	Funktionscode nicht unterstützt
8382	03	Fehler in der Datenlänge
8383	02	Datenadressfehler oder Zugriff außerhalb der Grenzen des Adressbereichs von MB_HOLD_REG
8384	03	Fehler im Datenwert
8385	03	Wert des Datendiagnosecodes wird nicht unterstützt (Funktionscode 08)

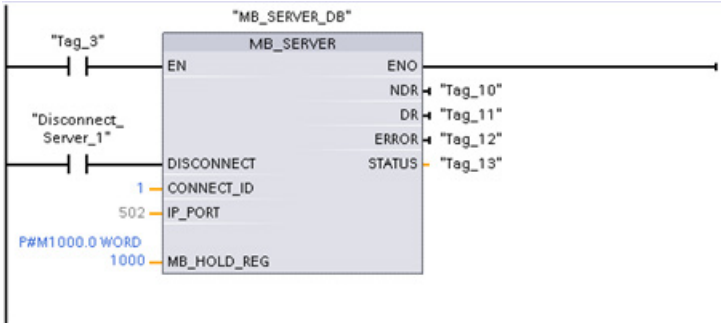
[1] Neben den oben aufgeführten MB_SERVER-Fehlern können auch von den zugrunde liegenden T-Bausteinanweisungen für die Kommunikation (TCON, TDISCON, TSEND und TRCV (Seite 688)) Fehler gemeldet werden.

13.8.4 Beispiel: MB_SERVER für mehrere TCP-Verbindungen in älteren Systemen

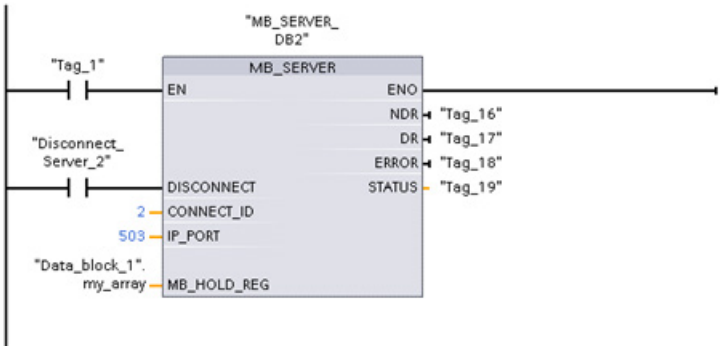
Sie können mehrere Verbindungen zum Modbus TCP-Server haben. Hierfür muss MB_SERVER für jede Verbindung unabhängig ausgeführt werden. Jede Verbindung muss einen unabhängigen Instanz-DB, eine Verbindungs-ID und einen IP-Port verwenden. Bei der S7-1200 ist nur je eine Verbindung pro IP-Port zulässig.

Um optimales Betriebsverhalten zu erzielen, muss MB_SERVER in jedem Programmzyklus für jede Verbindung ausgeführt werden.

Netzwerk 1: Verbindung Nr. 1 mit unabhängigem IP_PORT, Verbindungs-ID und Instanz-DB



Netzwerk 2: Verbindung Nr. 2 mit unabhängigem IP_PORT, Verbindungs-ID und Instanz-DB



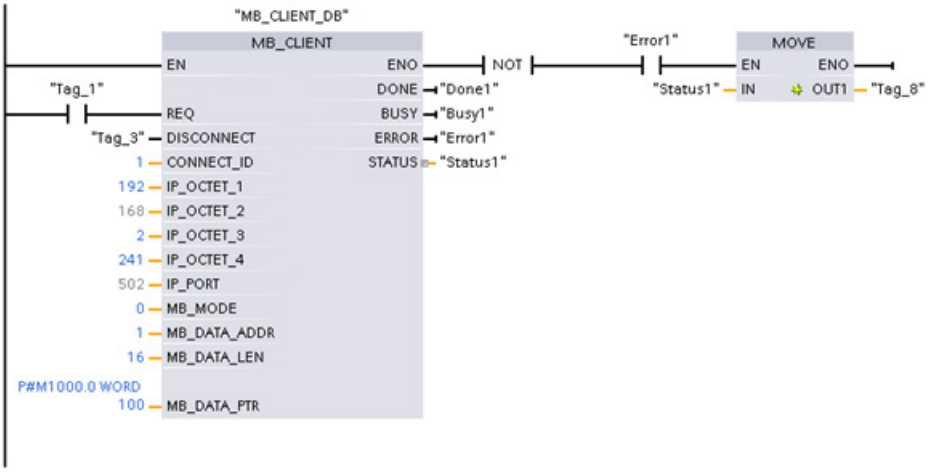
13.8.5 Beispiel: MB_CLIENT 1 in älteren Systemen: Mehrere Anforderungen mit gemeinsamer TCP-Verbindung

Mehrere Modbus-Clientanforderungen können über die gleiche Verbindung gesendet werden. Hierfür verwenden Sie den gleichen Instanz-DB, die gleiche Verbindungs-ID und Portnummer.

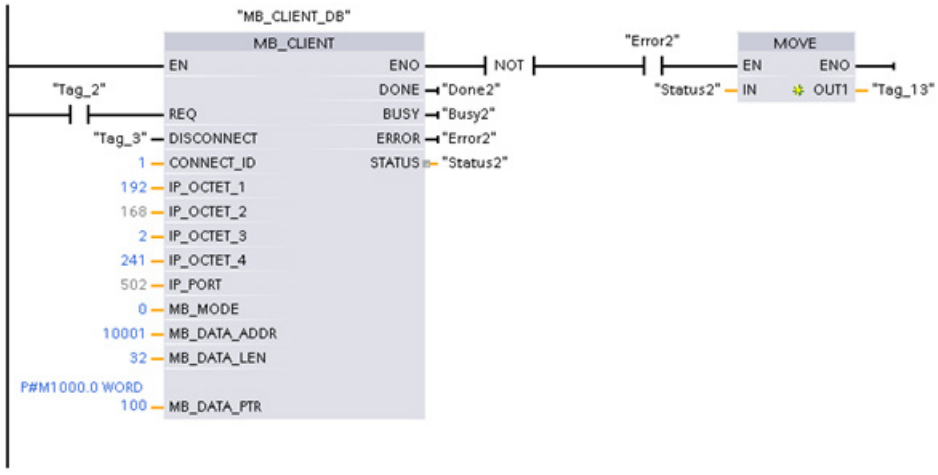
Es kann jeweils nur 1 Client aktiv sein. Nachdem ein Client seine Ausführung beendet hat, beginnt der nächste Client mit der Ausführung. Ihr Programm ist für die Reihenfolge der Ausführung verantwortlich.

Das Beispiel zeigt beide Clients, die in denselben Speicherbereich schreiben. Außerdem wird ein ausgegebener Fehler erfasst (optionale Funktion).

Netzwerk 1: Modbus-Funktion 1 - 16 Bits im Prozessabbild der Ausgänge lesen



Netzwerk 2: Modbus-Funktion 2 - 32 Bits im Prozessabbild der Eingänge lesen



13.8.6 Beispiel: MB_CLIENT 2 in älteren Systemen: Mehrere Anforderungen mit unterschiedlichen TCP-Verbindungen

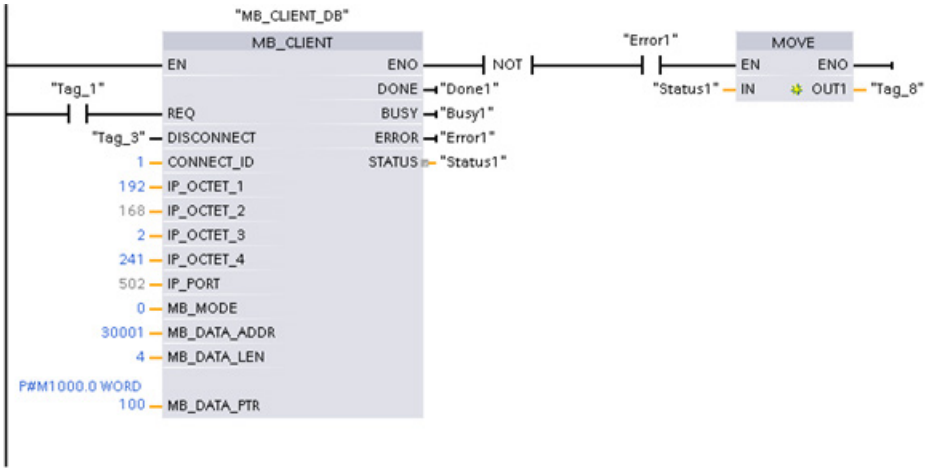
Mehrere Modbus-Clientanforderungen können über verschiedene Verbindungen gesendet werden. Hierfür müssen unterschiedliche Instanz-DBs, IP-Adressen und Verbindungs-IDs verwendet werden.

Die Portnummer muss unterschiedlich sein, wenn die Verbindungen mit demselben Modbus-Server aufgebaut werden. Bei Verbindungen zu unterschiedlichen Servern gibt es keine Einschränkungen hinsichtlich der Portnummer.

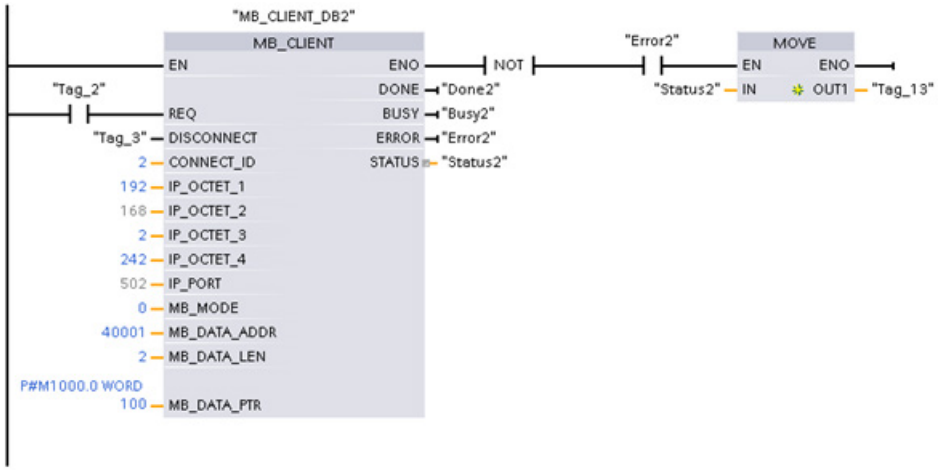
Das Beispiel zeigt beide Clients, die in denselben Speicherbereich schreiben. Außerdem wird ein ausgegebener Fehler erfasst (optionale Funktion).

Netzwerk 1:

Modbus-Funktion 4 - Eingangswörter lesen (im Speicher der S7-1200)



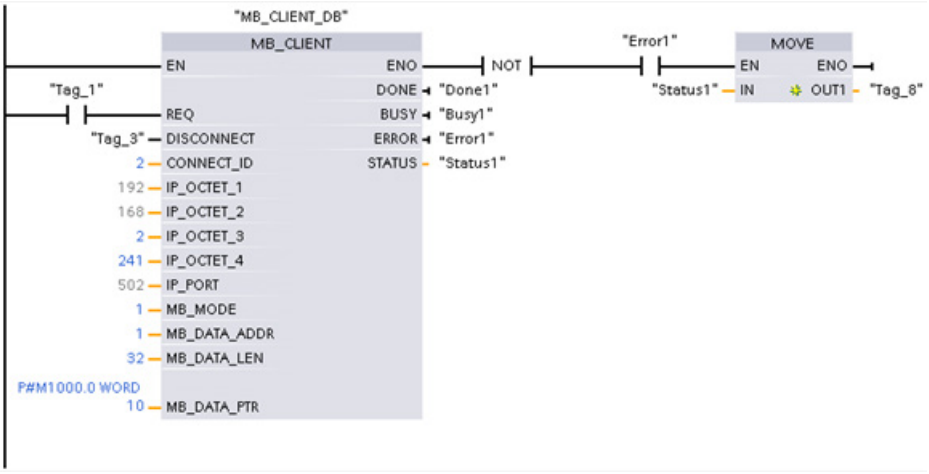
Netzwerk 2: Modbus-Funktion 3 - Wörter im Halteregister von einem Modbus-TCP-Server lesen



13.8.7 Beispiel: MB_CLIENT 3 in älteren Systemen: Schreibanforderung für das Prozessabbild der Ausgänge

Dieses Beispiel zeigt die Anforderung eines Modbus-Clients zum Schreiben in das Prozessabbild der Ausgänge der S7-1200.

Netzwerk 1: Modbus-Funktion 15 - Bits in das Prozessabbild der Ausgänge der S7-1200 schreiben

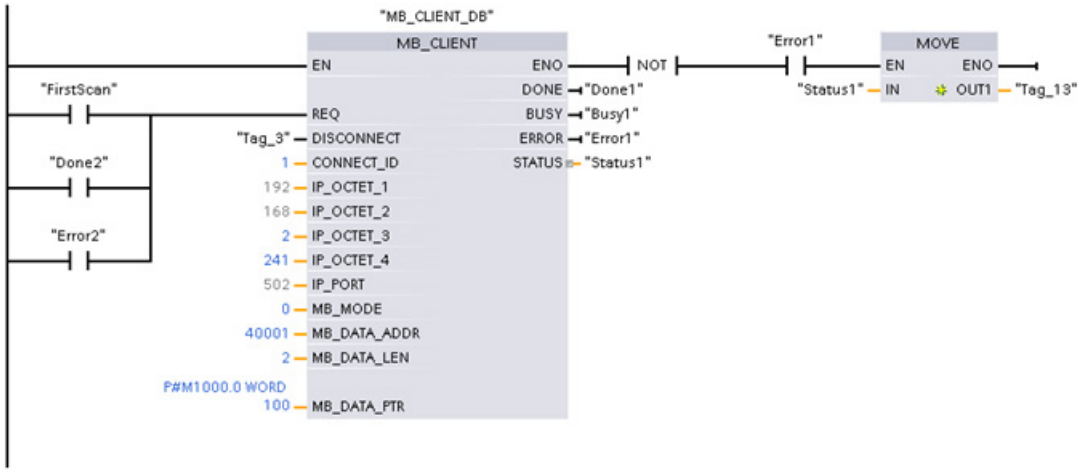


13.8.8 Beispiel: MB_CLIENT 4 in älteren Systemen: Mehrere Anforderungen koordinieren

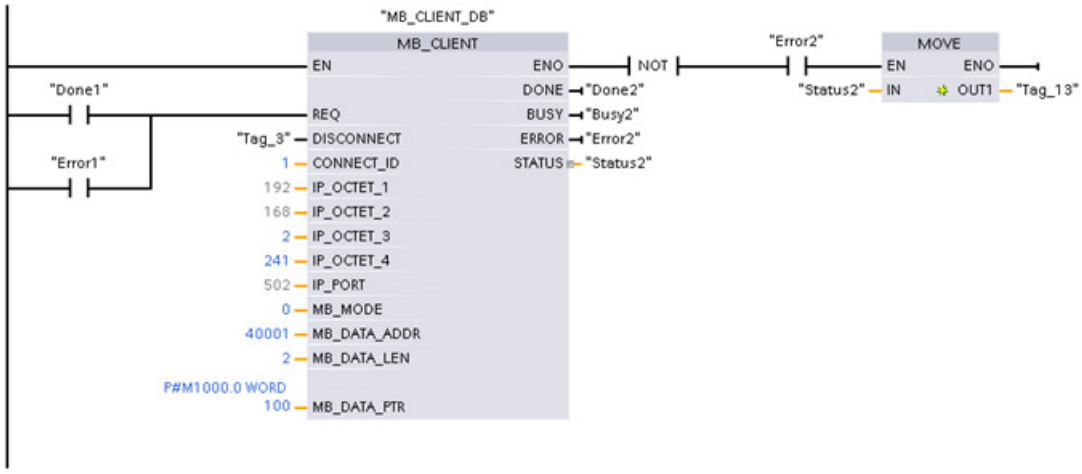
Sie müssen sicherstellen, dass die Ausführung jeder einzelnen Modbus TCP-Anforderung abgeschlossen wird. Diese Koordination muss durch Ihr Programm erfolgen. Das folgende Beispiel zeigt, wie die Ausführung mit Hilfe der Ausgänge der ersten und der zweiten Clientanforderung koordiniert werden kann.

Das Beispiel zeigt beide Clients, die in denselben Speicherbereich schreiben. Außerdem wird ein ausgegebener Fehler erfasst (optionale Funktion).

Netzwerk 1: Modbus-Funktion 3 - Wörter im Halteregister lesen



Netzwerk 2: Modbus-Funktion 3 - Wörter im Halteregister lesen



13.9 Modbus RTU-Kommunikation in älteren Systemen (nur CM/CB 1241)

13.9.1 Übersicht

Vor STEP 7 V13 SP1 und den S7-1200 V4.1-CPUs hatten die Modbus RTU-Kommunikationsanweisungen andere Namen und teilweise auch etwas andere Schnittstellen. Für beide Anweisungsarten gelten die allgemeinen Konzepte. Informationen zur Programmierung finden sich in den Beschreibungen zu den einzelnen älteren Modbus RTU-Anweisungen.

13.9.2 Version der Modbus RTU-Anweisungen auswählen

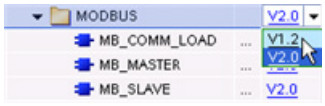
Es gibt zwei Versionen der Modbus RTU-Anweisungen in STEP 7:

- Version 1 war zunächst in STEP 7 Basic V10.5 verfügbar.
- Version 2 ist in STEP 7 Basic/Professional V11 verfügbar. Die Version 2 bietet die zusätzlichen Parameter REQ und DONE bei MB_COMM_LOAD. Außerdem gestattet der Parameter MB_ADDR bei MB_MASTER und MB_SLAVE nun einen UInt-Wert für die erweiterte Adressierung.

Zum Zweck der Kompatibilität und zur einfacheren Migration können Sie wählen, welche Version der Anweisungen Sie in Ihr Anwenderprogramm einfügen.

Sie können nicht beide Versionen der Anweisungen bei demselben Modul verwenden, Sie können jedoch für zwei verschiedene Module unterschiedliche Versionen der Anwendungen einsetzen. Die Modbus RTU-Anweisungen in Ihrem Programm müssen dieselbe Hauptversionsnummer haben (**1**.x, **2**.y oder **V**.z). Die einzelnen Anweisungen in einer Versionsgruppe können dann verschiedene Unterversionen aufweisen (1.**x**).

Klicken Sie in der Taskcard mit dem Anweisungsverzeichnis auf das Symbol, um die Überschriften und Spalten im Anweisungsverzeichnis zu aktivieren.



Um die Version einer Modbus-Anweisung zu ändern, wählen Sie die entsprechende Version in der Klappliste aus. Sie können die Gruppe oder einzelne Anweisungen auswählen.

Wenn Sie eine Modbus-Anweisung über das Anweisungsverzeichnis in Ihr Programm einfügen, wird in der Projektnavigation eine neue FB-Instanz angelegt. Die neue FB-Instanz finden Sie in der Projektnavigation unter PLC_x > Programmbausteine > Systembausteine > Programmressourcen.

Um die Version einer Modbus-Anweisung in einem Programm zu ermitteln, müssen Sie die Eigenschaften in der Projektnavigation aufrufen, nicht die Eigenschaften einer im Programmeditor angezeigten Box. Wählen Sie in der Projektnavigation die Instanz eines Modbus-FBs aus, klicken Sie mit der rechten Maustaste, wählen Sie "Eigenschaften" und dann die Seite "Informationen", um die Versionsnummer der Modbus-Anweisung anzuzeigen.

13.9.3 Ältere Modbus RTU-Anweisungen

13.9.3.1 Anweisung MB_COMM_LOAD (Port am PtP-Modul für Modbus RTU konfigurieren)

Tabelle 13- 140 Anweisung MB_COMM_LOAD

KOP/FUP	SCL	Beschreibung

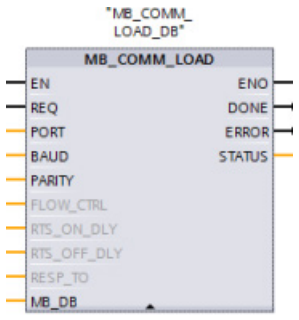
	`"MB_COMM_LOAD_DB"(`
	`REQ:=_bool_in,`
	`PORT:=_uint_in_,`
	`BAUD:=_udint_in_,`
	`PARITY:=_uint_in_,`
	`FLOW_CTRL:=_uint_in_,`
	`RTS_ON_DLY:=_uint_in_,`
	`RTS_OFF_DLY:=_uint_in_,`
	`RESP_TO:=_uint_in_,`
	`DONE=>_bool_out,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_DB:=_fbtref_inout_);`
		Die Anweisung MB_COMM_LOAD konfiguriert einen PtP-Port für die Kommunikation über das Modbus-RTU-Protokoll. Hardwareoptionen des Modbus-Ports: Einbau von max. drei CMs (RS485 oder RS232), plus eines CB (R4845). Wenn Sie die Anweisung MB_COMM_LOAD in Ihr Programm einfügen, wird automatisch ein Instanz-Datenbaustein zugewiesen.

Tabelle 13- 141 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet. (Nur Version 2.0)
PORT	IN	Port	Nachdem Sie das CM oder CB eingebaut und konfiguriert haben, erscheint die Portkennung in der Parameter-Klappliste am Box-Anschluss PORT. Der zugewiesene CM- oder CB-Portwert ist die Eigenschaft "Hardwarekennung" der Gerätekonfiguration. Der symbolische Name des Ports wird im Register "Systemkonstanten" der PLC-Variablentabelle zugewiesen.
BAUD	IN	UDInt	Auswahl der Baudrate: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200. Alle anderen Werte sind ungültig.
PARITY	IN	UInt	Auswahl der Parität: • 0 – Keine • 1 – Ungerade • 2 – Gerade

Parameter und Datentyp		Datentyp	Beschreibung
FLOW_CTRL [1]	IN	UInt	Auswahl der Flusskontrolle: • 0 – (Standard) Keine Flusskontrolle • 1 – Hardware-Flusskontrolle mit RTS immer EIN (gilt nicht bei RS485-Ports) • 2 – Hardware-Flusskontrolle mit RTS geschaltet
RTS_ON_DLY [1]	IN	UInt	Auswahl RTS-Einschaltverzögerung: • 0 – (Standard) Keine Verzögerung von "RTS aktiv", bis das erste Zeichnung der Meldung gesendet wird. • 1 bis 65535 – Verzögerung in Millisekunden von "RTS aktiv", bis das erste Zeichen der Meldung gesendet wird (gilt nicht bei RS485-Ports). RTS-Verzögerungen müssen unabhängig von der Auswahl FLOW_CTRL angewendet werden.
RTS_OFF_DLY [1]	IN	UInt	Auswahl RTS-Ausschaltverzögerung: • 0 – (Standard) Keine Verzögerung nach der Übertragung des letzten Zeichens bis "RTS inaktiv" • 1 bis 65535 – Verzögerung in Millisekunden von der Übertragung des letzten Zeichens bis "RTS inaktiv" (gilt nicht bei RS485-Ports). RTS-Verzögerungen müssen unabhängig von der Auswahl FLOW_CTRL angewendet werden.
RESP_TO [1]	IN	UInt	Antwort-Timeout: Zeit in Millisekunden, die MB_MASTER auf eine Antwort vom Slave wartet. Wenn der Slave während dieses Zeitraums nicht antwortet, wiederholt MB_MASTER die Anforderung oder beendet die Anforderung mit einem Fehler, wenn die angegebene Anzahl von Wiederholungen gesendet wurde. 5 ms bis 65535 ms (Standardwert = 1000 ms).
MB_DB	IN	Variant	Ein Verweis auf den Instanz-Datenbaustein der Anweisungen MB_MASTER oder MB_SLAVE. Nachdem Sie MB_SLAVE oder MB_MASTER in Ihr Programm eingefügt haben, ist die DB-Kennung in der Parameter-Klappliste am Anschluss der MB_DB-Box verfügbar.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde. (Nur Version 2.0)
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung

[1] Optionale Parameter für MB_COMM_LOAD (V 2.x oder höher). Klicken Sie auf den Pfeil im unteren Bereich der KOP/FUP-Box, um die Box zu erweitern und diese Parameter aufzunehmen.

MB_COMM_LOAD wird ausgeführt, um einen Port für das Modbus-RTU-Protokoll zu konfigurieren. Nachdem ein Port für das Modbus-RTU-Protokoll konfiguriert ist, kann er nur von den Anweisungen MB_MASTER oder MB_SLAVE verwendet werden.

Für die Konfiguration jedes Kommunikationsports, der für die Modbus-Kommunikation eingesetzt wird, muss eine Ausführung von MB_COMM_LOAD genutzt werden. Sie müssen jedem Port, den Sie nutzen, einen eindeutigen Instanz-DB MB_COMM_LOAD zuweisen. Sie können bis zu drei Kommunikationsmodule (RS232 oder RS485) und ein Kommunikationsboard (RS485) in die CPU einbauen. Sie können MB_COMM_LOAD aus einem Anlauf-OB aufrufen und einmal ausführen, oder Sie können den Aufruf für die einmalige Ausführung mit Hilfe des Systemmerkers für den ersten Zyklus (Seite 106) initiieren. Führen Sie MB_COMM_LOAD nur dann erneut aus, wenn Kommunikationsparameter wie Baudrate oder Parität geändert werden müssen.

Wenn Sie MB_MASTER oder MB_SLAVE in Ihr Programm einfügen, wird der Anweisung ein Instanz-Datenbaustein zugewiesen. Dieser Instanz-Datenbaustein wird referenziert, wenn Sie in der Anweisung MB_COMM_LOAD den Parameter MB_DB angeben.

MB_COMM_LOAD Datenbausteinvariablen

Die folgende Tabelle zeigt die öffentlichen statischen Variablen im Instanz-DB von MB_COMM_LOAD, die in Ihrem Programm verwendet werden können.

Tabelle 13- 142 Statische Variablen im Instanz-DB

Variable	Datentyp	Beschreibung
ICHAR_GAP	UInt	Verzögerung für den Zeichenabstand zwischen den Zeichen. Dieser Parameter wird in Millisekunden angegeben und dient dazu, den erwarteten Zeitraum zwischen den empfangenen Zeichen zu erhöhen. Die entsprechende Anzahl Bitzeiten für diesen Parameter wird zum Modbus-Standardwert von 35 Bitzeiten (3,5 Zeichenzeiten) addiert.
RETRIES	UInt	Die Anzahl der wiederholten Versuche, die der Master durchführt, bevor der Fehlercode 0x80C8 für "Keine Antwort" zurückgegeben wird.
STOP_BITS	USInt	Anzahl Stoppbits für jeden Frame eines Zeichens. Gültig sind die Werte 1 und 2.

Tabelle 13- 143 Bedingungscodes der Ausführung von MB_COMM_LOAD [1]

STATUS (W#16#)	Beschreibung
0000	Kein Fehler
8180	Ungültiger Wert der Port-ID (falscher Port/falsche Hardwarekennung des Kommunikationsmoduls)
8181	Ungültiger Wert für die Baudrate.
8182	Ungültiger Wert für die Parität.
8183	Ungültiger Wert für die Flusskontrolle.
8184	Ungültiger Wert für die Timeout-Zeit der Antwort (Antwort-Timeout ist kleiner als der Mindestwert von 5 ms)
8185	Der Parameter MB_DB ist kein Instanz-Datenbaustein einer Anweisung MB_MASTER oder MB_SLAVE.

[1] Neben den oben aufgeführten Fehlern der Anweisung MB_COMM_LOAD können auch die zugrunde liegenden PtP-Kommunikationsanweisungen Fehler zurückgeben.

13.9.3.2 Anweisung MB_MASTER (Über PtP-Port als Modbus RTU-Master kommunizieren)

Tabelle 13- 144 Anweisung MB_MASTER

KOP/FUP	SCL	Beschreibung

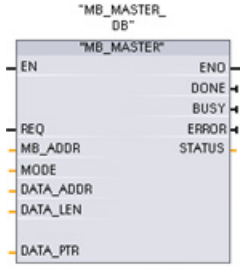
	`"MB_MASTER_DB"(`
	`REQ:=_bool_in_,`
	`MB_ADDR:=_uint_in_,`
	`MODE:=_usint_in_,`
	`DATA_ADDR:=_udint_in_,`
	`DATA_LEN:=_uint_in_,`
	`DONE=>_bool_out_,`
	`BUSY=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`DATA_PTR:=_variant_inout_);`
		Die Anweisung MB_MASTER kommuniziert als Modbus-Master über einen Port, der von einer vorherigen Ausführung der Anweisung MB_COMM_LOAD konfiguriert wurde. Wenn Sie die Anweisung MB_MASTER in Ihr Programm einfügen, wird automatisch ein Instanz-Datenbaustein zugewiesen. Dieser Instanz-Datenbaustein MB_MASTER wird verwendet, wenn Sie in der Anweisung MB_COMM_LOAD den Parameter MB_DB angeben.

Tabelle 13- 145 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
REQ	IN	Bool	0 = Keine Anforderung 1 = Anforderung, Daten an den Modbus-Slave zu senden
MB_ADDR	IN	V1.0: USInt V2.0: UInt	Modbus-RTU-Stationsadresse: Standard-Adressierungsbereich (1 bis 247) Erweiterter Adressierungsbereich (1 bis 65535) Der Wert 0 ist für den Broadcast einer Meldung an alle Modbus-Slaves reserviert. Für den Broadcast werden nur die Modbus-Funktionscodes 05, 06, 15 und 16 unterstützt.
MODE	IN	USInt	Auswahl Modus: Gibt die Art der Anforderung an (Lesen, Schreiben oder Diagnose). Weitere Informationen finden Sie in der Tabelle der Modbus-Funktionen unten.
DATA_ADDR	IN	UDInt	Anfangsadresse im Slave: Gibt die Anfangsadresse der Daten an, auf die im Modbus-Slave zugegriffen werden soll. Die gültigen Adressen finden Sie in der Tabelle der Modbus-Funktionen unten.
DATA_LEN	IN	UInt	Datenlänge: Gibt die Anzahl der Bits oder Wörter an, auf die diese Anforderung zugreifen soll. Die gültigen Längen finden Sie in der Tabelle der Modbus-Funktionen unten.
DATA_PTR	IN	Variant	Daten-Pointer: Zeigt auf die M- oder DB-Adresse (Standard-DB) für die zu schreibenden oder zu lesenden Daten.
DONE	OUT	Bool	Das Bit DONE ist einen Zyklus lang WAHR, nachdem die letzte Anforderung fehlerfrei ausgeführt wurde.
BUSY	OUT	Bool	• 0 – Keine Anweisung MB_MASTER in Bearbeitung • 1 – Anweisung MB_MASTER in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Parameter STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Ausführungsbedingung

Regeln für die Kommunikation des Modbus-Masters

- MB_COMM_LOAD muss ausgeführt werden, um einen Port zu konfigurieren, damit die Anweisung MB_MASTER mit diesem Port kommunizieren kann.
- Wenn ein Port zum Initiieren von Modbus-Masteranforderungen verwendet werden soll, darf dieser Port nicht von MB_SLAVE verwendet werden. Eine oder mehrere Instanzen von MB_MASTER können mit diesem Port verwendet werden. Doch alle Ausführungen von MB_MASTER müssen den gleichen Instanz-DB MB_MASTER für den Port verwenden.
- Die Modbus-Anweisungen nutzen keine Kommunikations-Alarmereignisse zum Steuern des Kommunikationsprozesses. Ihr Programm muss die Anweisung MB_MASTER auf abgeschlossene Sende- und Empfangsvorgänge abfragen.

- Es wird empfohlen, die Ausführung von MB_MASTER für einen bestimmten Port immer aus einem Programmzyklus-OB aufzurufen. Modbus-Master-Anweisungen können nur in jeweils einem Programmzyklus oder in jeweils einer zyklischen/zeitverzögerten Ausführungsschicht ausgeführt werden. Sie dürfen nicht in beiden Prioritätsschichten der Ausführung bearbeitet werden. Die Vorrangunterbrechung einer Modbus-Master-Anweisung durch eine andere Modbus-Master-Anweisung in einer Ausführungsschicht mit höherer Priorität führt zu nicht ordnungsgemäßem Betrieb. Die Modbus-Master-Anweisungen dürfen nicht in Anlauf-, Diagnose- oder Zeitfehlerausführungsschichten bearbeitet werden.
- Nachdem eine Masteranweisung eine Übertragung ausgelöst hat, muss diese Instanz mit aktiviertem Eingang EN fortlaufend ausgeführt werden, bis der Zustand DONE = 1 oder ERROR = 1 ausgegeben wird. Eine bestimmte MB_MASTER-Instanz wird als aktiv betrachtet, bis eines dieser beiden Ereignisse auftritt. Während die ursprüngliche Instanz aktiv ist, führt jeder Aufruf einer weiteren Instanz mit aktiviertem Eingang REQ zu einem Fehler. Wenn die fortlaufende Ausführung der ursprünglichen Instanz stoppt, bleibt der Anforderungszustand für die von der statischen Variable Blocked_Proc_Timeout angegebene Zeitdauer aktiv. Nach Ablauf dieser Zeitdauer wird die nächste aufgerufene Master-Anweisung mit aktiviertem Eingang REQ zur aktiven Instanz. Dies verhindert, dass eine einzige Modbus-Master-Anweisung ein Monopol inne hat oder den Zugriff auf den Port sperrt. Wenn die ursprüngliche aktive Instanz nicht während des von der statischen Variablen "Blocked_Proc_Timeout" angegebenen Zeitraums aktiviert wird, löscht die nächste Ausführung durch diese Instanz (ohne gesetzten Eingang REQ) den aktiven Zustand. Falls (REQ gesetzt ist), dann löst diese Ausführung eine neue Master-Anforderung aus, als wenn keine andere Instanz aktiv wäre.

Parameter REQ

0 = Keine Anforderung; 1 = Anforderung, Daten an den Modbus-Slave zu senden

Sie können diesen Eingang über einen pegel- oder flankengesteuerten Kontakt steuern. Immer wenn dieser Eingang aktiviert wird, wird eine Zustandsmaschine gestartet, um sicherzustellen, dass eine andere Anweisung MB_MASTER, die denselben Instanz-DB verwendet, erst dann eine Anforderung abgeben kann, wenn die aktuelle Anforderung abgearbeitet ist. Alle anderen Eingangszustände werden erfasst und intern für die aktuelle Anforderung gespeichert, bis die Antwort empfangen oder ein Fehler erkannt wird.

Wird dieselbe Instanz von MB_MASTER erneut mit dem REQ-Eingang = 1 ausgeführt, bevor die aktuelle Anforderung abgearbeitet ist, werden keine nachfolgenden Übertragungen durchgeführt. Wenn die Anforderung jedoch abgearbeitet ist, wird zu dem Zeitpunkt, zu dem MB_MASTER erneut mit REQ-Eingang = 1 ausgeführt wird, eine neue Anforderung abgegeben.

Über die Parameter DATA_ADDR und MODE wählen Sie den Modbus-Funktionstyp aus.

DATA_ADDR (Modbus-Anfangsadresse im Slave): Gibt die Anfangsadresse der Daten an, auf die im Modbus-Slave zugegriffen werden soll.

Die Anweisung MB_MASTER nutzt statt eines Funktionscodeeingangs einen Eingang MODE. Die Kombination aus MODE und Modbus-Adresse legt den Funktionscode fest, der in der eigentlichen Modbus-Meldung verwendet wird. Die folgende Tabelle zeigt die Entsprechung zwischen dem Parameter MODE, dem Modbus-Funktionscode und dem Modbus-Adressbereich.

Tabelle 13- 146 Modbus-Funktionen

MODE	Modbus-Funktion	Länge der Daten	Operation und Daten	Modbus-Adresse
0	01	1 bis 2000 1 bis 1992 [1]	Ausgangsbits lesen: 1 bis (1992 oder 2000) Bit pro Anforderung	1 bis 9999
0	02	1 bis 2000 1 bis 1992 [1]	Eingangsbits lesen: 1 bis (1992 oder 2000) Bit pro Anforderung	10001 bis 19999
0	03	1 bis 125 1 bis 124 [1]	Halteregister lesen: 1 bis (124 oder 125) Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535
0	04	1 bis 125 1 bis 124 [1]	Eingangswörter lesen: 1 bis (124 oder 125) Bits pro Anforderung	30001 bis 39999
1	05	1	Ein Ausgangsbit schreiben: Ein Bit pro Anforderung	1 bis 9999
1	06	1	Ein Halteregister schreiben: 1 Wort pro Anforderung	40001 bis 49999 oder 400001 bis 465535
1	15	2 bis 1968 2 bis 1960 [1]	Mehrere Ausgangsbits schreiben: 2 bis (1960 oder 1968) Bit pro Anforderung	1 bis 9999
1	16	2 bis 123 2 bis 122 [1]	Mehrere Halteregister schreiben: 2 bis (122 oder 123) Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535
2	15	1 bis 1968 2 bis 1960 [1]	Ein oder mehrere Ausgangsbits schreiben: 1 bis (1960 oder 1968) Bit pro Anforderung	1 bis 9999
2	16	1 bis 123 1 bis 122 [1]	Ein oder mehrere Halteregister schreiben: 1 bis (122 oder 123) Wörter pro Anforderung	40001 bis 49999 oder 400001 bis 465535
11	11	0	Statuswort und Ereigniszähler der Slave-Kommunikation lesen. Das Statuswort zeigt beschäftigt an (0 – nicht beschäftigt, 0xFFFF - beschäftigt). Der Ereigniszähler wird bei jeder erfolgreichen Abarbeitung einer Meldung inkrementiert. Beide Operanden DATA_ADDR und DATA_LEN von MB_MASTER werden bei dieser Funktion ignoriert.	
80	08	1	Slave-Zustand über Daten-Diagnosecode 0x0000 prüfen (Prüfschleifentest – Slave gibt ein Echo der Anforderung zurück) 1 Wort pro Anforderung	

MODE	Modbus-Funktion	Länge der Daten	Operation und Daten	Modbus-Adresse
81	08	1	Slave-Ereigniszähler über Daten-Diagnosecode 0x000A zurücksetzen 1 Wort pro Anforderung	
3 bis 10, 12 bis 79, 82 bis 255			Reserviert	

[1] Bei der erweiterten Adressierung ist die maximale Datenlänge je nach Datentyp der Funktion um 1 Byte oder 1 Wort geringer.

Parameter DATA_PTR

Der Parameter DATA_PTR zeigt auf die DB- oder M-Adresse, in die geschrieben oder aus der gelesen wird. Wenn Sie einen Datenbaustein verwenden, müssen Sie einen globalen Datenbaustein anlegen, der den Datenspeicher für Lese- und Schreibvorgänge auf Modbus-Slaves bereitstellt.

Hinweis

Der Datenbaustein DATA_PTR muss die direkte Adressierung zulassen

Der Datenbaustein muss die direkte (absolute) und die symbolische Adressierung zulassen. Wenn Sie den Datenbaustein anlegen, müssen Sie das Zugriffsattribut "Standard" auswählen.

Datenbausteinstrukturen für den Parameter DATA_PTR

- Diese Datentypen gelten für das **Lesen von Wörtern** der Modbus-Adressen 30001 bis 39999, 40001 bis 49999 und 400001 bis 465536 sowie für das **Schreiben von Wörtern** in die Modbus-Adressen 40001 bis 49999 und 400001 bis 465536.
  - Standard-Array der Datentypen WORD, UINT oder INT
  - Benannte Struktur vom Typ WORD, UINT oder INT, bei der jedes Element einen eindeutigen Namen und einen 16-Bit-Datentyp hat.
  - Benannte komplexe Struktur, bei der jedes Element einen eindeutigen Namen und einen 16- oder 32-Bit-Datentyp hat.
- Zum **Lesen** und Schreiben von Bits für die Modbus-Adressen 00001 bis 09999 und zum Lesen von Bits von 10001 bis 19999.
  - Standardfeld aus Booleschen Datentypen.
  - Benannte Boolesche Struktur aus eindeutig benannten Booleschen Variablen.

- Es ist zwar nicht erforderlich, doch empfehlenswert, dass jede Anweisung MB_MASTER einen eigenen getrennten Speicherbereich hat. Grund hierfür ist, dass die Möglichkeit der Datenzerstörung größer ist, wenn mehrere Anweisungen MB_MASTER in demselben Speicherbereich lesen und schreiben.
- Es ist nicht erforderlich, dass die Datenbereiche für DATA_PTR in demselben globalen Datenbaustein liegen. Sie können einen Datenbaustein mit mehreren Bereichen für Modbus-Lesevorgänge, einen Datenbaustein für Modbus-Schreibvorgänge oder einen Datenbaustein für jede Slave-Station anlegen.

Variablen im Datenbaustein des Modbus-Masters

Die folgende Tabelle zeigt die öffentlichen statischen Variablen im Instanz-DB von MB_MASTER, die in Ihrem Programm verwendet werden können.

Tabelle 13- 147 Statische Variablen im Instanz-DB

Variable	Datentyp	Anfangswert	Beschreibung
Blocked_Proc_Timeout	Real	3.0	Zeitdauer (in Sekunden), die auf eine blockierte Modbus-Master-Instanz gewartet werden soll, bevor diese Instanz als AKTIV entfernt wird. Dies kann beispielsweise vorkommen, wenn eine Master-Anforderung ausgegeben wurde und das Programm dann aufhört, die Masterfunktion aufzurufen, bevor es die Anforderung vollständig beendet hat. Der Zeitwert muss größer als 0 und kleiner als 55 Sekunden sein, ansonsten tritt ein Fehler auf. Der Standardwert ist 0,5 Sekunden.
Extended_Addressing	Bool	Falsch	Konfiguriert die Slave-Adressierung als einzelnes oder doppeltes Byte. Der Standardwert ist 0. (0 = Ein-Byte-Adresse, 1 = Zwei-Byte-Adresse)

Ihr Programm kann in die Variablen Blocked_Proc_Timeout und Extended_Addressing Werte schreiben, um die Modbus-Master-Operationen zu steuern. Im Abschnitt zur Anweisung MB_SLAVE finden Sie in der Beschreibung der Variablen HR_Start_Offset und Extended_Addressing ein Beispiel dafür, wie Sie diese Variablen im Programmiereditor nutzen, sowie weitere Einzelheiten zu erweiterten Modbus-Adressierung (Seite 1084).

Bedingungscodes

Tabelle 13- 148 Bedingungscodes für die Ausführung von MB_MASTER (Kommunikations- und Konfigurationsfehler) [1]

STATUS (W#16#)	Beschreibung
0000	Kein Fehler
80C8	Slave-Timeout. Prüfen Sie Baudrate, Parität und Verdrahtung des Slave.
80D1	Der Empfänger hat eine Flusskontrolle zur Unterbrechung einer laufenden Sendung angefordert, aber die Sendung innnerhalb der Wartezeit nicht wieder aktiviert. Dieser Fehler wird auch bei der Hardware-Flusskontrolle erzeugt, wenn der Empfänger innerhalb der Wartezeit kein CTS erkennt.
80D2	Die Sendeanforderung wurde abgebrochen, weil vom DCE kein DSR-Signal empfangen wird.
80E0	Die Meldung wurde beendet, weil der Empfangspuffer voll ist.
80E1	Die Meldung wurde wegen eines Paritätsfehlers beendet.

STATUS (W#16#)	Beschreibung
80E2	Die Meldung wurde wegen eines Telegrammfehlers beendet.
80E3	Die Meldung wurde wegen eines Überlauffehlers beendet.
80E4	Die Meldung wurde abgebrochen, weil die angegebene Länge die Gesamtpuffergröße überschreitet.
8180	Ungültiger Wert für die Port-ID oder Fehler bei der Anweisung MB_COMM_LOAD
8186	Ungültige Modbus-Stationsadresse
8188	Ungültiger Modus für die Broadcast-Anforderung angegeben
8189	Ungültiger Wert für die Datenadresse.
818A	Ungültiger Wert für die Datenlänge.
818B	Ungültiger Pointer auf die lokale Datenquelle bzw. das lokale Datenziel: Größe nicht richtig
818C	Ungültiger Pointer für DATA_PTR oder Blocked_Proc_Timeout ungültig: Bei dem Datenbereich muss es sich um einen DB (der die symbolische und direkte Adressierung gestattet) oder um den Speicherbereich der Merker handeln.
8200	Port ist durch Bearbeitung einer Sendeanforderung belegt.

Tabelle 13- 149 Bedingungscodes für die Ausführung von MB_MASTER (Modbus-Protokollfehler) [1]

STATUS (W#16#)	Antwortcode vom Slave	Modbus-Protokollfehler
8380	-	CRC-Fehler
8381	01	Funktionscode nicht unterstützt
8382	03	Fehler in der Datenlänge
8383	02	Datenadressfehler oder Adresse außerhalb des gültigen Bereichs von DATA_PTR
8384	Größer als 03	Fehler im Datenwert
8385	03	Wert des Datendiagnosecodes wird nicht unterstützt (Funktionscode 08)
8386	-	Funktionscode in der Antwort entspricht nicht dem Code in der Anforderung.
8387	-	Der falsche Slave hat reagiert.
8388	-	Die Slave-Antwort auf eine Schreibanforderung ist inkorrekt. Die vom Slave ausgegebene Schreibanforderung entspricht nicht dem, was der Master gesendet hat.

[1] Neben den oben aufgeführten Fehlern der Anweisung MB_MASTER können auch die zugrunde liegenden PtP-Kommunikationsanweisungen Fehler zurückgeben.

13.9.3.3 Anweisung MB_SLAVE (Über PtP-Port als Modbus RTU-Slave kommunizieren)

Tabelle 13- 150 Anweisung MB_SLAVE

KOP/FUP	SCL	Beschreibung

	`"MB_SLAVE_DB"(`
	`MB_ADDR:=_uint_in_,`
	`NDR=>_bool_out_,`
	`DR=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,`
	`MB_HOLD_REG:=_variant_inout_);`
		Mit der Anweisung MB_SLAVE kann Ihr Programm über einen PtP-Port eines CM (RS485 oder RS232) oder eines CB (RS485) als Modbus-Slave kommunizieren. Wenn ein dezentraler Modbus-RTU-Master eine Anforderung ausgibt, reagiert Ihr Anwenderprogramm auf die Anforderung, indem es die Anweisung MB_SLAVE ausführt. STEP 7 erstellt automatisch einen Instanz-DB, wenn Sie die Anweisung einfügen. Sie geben mit diesem MB_SLAVE_DB-Namen den Parameter MB_DB für die Anweisung MB_COMM_LOAD an.

Tabelle 13- 151 Datentypen für die Parameter

Parameter und Datentyp		Datentyp	Beschreibung
MB_ADDR	IN	V1.0: USInt V2.0: UInt	Stationsadresse des Modbus-Slaves: Standard-Adressierungsbereich (1 bis 247) Erweiterter Adressierungsbereich (0 bis 65535)
MB_HOLD_REG	IN	Variant	Pointer auf den Modbus-Halteregister-DB: Bei dem Modbus-Halteregister kann es sich um den Speicherbereich der Merker oder um einen Datenbaustein handeln.
NDR	OUT	Bool	Neue Daten bereit: • 0 – Keine neuen Daten • 1 – Gibt an, dass neue Daten vom Modbus-Master geschrieben wurden
DR	OUT	Bool	Daten lesen: • 0 – Keine Daten gelesen • 1 – Gibt an, dass Daten vom Modbus-Master gelesen wurden
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang WAHR, nachdem die letzte Anforderung mit Fehler beendet wurde. Wenn die Ausführung mit einem Fehler beendet wird, ist der Fehlercode im Parameter STATUS nur in dem einen Zyklus gültig, in dem ERROR = WAHR ist.
STATUS	OUT	Word	Fehlercode der Ausführung

Die Funktionscodes der Modbus-Kommunikation (1, 2, 4, 5 und 15) können Bits und Wörter direkt im Prozessabbild der Eingänge und im Prozessabbild der Ausgänge der CPU lesen und schreiben. Bei diesen Funktionscodes muss der Parameter MB_HOLD_REG als Datentyp größer als ein Byte definiert werden. Die folgende Tabelle zeigt die Beispielzuordnung der Modbus-Adressen zum Prozessabbild in der CPU.

Tabelle 13- 152 Zuordnung der Modbus-Adressen zum Prozessabbild

Modbus-Funktionen						S7-1200	
Codes	Funktion	Datenbereich	Adressbereich			Datenbereich	CPU-Adresse
01	Bits lesen	Ausgang	1	bis	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7
02	Bits lesen	Eingang	10001	bis	18192	Prozessabbild der Eingänge	E0.0 bis E1023.7
04	Wörter lesen	Eingang	30001	bis	30512	Prozessabbild der Eingänge	EW0 bis EW1022
05	Bit schreiben	Ausgang	1	bis	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7
15	Bits schreiben	Ausgang	1	bis	8192	Prozessabbild der Ausgänge	A0.0 bis A1023.7

Die Funktionscodes der Modbus-Kommunikation (3, 6, 16) nutzen ein Modbus-Halteregister, bei dem es sich um einen Adressbereich im Speicherbereich der Merker oder um einen Datenbaustein handeln kann. Der Typ des Haltesregisters wird vom Parameter MB_HOLD_REG der Anweisung MB_SLAVE angegeben.

Hinweis

Typ des Datenbausteins MB_HOLD_REG

Ein Datenbaustein mit Modbus-Halteregister muss die direkte (absolute) und die symbolische Adressierung zulassen. Wenn Sie den Datenbaustein anlegen, müssen Sie das Zugriffsattribut "Standard" auswählen.

Die folgende Tabelle zeigt Beispiele für die Zuordnung von Modbus-Adressen zum Halteregister, das für die Modbus-Funktionscodes 03 (Wörter lesen), 06 (Wort schreiben) und 16 (Wörter schreiben) verwendet wird. Die tatsächliche obere Grenze der DB-Adressen wird vom maximalen Arbeitsspeicher und vom maximalen M-Speicher des jeweiligen CPU-Modells festgelegt.

Tabelle 13- 153 Zuordnung von Modbus-Adressen zum CPU-Speicher

Modbus-Master-Adresse	Beispiele für Parameter von MB_HOLD_REG				
	MW100	DB10.DBw0	MW120	DB10.DBW50	"Rezept".Inhaltsstoff
40001	MW100	DB10.DBW0	MW120	DB10.DBW50	"Rezept".Inhaltsstoff[1]
40002	MW102	DB10.DBW2	MW122	DB10.DBW52	"Rezept".Inhaltsstoff[2]
40003	MW104	DB10.DBW4	MW124	DB10.DBW54	"Rezept".Inhaltsstoff[3]
40004	MW106	DB10.DBW6	MW126	DB10.DBW56	"Rezept".Inhaltsstoff[4]
40005	MW108	DB10.DBW8	MW128	DB10.DBW58	"Rezept".Inhaltsstoff[5]

Tabelle 13- 154 Diagnosefunktionen

Modbus-Diagnosefunktionen von MB_SLAVE der S7-1200		
Codes	Teilfunktion	Beschreibung
08	0000H	Echotest Abfragedaten ausgeben: Die Anweisung MB_SLAVE gibt einem Modbus-Master das Echo eines empfangenen Datenworts zurück.
08	000AH	Kommunikationsereigniszähler löschen: Die Anweisung MB_SLAVE löscht den Kommunikationsereigniszähler, der für Modbus-Funktion 11 verwendet wird.
11		Kommunikationsereigniszähler abrufen: Die Anweisung MB_SLAVE nutzt einen internen Kommunikationsereigniszähler, um die Anzahl erfolgreicher Modbus-Lese- und Modbus-Schreibanforderungen, die an den Modbus-Slave gesendet werden, zu erfassen. Der Zähler wird bei Funktion 8, Funktion 11 und Broadcast-Anforderungen nicht hochgezählt. Außerdem wird bei Anforderungen, die zu Kommunikationsfehlern führen (z.B. Paritäts- oder CRC-Fehler), nicht hochgezählt.

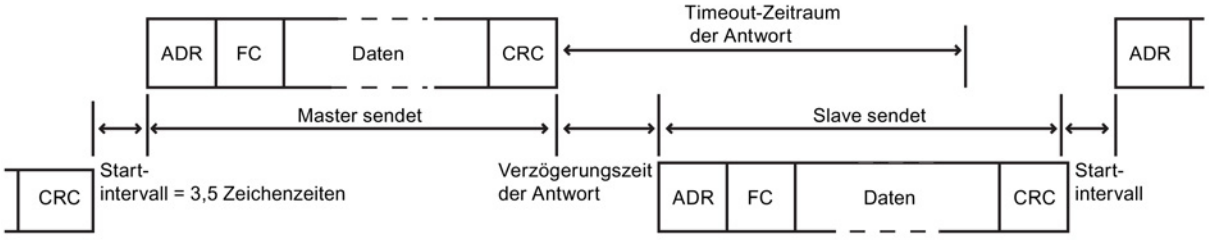
Die Anweisung MB_SLAVE unterstützt Broadcast-Schreibanforderungen von Modbus-Mastern, solange die Anforderungen den Zugriff auf gültige Adressen umfassen. Für Funktionscodes, die bei der Broadcast-Funktion nicht unterstützt werden, erzeugt MB_SLAVE den Fehlercode 0x8188.

Regeln für die Modbus-Slave-Kommunikation

- MB_COMM_LOAD muss ausgeführt werden, um einen Port zu konfigurieren, damit die Anweisung MB_SLAVE über diesen Port kommunizieren kann.
- Wenn ein Port als Slave auf einen Modbus-Master reagieren soll, darf dieser Port nicht mit der Anweisung MB_MASTER programmiert werden.
- Nur eine Instanz von MB_SLAVE kann mit einem bestimmten Port verwendet werden, andernfalls kann unerwartetes Verhalten auftreten.
- Die Modbus-Anweisungen nutzen keine Kommunikations-Alarmereignisse zum Steuern des Kommunikationsprozesses. Ihr Programm muss den Kommunikationsprozess durch Abfrage der Anweisung MB_SLAVE auf abgeschlossene Sende- und Empfangsvorgänge steuern.
- Die Anweisung MB_SLAVE muss regelmäßig mit einer Häufigkeit ausgeführt werden, die eine zeitnahe Antwort auf eingehende Anforderungen eines Modbus-Masters ermöglicht. Es wird empfohlen, MB_SLAVE in jedem Zyklus aus einem Programmzyklus-OB auszuführen. MB_SLAVE kann aus einem Weckalarm-OB ausgeführt werden, doch ist dies nicht empfehlenswert, weil übermäßige Zeitverzögerungen im Interruptprogramm die Ausführung anderer Interruptprogramme temporär blockieren können.

Zeitsteuerung des Modbus-Signals

MB_SLAVE muss regelmäßig ausgeführt werden, um jede Anforderung des Modbus-Masters zu empfangen und entsprechend zu antworten. Die Häufigkeit der Ausführung von MB_SLAVE richtet sich nach dem vom Modbus-Master vorgegebenen Timeout-Wert für die Antwort. Dies ist in der nachstehenden Abbildung dargestellt.



Der Timeout-Zeitraum für die Antwort RESP_TO ist die Zeitdauer, die ein Modbus-Master auf den Beginn einer Antwort von einem Modbus-Slave wartet. Dieser Zeitraum wird nicht vom Modbus-Protokoll definiert, sondern von einem Parameter des jeweiligen Modbus-Masters. Die Häufigkeit der Ausführung (die Zeit zwischen einer Ausführung und der nächsten Ausführung) von MB_SLAVE muss sich nach den jeweiligen Parametern des Modbus-Masters richten. Sie sollten MB_SLAVE mindestens zweimal während des Timeout-Zeitraums für die Antwort des Modbus-Masters ausführen.

Variablen des Modbus-Slave

Diese Tabelle zeigt die öffentlichen statischen Variablen im Instanz-Datenbaustein von MB_SLAVE, die in Ihrem Programm verwendet werden können.

Tabelle 13- 155 Variablen des Modbus-Slave

Variable	Datentyp	Beschreibung
Request_Count	Word	Die Anzahl aller von diesem Slave empfangenen Anforderungen
Slave_Message_Count	Word	Die Anzahl der für diesen spezifischen Slave empfangenen Anforderungen
Bad_CRC_Count	Word	Die Anzahl der empfangenen Anforderungen, die einen CRC-Fehler aufweisen
Broadcast_Count	Word	Die Anzahl der empfangenen Broadcast-Anforderungen
Exception_Count	Word	Modbus-spezifische Fehler, die eine zurückgegebene Ausnahme erfordern
Success_Count	Word	Die Anzahl der für diesen spezifischen Slave empfangenen Anforderungen ohne Protokollfehler
HR_Start_Offset	Word	Gibt die Anfangsadresse des Modbus-Halteregisters an (Standard = 0)
Extended_Addressing	Bool	Konfiguriert die Slave-Adressierung als einzelnes oder doppeltes Byte (0 = einzelne Byte-Adresse, 1 = doppelte Byte-Adresse, Standard = 0)

Ihr Programm kann in die Variablen HR_Start_Offset und Extended_Addressing Werte schreiben und die Modbus-Slave-Operationen steuern. Die anderen Variablen können gelesen werden, um den Modbus-Zustand zu überwachen.

HR_Start_Offset

Die Adressen des Modbus-Halteregisters beginnen bei 40001 oder 400001. Diese Adressen entsprechen der Anfangsadresse des Halteregisters im Zielsystemspeicher. Sie können jedoch die Variable HR_Start_Offset konfigurieren, um eine andere Anfangsadresse als 40001 oder 400001 für das Modbus-Halteregister zu konfigurieren.

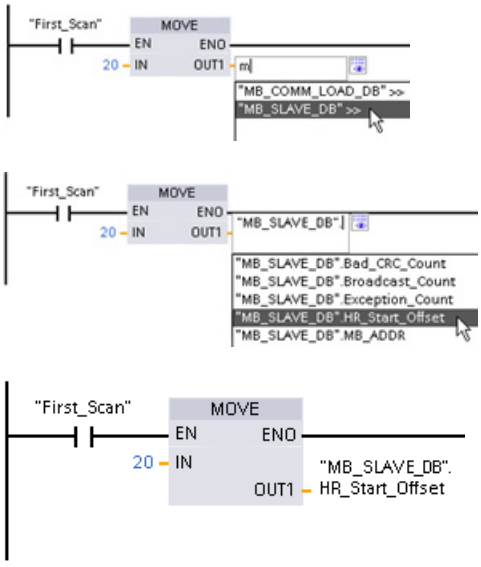
Sie können z. B. ein Halteregister mit Beginn an MW100 und einer Länge von 100 Wörtern konfigurieren. Mit einem Versatz von 20 geben Sie eine Anfangsadresse des Halteregisters von 40021 statt 40001 an. Jede Adresse unter 40021 und über 400119 führt zu einem Adressierungsfehler.

Tabelle 13- 156 Beispiel für die Adressierung des Modbus-Halteregisters

HR_Start_Offset	Adresse	Minimum	Maximum
0	Modbus-Adresse (Wort)	40001	40099
	S7-1200 Adresse	MW100	MW298
20	Modbus-Adresse (Wort)	40021	40119
	S7-1200 Adresse	MW100	MW298

HR_Start_Offset ist ein Wortwert, der die Anfangsadresse des Modbus-Halteregisters angibt und im Instanz-Datenbaustein MB_SLAVE gespeichert ist. Sie können diese öffentliche statische Variable über die Parameter-Klappliste auswählen, nachdem Sie MB_SLAVE in Ihr Programm eingefügt haben.

Wenn Sie beispielsweise MB_SLAVE in ein KOP-Netzwerk eingefügt haben, können Sie in ein vorheriges Netzwerk gehen und den Wert HR_Start_Offset zuweisen. Der Wert muss vor der Ausführung von MB_SLAVE zugewiesen werden.



Modbus-Slave-Variable über den Standard-DB-Namen eingeben:

1. Positionieren Sie den Cursor im Parameterfeld und geben Sie das Zeichen m ein.
2. Wählen Sie in der Klappliste "MB_SLAVE_DB" aus.
3. Positionieren Sie den Cursor rechts vom DB-Namen (nach den Anführungszeichen) und geben Sie einen Punkt ein.
4. Wählen Sie in der Klappliste "MB_SLAVE_DB.HR_Start_Offset" aus.

Extended_Addressing

Auf die Extended_Addressing-Variable wird in ähnlicher Weise zugegriffen wie auf die oben beschriebene HR_Start_Offset-Referenz, außer dass es sich bei der Extended_Addressing-Variable um einen Booleschen Wert handelt. Der Boolesche Wert muss von einer Ausgangsspule und kann nicht von einer Box "Verschieben" geschrieben werden.

Für die Modbus-Slave-Adressierung kann ein einzelnes Byte (dies ist der Modbus-Standard) oder ein doppeltes Byte konfiguriert werden. Die erweiterte Adressierung dient zur Adressierung von mehr als 247 Geräten in einem einzigen Netzwerk. Wenn Sie sich für die erweiterte Adressierung entscheiden, können Sie maximal 64.000 Adressen ansprechen. Im Folgenden wird ein Telegramm der Modbus-Funktion 1 als Beispiel gezeigt.

Tabelle 13- 157 Slave-Adresse mit einem Byte (Byte 0)

Funktion 1	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	
Anforderung	Slave-Adresse	F-Code	Anfangsadresse		Länge der Spulen		
Gültige Antwort	Slave-Adresse	F-Code	Länge	Spulendaten			
Fehlerantwort	Slave-Adresse	0x81	E-Code				

Tabelle 13- 158 Slave-Adresse mit zwei Bytes (Byte 0 und Byte 1)

	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6
Anforderung	Slave-Adresse		F-Code	Anfangsadresse		Länge der Spulen	
Gültige Antwort	Slave-Adresse		F-Code	Länge	Spulendaten		
Fehlerantwort	Slave-Adresse		0x81	E-Code			

Bedingungscodes

Tabelle 13- 159 Bedingungscodes für die Ausführung von MB_SLAVE (Kommunikations- und Konfigurationsfehler) [1]

STATUS (W#16#)	Beschreibung
80D1	Der Empfänger hat eine Flusskontrolle zur Unterbrechung einer laufenden Sendung angefordert, aber die Sendung innnerhalb der Wartezeit nicht wieder aktiviert. Dieser Fehler wird auch bei der Hardware-Flusskontrolle erzeugt, wenn der Empfänger innerhalb der Wartezeit kein CTS erkennt.
80D2	Die Sendeanforderung wurde abgebrochen, weil vom DCE kein DSR-Signal empfangen wird.
80E0	Die Meldung wurde beendet, weil der Empfangspuffer voll ist.
80E1	Die Meldung wurde wegen eines Paritätsfehlers beendet.
80E2	Die Meldung wurde wegen eines Telegrammfehlers beendet.
80E3	Die Meldung wurde wegen eines Überlauffehlers beendet.
80E4	Die Meldung wurde abgebrochen, weil die angegebene Länge die Gesamtpuffergröße überschreitet.
8180	Ungültiger Wert für die Port-ID oder Fehler bei der Anweisung MB_COMM_LOAD
8186	Ungültige Modbus-Stationsadresse
8187	Ungültiger Pointer auf MB_HOLD_REG-DB: Bereich ist zu klein
818C	Ungültiger Pointer MB_HOLD_REG auf M-Speicher oder DB (DB-Bereich muss symbolische und direkte Adressen zulassen)

Tabelle 13- 160 Bedingungscodes für die Ausführung von MB_SLAVE (Modbus-Protokollfehler) [1]

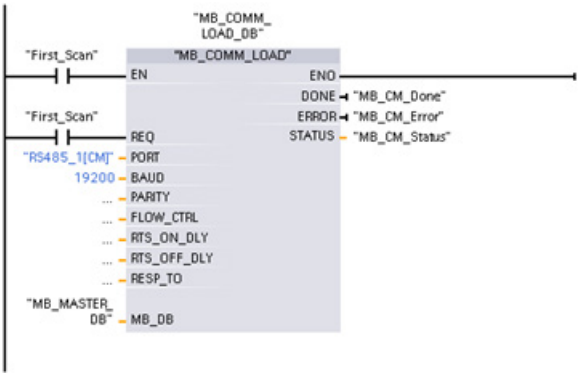
STATUS (W#16#)	Antwortcode vom Slave	Modbus-Protokollfehler
8380	Keine Antwort	CRC-Fehler
8381	01	Funktioncode nicht unterstützt oder nicht unterstützt in Broadcasts
8382	03	Fehler in der Datenlänge
8383	02	Datenadressfehler oder Adresse außerhalb des gültigen Bereichs von DATA_PTR
8384	03	Fehler im Datenwert
8385	03	Wert des Datendiagnosecodes wird nicht unterstützt (Funktionscode 08)

[1] Neben den oben aufgeführten Fehlern der Anweisung MB_SLAVE können auch die zugrunde liegenden PtP-Kommunikationsanweisungen Fehler zurückgeben.

13.9.4 Beispiel: Beispielprogramm für einen Modbus RTU-Master in älteren Systemen

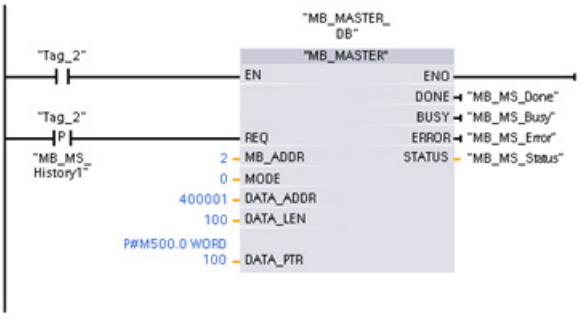
MB_COMM_LOAD wird während des Anlaufs über den Merker des ersten Zyklus initialisiert. Die Ausführung von MB_COMM_LOAD auf diese Weise darf nur durchgeführt werden, wenn sich die Konfiguration des seriellen Ports während der Laufzeit nicht ändert.

Netzwerk 1: Parameter des RS485-Moduls nur einmal während des ersten Zyklus initialisieren.

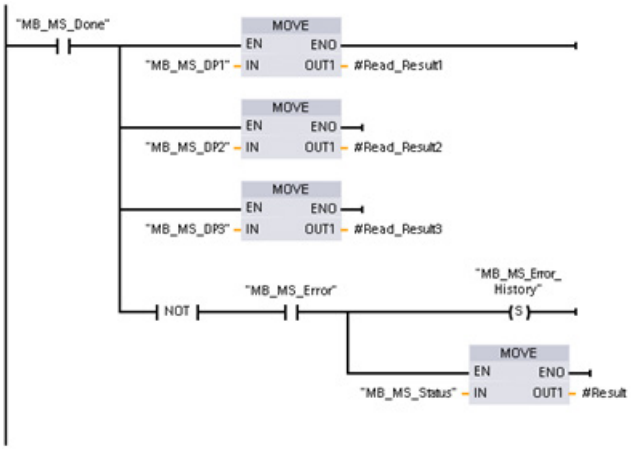


Eine Anweisung MB_MASTER dient im Programmzyklus-OB zur Kommunikation mit einem Slave. Weitere Anweisungen MB_MASTER können im Programmzyklus-OB für die Kommunikation mit anderen Slaves genutzt werden, oder es kann ein MB_MASTER-FB wiederverwendet werden, um mit weiteren Slaves zu kommunizieren.

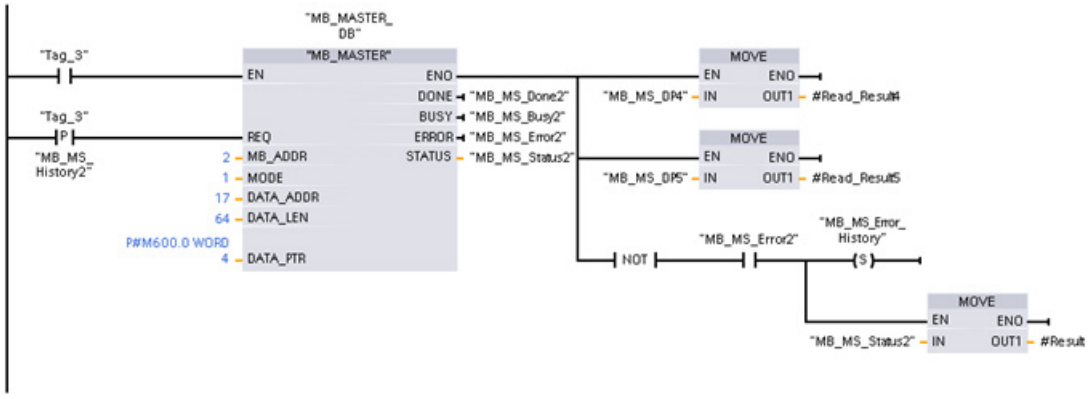
Netzwerk 2: 100 Wörter aus dem Halteregister des Slave lesen.



Netzwerk 3: Dies ist ein optionales Netzwerk, das die Werte der ersten 3 Wörter anzeigt, sobald die Leseoperation durchgeführt wurde.



Netzwerk 4: 64 Bits in das Prozessabbild der Ausgänge schreiben, Beginn an Slave-Adresse A2.0.

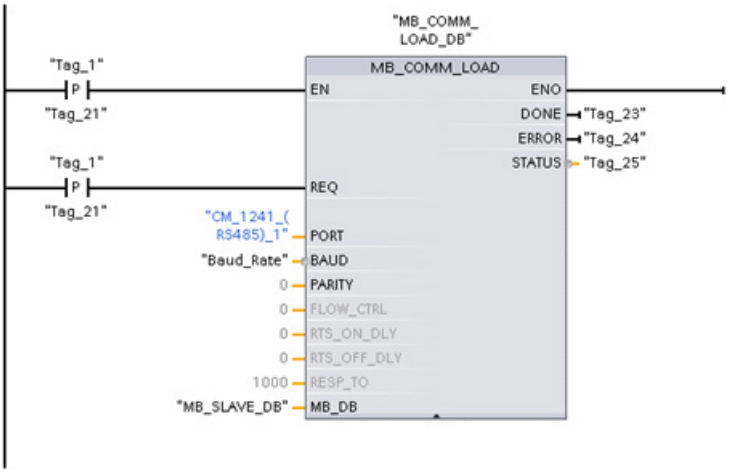


13.9.5 Beispiel: Beispielprogramm für einen Modbus RTU-Slave in älteren Systemen

Die im Folgenden gezeigte Anweisung MB_COMM_LOAD wird bei jeder Aktivierung von "Variable_1" initialisiert.

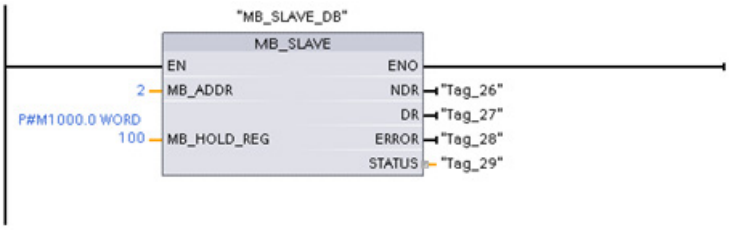
Die Ausführung von MB_COMM_LOAD auf diese Weise darf nur durchgeführt werden, wenn sich die Konfiguration des seriellen Ports als Folge der HMI-Konfiguration während der Laufzeit ändert.

Netzwerk 1: RS485-Modulparameter bei jeder Änderung durch ein HMI-Gerät initialisieren.



Die im Folgenden gezeigte Anweisung MB_SLAVE befindet sich in einem zyklischen OB, der alle 10 ms ausgeführt wird. Dies bietet zwar nicht die absolut schnellste Reaktion durch den Slave, doch es bietet ein gutes Betriebsverhalten bei 9600 Baud für kurze Meldungen (bis zu 20 Byte in der Anforderung).

Netzwerk 2: In jedem Zyklus auf Anforderungen des Modbus-Masters prüfen. Für das Modbus-Halteregister sind 100 Wörter mit Beginn an MW1000 konfiguriert.



13.10 Telecontrol und TeleService mit dem CP 1242-7

13.10.1 Übersicht über Telecontrol-CPs

TeleControl-CPs für die S71200

Für TeleControl-Anwendungen sind die folgenden Kommunikationsprozessoren erhältlich:

- **CP 1243-1**

  Artikelnummer 6GK7 243-1BX30-0XE0

  Kommunikationsprozessor zum Anschließen der SIMATIC S7-1200 über die öffentliche Infrastruktur (z. B. DSL) an eine Leitstelle mit TeleControl Server Basic (TCSB Version V3).

  Mithilfe von VPN-Technologie und der Firewall ermöglicht der CP geschützten Zugriff auf die S7-1200.

  Der CP kann als zusätzliche Ethernet-Schnittstelle der CPU für die S7-Kommunikation verwendet werden.

  Die Kommunikation zwischen CP und CPU erfolgt über konfigurierbare Datenpunkte, die auf PLC-Variablen zugreifen.

- **CP 1243-1 DNP3**

  Artikelnummer 6GK7 243-1JX30-0XE0

  Kommunikationsprozessor zum Anschließen der SIMATIC S7-1200 an Leitstellen mittels DNP3-Protokoll.

  Die Kommunikation zwischen CP und CPU erfolgt über konfigurierbare Datenpunkte, die auf PLC-Variablen zugreifen.

- **CP 1243-1 IEC**

  Artikelnummer 6GK7 243-1PX30-0XE0

  Kommunikationsprozessor zum Anschließen der SIMATIC S7-1200 an Leitstellen mittels IEC 60870-5-Protokoll.

  Die Kommunikation zwischen CP und CPU erfolgt über konfigurierbare Datenpunkte, die auf PLC-Variablen zugreifen.

- **CP 1242-7**

  Artikelnummer 6GK7 242-7KX30-0XE0

  Kommunikationsprozessor zum Anschließen der SIMATIC S7-1200 über Mobilfunknetze (GPRS) und die öffentliche Infrastruktur (DSL) an eine Leitstelle mit TeleControl Server Basic.

- **CP 1242-7 GPRS V2**

  Artikelnummer 6GK7 242-7KX31-0XE0

  Kommunikationsprozessor zum Anschließen der SIMATIC S7-1200 über Mobilfunknetze (GPRS) und die öffentliche Infrastruktur (DSL) an eine Leitstelle mit TeleControl Server Basic (TCSB Version V3).

  Mithilfe von VPN-Technologie und der Firewall ermöglicht der CP geschützten Zugriff auf die S7-1200.

  Der CP kann als zusätzliche Ethernet-Schnittstelle der CPU für die S7-Kommunikation verwendet werden.

  Die Kommunikation zwischen CP und CPU erfolgt über konfigurierbare Datenpunkte, die auf PLC-Variablen zugreifen.

- **CP 1243-7 LTE-xx**

  Kommunikationsprozessor zum Anschließen der SIMATIC S7-1200 über Mobilfunknetze (GPRS) und die öffentliche Infrastruktur (DSL) an eine Leitstelle mit TeleControl Server Basic (TCSB Version V3).

  Unterstützung der folgenden Mobilfunkspezifikationen: GSM/GPRS, UMTS (G3), LTE

  Damit der CP auch in Ländern mit unterschiedlichen Mobilfunkspezifikationen eingesetzt werden kann, steht er in zwei Varianten zur Verfügung:

  – CP 1243-7 LTE-US

    Nordamerikanischer Standard

    Artikelnummer 6GK7 243-7SX30-0XE0

  – CP 1243-7 LTE-EU

    Westeuropäischer Standard

    Artikelnummer 6GK7 243-7KX30-0XE0

  Mithilfe von VPN-Technologie und der Firewall ermöglicht der CP geschützten Zugriff auf die S7-1200.

  Der CP kann als zusätzliche Ethernet-Schnittstelle der CPU für die S7-Kommunikation verwendet werden.

  Die Kommunikation zwischen CP und CPU erfolgt über konfigurierbare Datenpunkte, die auf PLC-Variablen zugreifen.

13.10.2 Anschluss an ein GSM-Netz

IP-basierte WAN-Kommunikation über GPRS

Mit Hilfe des Kommunikationsprozessors CP 1242-7 lässt sich die S7-1200 an GSM-Netze anschließen. Der CP 1242-7 ermöglicht die WAN-Kommunikation von entfernten Stationen mit einer Zentrale und die Kommunikation zwischen Stationen.

Die Kommunikation zwischen Stationen ist nur über das GSM-Netz möglich. Für die Kommunikation einer entfernten Station mit einer zentralen Warte muss in der Zentrale ein PC mit Internet-Anschluss zur Verfügung stehen.

Der CP 1242-7 unterstützt folgende Dienste für die Kommunikation über das GSM-Netz:

- GPRS (General Packet Radio Service)

  Der paketorientierte Dienst der Datenübertragung "GPRS" wird über das GSM-Netz abgewickelt.

- Kurzmitteilungsdienst (SMS, Short Message Service)

  Der CP 1242-7 kann Meldungen als SMS empfangen und versenden. Kommunikationspartner kann ein Mobiltelefon oder eine S7-1200 sein.

Der CP 1242-7 ist weltweit für den industrieller Einsatz geeignet und unterstützt folgende Frequenzbänder:

- 850 MHz
- 900 MHz
- 1.800 MHz
- 1.900 MHz

Voraussetzungen

Die Ausrüstung der Stationen oder der Zentrale hängt vom jeweiligen Anwendungsfall ab.

- Für die Kommunikation mit oder über eine zentrale Warte benötigt die Zentrale einen PC mit Internet-Anschluss.
- Für eine entfernte S7-1200-Station mit CP 1242-7, die die Kommunikation über das GSM-Netz nutzen soll, sind neben der Stationsausrüstung folgende Voraussetzungen erforderlich:
  - Ein Vertrag mit einem geeigneten GSM-Netzbetreiber

    Wenn GPRS genutzt werden soll, dann muss der Vertrag die Nutzung des GPRS-Dienstes ermöglichen.

    Bei direkter Kommunikation zwischen Stationen nur über das GSM-Netz muss der GSM-Netzbetreiber den CPs eine feste IP-Adresse zuweisen. In diesem Fall läuft die Kommunikation zwischen den Stationen nicht über die Zentrale.
  - Die zum Vertrag gehörende SIM-Karte

    Die SIM-Karte wird in den CP 1242-7 gesteckt.
  - Lokale Verfügbarkeit eines GSM-Netzes im Bereich der Station

13.10.3 Anwendungen des CP 1242-7

Folgende Anwendungsfälle sind für den CP 1242-7 möglich:

Telecontrol-Anwendungen

- Versenden von Meldungen per SMS

  Über den CP 1242-7 empfängt die CPU einer entfernten S7-1200-Station SMS-Nachrichten aus dem GSM-Netz oder verschickt Meldungen per SMS an ein projektiertes Mobiltelefon oder eine S7-1200.

- Kommunikation mit einer Zentrale

  Entfernte S7-1200-Stationen kommunizieren über das GSM-Netz und das Internet mit einem Telecontrol-Server in der Zentrale. Für die Datenübertragung per GPRS ist auf dem Telecontrol-Server in der Zentrale die Anwendung "TELECONTROL SERVER BASIC" installiert. Über die integrierte OPC-Server-Funktion kommuniziert der Telecontrol-Server mit einem übergeordneten zentralen Leitsystem.

- Kommunikation zwischen S7-1200-Stationen über ein GSM-Netz

  Die Kommunikation zwischen entfernten Stationen und einem CP 1242-7 kann auf zwei unterschiedliche Arten ablaufen:

  – Kommunikation zwischen Stationen über eine Zentrale

    In dieser Konfiguration wird eine permanente gesicherte Verbindung zwischen miteinander kommunizierenden S7-1200-Stationen und dem Telecontrol-Server in der Zentrale aufgebaut. Die Kommunikation zwischen den Stationen läuft über den Telecontrol-Server. Der CP 1242-7 arbeitet in Betriebsart "Telecontrol".

  – Direkte Kommunikation zwischen den Stationen

    Für die direkte Kommunikation zwischen Stationen ohne den Umweg über eine Zentrale werden SIM-Karten mit fester IP-Adresse eingesetzt, die es ermöglichen, die Stationen direkt anzusprechen. Die möglichen Kommunikationsdienste und Sicherheitsfunktionen (z. B. VPN) hängen dabei vom Angebot des Netzwerkbetreibers ab. Der CP 1242-7 arbeitet in Betriebsart "GPRS direkt".

TeleService über GPRS

Zwischen einer Engineering-Station mit STEP 7 und einer entfernten S7-1200-Station mit CP 1242-7 kann eine TeleService-Verbindung über das GSM-Netz und das Internet aufgebaut werden. Die Verbindung läuft von der Engineering-Station über einen Telecontrol-Server oder über ein TeleService-Gateway, die als Vermittler die Telegramme weiterleiten und die Autorisierung durchführen. Diese PCs nutzen die Funktionen der Anwendung TELECONTROL SERVER BASIC.

Die TeleService-Verbindung können Sie für folgende Zwecke nutzen:

- Laden von Projektierungs- oder Programmdaten aus dem STEP 7-Projekt in die Station
- Abfragen von Diagnosedaten aus der Station

13.10.4 Weitere Eigenschaften des CP-1242-7

Weitere Dienste und Funktionen des CP 1242-7

- Uhrzeitsynchronisation des CP über Internet

  Die Uhrzeit des CP können Sie folgendermaßen einstellen:

  – In der Betriebsart "Telecontrol" wird die Uhrzeit vom Telecontrol-Server übertragen. Der CP stellt damit seine Uhrzeit ein.

  – In der Betriebsart "GPRS direkt" kann der CP die Uhrzeit über SNTP anfordern.

  Zur Synchronisation der CPU-Uhrzeit können Sie die aktuelle Uhrzeit mithilfe eines Bausteins aus dem CP auslesen.
- Zwischenspeicherung der zu sendenden Telegramme bei Verbindungsproblemen
- Erhöhte Verfügbarkeit durch die Möglichkeit zum Verbinden mit einem Ersatz-Telecontrol-Server
- Optimiertes Datenvolumen (temporäre Verbindung)

  Alternativ zu einer permanenten Verbindung zum Telecontrol-Server kann der CP in STEP 7 mit einer temporären Verbindung zum Telecontrol-Server projektiert werden. In diesem Fall wird eine Verbindung zum Telecontrol-Server nur bei Bedarf aufgebaut.
- Protokollierung des Datenvolumens

  Die übertragenen Datenmengen werden protokolliert und können zu weiteren Zwecken ausgewertet werden.

13.10.5 Projektierung und elektrische Anschlüsse

Projektierung und Baugruppentausch

Für die Projektierung der Baugruppe ist folgendes Projektierungswerkzeug erforderlich:

STEP 7 Version V11.0 SP1 oder höher

Für STEP 7 V11.0 SP1 benötigen Sie zusätzlich das Support Package "CP 1242-7" (HSP0003001).

Für die Prozessdatenübertragung per GPRS verwenden Sie im Anwenderprogramm der Station die Telecontrol-Kommunikationsanweisungen.

Die Projektierungsdaten des CP 1242-7 werden auf der jeweils lokalen CPU gespeichert. Dadurch ist bei Bedarf der einfache Austausch dieser Baugruppe möglich.

Sie können bis zu drei Baugruppen des Typs CP 1242-7 pro S7-1200 stecken. Damit lassen sich beispielsweise redundante Kommunikationspfade aufbauen.

Elektrische Anschlüsse

- Spannungsversorgung des CP 1242-7

  Der CP hat einen eigenen Anschluss für die externe Spannungsversorgung 24 V DC.

- Funkschnittstelle für das GSM-Netz

  Für die GSM-Kommunikation ist eine externe Antenne erforderlich. Diese wird über die SMA-Buchse des CP angeschlossen.

13.10.6 Weitere Informationen

Weitere Informationen

Detaillierte Informationen enthält das Handbuch des CP 1242-7. Dieses finden Sie im Internet auf den Seiten des Siemens Industrial Automation Customer Support unter der folgenden Beitrags-ID:

45605894 (http://support.automation.siemens.com/WW/view/de/45605894)

13.10.7 Zubehör

GSM/GPRS-Antenne ANT794-4MR

Für den Einsatz in GSM/GPRS-Netzen stehen folgende Antennen zur Montage im Innen- oder Außenbereich zur Verfügung:

- Quadband-Antenne ANT794-4MR

Kurzbezeichnung	Bestell-Nr.	Erläuterung
ANT794-4MR	6NH9 860-1AA00	Quadband-Antenne (900, 1800/1900 MHz, UMTS); witterungsbeständig für Innen- und Außenbereich; 5-m-Anschlusskabel fest mit der Antenne verbunden; SMA-Stecker; inkl. Montagewinkel, Schrauben, Dübel

- Flachantenne ANT794-3M

Kurzbezeichnung	Bestell-Nr.	Erläuterung
ANT794-3M	6NH9 870-1AA00	Flachantenne (900, 1800/1900 MHz); witterungsbeständig für Innen- und Außenbereich; 1,2-m-Anschlusskabel fest mit der Antenne verbunden; SMA-Stecker; inkl. Klebepad, Schraubbefestigung möglich

Die Antennen sind separat zu bestellen.

13.10.8 Verweis auf Handbuch zur GSM-Antenne

Weitere Informationen

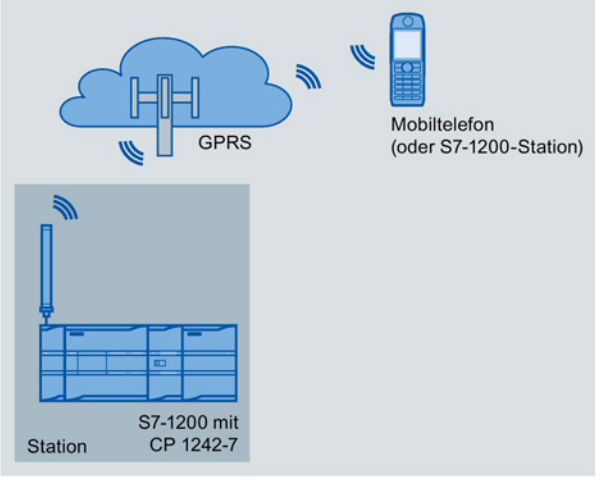
Ausführliche Informationen sind im Gerätehandbuch zu finden. Dieses finden Sie im Internet auf den Seiten des Siemens Industrial Automation Customer Support unter der folgenden Beitrags-ID:

23119005 (http://support.automation.siemens.com/WW/view/de/23119005)

13.10.9 Konfigurationsbeispiele für Telecontrol

Im Folgenden finden Sie einige Konfigurationsbeispiele für Stationen mit CP 1242-7.

Versenden von Meldungen per SMS



Eine SIMATIC S7-1200 mit CP 1242-7 kann Meldungen per SMS an ein Mobiltelefon oder eine projektierte S7-1200-Station versenden.

Telecontrol durch eine Zentrale

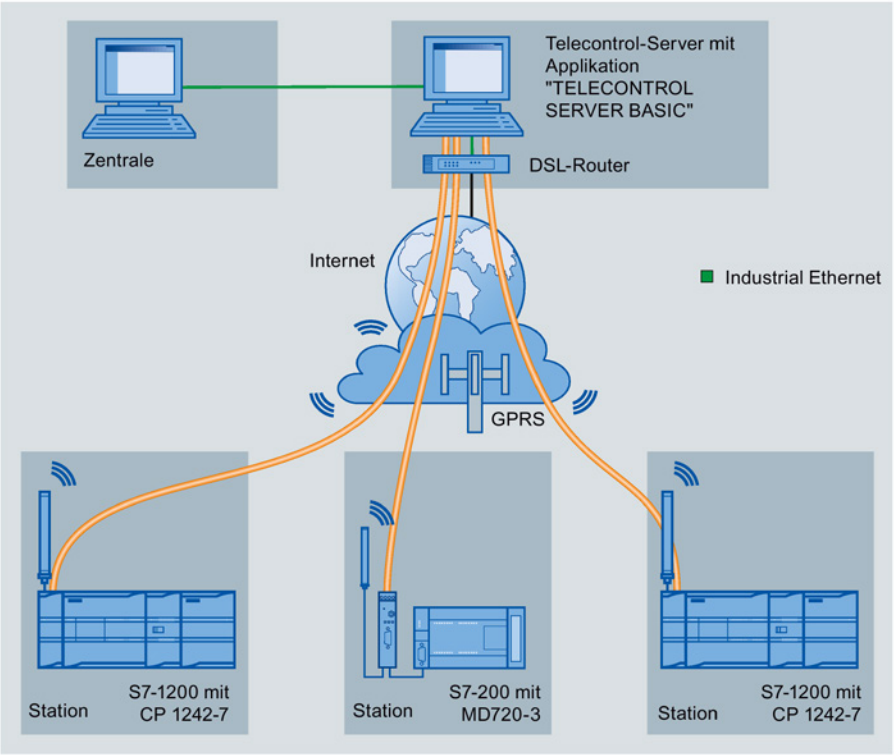


Bild 13-1 Kommunikation zwischen S7-1200-Stationen und einer Zentrale

Bei den Telecontrol-Anwendungen kommunizieren SIMATIC S7-1200-Stationen mit CP 1242-7 über das GSM-Netz und das Internet mit einer Zentrale. Auf dem Telecontrol-Server in der Zentrale wird die Anwendung "TELECONTROL SERVER BASIC" (TCSB) installiert. Damit ergeben sich folgende Anwendungsfälle:

- Telecontrol-Kommunikation zwischen Station und Zentrale

  In diesem Anwendungsfall werden Daten aus dem Feld von den Stationen über das GSM-Netz und das Internet an den Telecontrol-Server in der Zentrale gesendet. Der Telecontrol-Server dient der Überwachung der entfernten Stationen.

- Kommunikation zwischen Station und einer Leitzentrale mit OPC-Client

  Wie im ersten Fall kommunizieren die Stationen mit dem Telecontrol-Server. Mithilfe des integrierten OPC-Servers tauscht der Telecontrol-Server die Daten mit dem OPC-Client der Leitzentrale aus.

  OPC-Client und Telecontrol-Server können sich auch auf einem einzigen Rechner befinden, beispielsweise wenn TCSB auf einem Leitstellenrechner mit WinCC installiert wird.

- Kommunikation zwischen Stationen über eine Zentrale

  Kommunikation ist möglich mit S7-Stationen, die auch mit einem CP 1242-7 ausgerüstet sind.

  Für die Kommunikation zwischen Stationen leitet der Telecontrol-Server die Telegramme der Sender-Station an die Empfänger-Station weiter.

Direkte Kommunikation zwischen Stationen

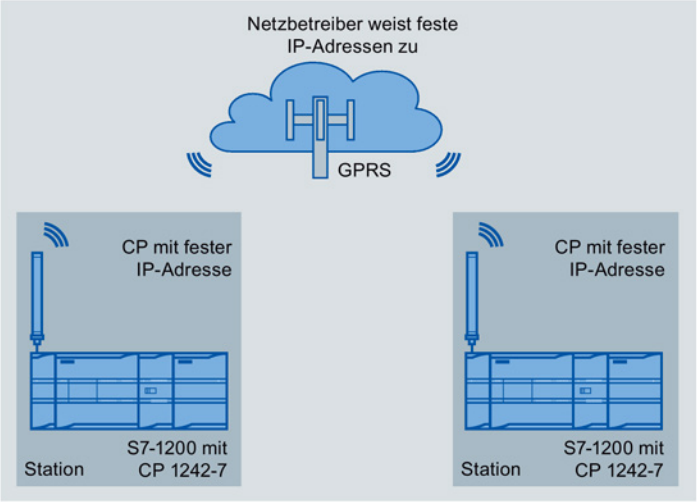


Bild 13-2 Direkte Kommunikation von zwei S7-1200-Stationen

In dieser Konfiguration kommunizieren zwei SIMATIC S7-1200-Stationen mithilfe des CP 1242-7 über das GSM-Netz direkt miteinander. Jeder CP 1242-7 hat eine feste IP-Adresse. Der entsprechende Dienst des GSM-Netzbetreibers muss dies ermöglichen.

TeleService über GPRS

Bei TeleService über GPRS kommuniziert eine Engineering-Station, auf der STEP 7 installiert ist, über das GSM-Netz und das Internet mit dem CP 1242-7 in der S7-1200.

Da eine Firewall in der Regel für Verbindungsanforderungen von außen geschlossen ist, wird eine Vermittlerstation zwischen entfernter Station und Engineering-Station benötigt. Diese Vermittlerstation kann ein Telecontrol-Server oder, falls in der Konfiguration kein Telecontrol-Server vorhanden ist, ein TeleService-Gateway sein.

TeleService mit Telecontrol-Server

Die Verbindung läuft über den Telecontrol-Server.

- Engineering-Station und Telecontrol-Server sind über Intranet (LAN) oder Internet verbunden.
- Telecontrol-Server und entfernte Station sind über das Internet und das GSM-Netz verbunden.

Engineering-Station und Telecontrol-Server können auch der gleiche Rechner sein, d. h, STEP 7 und TCSB sind auf dem gleichen Rechner installiert.

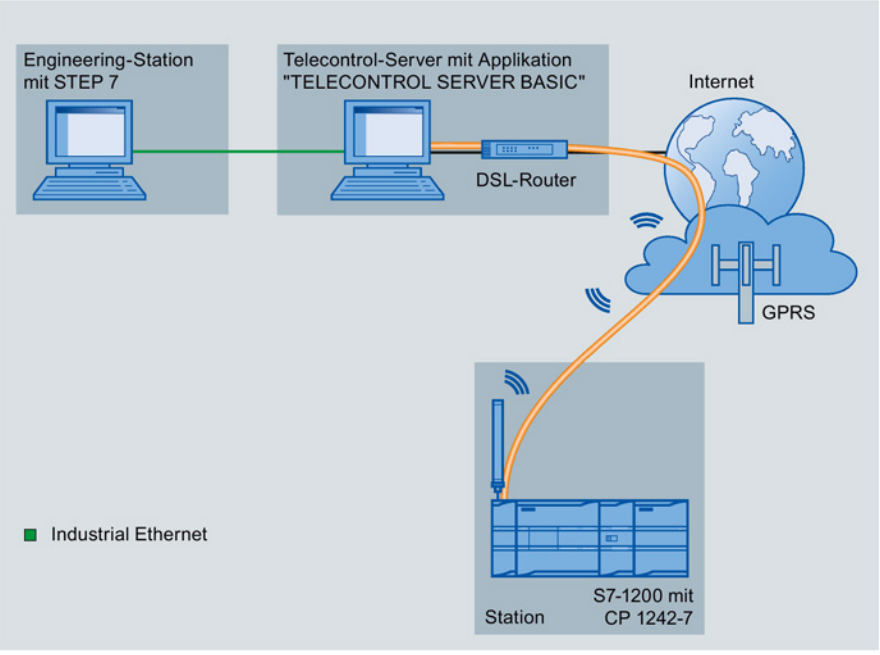


Bild 13-3 TeleService über GPRS in einer Konfiguration mit Telecontrol-Server

TeleService ohne Telecontrol-Server

Die Verbindung läuft über das TeleService-Gateway.

Die Verbindung zwischen Engineering-Station und TeleService-Gateway kann lokal über LAN oder über das Internet laufen.

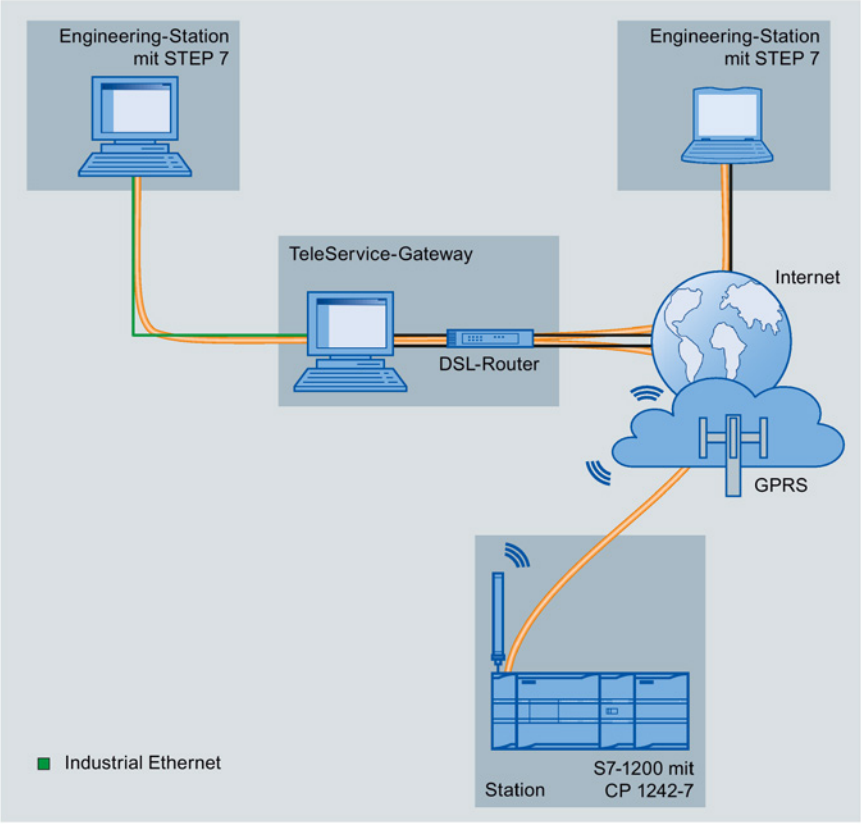


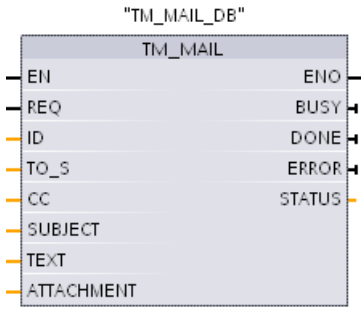
Bild 13-4 TeleService über GPRS in einer Konfiguration mit TeleService-Gateway

TeleService-Kommunikation (SMTP-E-Mail) 14

14.1 Anweisung TM_Mail (E-Mail senden)

Tabelle 14- 1 Anweisung TM_MAIL

KOP/FUP	SCL	Beschreibung

	`"TM_MAIL_DB"(`
	`REQ:=_bool_in_,`
	`ID:=_int_in_,`
	`TO_S:=_string_in_,`
	`CC:=_string_in_,`
	`SUBJECT:=_string_in_,`
	`TEXT:= _string_in_,`
	`ATTACHMENT:=_variant_in_,`
	`BUSY=>_bool_out_,`
	`DONE=>_bool_out_,`
	`ERROR=>_bool_out_,`
	`STATUS=>_word_out_,);`
		Die Anweisung TM_MAIL sendet mittels SMTP (Simple Mail Transfer Protocol) eine E-Mail über TCP/IP über die Industrial Ethernet-Verbindung der CPU. Wo keine Ethernet-basierte Internetverbindung verfügbar ist, kann ein optionaler Teleservice-Adapter für die Verbindung mit Telefonfestnetzleitungen verwendet werden. TM_MAIL wird asynchron ausgeführt und der Auftrag erstreckt sich über mehrere Aufrufe von TM_MAIL. Wenn Sie TM_MAIL aufrufen, müssen Sie einen Instanz-DB zuweisen. **Der Instanz- DB darf nicht als remanent definiert sein**. Dadurch wird sichergestellt, dass der Instanz-DB beim Wechsel der CPU von STOP nach RUN initialisiert wird und dass eine neue Ausführung von TM_MAIL ausgelöst werden kann.

[1] STEP 7 erstellt automatisch den Instanz-DB, wenn Sie die Anweisung einfügen.

Sie starten den Sendevorgang einer E-Mail mit einer positiven Flanke von 0 nach 1 am Eingangsparameter REQ. Die folgende Tabelle zeigt die Beziehungen zwischen BUSY, DONE und ERROR. Sie können den Fortschritt der Ausführung von TM_MAIL überwachen und die Fertigstellung erkennen, indem Sie diese Parameter bei aufeinander folgenden Aufrufen auswerten.

Die Ausgangsparameter DONE, ERROR, STATUS, und SFC_STATUS sind nur einen Zyklus lang gültig, wenn der Zustand des Ausgangsparameters BUSY von 1 auf 0 wechselt. Ihre Programmlogik muss temporäre Ausgangszustandswerte speichern, damit Sie in nachfolgenden Programmausführungszyklen Zustandsänderungen erkennen können.

Tabelle 14- 2 Interaktion zwischen den Parametern Done, Busy und Error

DONE	BUSY	ERROR	Beschreibung
Irrelevant	1	Irrelevant	Auftrag ist in Bearbeitung.
1	0	0	Der Auftrag wurde erfolgreich ausgeführt.
0	0	1	Auftrag wurde mit Fehler beendet. Die Ursache des Fehlers finden Sie im Parameter STATUS.
0	0	0	Kein Auftrag in Bearbeitung.

Wenn die CPU während der Ausführung von TM_MAIL in STOP versetzt wird, wird die Kommunikationsverbindung zum E-Mailserver beendet. Die Kommunikationsverbindung zum E-Mailserver geht auch verloren, wenn bei der Kommunikation der CPU auf dem Industrial Ethernet-Bus Probleme auftreten. In diesen Fällen wird der Sendevorgang unterbrochen und die E-Mail erreicht den Empfänger nicht.

ACHTUNG
Anwenderprogramme ändern Das Löschen und Ersetzen von Programmbausteinen, die Aufrufe von TM_MAIL oder Aufrufe des Instanz-DBs von TM_MAIL können die Verknüpfung von Programmbausteinen unterbrechen. Wenn Sie verknüpfte Programmbausteine nicht speichern, können die TPC/IP-Kommunikationsfunktionen einen undefinierten Zustand einnehmen, der möglicherweise zu Sachschaden führt. Nach der Übertragung eines geänderten Programmbausteins müssen Sie einen CPU-Neustart (Warmstart) oder einen Kaltstart durchführen. Um die Verknüpfung von Programmbausteinen nicht zu unterbrechen, ändern Sie die Teile Ihres Anwenderprogramms, die sich direkt auf die TM_MAIL-Aufrufe auswirken, nur in den folgenden Fällen: • Sich die CPU in STOP befindet • Keine E-Mail gesendet wird (REQ und BUSY = 0)

Datenkonsistenz

Der Eingangsparameter ADDR_MAIL_SERVER wird gelesen, wenn der Vorgang gestartet wird. Ein neuer Wert wird erst wirksam, wenn der gegenwärtige Vorgang beendet ist und eine neue Anweisung TM_MAIL initiiert wird.

Im Gegensatz dazu werden die Parameter WATCH_DOG_TIME, TO_S, CC, FROM, SUBJECT, TEXT, ATTACHMENT, USERNAME und PASSWORD während der Ausführung von TM_MAIL gelesen und können erst geändert werden, wenn der Auftrag beendet ist (BUSY = 0)

Wählverbindung: IE-Parameter des TS-Adapters konfigurieren

Sie müssen die IE-Parameter des Teleservice-Adapters für ausgehende Aufrufe konfigurieren, um eine Wählverbindung zum Server Ihres Internet-Dienstanbieters herzustellen. Wenn Sie das Attribut für den Aufruf auf Anforderung einrichten, dann wird die Verbindung nur hergestellt, wenn eine E-Mail gesendet wird. Bei einer analogen Modemverbindung ist für den Verbindungsvorgang mehr Zeit erforderlich (ca. eine zusätzliche Minute). Sie müssen diese zusätzliche Zeit in den Wert WATCH_DOG_TIME einschließen.

Tabelle 14- 3 Datentypen für die Parameter

Parameter und Datentyp		Datentypen	Beschreibung
REQ	IN	Bool	Die Anweisung wird durch eine steigende Flanke (0 nach 1) gestartet.
ID	IN	Int	Verbindungskennung: Siehe ID-Parameter der Anweisungen TCON, TDISCON, TSEND und TRCV. Eine Zahl, die nicht für weitere Instanzen dieser Anweisung im Anwenderprogramm verwendet wird, muss verwendet werden.
TO_S	IN	String	Empfängeradressen: STRING-Daten mit einer maximalen Länge von 240 Zeichen.
CC	IN	String	Empfängeradressen für CC-Kopie (optional): STRING-Daten mit einer maximalen Länge von 240 Zeichen.
SUBJECT	IN	String	Betreff der E-Mail: STRING-Daten mit einer maximalen Länge von 240 Zeichen.
TEXT	IN	String	Text der E-Mail (optional): STRING-Daten mit einer maximalen Länge von 240 Zeichen. Enthält dieser Parameter eine leere Zeichenkette, wird die E-Mail ohne Textnachricht gesendet.
ATTACHMENT	IN	Variant	Pointer auf E-Mail-Anhang: Byte-, Wort- oder Doppelwortdaten mit einer maximalen Länge von 65.534 Byte. Wird kein Wert zugewiesen, wird die E-Mail ohne Anhang gesendet.
DONE	OUT	Bool	• 0 - Auftrag noch nicht gestartet oder wird noch ausgeführt. • 1 - Auftrag wurde fehlerfrei ausgeführt.
BUSY	OUT	Bool	• 0 - Kein Vorgang in Bearbeitung • 1- Vorgang in Bearbeitung
ERROR	OUT	Bool	Das Bit ERROR ist einen Zyklus lang = 1, nachdem die letzte Anforderung mit Fehler beendet wurde. Der Fehlercode im Ausgang STATUS ist nur in dem einen Zyklus gültig, in dem ERROR = 1 ist.
STATUS	OUT	Word	Rückgabewert oder Fehlerinformationen der Anweisung TM_MAIL.

Parameter und Datentyp		Datentypen	Beschreibung
ADDR_MAIL_SERVER	[1] Static	DWord	IP-Adresse des Mailservers: Sie müssen jedes Fragment der IP-Adresse als Oktett aus zwei 4-Bit-Hexadezimalzeichen zuweisen. Ist das Fragment der IP-Adresse = Dezimalwert 10, was dem Hexadezimalwert A entspricht, dann müssen Sie für dieses Oktett "0A" eingeben. Beispiel: IP-Adresse = 192.168.0.10 ADDR_MAIL_SERVER = DW#16#C0A8000A, wobei gilt: • 192 = 16#C0, • 168 =16#A8 • 0 = 16#00 • 10 = 16#0A
WATCH_DOG_TIME	[1] Static	Time	Die maximale Zeitdauer, während der TM_MAIL den gesamten SMTP-Prozess abschließen muss, von der Einleitung der Verbindung zum SMTP bis zum Ende der SMTP-Übertragung. Wird dieser Zeitraum überschritten, wird die Ausführung von TM_MAIL mit einem Fehler beendet. Die tatsächliche Zeitverzögerung, bis TM_MAIL endet und der Fehler ausgegeben wird, kann die WATCH_DOG_TIME überschreiten, weil zum Trennen der Verbindung zusätzliche Zeit erforderlich ist. Sie sollten zunächst einen Zeitraum von 2 Minuten einrichten. Dieser Zeitraum kann bei einer ISDN-Telefonverbindung sehr viel kleiner sein.
USERNAME	[1] Static	String	Benutzername des Mailkontos: STRING-Daten mit einer maximalen Länge von 180 Zeichen.
PASSWORD	[1] Static	String	Passwort für den Mailserver: STRING-Daten mit einer maximalen Länge von 180 Zeichen.
FROM	[1] Static	String	Adresse des Absenders: STRING mit einer maximalen Länge von 240 Zeichen
SFC_STATUS	[1] Static	Word	Bedingungscode der Ausführung der aufgerufenen Kommunikationsbausteine

[1] Die Werte dieser Parameter werden nicht bei jedem Aufruf von TM_MAIL modifiziert. Die Werte werden im Instanz-Datenbaustein TM_MAIL zugewiesen und nur einmal referenziert, im ersten Aufruf von TM_MAIL.

SMTP-Authentifizierung

TM_MAIL unterstützt das SMTP-AUTH-LOGIN-Authentifizierungsverfahren. Informationen zu diesem Authentifizierungsverfahren finden Sie im Handbuch des Mailservers oder auf der Website Ihres Internet-Dienstanbieters.

Das AUTH LOGIN-Authentifizierungsverfahren verwendet die Parameter USERNAME und PASSWORD der Anweisung TM_MAIL, um eine Verbindung zum Mailserver herzustellen. Der Benutzername und das Passwort müssen zuvor in einem E-Mail-Konto auf einem E-Mailserver eingerichtet sein.

Wenn dem Parameter USERNAME kein Wert zugewiesen ist, wird das AUTH LOGIN-Authentifizierungsverfahren nicht verwendet und die E-Mail wird ohne Authentifizierung gesendet.

TO_S:, CC: und FROM: Parameter

Die Parameter TO_S:, CC: und FROM: sind Zeichenketten, wie in den folgenden Beispielen gezeigt:

TO: <wenna@mydomain.com>, <ruby@mydomain.com>,

CC: <admin@mydomain.com>, <judy@mydomain.com>,

FROM: <admin@mydomain.com>

Die folgenden Regeln müssen bei der Eingabe dieser Zeichenfolgen beachtet werden:

- Die Zeichen "TO:", "CC:" und "FROM:" müssen eingegeben werden, einschließlich des Doppelpunkts.
- Ein Leerzeichen und eine öffnende spitze Klammer "<" müssen vor jeder Adresse stehen. Beispielsweise muss sich zwischen "TO:" und <E-Mail-Adresse> ein Leerzeichen befinden.
- Eine abschließende spitze Klammer ">" muss nach jeder Adresse eingegeben werden.
- Bei den Adressen TO_S: und CC: muss nach jeder Adresse ein Komma "," eingegeben werden. Beispielsweise ist das Komma nach der einzelnen E-Mail-Adresse in "TO: <email address>," erforderlich.
- Für die Eingabe FROM: darf nur eine E-Mail-Adresse angegeben werden, ohne Komma am Ende.

Wegen des Laufzeitmodus und der Speicherauslastung wird für die Daten TO_S:, CC: und FROM: der Anweisung TM_MAIL keine Syntaxprüfung durchgeführt. Wenn die obigen Formatregeln nicht exakt eingehalten werden, schlägt die Übertragung des SMTP-E-Mailservers fehl.

Parameter STATUS und SFC_STATUS

Die von TM_MAIL zurückgegebenen Bedingungscodes der Ausführung können wie folgt klassifiziert werden:

- W#16#0000: Ausführung von TM_MAIL wurde erfolgreich beendet.
- W#16#7xxx: Zustand der Ausführung von TM_MAIL.
- W#16#8xxx: Fehler in einem internen Aufruf eines Kommunikationsgeräts oder des Mailservers.

Die folgende Tabelle zeigt die Bedingungscodes der Ausführung von TM_MAIL, mit Ausnahme der Bedingungscodes von intern aufgerufenen Kommunikationsmodulen.

Hinweis

Anforderungen an den E-Mailserver

TM_MAIL kann nur mittels SMTP über Port 25 mit einem E-Mailserver kommunizieren. Die zugewiesene Portnummer kann nicht geändert werden.

Die meisten IT-Abteilungen und externen E-Mailserver blockieren mittlerweile Port 25, um zu verhindern, dass ein mit einem Virus infizierter PC zu einem gefährlichen E-Mail-Generator wird.

Sie können über SMTP eine Verbindung mit einem internen E-Mailserver herstellen und den internen Server die aktuellen Sicherheitserweiterungen verwalten lassen, die erforderlich sind, um E-Mails über das Internet an einen externen E-Mailserver weiterzuleiten.

Beispiel: Konfiguration eines internen E-Mailservers

Wenn Sie Microsoft Exchange als internen E-Mailserver nutzen, können Sie den Server so konfigurieren, dass der SMTP-Zugriff von der IP-Adresse, die dem S7-1200 Zielsystem zugewiesen ist, zulässig ist. Exchange-Verwaltungskonsole konfigurieren: Serverkonfiguration > Hub-Transport > Empfangsconnectors > IP-Relay. Im Register "Netzwerke" gibt es das Feld "E-Mail von Remoteservern mit folgenden IP-Adressen empfangen". Hier geben Sie die IP-Adresse des PLC-Geräts ein, das die Anweisung TM_MAIL ausführt. Für diesen Verbindungstyp ist bei einem internen Microsoft Exchange-Server keine Authentifizierung erforderlich.

Konfiguration des E-Mailservers

TM_MAIL kann nur einen E-Mailserver verwenden, der die Kommunikation über Port 25, SMTP und die AUTH-LOGIN-Authentifizierung (optional) gestattet.

Richten Sie ein kompatibles Konto auf dem E-Mailserver ein, um die Remote-SMTP-Anmeldung zu akzeptieren. Bearbeiten Sie anschließend den Instanz-DB für TM_MAIL so, dass die Zeichenketten TM_MAIL USERNAME und PASSWORD eingegeben werden, über die die Verbindung mit Ihrem E-Mail-Konto authentifiziert wird.

Tabelle 14- 4 Bedingungscodes

STATUS (W#16#...):	SFC_STATUS (W#16#...):	Beschreibung
0000	-	Die Ausführung von TM_MAIL wurde fehlerfrei beendet. Dieser STATUS-Code 0 gewährleistet nicht, dass tatsächlich eine E-Mail gesendet wurde (siehe Punkt 1 im auf diese Tabelle folgenden Hinweis).
7001	-	TM_MAIL ist aktiv (BUSY = 1).
7002	7002	TM_MAIL ist aktiv (BUSY = 1).
8xxx	xxxx	Die Ausführung von TM_MAIL wurde mit einem Fehler in den internen Aufrufen der Kommunikationsanweisung beendet. Weitere Informationen zum Parameter SFC_STATUS finden Sie in den Beschreibungen des Parameters STATUS der zugrunde liegenden offenen PROFINET-Benutzerkommunikationsanweisungen.
8010	xxxx	Verbindung konnte nicht aufgebaut werden: Weitere Informationen zum Parameter SFC_STATUS finden Sie in der Beschreibung des Parameters STATUS der Anweisung TCON.
8011	xxxx	Fehler beim Senden der Daten: Weitere Informationen zum Parameter SFC_STATUS finden Sie in der Beschreibung des Parameters STATUS der Anweisung TSEND.
8012	xxxx	Fehler beim Empfangen der Daten: Weitere Informationen zum Parameter SFC_STATUS finden Sie in der Beschreibung des Parameters STATUS der Anweisung TRCV.
8013	xxxx	Verbindung konnte nicht aufgebaut werden: Weitere Informationen zum Auswerten des Parameters SFC_STATUS finden Sie in den Beschreibungen des Parameters STATUS der Anweisungen TCON und TDISCON.
8014	-	Verbindung konnte nicht aufgebaut werden: Sie haben möglicherweise eine inkorrekte IP-Adresse des Mailservers (ADDR_MAIL_SERVER) oder zu wenig Zeit (WATCH_DOG_TIME) für die Verbindung angegeben. Es ist auch möglich, dass die CPU keine Verbindung zum Netzwerk hat oder die CPU-Konfiguration inkorrekt ist.
8015	-	Ungültiger Pointer für den Parameter ATTACHMENT: Verwenden Sie einen Variant-Pointer mit Datentyp- und Längenzuweisung. Beispiel: "P#DB.DBX0.0" ist falsch und "P#DB.DBX0.0 Byte 256" ist richtig.
82xx, 84xx, 85xx	-	Die Fehlermeldung kommt vom Mailserver und entspricht der Fehlernummer "8" des SMTP-Protokolls. Siehe Punkt 2 in dem auf diese Tabelle folgenden Hinweis.
8450	-	Vorgang wird nicht ausgeführt: Mailbox ist nicht verfügbar, wiederholen Sie den Vorgang später.
8451	-	Vorgang abgebrochen: Lokaler Fehler bei der Bearbeitung. Wiederholen Sie den Vorgang später.

STATUS (W#16#...):	SFC_STATUS (W#16#...):	Beschreibung
8500	-	Fehler in der Befehlssyntax: Die Ursache kann sein, dass der E-Mailserver den LOGIN-Authentifizierungsvorgang nicht unterstützt. Prüfen Sie die Parameter von TM_MAIL. Versuchen Sie, eine E-Mail ohne Authentifizierung zu senden. Versuchen Sie, den Parameter USERNAME durch eine leere Zeichenkette zu ersetzen.
8501	-	Syntaxfehler: Inkorrekter Parameter oder inkorrektes Argument; Sie haben möglicherweise eine inkorrekte Adresse in den Parameter TO_S oder CC eingegeben.
8502	-	Befehl ist unbekannt oder nicht implementiert: Prüfen Sie Ihre Eingaben, insbesondere den Parameter FROM. Möglicherweise ist dieser unvollständig und Sie haben eines der Zeichen "@" oder "." weggelassen.
8535	-	SMTP-Authentifizierung ist unvollständig. Sie haben möglicherweise einen inkorrekten Benutzernamen oder ein inkorrektes Passwort eingegeben.
8550	-	Der Mailserver ist nicht erreichbar, oder Sie haben keine Zugriffsrechte. Sie haben möglicherweise einen inkorrekten Benutzernamen oder ein inkorrektes Passwort eingegeben oder Ihr Mailserver unterstützt keinen Zugriff über Anmeldung. Eine weitere Ursache für diesen Fehler kann eine fehlerhafte Eingabe des Domainnamens nach dem Zeichen "@" in den Parametern TO_S oder CC sein.
8552	-	Vorgang abgebrochen: Zugeordnete Speichergröße überschritten; wiederholen Sie den Vorgang später.
8554	-	Übertragung fehlgeschlagen: Wiederholen Sie den Vorgang später.

Hinweis

Mögliche nicht gemeldete E-Mail-Übertragungsfehler

- Die fehlerhafte Angabe einer Empfängeradresse erzeugt bei TM_MAIL keinen STATUS-Fehler. In einem solchen Fall ist nicht gewährleistet, dass die zusätzlichen Empfänger (mit korrekten E-Mail-Adressen) die E-Mail empfangen.
- Weitere Informationen zu SMTP-Fehlercodes finden Sie im Internet oder in der Fehlerdokumentation des Mailservers. Sie können die letzte Fehlermeldung des Mailservers auch auslesen. Die Fehlermeldung wird im Parameter buffer1 des Instanz-DBs von TM_MAIL gespeichert.

Online- und Diagnose-Tools 15

15.1 Status-LEDs

Die CPU und die E/A-Module nutzen LEDs, um Informationen über den Betriebszustand des Moduls oder der E/A zu liefern.

Status-LEDs an einer CPU

Die CPU bietet die folgenden Statusanzeigen:

- STOP/RUN
  - Gelbes Dauerlicht zeigt den Betriebszustand STOP an
  - Grünes Dauerlicht zeigt den Betriebszustand RUN an
  - Blinken (abwechselnd grün und gelb) zeigt an, dass die CPU in der Betriebsart STARTUP ist
- ERROR
  - Rotes Blinklicht zeigt einen Fehler an, z.B. einen internen Fehler in der CPU, einen Fehler der Memory Card oder einen Konfigurationsfehler (nicht übereinstimmende Module)
  - Fehlerzustand:
    - Rotes Dauerlicht zeigt defekte Hardware an
    - Alle LEDs blinken, wenn der Fehler in der Firmware erkannt wird
- Wenn Sie eine Memory Card stecken, blinkt die LED MAINT (Wartung). Die CPU wechselt dann in den Betriebszustand STOP. Nachdem die CPU in den Betriebszustand STOP gegangen ist, führen Sie eine der folgenden Funktionen durch, um die Auswertung der Memory Card zu starten:
  - Versetzen Sie die CPU in den Betriebszustand RUN
  - Führen Sie ein Urlöschen durch (MRES)
  - Schalten Sie die CPU aus und wieder ein

Den Zustand der LEDs können Sie auch mit der Anweisung LED (Seite 392) ermitteln.

Tabelle 15- 1 Status-LEDs für eine CPU

Beschreibung	STOP/RUN Gelb/Grün	ERROR Rot	MAINT Gelb
Netz aus	Aus	Aus	Aus
Anlauf, Selbsttest oder Firmware-Aktualisierung	Blinken (abwechselnd gelb und grün)	-	Aus
Betriebszustand STOP	Ein (gelb)	-	-
Betriebszustand RUN	Ein (grün)	-	-
Ziehen Sie die Memory Card	Ein (gelb)	-	Blinkt
Fehler	Ein (gelb oder grün)	Blinkt	-
Wartung erforderlich • Geforcte E/A • Batteriewechsel erforderlich (bei installiertem Batterieboard)	Ein (gelb oder grün)	-	Ein
Hardware defekt	Ein (gelb)	Ein	Aus
LED-Test oder CPU-Firmware defekt	Blinken (abwechselnd gelb und grün)	Blinkt	Blinkt
Unbekannte oder inkompatible Version der CPU-Konfiguration	Ein (gelb)	Blinkt	Blinkt

Hinweis

Fehler "Unbekannte oder inkompatible Version der CPU-Konfiguration"

Wenn Sie versuchen, ein S7-1200 V3.0-Programm in eine S7-1200 V4.0-CPU zu laden, verursacht dies einen CPU-Fehler und die CPU zeigt eine entsprechende Fehlermeldung im Diagnosepuffer an. Wenn Sie diesen Zustand durch Verwendung einer Übertragungskarte (Seite 139) mit einer ungültigen Programmversion hervorgerufen haben, entnehmen Sie die Karte und versetzen die CPU von STOP in RUN, führen ein Urlöschen (MRES) durch oder schalten die CPU aus und wieder ein. Wenn Sie diesen Zustand durch Laden eines ungültigen Programms hervorgerufen haben, setzen Sie die CPU auf die Werkseinstellungen zurück (Seite 1123). Nachdem Sie den Fehlerzustand der CPU behoben haben, können Sie ein gültiges CPU-Programm V4.0 laden.

Die CPU bietet auch zwei LEDs, die den Zustand der PROFINET-Kommunikation anzeigen. Öffnen Sie die untere Abdeckklappe der Klemmenleiste, um die PROFINET-LEDs zu sehen.

- Link (grün) wird eingeschaltet, um eine erfolgreiche Verbindung anzuzeigen
- Rx/Tx (gelb) wird eingeschaltet, um Übertragungsaktivität anzuzeigen

Die CPU und jedes digitale Signalmodul (SM) bieten eine I/O Channel -LED für jeden digitalen Eingang und Ausgang. I/O Channel (grün) wird ein- oder ausgeschaltet, um den Zustand des jeweiligen Eingangs oder Ausgangs anzuzeigen.

Verhalten der S7-1200 nach einem schweren Fehler

Wenn die CPU-Firmware einen schweren Fehler erkennt, wird ein Neustart im Fehlermodus versucht und der Fehlermodus durch beständiges Blinken der LEDs STOP/RUN, ERROR und MAINT angezeigt. Das Anwenderprogramm und die Hardwarekonfiguration werden nach dem Neustart im Fehlermodus nicht geladen.

Wenn die CPU den Neustart im Fehlermodus erfolgreich beendet, werden die Ausgänge von CPU und Signalboard auf 0 gesetzt und die Ausgänge der Signalmodule im zentralen Baugruppenträger sowie der dezentralen E/A werden auf die konfigurierte "Reaktion auf CPU-STOP" gesetzt.

Schlägt der Neustart im Fehlermodus fehl (z. B. aufgrund eines Hardwarefehlers), sind die LEDs STOP und ERROR eingeschaltet und die LED MAINT ist ausgeschaltet.

⚠ WARNUNG

Betrieb im Fehlerzustand kann nicht gewährleistet werden

Steuerungen können bei unsicheren Betriebszuständen ausfallen und dadurch unkontrolliertes Betriebsverhalten der gesteuerten Geräte verursachen. Derartiges unerwartetes Betriebsverhalten des Automatisierungssystems kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Sorgen Sie daher für eine NOTAUSFunktion, elektromechanische oder andere redundante Sicherheitseinrichtungen, die von Ihrem PLC unabhängig sind.

Status-LEDs an einem SM

Außerdem bietet jedes digitale SM eine DIAG-LED, die den Zustand des Moduls anzeigt:

- Grün zeigt an, dass das Modul betriebsbereit ist
- Rot zeigt an, dass das Modul defekt oder nicht betriebsbereit ist

Jedes analoge SM bietet eine I/O Channel -LED für jeden der analogen Eingänge und Ausgänge.

- Grün zeigt an, dass der Kanal konfiguriert wurde und aktiv ist
- Rot zeigt einen Fehlerzustand des jeweiligen analogen Eingangs oder Ausgangs an

Außerdem bietet jedes analoge SM eine DIAG-LED, die den Zustand des Moduls anzeigt:

- Grün zeigt an, dass das Modul betriebsbereit ist
- Rot zeigt an, dass das Modul defekt oder nicht betriebsbereit ist

Das SM erkennt das Vorhandensein bzw. die Abwesenheit von Modulspannung (feldseitige Spannung, sofern erforderlich).

Tabelle 15- 2 Status-LEDs für ein Signalmodul (SM)

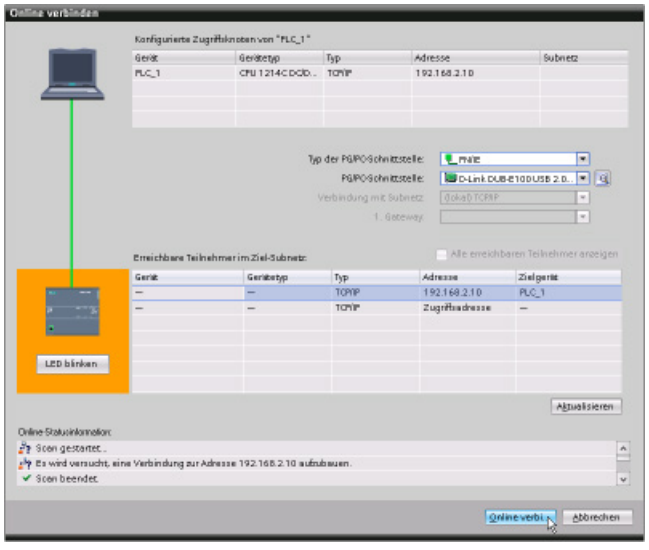
Beschreibung	DIAG (Gelb/Grün)	I/O Channel (Gelb/Grün)
Feldseitige Spannung ist aus	Rot blinkend	Rot blinkend
Nicht konfiguriert oder Aktualisierung in Bearbeitung	Grün blinkend	Aus
Modul fehlerfrei konfiguriert	Ein (grün)	Ein (grün)
Fehlerbedingung	Rot blinkend	-
E/A-Fehler (bei aktivierter Diagnose)	-	Rot blinkend
E/A-Fehler (bei deaktivierter Diagnose)	-	Ein (grün)

15.2 Online-Verbindung mit einer CPU herstellen

Zum Laden von Programmen und Projektdaten sowie für die folgenden Tätigkeiten ist eine Online-Verbindung zwischen dem Programmiergerät und der CPU erforderlich:

- Testen von Anwenderprogrammen
- Anzeigen und Ändern des Betriebszustands der CPU (Seite 1125)
- Anzeigen und Einstellen von Datum und Uhrzeit der CPU (Seite 1122)
- Anzeigen der Modulinformationen
- Vergleichen und Synchronisieren (Seite 1127) von Offline- und Online-Programmbausteinen
- Laden von Programmbausteinen in die und aus der CPU
- Anzeigen von Diagnose und Diagnosepuffer (Seite 1126)
- Mit einer Beobachtungstabelle (Seite 1131) das Anwenderprogramm durch Beobachten und Steuern von Werten testen
- Mit einer Forcetabelle Werte in der CPU forcen (Seite 1135)

Um eine Online-Verbindung zu einer konfigurierten CPU herzustellen, klicken Sie in der Projektnavigation auf die CPU und in der Projektansicht auf die Schaltfläche "Online verbinden":



Wenn Sie zum ersten Mal mit dieser CPU online gehen, müssen Sie den Typ der PG/PC-Schnittstelle und die spezifische PG/PC-Schnittstelle im Dialog "Online verbinden" auswählen, bevor Sie eine Online-Verbindung zu einer CPU auf dieser Schnittstelle herstellen.

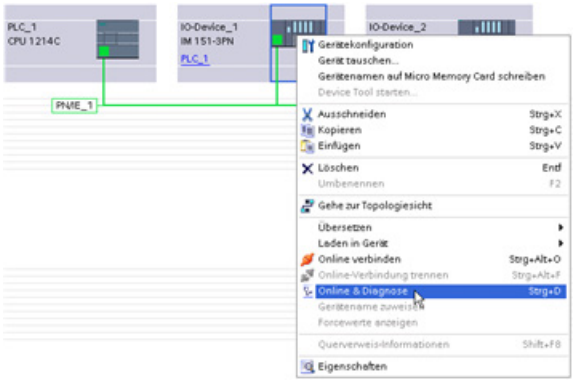
Das Programmiergerät muss an die CPU angeschlossen sein. Orangefarbene Rahmen weisen auf eine Online-Verbindung hin. Nun können Sie die Tools unter "Online & Diagnose" in der Projektnavigation und die Taskcard "Online-Tools" verwenden.

15.3 Einem PROFINET IO-Gerät online einen Namen zuweisen

Den Geräten in Ihrem PROFINET-Netzwerk muss ein Name zugewiesen sein, damit Sie eine Verbindung zur CPU herstellen können. Sie weisen Ihren PROFINET-Geräten im Editor "Geräte & Netze" Namen zu, wenn die Geräte noch keinen Namen haben oder wenn der Name eines Geräts geändert werden soll.

Den Namen eines PROFINET IO-Geräts müssen Sie sowohl im STEP 7-Projekt als auch über das Werkzeug "Online & Diagnose" im Konfigurationsspeicher des PROFINET IO-Geräts zuweisen (z. B. im Konfigurationsspeicher eines ET200 S-Schnittstellenmoduls). Fehlt ein Name oder entsprechen sich die Namen an den beiden Speicherorten nicht, funktioniert der Modus für den PROFINET IO-Datenaustausch nicht.

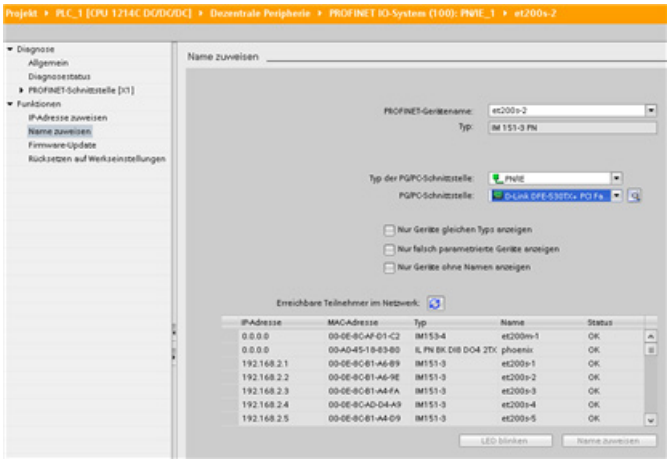
1. Klicken Sie im Editor "Geräte & Netze" mit der rechten Maustaste auf das gewünschte PROFINET IO-Gerät und wählen Sie "Online & Diagnose".



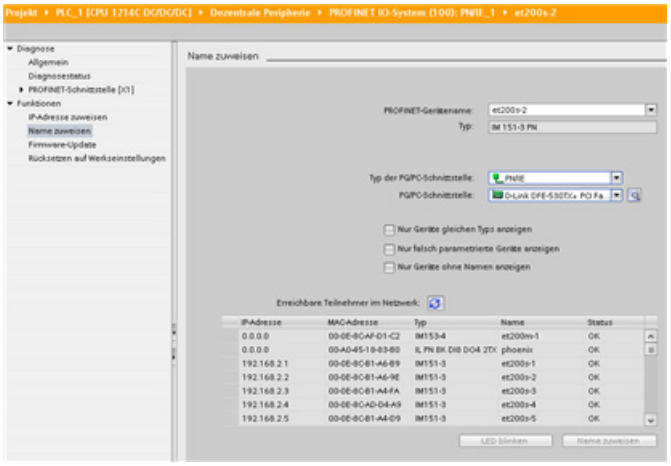
2. Wählen Sie im Dialog "Online & Diagnose" die folgenden Menübefehle:

- "Funktionen"
- "Name zuweisen"

Klicken Sie auf das Symbol "Erreichbare Teilnehmer im Netz", um alle PROFINET IO-Geräte im Netzwerk anzuzeigen.

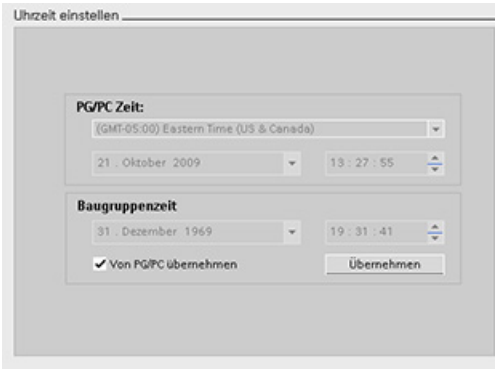


3. Klicken Sie in der daraufhin angezeigten Liste auf das gewünschte PROFINET IO-Gerät und dann auf die Schaltfläche "Name zuweisen", um den Namen in den Konfigurationsspeicher des PROFINET IO-Geräts zu schreiben.



15.4 Einstellen der IP-Adresse und der Uhrzeit

Sie können die IP-Adresse (Seite 645) und die Uhrzeit der Online-CPU einstellen. Nach dem Zugriff auf "Online & Diagnose" in der Projektnavigation einer Online-CPU können Sie die IP-Adresse anzeigen oder ändern. Ebenso können Sie Datum und Uhrzeit der CPU online aufrufen und ändern.



Hinweis

Diese Funktion ist nur für eine CPU verfügbar, die entweder nur eine MAC-Adresse hat (der noch keine IP-Adresse zugewiesen wurde) oder die auf die Werkseinstellungen zurückgesetzt wurde.

15.5 Auf Werkseinstellungen zurücksetzen

Unter den folgenden Bedingungen können Sie eine S7-1200 auf die ursprünglichen Werkseinstellungen zurücksetzen:

- Die CPU hat eine Online-Verbindung.
- Die CPU befindet sich im Betriebszustand STOP.

Hinweis

Wenn sich die CPU im Betriebszustand RUN befindet und Sie den Rücksetzvorgang starten, können Sie die CPU nach Bestätigung einer Eingabeaufforderung in den Betriebszustand STOP versetzen.

Vorgehen

Um eine CPU auf die Werkseinstellungen zurückzusetzen, gehen Sie wie folgt vor:

1. Öffnen Sie die Sicht "Online und Diagnose" der CPU.
2. Wählen Sie im Ordner "Funktionen" den Eintrag "Auf Werkseinstellungen zurücksetzen".
3. Aktivieren Sie das Kontrollkästchen "IP-Adresse beibehalten", wenn Sie die IP-Adresse beibehalten möchten, oder das Kontrollkästchen "IP-Adresse löschen", wenn Sie die IP-Adresse löschen möchten.
4. Klicken Sie auf die Schaltfläche "Zurücksetzen".
5. Bestätigen Sie die Eingabeaufforderung mit "OK".

Ergebnis

Das Modul wird ggf. in den Betriebszustand STOP versetzt und auf die Werkseinstellungen zurückgesetzt: Die CPU führt die folgenden Vorgänge durch:

Mit in der CPU gesteckter Memory Card	Ohne in die CPU gesteckte Memory Card
• Der Diagnosepuffer wird gelöscht • Die Uhrzeit wird zurückgesetzt • Der Arbeitsspeicher wird von der Memory Card wiederhergestellt • Alle Operandenbereiche werden auf konfigurierte Anfangswerte gesetzt • Alle Parameter werden auf ihre konfigurierten Werte gesetzt • Die IP-Adresse wird abhängig von Ihrer Auswahl beibehalten oder gelöscht. (Die MAC-Adresse ist fest zugewiesen und wird niemals geändert.)[1] • Sofern vorhanden wird der Steuerdatensatz (Seite 157) gelöscht	• Der Diagnosepuffer wird gelöscht • Die Uhrzeit wird zurückgesetzt • Der Arbeitsspeicher und interne Ladespeicher werden gelöscht • Alle Operandenbereiche werden auf konfigurierte Anfangswerte gesetzt • Alle Parameter werden auf ihre konfigurierten Werte gesetzt • Die IP-Adresse wird abhängig von Ihrer Auswahl beibehalten oder gelöscht. (Die MAC-Adresse ist fest zugewiesen und wird niemals geändert.)[1] • Sofern vorhanden wird der Steuerdatensatz gelöscht

Mit in der CPU gesteckter Memory Card	Ohne in die CPU gesteckte Memory Card

[1] Wenn Sie "IP-Adresse beibehalten" ausgewählt haben, setzt die CPU die IP-Adresse, Subnetzmaske und Routeradresse (falls verwendet) auf die Einstellungen in Ihrer Hardwarekonfiguration zurück, sofern Sie diese Werte nicht über das Anwenderpogramm oder ein anderes Werkzeug geändert haben. Dann stellt die CPU die geänderten Werte wieder her.

15.6 Firmware aktualisieren

Sie können die Firmware der angeschlossenen CPU mit den Online- und Diagnosetools von STEP 7 aktualisieren.

Für ein Firmware-Update gehen Sie folgendermaßen vor:

1. Öffnen Sie die Online- und Diagnoseansicht der angeschlossenen CPU.
2. Wählen Sie im Ordner "Funktionen" die Option "Firmware-Update".
3. Klicken Sie auf die Such-Schaltfläche und navigieren Sie zum Speicherort der Datei mit dem Firmware-Update. Dies kann ein Speicherort auf Ihrer Festplatte sein, in den Sie eine Firmware-Update-Datei für S7-1200 (http://support.automation.siemens.com/WW/view/de/34612486/133100) von der Service- und Support-Webseite (http://www.siemens.com/automation/) heruntergeladen haben.
4. Wählen Sie eine Datei, die für Ihr Modul geeignet ist. In der Tabelle werden die kompatiblen Module für die ausgewählte Datei angezeigt.
5. Klicken Sie auf die Schaltfläche "Aktualisierung ausführen". Folgen Sie ggf. den Anweisungen in den Dialogen, um den Betriebszustand Ihrer CPU zu ändern.

Während das Firmware-Update geladen wird, wird der Fortschritt in STEP 7 angezeigt Nach beendetem Update werden Sie aufgefordert, das Modul mit der neuen Firmware zu starten.

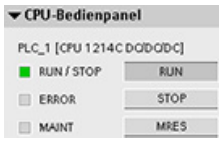
Hinweis

Wollen Sie das Modul mit der neuen Firmware nicht starten, bleibt die alte Firmware aktiv, bis das Modul, zum Beispiel durch Aus- und Einschalten, zurückgesetzt wird. Die neue Firmware wird erst nach dem Rücksetzen des Moduls aktiv.

Sie können ein Firmware-Update auch mit einer der folgenden Methoden ausführen:

- Mit einer Memory Card (Seite 146)
- Über die Standard-Webseite "Modulinformationen" des Webservers (Seite 837)

15.7 Bedienpanel für die Online-CPU



Im CPU-Bedienpanel wird der Betriebszustand (STOP oder RUN) der Online-CPU angezeigt. Hier wird auch angezeigt, ob ein Fehler in der CPU aufgetreten ist oder ob Werte geforct wurden.

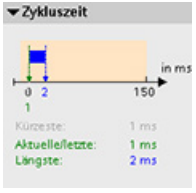
Im CPU-Bedienpanel der Taskcard "Online-Tools" können Sie den Betriebszustand einer Online-CPU ändern. Die Taskcard "Online-Tools" ist verfügbar, wenn die CPU online ist.

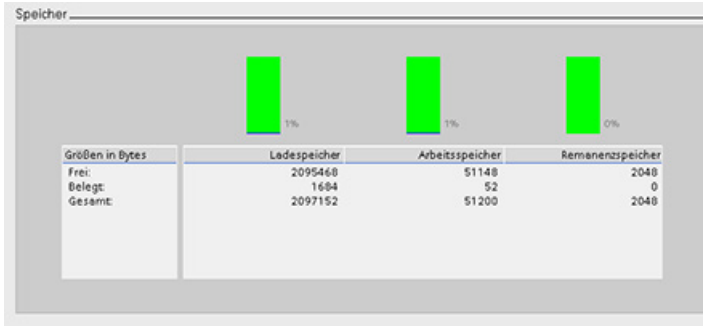
15.8 Überwachung von Zykluszeit und Speicherauslastung

Sie können die Zykluszeit und die Speicherauslastung einer Online-CPU überwachen.

Nach dem Herstellen der Verbindung zur Online-CPU öffnen Sie die Taskcard "Online-Tools", um die folgenden Messwerte anzuzeigen:

- Zykluszeit
- Speicherauslastung

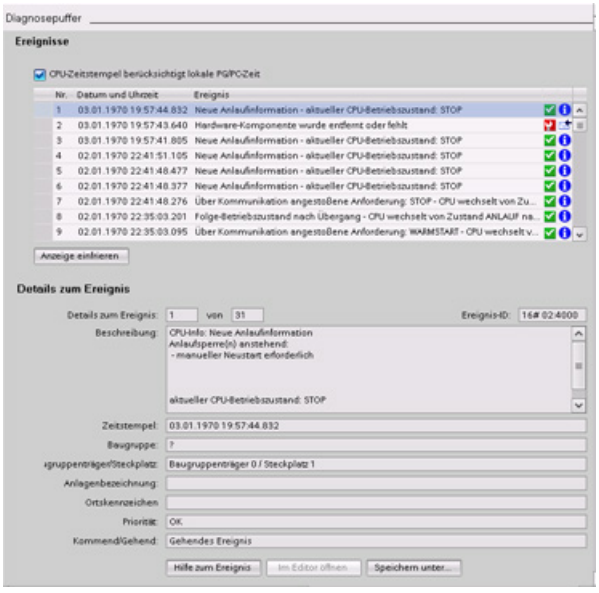




15.9 Diagnoseereignisse in der CPU anzeigen

Im Diagnosepuffer können Sie die letzten Ereignisse in der CPU betrachten. Der Diagnosepuffer ist in der Projektnavigation über "Online & Diagnose" für eine Online-CPU verfügbar. Er enthält die folgenden Einträge:

- Diagnoseereignisse
- Änderungen im Betriebszustand der CPU (Wechsel zwischen STOP und RUN)



Der erste Eintrag entspricht dem jüngsten Ereignis. Jeder Eintrag im Diagnosepuffer enthält das Datum und die Uhrzeit, zu denen das Ereignis erfasst wurde, und eine Beschreibung.

Wie viele Einträge maximal möglich sind, hängt von der CPU ab. Es werden bis zu 50 Einträge unterstützt.

Nur die letzten 10 Ereignisse im Diagnosepuffer werden nullspannungsfest gespeichert. Wird die CPU auf die Werkseinstellungen zurückgesetzt, so werden dadurch die Einträge im Diagnosepuffer gelöscht.

Die Diagnoseinformationen können Sie auch mit der Anweisung GET_DIAG (Seite 405) erfassen.

15.10 Vergleichen von Offline- und Online-CPUs

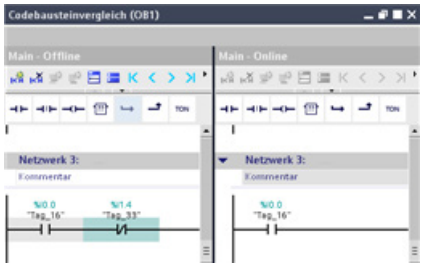
Sie können die Codebausteine in einer Online-CPU mit den Codebausteinen Ihres Projekts vergleichen. Wenn die Codebausteine Ihres Projekts nicht den Codebausteinen der Online-CPU entsprechen, haben Sie im Editor "Vergleichen" die Möglichkeit, Ihr Projekt mit der Online-CPU abzugleichen. Laden Sie dazu entweder die Codebausteine Ihres Projekts in die CPU oder löschen Sie die Bausteine aus dem Projekt, die in der Online-CPU nicht vorhanden sind.

Wählen Sie die CPU in Ihrem Projekt aus.

Wählen Sie im Editor "Vergleichen" den Befehl "Offline/online vergleichen". (Rufen Sie den Befehl entweder über das Menü "Werkzeuge" oder durch Rechtsklick auf die CPU in Ihrem Projekt auf.)

Klicken Sie in die Spalte "Aktion" eines Objekts, um auszuwählen, ob das Objekt gelöscht, keine Maßnahme durchgeführt oder das Objekt in das Gerät geladen werden soll.

Durch Klicken auf die Schaltfläche "Synchronisieren" laden Sie die Codebausteine.



Klicken Sie in der Spalte "Vergleichen mit" mit der rechten Maustaste auf ein Objekt und wählen Sie "Detaillierten Vergleich starten", um die Codebausteine nebeneinander anzuzeigen.

Bei diesem Detailvergleich werden die Unterschiede zwischen den Codebausteinen der Online-CPU und den Codebausteinen der CPU in Ihrem Projekt hervorgehoben.

15.11 Werte in der CPU beobachten und steuern

STEP 7 bietet Online-Funktionen zum Beobachten der CPU:

- Sie können die aktuellen Werte der Variablen anzeigen oder beobachten. Die Beobachtungsfunktion ändert den Programmablauf nicht. Sie liefert Ihnen Informationen zum Programmablauf und den Daten des Programms in der CPU.
- Sie können den Ablauf und die Daten des Anwenderprogramms auch mit anderen Funktionen steuern:
  - Sie können den Wert der Variablen in der Online-CPU steuern, um zu ermitteln, wie das Anwenderprogramm reagiert.
  - Sie können einen Peripherieausgang (wie A0.1:P oder "Start":P) auf einen bestimmten Wert forcen.
  - Sie können Ausgänge im Betriebszustand STOP freischalten.

Hinweis

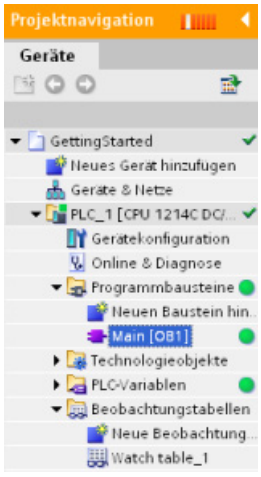
Verwenden Sie die Steuerfunktionen stets mit Vorsicht. Diese Funktionen können die Ausführung des Anwender-/Systemprogramms schwerwiegend beeinflussen.

Tabelle 15- 3 Online-Funktionen der STEP 7-Editoren

Editor	Beobachten	Steuern	Forcen
Beobachtungstabelle	Ja	Ja	Nein
Forcetabelle	Ja	Nein	Ja
Programmiereditor	Ja	Ja	Nein
Variablentabelle	Ja	Nein	Nein
DB-Editor	Ja	Nein	Nein

15.11.1 Online gehen, um die Werte in der CPU zu beobachten

Die Beobachtung der Variablen setzt eine Online-Verbindung zur CPU voraus. Klicken Sie in der Funktionsleiste einfach auf die Schaltfläche "Online verbinden".



Wenn Sie eine Verbindung zur CPU hergestellt haben, stellt STEP 7 die Überschriften der Arbeitsbereiche orangefarben dar.

In der Projektnavigation wird ein Vergleich des Offline-Projekts mit der Online-CPU angezeigt. Ein grüner Kreis bedeutet, dass die CPU und das Projekt synchronisiert sind, d. h. beide haben dieselbe Konfiguration und dasselbe Anwenderprogramm.

Variablentabellen zeigen die Variablen. Beobachtungstabellen können auch die Variablen anzeigen, ebenso wie direkte Adressen.

Um die Ausführung des Anwenderprogramms zu beobachten und die Werte der Variablen anzuzeigen, klicken Sie in der Funktionsleiste auf die Schaltfläche "Alle beobachten".

Im Feld "Beobachtungswert" wird für jede Variable der Wert angezeigt.

15.11.2 Zustand im Programmiereditor anzeigen

Sie können den Zustand von bis zu 50 Variablen im KOP- und FUP-Editor beobachten. Rufen Sie den KOP-Editor über die Editorleiste auf. Über die Editorleiste können Sie zwischen den geöffneten Editoren umschalten, ohne die Editoren öffnen oder schließen zu müssen.

Klicken Sie im Programmiereditor in der Funktionsleiste auf die Schaltfläche "Beobachten ein/aus", um den Zustand Ihres Anwenderprogramms anzuzeigen.

Im Netzwerk im Programmiereditor wird der Signalfluss grün dargestellt.

Sie können auch mit der rechten Mastaste auf die Anweisung oder den Parameter klicken, um den Wert der Anweisung zu ändern.

15.11.3 Online-Werte eines DBs erfassen, um die Startwerte zurückzusetzen

Sie können die aktuellen Werte, die in einer Online-CPU beobachtet werden, erfassen und daraus Startwerte für einen globalen DB machen.

- Sie benötigen eine Online-Verbindung zur CPU.
- Die CPU muss sich im Betriebszustand RUN befinden.
- Sie müssen den DB in STEP 7 geöffnet haben.

Mit der Schaltfläche "Momentabbild der beobachteten Werte anzeigen" erfassen Sie die aktuellen Werte der ausgewählten Variablen im DB. Sie können diese Werte dann in die Spalte "Startwert" des DBs kopieren.

1. Klicken Sie im DB-Editor auf die Schaltfläche "Alle Variablen beobachten". Die Spalte "Beobachtungswert" zeigt die aktuellen Datenwerte an.
2. Klicken Sie auf die Schaltfläche "Momentabbild der beobachteten Werte anzeigen", um die aktuellen Werte in der Spalte "Momentabbild" anzuzeigen.
3. Klicken Sie auf die Schaltfläche "Alle beobachten", um die Beobachtung der Daten in der CPU zu beenden.
4. Kopieren Sie einen Wert aus der Spalte "Momentabbild" für eine Variable.
   - Wählen Sie einen zu kopierenden Wert aus.
   - Klicken Sie mit der rechten Maustaste auf den ausgewählten Wert, um das Kontextmenü aufzurufen.
   - Wählen Sie den Befehl "Kopieren".

5. Fügen Sie den kopierten Wert in die entsprechende Spalte "Startwert" der Variablen ein. (Klicken Sie mit der rechten Maustaste auf die Zelle und wählen Sie im Kontextmenü den Befehl "Einfügen".)
6. Speichern Sie das Projekt, um die kopierten Werte als neue Startwerte für den DB zu speichern.
7. Übersetzen Sie den DB und laden Sie ihn in die CPU. Der DB verwendet die neuen Startwerte, nachdem die CPU in den Betriebszustand RUN geht.

Hinweis

Die Werte, die in der Spalte "Beobachtungswert" angezeigt werden, sind immer aus der CPU kopiert. STEP 7 prüft nicht, ob alle Werte aus demselben Zykus der CPU stammen.

15.11.4 Werte in der CPU über die Beobachtungstabelle beobachten und steuern

Mit Hilfe einer Beobachtungstabelle können Sie die Datenpunkte beobachten und steuern, während die CPU Ihr Programm ausführt. Bei diesen Datenpunkten kann es sich je nach Beobachtungs- oder Steuerfunktion um das Prozessabbild (E oder A), um M, DB oder physische Eingänge (E_:P) handeln. Sie können die physischen Ausgänge (A_:P) nicht genau beobachten, weil die Beobachtungsfunktion nur den letzten geschriebenen Wert aus dem Speicherbereich A anzeigen kann und nicht den tatsächlichen Wert aus den physischen Ausgängen liest.

Die Beobachtungsfunktion ändert den Programmablauf nicht. Sie liefert Ihnen Informationen zum Programmablauf und den Daten des Programms in der CPU.

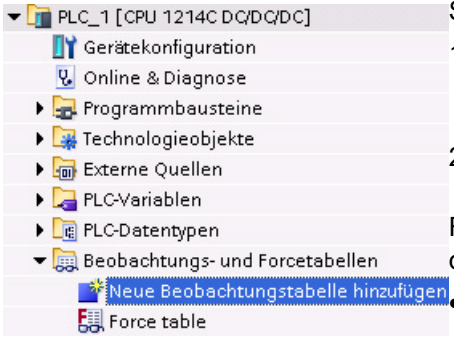
Die Steuerfunktionen ermöglichen es Ihnen, den Ablauf und die Daten des Programms zu steuern. Verwenden Sie die Steuerfunktionen stets mit Vorsicht. Diese Funktionen können die Ausführung des Anwender-/Systemprogramms schwerwiegend beeinflussen. Bei den drei Steuerfunktionen handelt es sich um Steuern, Forcen und Ausgänge in STOP freischalten.

Sie können mit der Beobachtungstabelle die folgenden Online-Funktionen ausführen:

- Status der Variablen beobachten
- Werte der einzelnen Variablen bearbeiten

Sie können auswählen, wann eine Variable beobachtet oder geändert werden soll:

- Zyklusbeginn: Der Wert wird bei Zyklusbeginn gelesen oder geschrieben
- Zyklusende: Der Wert wird bei Zyklusende gelesen oder geschrieben
- Umschalten nach Stop



So erstellen Sie eine Beobachtungstabelle:

1. Öffnen Sie mit Doppelklick auf "Neue Beobachtungstabelle hinzufügen" eine neue Beobachtungstabelle.
2. Geben Sie den Namen einer Variablen ein, die in der Beobachtungstabelle hinzugefügt werden soll.

Für die Beobachtung der Variablen stehen die folgenden Möglichkeiten zur Verfügung:

- Alle beobachten: Mit diesem Befehl wird die Beobachtung der sichtbaren Variablen in der aktiven Beobachtungstabelle gestartet.
- Jetzt beobachten: Mit diesem Befehl wird die Beobachtung der sichtbaren Variablen in der aktiven Beobachtungstabelle gestartet. Die Beobachtungstabelle beobachtet die Variablen sofort und nur einmal.

Zum Ändern der Variablen stehen die folgenden Möglichkeiten zur Verfügung:

- "Steuern auf 0" setzt den Wert der ausgewählten Adresse auf "0".
- "Steuern auf 1" setzt den Wert der ausgewählten Adresse auf "1".
- "Steuern jetzt" ändert den Wert der ausgewählten Adresse sofort und nur für einen Zyklus.
- "Steuern mit Trigger" ändert die Werte für die ausgewählten Adressen.

  Diese Funktion erzeugt keine Rückmeldung, mit der die Änderung der ausgewählten Adressen bestätigt wird. Wird eine Bestätigung der Änderung benötigt, so ist die Funktion "Steuern jetzt" zu verwenden.
- "PA freischalten" deaktiviert den Befehl zur Sperrung von Ausgängen und steht nur zur Verfügung, wenn die CPU im Betriebszustand STOP ist.

Diese Beobachtung der Variablen setzt jedoch eine Online-Verbindung zur CPU voraus.

	i	Name	Adresse	Anzeigeformat	Beobachtungswert	Beobachten mit Trigger	Steuern mit Trigger	Steuerwert	
1		"Start"	%I0.0	Bool		Permanent	Permanent		
2		"Stop"	%I0.1	Bool		Permanent	Permanent		
3		"Running"	%M0.0	Bool		Permanent	Permanent		

Mit den Schaltflächen am oberen Rand der Beobachtungstabelle können Sie die verschiedenen Funktionen auswählen.

Geben Sie den Namen der zu beobachtenden Variablen ein und wählen Sie in der Klappliste ein Anzeigeformat. Besteht eine Online-Verbindung zur CPU, wird durch Anklicken der Schaltfläche "Beobachten" der Istwert des Datenpunkts im Feld "Beobachtungswert" angezeigt.

15.11.4.1 Variablen mit Trigger beobachten oder steuern

Der Trigger legt fest, an welchem Punkt im Zyklus die ausgewählte Adresse beobachtet oder gesteuert wird.

Tabelle 15- 4 Trigger-Arten

Trigger	Beschreibung
Permanent	Die Daten werden ständig erfasst
Zu Beginn des Zyklus	Permanent: Die Daten werden zu Beginn des Zyklus, nachdem die CPU die Eingänge gelesen hat, ständig erfasst
	Einmalig: Die Daten werden zu Beginn des Zyklus, nachdem die CPU die Eingänge gelesen hat, erfasst
Am Ende des Zyklus	Permanent: Die Daten werden am Ende des Zyklus, bevor die CPU in die Ausgänge schreibt, ständig erfasst
	Einmalig: Die Daten werden am Ende des Zyklus, bevor die CPU in die Ausgänge schreibt, erfasst
Beim Wechsel in STOP	Permanent: Die Daten werden ständig erfasst, wenn die CPU in STOP wechselt
	Einmalig: Die Daten werden einmal erfasst, nachdem die CPU in STOP wechselt

Um eine PLC-Variable mit einem bestimmten Trigger zu steuern, wählen Sie entweder den Anfang oder das Ende des Zyklus aus.

- Steuern eines Ausgangs: Das beste Triggerereignis zum Steuern eines Ausgangs ist am Ende des Zyklus, unmittelbar bevor die CPU in die Ausgänge schreibt.

  Beobachten Sie den Wert der Ausgänge am Anfang des Zyklus, um zu ermitteln, welcher Wert in die physischen Ausgänge geschrieben wird. Beobachten Sie auch die Ausgänge, bevor die CPU die Werte in die physischen Ausgänge schreibt, um die Programmlogik zu prüfen und mit dem tatsächlichen E/A-Verhalten zu vergleichen.

- Steuern eines Eingangs: Das beste Triggerereignis zum Steuern eines Eingangs ist am Anfang des Zyklus, unmittelbar nachdem die CPU die Eingänge gelesen hat und bevor das Anwenderprogramm die Eingangswerte nutzt.

  Wenn Sie den Verdacht haben, dass sich Werte während des Zyklus ändern, kann es sinnvoll sein, den Wert der Eingänge am Zyklusende zu überwachen, um sicherzugehen, dass sich die Werte im Zyklusverlauf nicht geändert haben. Unterscheiden sich die Werte, schreibt Ihr Anwenderprogramm möglicherweise in einen falschen Eingang.

Um zu ermitteln, warum die CPU in STOP gegangen ist, erfassen Sie mit dem Trigger "Wechsel in STOP" die letzten Prozesswerte.

15.11.4.2 Ausgänge im Betriebszustand STOP freischalten

Mit der Beobachtungstabelle können Sie in die Ausgänge schreiben, wenn sich die CPU im Betriebszustand STOP befindet. Mithilfe dieser Funktionalität können Sie die Verdrahtung der Ausgänge prüfen und sicherstellen, dass der an eine Ausgangsklemme angeschlossene Draht das Signal 1 oder 0 an die Klemme des angeschlossenen Prozessgeräts weitergibt.

⚠WARNUNG

Gefahren beim Schreiben in physische Ausgänge im Betriebszustand STOP

Auch wenn sich die CPU im Betriebszustand STOP befindet, kann die Freischaltung eines physischen Ausgangs den daran angeschlossenen Prozesspunkt aktivieren und zu unerwünschtem Verhalten der Geräte führen. Dies kann tödliche oder schwere Verletzungen zur Folge haben.

Vor dem Schreiben in einen Ausgang in der Beobachtungstabelle ist daher sicherzustellen, dass eine Änderung des physischen Ausgangswerts kein unerwünschtes Verhalten von Geräten hervorrufen kann. Beachten Sie immer die Sicherheitsvorkehrungen für Ihre Prozessanlagen.

Sie können den Zustand der Ausgänge im Betriebszustand STOP ändern, wenn die Ausgänge aktiviert sind. Sind die Ausgänge deaktiviert, können Sie die Ausgänge im Betriebszustand STOP nicht steuern. Gehen Sie wie folgt vor, um die Änderung von Ausgangswerten aus der Beobachtungstabelle im Betriebszustand STOP zu ermöglichen:

1. Wählen Sie im Menü "Online" den Menübefehl "Erweiterter Modus".
2. Wählen Sie im Menü "Online" den Befehl "Steuern" und dann die Option "PA freischalten" oder klicken Sie im Kontextmenü mit der rechten Maustate auf die entsprechende Zeile der Beobachtungstabelle.

Wenn Sie dezentrale Peripherie konfiguriert haben, können Sie in der Betriebsart STOP keine Ausgänge aktivieren. Versuchen Sie es trotzdem, wird ein Fehler gemeldet.

Wenn die CPU in den Betriebszustand RUN wechselt, wird die Option "PA freischalten" deaktiviert.

Wenn Eingänge oder Ausgänge geforct sind, kann die CPU im Betriebszustand STOP keine Ausgänge aktivieren. Die Force-Funktion muss zunächst beendet werden.

15.11.5 Werte in der CPU forcen

15.11.5.1 Arbeiten mit der Forcetabelle

Eine Forcetabelle bietet die Funktion "Forcen", die den Wert eines Eingangs oder Ausgangs zwangsweise auf einen vorgegebenen Wert für die Adresse des Peripherieeingangs bzw. -ausgangs setzt. Das Forcen wird im Prozessabbild der Eingänge vor der Ausführung des Anwenderprogramms und im Prozessabbild der Ausgänge vor dem Schreiben der Ausgänge in die Module durchgeführt.

Hinweis

Die Forcewerte werden in der CPU und nicht in der Forcetabelle gespeichert.

Sie können keinen Eingang (Adresse "E") oder Ausgang (Adresse "A") forcen. Sie können jedoch einen Peripherieeingang oder einen Peripherieausgang forcen. Die Forcetabelle hängt automatisch ein ":P" an die Adresse an (Beispiel: "On":P oder "Run":P).

	i	Name	Adresse	Anzeigeformat	Beobachtungswert	Forcewert	F
1		"On":P	%I0.0:P	Bool		TRUE	
2		"Off":P	%I0.1:P	Bool			
3		"Run":P	%Q0.1:P	Bool			

Geben Sie den Wert für den zu forcenden Eingang oder Ausgang in die Zelle "Forcewert" ein. Sie können dann das Kontrollkästchen in der Spalte "Forcen" aktivieren, um das Forcen des Eingangs oder Ausgangs zu aktivieren.

Klicken Sie auf die Schaltfläche "Forcen starten oder ersetzen", um die Werte der Variablen in der Forcetabelle zu forcen. Klicken Sie auf die Schaltfläche "Forcen beenden", um die Werte der Variablen zurückzusetzen.

In der Forcetabelle können Sie den Status des geforcten Werts eines Eingangs beobachten. Den geforcten Wert eines Ausgangs können Sie jedoch nicht beobachten.

Sie können den Zustand der geforcten Werte auch im Programmiereditor anzeigen.

Hinweis

Wenn ein Eingang oder Ausgang in einer Forcetabelle geforct wird, werden die Forceaktionen Teil der Projektkonfiguration. Beim Schließen von STEP 7 bleiben die geforcten Elemente im CPU-Programm so lange aktiv, bis sie gelöscht werden. Um diese geforcten Elemente zu löschen, müssen Sie über STEP 7 eine Verbindung zur Online-CPU herstellen und dann mithilfe der Forcetabelle die Forcefunktion für diese Elemente deaktivieren oder stoppen.

15.11.5.2 Funktionsweise der Forcefunktion

Die CPU gestattet Ihnen das Forcen von Eingängen und Ausgängen, indem Sie in der Forcetabelle die Adresse der physischen Eingänge und Ausgänge (E_:P oder A_:P) angeben und dann die Forcefunktion starten.

Im Programm werden die gelesenen Werte der physischen Eingänge durch den Forcewert überschrieben. Das Programm nutzt den geforcten Wert während der Bearbeitung. Wenn das Programm in einen physischen Ausgang schreibt, wird der Ausgangswert durch den Forcewert überschrieben. Der geforcte Wert erscheint am physischen Ausgang und wird im Prozess verwendet.

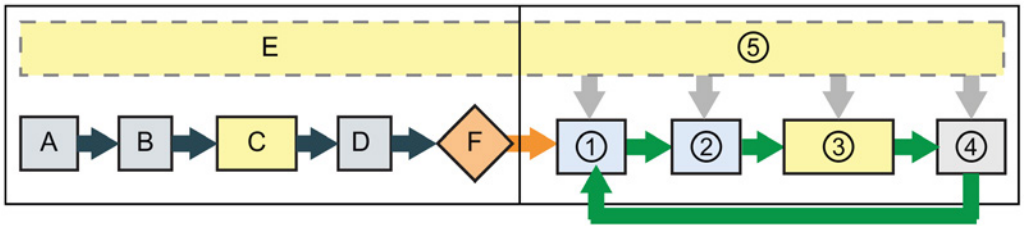
Wenn ein Eingang oder Ausgang in der Forcetabelle geforct wird, werden die Forceaktionen Teil des Anwenderprogramms. Auch wenn Sie die Programmiersoftware schließen, bleiben die geforcten Werte im ausgeführten CPU-Programm solange aktiv, bis Sie wieder in der Programmiersoftware online gehen und die Forcefunktion beenden. Programme mit geforcten Ein- und Ausgängen, die von einer Memory Card in eine andere CPU geladen werden, forcen auch weiterhin die im Programm ausgewählten Ein- und Ausgänge.

Wenn die CPU das Anwenderprogramm auf einer schreibgeschützten Memory Card ausführt, können Sie das Forcen von E/A nicht über eine Beobachtungstabelle auslösen oder ändern, weil Sie die Werte in dem schreibgeschützten Anwenderprogramm nicht überschreiben können. Jeder Versuch, die schreibgeschützten Werte zu forcen, führt zu einem Fehler. Bei Verwendung einer Memory Card zum Übertragen eines Anwenderprogramms werden auf dieser Memory Card gespeicherte geforcte Elemente mit an die CPU übertragen.

Hinweis

Zu HSC, PWM und PTO zugewiesene digitale E/A können nicht geforct werden

Die vom schnellen Zähler (HSC), von der Impulsdauermodulation (PWM) und von der Impulsfolge (PTO) verwendeten E/A werden während der Konfiguration zugewiesen. Wenn diesen Funktionen digitale E/A zugewiesen werden, können die Werte der Adressen der zugewiesenen E/A nicht durch die Forcefunktion der Forcetabelle geändert werden.



Anlauf

A Das Löschen des Speicherbereichs E wird von der Forcefunktion nicht beeinflusst.

B Die Initialisierung der Ausgangswerte wird von der Forcefunktion nicht beeinflusst.

C Während der Ausführung der Anlauf-OBs schaltet die CPU den Forcewert auf, wenn das Anwenderprogramm auf den physischen Eingang zugreift.

D Das Speichern von Alarmereignissen in der Warteschlange wird nicht beeinflusst.

E Die Freigabe des Schreibens in die Ausgänge wird nicht beeinflusst.

RUN

① Beim Schreiben von A-Speicher in die physischen Ausgänge schaltet die CPU den Forcewert bei der Aktualisierung der Ausgänge auf.

② Beim Lesen der physischen Eingänge wendet die CPU die Forcewerte an, kurz bevor die Eingänge in den Speicherbereich E kopiert werden.

③ Während der Ausführung der Anwenderprogramms (Programmzyklus-OBs) schaltet die CPU den Forcewert auf, wenn das Anwenderprogramm auf den physischen Eingang zugreift oder in den physischen Ausgang schreibt.

④ Die Behandlung von Kommunikationsanforderungen und die Selbsttestdiagnose werden von der Forcefunktion nicht beeinflusst.

⑤ Die Verarbeitung von Alarmen während eines beliebigen Teils des Zyklus wird nicht beeinflusst.

15.12 Laden im Betriebszustand RUN

Die CPU unterstützt das "Laden im Betriebszustand RUN". Diese Funktion soll Ihnen ermöglichen, kleinere Änderungen am Anwenderprogramm vorzunehmen, ohne den vom Programm gesteuerten Prozess zu stören. Diese Funktion ermöglicht jedoch auch größere Programmänderungen, die den Prozess beeinträchtigen oder sogar gefährlich werden können.

⚠WARNUNG

Risiken beim Laden im Betriebszustand RUN

Wenn Sie im Betriebszustand RUN Änderungen in die CPU laden, wirken sich die Änderungen sofort auf den Prozess aus. Wenn Sie das Programm im Betriebszustand RUN ändern, kann dies zu unerwartetem Verhalten im Prozess führen und Tod, schwere Körperverletzungen und/oder Sachschaden können die Folge sein.

Nur dazu befugtes Personal mit Kenntnis der Auswirkungen einer Programmbearbeitung in RUN auf das Prozessverhalten darf einen Ladevorgang im Betriebszustand RUN durchführen.

Die Funktion zum "Laden im Betriebszustand RUN" ermöglicht Ihnen, Änderungen an einem Programm vorzunehmen und sie in die CPU zu laden, ohne nach STOP wechseln zu müssen.

- Sie können kleinere Änderungen am aktuellen Prozess vornehmen (z. B. eine Parameterwertänderung), ohne den Prozess herunterfahren zu müssen.
- Außerdem können Sie mit dieser Funktion Programmfehler schneller beheben (z. B. die Logik für einen Schließerkontakt oder Öffnerkontakt invertieren).

Sie können die folgenden Änderungen an Programmbausteinen und Variablen vornehmen und sie im Betriebszustand RUN laden:

- Funktionen (FCs), Funktionsbausteine (FBs) und Variablentabellen erstellen, überschreiben und löschen
- Datenbausteine (DBs) und Instanz-Datenbausteine für Funktionsbausteine (FBs) erstellen, löschen und überschreiben. Sie können DB-Strukturen hinzufügen und im Betriebszustand RUN laden. Je nach Ihren Konfigurationseinstellungen (Seite 1144) kann die CPU die Werte vorhandener Bausteinvariablen speichern und die neuen Datenbausteinvariablen mit den Anfangswerten initialisieren oder alle Datenbausteinvariablen auf die Anfangswerte zurücksetzen. Im Betriebszustand RUN können Sie keinen Webserver-DB (Steuerung oder Fragment) laden.
- Organisationsbausteine (OBs) überschreiben; es können jedoch keine OBs erstellt oder gelöscht werden.

Sie können maximal zwanzig Bausteine gleichzeitig im Betriebszustand RUN laden. Müssen mehr als zwanzig Bausteine geladen werden, ist die CPU in STOP zu setzen.

Wenn Sie Änderungen in einen realen Prozess laden (im Unterschied zu einem simulierten Prozess wie z. B. bei der Fehlerbehebung in einem Programm), sollten Sie vor dem Laden unbedingt in Gedanken die möglichen Folgen für die Sicherheit der Maschinen und Maschinenbediener durchspielen.

Hinweis

Befindet sich die CPU im Betriebszustand RUN und wurden Programmänderungen vorgenommen, versucht STEP 7 zunächst stets, die Änderungen im Betriebszustand RUN zu laden. Möchten Sie dies verhindern, müssen Sie die CPU in STOP versetzen.

Werden die vorgenommenen Änderungen nicht von der Funktion "Laden im Betriebszustand RUN" unterstützt, werden Sie von STEP 7 aufgefordert, die CPU in den Betriebszustand STOP zu versetzen.

15.12.1 Voraussetzungen für "Laden im Betriebszustand RUN"

Sie können Ihre Programmänderungen nur dann im Betriebszustand RUN in die CPU laden, wenn die folgenden Voraussetzungen erfüllt sind:

- Ihre CPU-Version weist mindestens V3.0 auf

  Hinweis

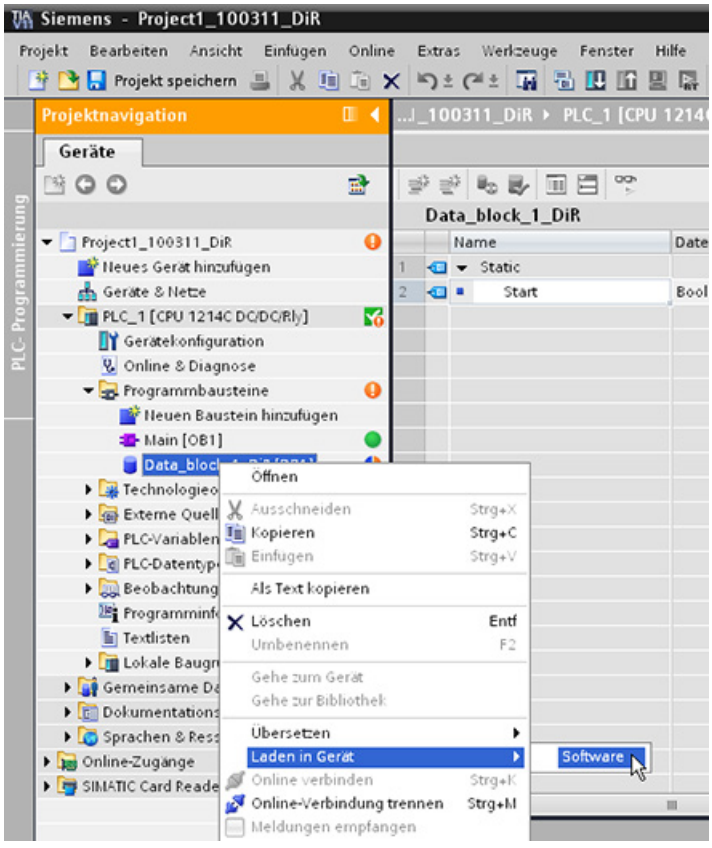
  Um vorhandene Bausteine zu ändern und die erweiterte Bausteinschnittstelle in RUN zu laden (Seite 1144), benötigen Sie eine CPU-Version von V4.0 oder höher.

- Ihr Programm muss sich erfolgreich übersetzen lassen.
- Sie müssen die Kommunikation zwischen dem Programmiergerät, auf dem STEP 7 läuft, und der CPU erfolgreich aufgebaut haben.

15.12.2 Ändern des Programms im Betriebszustand RUN

Um Programmänderungen in RUN vorzunehmen, vergewissern Sie sich zuerst, dass die CPU und das Programm die Voraussetzungen (Seite 1139) erfüllen, und gehen Sie dann wie folgt vor:

1. Um Ihr Programm im Betriebszustand RUN zu laden, gehen Sie auf eine der folgenden Arten vor:
   - Wählen Sie im Menü "Online" den Befehl "Laden in Gerät".
   - Klicken Sie in der Symbolleiste auf die Schaltfläche "Laden in Gerät".
   - Klicken Sie in der "Projektnavigation" mit der rechten Maustaste auf "Programmbausteine" und wählen Sie den Befehl "Laden in Gerät > Software".



   Wenn das Programm erfolgreich übersetzt wird, lädt STEP 7 das Programm in die CPU.

2. Wenn STEP 7 Sie auffordert, das Programm zu laden oder den Vorgang abzubrechen, klicken Sie auf "Laden", um das Programm in die CPU zu laden.

15.12.3 Ausgewählte Bausteine laden

Im Ordner "Programmbausteine" können Sie einen einzelnen oder mehrere Bausteine zum Laden auswählen.

Wenn Sie einen einzelnen Baustein zum Laden auswählen, wird in der Spalte "Aktion" als einzige Option "Konsistent laden" angezeigt.
Sie können die Kategoriezeile erweitern, um zu prüfen, welche Bausteine geladen werden müssen. In diesem Beispiel wurde eine geringfügige Änderung am Offline-Baustein vorgenommen und es müssen keine anderen Bausteine geladen werden.

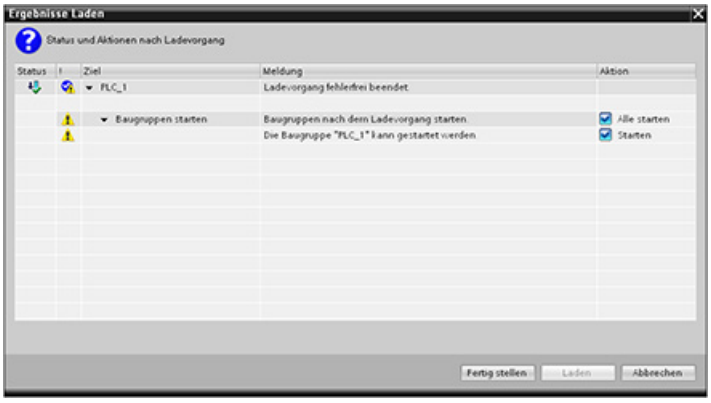
In diesem Beispiel müssen mehrere Bausteine geladen werden.

Hinweis

Sie können maximal zwanzig Bausteine gleichzeitig im Betriebszustand RUN laden. Müssen mehr als zwanzig Bausteine geladen werden, ist die CPU in STOP zu setzen.

Wenn Sie versuchen, die Bausteine in RUN zu laden, doch das System erkennt vor dem tatsächlichen Ladevorgang, dass dies nicht möglich ist, erscheint im Dialog eine Zeile zum Stoppen der Module.

Klicken Sie auf die Schaltfläche "Laden". Daraufhin wird der Dialog "Ladeergebnisse" angezeigt. Klicken Sie auf die Schaltfläche "Fertig stellen", um den Ladevorgang abzuschließen.



15.12.4 Einen einzelnen ausgewählten Baustein mit einem Übersetzungsfehler in einem anderen Baustein laden

Wenn Sie versuchen, einen konsistenten Ladevorgang mit einem Übersetzungsfehler in einem anderen Baustein durchzuführen, zeigt der Dialog einen Fehler an und die Schaltfläche zum Laden wird deaktiviert.

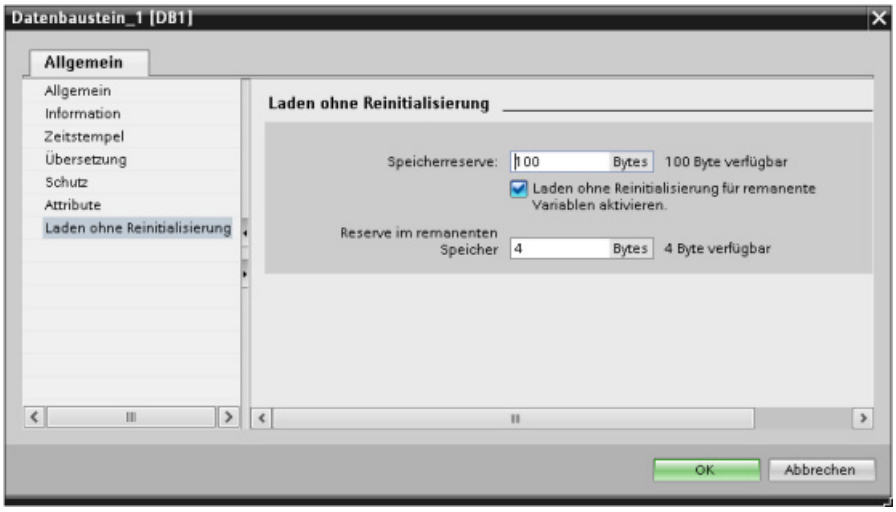
Sie müssen zunächst den Übersetzungsfehler in dem anderen Baustein korrigieren. Danach wird die Schaltfläche "Laden" wieder aktiviert.

15.12.5 Bestehende Bausteine im Betriebszustand RUN ändern und ins Zielsystem laden

Mit der Funktion zum Laden im Betriebszustand RUN können Sie Variablen in Datenbausteinen und Funktionsbausteinen hinzufügen und ändern und dann den geänderten Baustein im Betriebszustand RUN in die CPU laden.

Laden ohne Reinitialisierung

Jeder DB und FB hat eine Speicherreserve, die Sie nutzen können, um dem Baustein Variablen hinzuzufügen und ihn nachfolgend im Betriebszustand RUN in die CPU zu laden. Standardmäßig beträgt die Anfangsgröße der Speicherreserve 100 Byte. Sie können im Rahmen der Größe der Speicherreserve zusätzliche Variablen zu Ihren Daten hinzufügen und den erweiterten Baustein im Betriebszustand RUN in die CPU laden. Sie können die Speicherreserve auch vergrößern, wenn Sie mehr Speicher für zusätzliche Variablen in Ihrem Baustein benötigen. Wenn Sie mehr Variablen hinzufügen als der Speicher aufnehmen kann, können Sie den erweiterten Baustein nicht im Betriebszustand RUN in die CPU laden.



Mit der Funktion "Laden ohne Reinitialisierung" können Sie einen Datenbaustein erweitern, indem Sie mehr Datenbausteinvariablen hinzufügen und den erweiterten Datenbaustein im Betriebszustand RUN in die CPU laden. Auf diese Weise können Sie einem Datenbaustein Variablen hinzufügen und den DB ohne Reinitialisierung in Ihr Programm laden. Die CPU speichert die Werte der bestehenden Datenbausteinvariablen und initialisiert die neu hinzugefügten Variablen mit ihren Startwerten.

Um diese Funktion für ein Online-Projekt mit der CPU im Betriebszustand RUN zu aktivieren, gehen Sie wie folgt vor:

1. Öffnen Sie im Ordner der Programmbausteine in der STEP 7-Projektnavigation den Baustein.
2. Klicken Sie im Bausteineditor auf die Schaltfläche "Laden ohne Reinitialisierung", um die Funktion zu aktivieren. (Das Symbol ist im aktivierten Zustand umrahmt:)
3. Bestätigen Sie Ihre Auswahl mit "OK".
4. Fügen Sie der Bausteinschnittstelle Variablen hinzu und laden Sie den Baustein im Betriebszustand RUN. Sie können so viele neue Variablen hinzufügen und laden, wie Ihre Speicherreserve zulässt.

Wenn Sie Ihrem Baustein mehr Bytes hinzugefügt haben, als Sie für die Speicherreserve konfiguriert haben, zeigt STEP 7 einen Fehler an, sobald Sie versuchen, den Baustein im Betriebszustand RUN zu laden. Sie müssen die Bausteineigenschaften bearbeiten und die Speicherreserve vergrößern. Solange die Funktion "Laden ohne Reinitialisierung" aktiviert ist, können Sie keine bestehenden Einträge löschen und die Speicherreserve des Bausteins nicht ändern. Um die Funktion "Laden ohne Reinitialisierung" zu deaktivieren, gehen Sie folgendermaßen vor:

1. Klicken Sie im Bausteineditor auf die Schaltfläche "Laden ohne Reinitialisierung", um die Funktion zu deaktivieren. (Das Symbol ist im deaktivierten Zustand nicht umrahmt:)
2. Bestätigen Sie Ihre Auswahl mit "OK".
3. Laden Sie den Baustein in die CPU. Im Dialogfeld zum Laden müssen Sie "Reinitialisieren" auswählen, um den erweiterten Baustein zu laden.

Beim Laden werden dann alle bestehenden und neuen Bausteinvariablen mit ihren Startwerten reinitialisiert.

Remanente Bausteinvariablen laden

Um remanente Bausteinvariablen im Betriebszustand RUN laden zu können, ist die Zuordnung einer remanenten Speicherreserve erforderlich. Zum Konfigurieren dieser remanenten Speicherreserve gehen Sie folgendermaßen vor:

1. Klicken Sie im Ordner der Programmbausteine in der STEP 7-Projektnavigation mit der rechten Maustaste auf den Baustein und wählen Sie im Kontextmenü "Eigenschaften" aus.
2. Wählen Sie die Eigenschaft "Laden ohne Reinitialisierung" aus.
3. Aktivieren Sie das Kontrollkästchen für "Laden ohne Reinitialisierung für remanente Variablen aktivieren".
4. Konfigurieren Sie die für die remanente Speicherreserve verfügbare Anzahl von Bytes.
5. Bestätigen Sie Ihre Änderungen mit "OK".
6. Fügen Sie dem Datenbaustein remanente Datenbausteinvariablen hinzu und laden Sie den Datenbaustein im Betriebszustand RUN. Sie können so viele neue remanente Datenbausteinvariablen hinzufügen und laden, wie Ihre remanente Speicherreserve zulässt.

Wenn Sie Ihrem Datenbaustein mehr remanente Bytes hinzugefügt haben, als Sie für die Speicherreserve konfiguriert haben, zeigt STEP 7 einen Fehler an, sobald Sie versuchen, den Baustein im Betriebszustand RUN zu laden. Sie können remanente Bausteinvariablen nur im Rahmen der Größe der remanenten Speicherreserve hinzufügen, damit Sie die Variablen im Betriebszustand RUN laden können.

Wenn Sie die erweiterten remanenten Bausteinvariablen laden, enthalten die Variablen ihre aktuellen Werte.

Größe der Speicherreserve für neue Bausteine konfigurieren

Die Standardgröße der Speicherreserve für neue Datenbausteine beträgt 100 Byte. Wenn Sie einen neuen Baustein erstellen, umfasst dieser eine Reserve von 100 Byte. Wenn Sie neue Bausteine mit einer Speicherreserve in einer anderen Größe erstellen möchten, können Sie die Einstellung in den PLC-Programmiereinstellungen ändern:

1. Wählen Sie in STEP 7 den Menübefehl **Optionen > Einstellungen** aus.
2. Erweitern Sie im Dialogfeld "Einstellungen" den Bereich "PLC-Programmierung" und wählen Sie "Allgemein".
3. Geben Sie im Abschnitt "Laden ohne Reinitialisierung" die Anzahl von Bytes für die Speicherreserve ein.

Beim Erstellen neuer Bausteine verwendet STEP 7 die von Ihnen für neue Bausteine konfigurierte Speicherreserve.

Einschränkungen

Die folgenden Einschränkungen gelten für das Bearbeiten und Laden von Bausteinen im Betriebszustand RUN:

- Das Erweitern der Bausteinschnittstelle und das Hinzufügen neuer Variablen und das Laden im Betriebszustand RUN sind nur bei optimierten Bausteinen (Seite 186) möglich.
- Ohne Reinitialisierung können Sie nicht die Struktur eines Bausteins ändern und den geänderten Baustein im Betriebszustand RUN laden. Für das Hinzufügen neuer Elemente zu einer Struct (Seite 128)-Variable, für das Ändern von Variablennamen, Array-Größen, Datentypen oder des remanenten Zustands ist es erforderlich, den Baustein beim Laden im Betriebszustand RUN zu reinitialisieren. Die einzigen Änderungen an bestehenden Bausteinvariablen, die Sie durchführen und trotzdem den Baustein ohne Reinitialisierung in RUN laden können, sind Änderungen an Startwerten (Datenbausteine), Standardwerten (Funktionsbausteine) oder Kommentaren.

  Hinweis

  Die Möglichkeit, Bausteine im Betriebszustand RUN zu ändern und zu laden, ist ab der Version 4.0 der S7-1200 CPU neu. Vor V4.0 konnten Sie geänderte Bausteine lediglich im Betriebszustand STOP laden.

- Sie können maximal so viele neue Bausteinvariablen im Betriebszustand RUN laden, wie die Speicherreserve aufnehmen kann.
- Sie können maximal so viele neue remanente Bausteinvariablen im Betriebszustand RUN laden, wie die remanente Speicherreserve aufnehmen kann.

Siehe auch

Geräteaustausch und Ersatzteilkompatibilität (Seite 1349)

15.12.6 Systemreaktion bei fehlgeschlagenem Ladevorgang

Tritt während des ersten Ladevorgangs in RUN ein Netzwerkverbindungsfehler auf, zeigt STEP 7 den im Folgenden abgebildeten Dialog "Vorschau laden" an:

15.12.7 Sicherheitsaspekte beim Laden im Betriebszustand RUN

Bevor Sie das Programm im Betriebszustand RUN laden, bedenken Sie die Auswirkungen Ihrer im Betriebszustand RUN vorgenommenen Änderungen auf den Betrieb der CPU in folgenden Fällen:

- Wenn Sie die Steuerungslogik für einen Ausgang gelöscht haben, behält die CPU den letzten Zustand des Ausgangs bei, bis die CPU ausgeschaltet oder in den Betriebszustand STOP versetzt wird.
- Wenn Sie einen schnellen Zähler oder eine Impulsausgabe gelöscht haben und eine der beiden Funktionen in Betrieb ist, läuft der schnelle Zähler bzw. die Impulsausgabe bis zum nächsten Ausschalten bzw. bis zum nächsten Übergang in STOP weiter.
- Logik, die durch den Merker des ersten Zyklus aktiviert wird, wird erst nach dem nächsten Einschalten bzw. nach dem nächsten Wechsel von STOP nach RUN ausgeführt. Der Merker des ersten Zyklus wird nur durch den Wechsel in den Betriebszustand RUN gesetzt und wird nicht durch das Laden im Betriebszustand RUN beeinflusst.
- Die aktuellen Werte von Datenbausteinen (DBs) und/oder Variablen können überschrieben werden.

Hinweis

Damit Sie Ihr Programm im Betriebszustand RUN laden können, muss die CPU das Laden im Betriebszustand RUN unterstützen. Das Programm muss fehlerfrei übersetzt werden und die Kommunikation zwischen STEP 7 und der CPU muss fehlerfrei sein.

Sie können die folgenden Änderungen an Programmbausteinen und Variablen vornehmen und sie im Betriebszustand RUN laden:

- Funktionen (FCs), Funktionsbausteine (FBs) und Variablentabellen erstellen, überschreiben und löschen
- Datenbausteine (DBs) erstellen und löschen; DB-Strukturänderungen können jedoch nicht überschrieben werden. Ausgangswerte des DBs können überschrieben werden. Im Betriebszustand RUN können Sie keinen Webserver-DB (Steuerung oder Fragment) laden.
- Organisationsbausteine (OBs) überschreiben; es können jedoch keine OBs erstellt oder gelöscht werden.

Sie können maximal zwanzig Bausteine gleichzeitig im Betriebszustand RUN laden. Müssen mehr als zwanzig Bausteine geladen werden, ist die CPU in STOP zu setzen.

Sobald Sie einen Ladevorgang gestartet haben, können Sie erst dann weitere Vorgänge in STEP 7 durchführen, wenn der Ladevorgang beendet ist.

Anweisungen, die beim "Laden im Betriebszustand RUN" fehlschlagen können

Die folgenden Anweisungen melden möglicherweise kurzzeitige Fehler, wenn Änderungen aus dem Laden in RUN in der CPU aktiviert werden. Der Fehler tritt auf, wenn die Anweisung eingeleitet wird, während die CPU die Aktivierung der geladenen Änderungen vorbereitet. Während dieses Zeitraums unterbricht die CPU den weiteren Zugriff des Anwenderprogramms auf den Ladespeicher, solange sie noch in Bearbeitung befindliche Zugriffe des Anwenderprogramms auf den Ladespeicher abarbeitet. Dies geschieht, damit geladene Änderungen konsistent aktiviert werden können.

Anweisung	Reaktion, solange die Aktivierung ansteht
DataLogCreate	STATUS = W#16#80C0, ERROR = WAHR
DataLogOpen	STATUS = W#16#80C0, ERROR = WAHR
DataLogWrite	STATUS = W#16#80C0, ERROR = WAHR
DataLogClose	STATUS = W#16#80C0, ERROR = WAHR
DataLogNewFile	STATUS = W#16#80C0, ERROR = WAHR
READ_DBL	RET_VAL = W#16#82C0
WRIT_DBL	RET_VAL = W#16#82C0
RTM	RET_VAL = 0x80C0

In allen Fällen ist das VKE der Anweisung falsch, wenn der Fehler auftritt. Der Fehler ist temporär. Wenn er auftritt, führen Sie die Anweisung zu einem späteren Zeitpunkt aus.

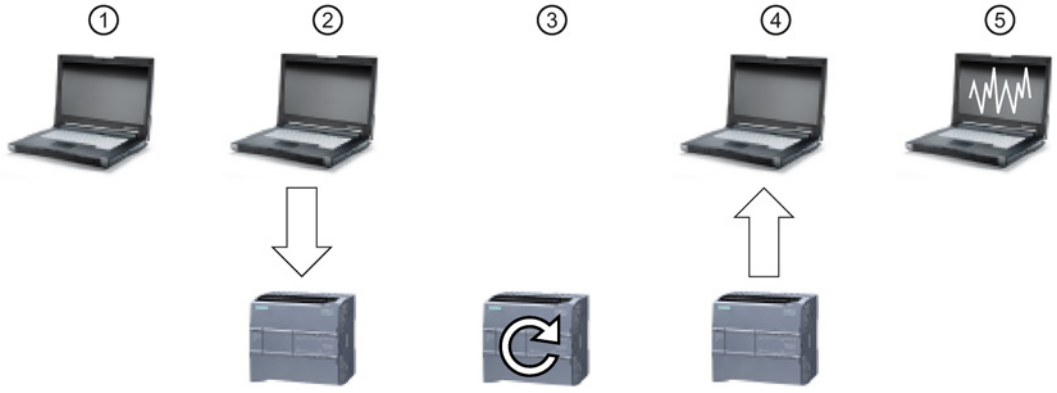
Hinweis

Versuchen Sie nicht, die Anweisung während der aktuellen Bearbeitung des OBs auszuführen.

15.13 CPU-Daten bei Auslösebedingungen verfolgen und aufzeichnen

STEP 7 bietet Trace- und Logic-Analyzer-Funktionen, mit denen Sie Variablen für die PLC konfigurieren können, die Sie verfolgen und aufzeichnen wollen. Die aufgezeichneten Trace-Daten können Sie dann in Ihr Programmiergerät laden und mit den STEP 7-Tools analysieren, verwalten und grafisch aufbereiten. In Ordner "Traces" in der STEP 7-Projektnavigation können Sie Traces anlegen und verwalten.

Die folgende Abbildung zeigt den Ablauf der Trace-Funktion:



① Richten Sie die Trace-Funktion im Trace-Editor von STEP 7 ein. Sie können die aufzuzeichnenden Datenwerte, die Aufzeichnungsdauer und -häufigkeit sowie die Auslösebedingung festlegen.

② Übertragen Sie die Trace-Konfiguration von STEP 7 in die PLC.

③ Die PLC führt das Programm aus und startet die Aufzeichnung der Trace-Daten, sobald die Auslösebedingung eintritt.

④ Übertragen Sie die aufgezeichneten Werte von der PLC in STEP 7.

⑤ Mit den Tools in STEP 7 können Sie die Daten analysieren, grafisch darstellen und speichern.

Die maximale Trace-Größe beträgt 512 kBytes pro Trace.

Zugriff auf Beispiele

Einzelheiten zur Trace-Programmierung, zum Laden der Konfiguration, Hochladen der Trace-Daten und Darstellung der Daten im Logic Analyzer finden Sie im Informationssystem von STEP 7. Kapitel "Online- und Diagnosefunktionen nutzen" > "Trace- und Logikanalysatorfunktion nutzen" enthält ausführliche Beispiele.

Außerdem ist auch das Online-Handbuch "Industry Automation SINAMICS/SIMATIC Trace- und Logikanalysatorfunktion nutzen" (http://support.automation.siemens.com/WW/view/de/64897128) eine ausgezeichnete Hilfe.

Technische Daten

A

A.1 Allgemeine technische Daten

Erfüllte Normen

Der Aufbau des Automatisierungssystems S7-1200 erfüllt die folgenden Normen und Prüfvorschriften. Die Prüfkriterien für S7-1200 beruhen auf diesen Normen und Prüfvorschriften.

Beachten Sie, dass möglicherweise nicht alle S7-1200 Varianten nach diesen Normen zertifiziert sind und dass sich der Zertifizierungszustand ohne Ankündigung ändern kann. Es liegt in der Verantwortung des Anwenders, geltende Zertifizierungen anhand der auf dem Produkt angebrachten Zulassungen zu ermitteln. Wenden Sie sich an Ihre Siemens-Vertretung, wenn Sie eine Liste mit den aktuellen Zulassungen für die einzelnen Bestellnummern benötigen.

CE-Zulassung

CE

Das Automatisierungssystem S7-1200 erfüllt die Anforderungen und sicherheitsrelevanten Ziele der folgenden EU-Richtlinien und entspricht den harmonisierten europäischen Normen (EN) für speicherprogrammierbare Steuerungen, die in den Amtsblättern der EU aufgeführt sind.

- EU-Richtlinie 2006/95/EG (Niederspannungs-Richtlinie) "Elektrische Betriebsmittel für die Verwendung innerhalb bestimmter Spannungsgrenzen"
  - EN 61131-2:2007 Speicherprogrammierbare Steuerungen - Betriebsmittelanforderungen und Prüfungen
- EU-Richtlinie 2004/108/EG (EMV-Richtlinie) "Elektromagnetische Verträglichkeit"
  - Störaussendung
    EN 61000-6-4:2007+A1:2011: Industriebereich
  - Funkentstörung
    EN 61000-6-2:2005: Industriebereich
- EG-Richtlinie 94/9/EG (ATEX) "Geräte und Schutzsysteme zur bestimmungsgemäßen Verwendung in explosionsgefährdeten Bereichen"
  - EN 60079-15:2010: Schutzart 'n':

Die CE-Konformitätserklärung steht allen zuständigen Behörden zur Verfügung bei der:

Siemens AG
Sector Industry
I IA AS FA DH AMB
Postfach 1963
D-92209 Amberg

cULus-Zulassung

Underwriters Laboratories, Inc. erfüllt:

- Underwriters Laboratories, Inc.: UL 508 Listed (Industriesteuerungsgeräte)
- Canadian Standards Association: CSA C22.2 Nummer 142 (Prozesssteuerungsgeräte)

Hinweis

Die Produktreihe SIMATIC S7-1200 entspricht der CSA-Norm.

Das cULus-Zeichen zeigt an, dass die S7-1200 von Underwriters Laboratories (UL) nach den Normen UL 508 und CSA 22.2 Nr. 142 geprüft und zugelassen wurde.

FM-Zertifizierung

Factory Mutual Research (FM)
Zertifizierungsnorm Klasse Nummer 3600 und 3611
Zugelassen für den Einsatz in:
Class I, Division 2, Gas Group A, B, C, D, Temperature Class T3C Ta = 60 °C
Class I, Zone 2, IIC, Temperature Class T3 Ta = 60 °C
Canadian Class I, Zone 2 Installation nach CEC 18-150

WICHTIGE AUSNAHME: Die Anzahl von Eingängen und Ausgängen, die gleichzeitig eingeschaltet sein dürfen, finden Sie in den technischen Daten. Einige Modelle sind auf Ta = 60 °C herabgesetzt.

⚠WARNUNG

Der Austausch von Komponenten kann die Eignung für Class I, Division 2 und Zone 2 beeinträchtigen.

Reparatur von Geräten darf nur von einem autorisierten Siemens Service Center durchgeführt werden.

IECEx-Zulassung

EN 60079-0: Explosionsfähige Atmosphäre – Allgemeine Anforderungen

EN60079-15: Elektrische Betriebsmittel für gasexplosionsgefährdete Bereiche;
Schutzart 'nA'
IECEX FMG14.0012X
Ex nA IIC Tx Gc

Die IECEx-Zulassung ist möglicherweise mit der FM-Zulassung für Gefahrenbereiche auf dem Produkt angegeben.

Nur Produkte mit IECEx-Kennzeichnung sind zugelassen. Wenden Sie sich an Ihre Siemens-Vertretung, wenn Sie eine Liste mit den aktuellen Zulassungen für die einzelnen Bestellnummern benötigen.

Relaismodelle haben keine IECEx-Zulassungen.

Die Temperaturauslegung finden Sie bei der spezifischen Produktbezeichnung.

Bauen Sie die Module in einem geeigneten Gehäuse mit einer Schutzart von mindestens IP54 nach IEC 60079-15 ein.

ATEX-Zulassung

Die ATEX-Zulassung gilt nur für DC-Varianten. Die ATEX-Zulassung gilt nicht für AC- und Relaisvarianten.

EN 60079-0:2009: Explosionsfähige Atmosphäre - Allgemeine Anforderungen

EN 60079-15:2010: Elektrische Betriebsmittel für gasexplosionsgefährdete Bereiche;
Schutzart 'nA'
II 3 G Ex nA IIC T4 oder T3 Gc

Bauen Sie die Module in einem geeigneten Gehäuse mit einer Schutzart von mindestens IP54 nach EN 60529 oder an einem Standort mit ähnlicher Schutzart ein.

Die angeschlossenen Kabel und Leitungen müssen unter den Nennbedingungen für die tatsächlich gemessene Temperatur ausgelegt sein.

Im Aufbau muss gewährleistet sein, dass kurzzeitige Störspannungen (Transienten) auf weniger als 119 V begrenzt sind. Siehe "Stoßwellenfestigkeit" in diesem Abschnitt.

WICHTIGE AUSNAHME: Die Anzahl der Eingänge und Ausgänge, die gleichzeitig eingeschaltet sein dürfen, entnehmen Sie den technischen Daten. Einige Modelle sind auf Ta = 60 °C herabgesetzt.

C-Tick-Zulassung

Das Automatisierungssystem S7-1200 erfüllt die Anforderungen der Normen nach AS/NZS CISPR16 (Klasse A).

Koreanische Zertifizierung

Das Automatisierungssystem S7-1200 erfüllt die Anforderungen der Koreanischen Zertifizierung (KC-Kennzeichen). Es wurde als Gerät der Klasse A eingestuft und ist für industrielle Anwendungen und nicht für die private Nutzung gedacht.

Zulassung für die Eurasische Zollunion (Belarus, Kasachstan, Russische Föderation)

EAC (Eurasische Konformität): Deklaration der Konformität nach den technischen Vorschriften der Zollunion (Technical Regulation of Customs Union, TR CU)

Zulassung für das Seewesen

Die S71200 Produkte werden regelmäßig für die Zulassungen hinsichtlich bestimmter Märkte und Anwendungen bei bestimmten Behörden eingereicht. Wenden Sie sich an Ihre Siemens-Vertretung, wenn Sie eine Liste mit den aktuellen Zulassungen für die einzelnen Bestellnummern benötigen.

Klassifizierungsgesellschaften:

- ABS (American Bureau of Shipping)
- BV (Bureau Veritas)
- DNV (Det Norske Veritas)
- GL (Germanischer Lloyd)
- LRS (Lloyds Register of Shipping)
- Class NK (Nippon Kaiji Kyokai)
- Korean Register of Shipping

Industrieumgebungen

Das Automatisierungssystem S7-1200 wurde für den Einsatz in Industrieumgebungen entwickelt.

Tabelle A- 1 Industrieumgebungen

Anwendungsgebiet	Anforderungen an die Störaussendung	Anforderungen an die Entstörung	Anforderungen an die Störfestigkeit
Industrie	EN 61000-6-4:2007+A1:2011	EN 61000-6-2:2005	EN 61000-6-2:2005

Hinweis

Das Automatisierungssystem S7-1200 wurde für den Einsatz in Industrieumgebungen entwickelt; beim Einsatz in Wohngebieten kann es Auswirkungen auf den Funk- oder TV-Empfang haben. Wird S7-1200 in Wohngebieten genutzt, so muss gewährleistet werden, dass seine Störaussendung innerhalb der Grenzwerte der Klasse B gemäß Norm EN 55011 bleibt.

Geeignete Maßnahmen für die Herstellung der Funkentstörung der Klasse B sind zum Beispiel:

- Einbau des S7-1200 in einem geerdeten Schaltschrank
- Verwendung von Rauschfiltern in den Zuleitungen

Es muss sichergestellt werden, dass die Störaussendung den Anforderungen an Klasse B der Norm 55011 entspricht.

Einzelne Abnahme ist erforderlich (die endgültige Installation muss alle Sicherheits- und EMV-Anforderungen für Installationen in Wohnumgebungen erfüllen).

Elektromagnetische Verträglichkeit (EMV)

Die elektromagnetische Verträglichkeit (EMV) eines elektrischen Geräts ist dessen Fähigkeit, in einer elektromagnetischen Umgebung bestimmungsgemäß zu funktionieren und keine elektromagnetischen Störungen auszusenden, die den Betrieb anderer elektrischer Geräte in der Umgebung beeinträchtigen könnten.

Tabelle A- 2 Störfestigkeit EN 61000-6-2

Elektromagnetische Verträglichkeit - Entstörung nach EN 61000-6-2	
EN 61000-4-2 Elektrostatische Entladung	8 kV Entladung durch die Luft an allen Oberflächen 6 kV Entladung durch Kontakt mit freiliegenden leitenden Oberflächen
EN 61000-4-3 Prüfung der Störfestigkeit gegen hochfrequente elektromagnetische Felder	80 bis 1000 MHz, 10 V/m, 80 % AM bei 1 kHz 1,4 bis 2,0 GHz, 3 V/m, 80 % AM bei 1 kHz 2,0 bis 2,7 GHz, 1 V/m, 80 % AM bei 1 kHz
EN 61000-4-4 Schnelle transiente Störgröße	2 kV, 5 kHz bei Kopplungsnetz zu AC und DCSystemspannung 2 kV, 5 kHz bei Kopplungsklemme zu Ein/Ausgängen
EN 6100-4-5 Stoßwellenfestigkeit	AC-Systeme - 2 kV Gleichtakt, 1 kV Gegentakt DC-Systeme - 2 kV Gleichtakt, 1 kV Gegentakt Zu DC-Systemen siehe "Stoßwellenfestigkeit" unten.
EN 61000-4-6 Leitungsgeführte Störungen	150 kHz bis 80 MHz, 10 V effektiv, 80% AM bei 1 kHz
EN 61000-4-11 Spannungseinbrüche	AC-Systeme 0 % für 1 Zyklus, 40 % für 12 Zyklen und 70 % für 30 Zyklen bei 60 Hz

Stoßwellenfestigkeit

Verdrahtungssysteme, die Einkopplungen durch Blitzschlag ausgesetzt sind, müssen mit einem externen Schutz versehen sein. Eine Spezifikation zur Bewertung des Schutzes vor Blitzstoßspannungen ist in EN 61000-4-5 enthalten, wobei die Betriebsgrenzen in EN 61000-6-2 festgelegt werden. S7-1200 DC-CPUs und Signalmodule erfordern externen Schutz, um den sicheren Betrieb aufrecht zu erhalten, wenn Stoßspannungen gemäß Definition in dieser Norm anliegen.

Nachstehend sind einige Geräte aufgeführt, die die erforderliche Stoßwellenfestigkeit gewährleisten. Diese Geräte bieten nur dann Schutz, wenn sie gemäß den Herstellerempfehlungen ordnungsgemäß installiert sind. Geräte anderer Hersteller mit gleichen oder besseren technischen Daten können ebenfalls verwendet werden:

Tabelle A- 3 Geräte, die Stoßwellenfestigkeit unterstützen

Subsystem	Schutzgerät
Leistung +24 V DC	BLITZDUCTOR VT, BVT AVD 24, Teilenummer 918 422
Industrial Ethernet	DEHNpatch DPA M CLE RJ45B 48, Teilenummer 929 121
RS-485	BLITZDUCTOR XT, Basiseinheit BXT BAS, Teilenummer 920 300
	BLITZDUCTOR XT, Modul BXT ML2 BD HFS 5, Teilenummer 920 271
RS-232	BLITZDUCTOR XT, Basiseinheit BXT BAS, Teilenummer 920 300
	BLITZDUCTOR XT, Modul BXT ML2 BE S 12, Teilenummer 920 222
Digitaleingänge +24 V DC	DEHN, Inc., Type DCO SD2 E 24, Teilenummer 917 988
Digitale Ausgänge und Geberversorgung +24 V DC	DEHN, Inc., Type DCO SD2 E 24, Teilenummer 917 988
Analoge E/A	DEHN, Inc., Type DCO SD2 E 12, Teilenummer 917 987
Relaisausgänge	Kein Schutz erforderlich

Tabelle A- 4 Leitungsgeführte und abgestrahlte Störaussendungen nach EN 61000-6-4

Elektromagnetische Verträglichkeit - Leitungsgeführte und abgestrahlte Störaussendungen nach EN 61000-6-4		
Leitungsgeführte Störaussendungen EN 55011, Klasse A, Gruppe 1	0,15 MHz bis 0,5 MHz	<79 dB (µV) Quasi-Spitze; <66 dB (µV) Mittelwert
	0,5 MHz bis 5 MHz	<73 dB (µV) Quasi-Spitze; <60 dB (µV) Mittelwert
	5 MHz bis 30 MHz	<73 dB (µV) Quasi-Spitze; <60 dB (µV) Mittelwert
Abgestrahlte Störaussendungen EN 55011, Klasse A, Gruppe 1	30 MHz bis 230 MHz	<40 dB (µV/m) Quasi-Spitze; gemessen in einer Entfernung von 10 m
	230 MHz bis 1 GHz	<47 dB (µV/m) Quasi-Spitze; gemessen in einer Entfernung von 10 m
	1 GHz bis 3 GHz	< 76dB (µV/m) Quasi-Spitze; gemessen in einer Entfernung von 10 m

Umgebungsbedingungen

Tabelle A- 5 Transport und Lagerung

Umgebungsbedingungen - Transport und Lagerung	
EN 6006822, Test Bb, trockene Wärme und EN 6006821 Test Ab, Kälte	-40 °C bis +70 °C
EN 60068230, Test Db, feuchte Wärme	25 °C bis 55 °C, 95 % Luftfeuchtigkeit
EN 60068-2-14, Test Na, Temperatur-schock	-40 °C bis +70 °C, Haltezeit 3 Stunden, 5 Zyklen
EN 60068-2-32 Freier Fall	0,3 m, 5 Mal, in Versandverpackung
Atmosphärischer Druck	1080 bis 660 hPa (entspricht einer Höhe von -1000 bis 3500 m)

Tabelle A- 6 Betriebsbedingungen

Umgebungsbedingungen - Betrieb	
Umgebungstemperaturen (Lufteinlass 25 mm unterhalb des Geräts)	-20 °C bis 60 °C C horizontale Montage -20 °C bis 50 °C vertikale Montage 95 % Luftfeuchtigkeit, nicht kondensierend Sofern nicht anders angegeben
Atmosphärischer Druck	1080 bis 795 hPa (entspricht einer Höhe von -1000 bis 2000m)
Konzentration von Schmutzstoffen	$S0_2$: < 0,5 ppm; H_2S: < 0,1 ppm; rel. LF < 60% nicht kondensierend
	ISA-S71.04 Stärkegrad G1, G2, G3
EN 60068-2-14, Test Nb, Temperaturver-änderung	5 °C bis 55 °C, 3 °C/Minute
EN 60068227 Mechanische Stoßbean-spruchung	15 G, Impuls 11 ms, 6 Stöße auf jeder der 3 Achsen
EN 6006826 Sinusschwingung	Hutschienenmontage: 3,5 mm von 5-9 Hz, 1 G von 9-150 Hz Schalttafeleinbau: 7,0 mm von 5 bis 9 Hz; 2 G von 9 bis 150 Hz 10 Ablenkungen je Achse, 1 Oktave/Minute

Tabelle A- 7 HochspannungsIsolationsprüfung

HochspannungsIsolationsprüfung	
Stromkreis mit 24V-DC-/5VDC-Nennspannung	520 V DC (Typprüfung der optischen Potentialtrennungsgrenzen)
115-V-AC-/230VAC-Stromkreis an Erde	1500 V AC
115-V-AC-/230VAC-Stromkreis an 115-V-AC-/230VAC-Stromkreis	1500 V AC
115-V-AC-/230VAC-Stromkreis an 24-V-DC-/5VDC-Stromkreis	1500 V AC (3000 V AC / 4242 V DC Typprüfung)
Ethernet-Port an 24-V-DC-/5-V-DC-Stromkreise und Erde[1]	1500 V AC (nur Typprüfung)

[1] Die Potentialtrennung des Ethernet-Ports ist dafür ausgelegt, die Gefahr während kurzfristiger Netzausfälle von gefährlichen Spannungen zu begrenzen. Sie entspricht nicht den Sicherheitsanforderungen der routinemäßigen Potentialtrennung von AC-Netzspannungen.

Schutzart

- Schutzklasse II nach EN 61131-2 (Schutzleiter nicht erforderlich)

Schutzgrad

- IP20 Mechanischer Schutz, EN 60529
- Schutz gegen direkte Berührung von Hochspannung wie mit genormter Sonde ermittelt. Externer Schutz erforderlich gegen Staub, Schmutz, Wasser und Fremdkörper mit einem Durchmesser von < 12,5 mm.

Bemessungsspannungen

Tabelle A- 8 Bemessungsspannungen

Nennspannung	**Toleranz**
24 V DC	20,4 V DC bis 28,8 V DC
120/230 V AC	85 bis 264 V AC, 47 bis 63 Hz

Verpolschutz

Verpolschutz ist vorhanden bei allen Klemmenpaaren mit +24-V-DC-Spannungsversorgung oder anwenderseitiger Eingangsspannung für CPUs, Signalmodule (SMs) und Signalboards (SBs). Trotzdem sind Beschädigungen des System weiterhin dadurch möglich, dass unterschiedliche Klemmenpaare mit entgegengesetzter Polarität verdrahtet werden.

Einige der 24-V-DC-Eingangsports des S7-1200 Systems sind miteinander verbunden, wobei ein logischer Bezugsleiter mehrere M-Klemmen verbindet. Beispielsweise sind die folgenden Stromkreise miteinander verbunden, sofern sie in den Datenblättern als "nicht potentialgetrennt" angegeben sind: die 24-V-DC-Versorgung der CPU, die Sensorleistung der CPU, der Leistungseingang für die Relaisspule eines SM und die Versorgung eines nicht potentialgetrennten Analogeingangs. Alle nicht potentialgetrennten M-Klemmen müssen an dasselbe externe Bezugspotential angeschlossen werden.

⚠WARNUNG

Wenn Sie nicht potentialgetrennte M-Klemmen an verschiedene Bezugspotentiale anschließen, verursacht dies unbeabsichtigten Stromfluss, der zu Beschädigung oder unvorhersehbarem Betriebsverhalten des Zielsystems und angeschlossener Geräte führen kann.

Die Nichteinhaltung dieser Richtlinien kann Schaden oder unvorhersehbaren Betrieb verursachen, was zu Tod oder schwerer Körperverletzung und/oder Sachschaden führen kann.

Schließen Sie stets alle nicht potentialgetrennten M-Klemmen in einem S7-1200 System an dasselbe Bezugspotential an.

DC-Ausgänge

Es stehen keine kurzschlussfesten Schaltungen für die Gleichspannungsausgänge an CPUs, Signalmodulen (SMs) und Signalboards (SBs) zur Verfügung.

Lebensdauer eines Relais

Die typischen Leistungsdaten, die anhand von Beispieltests geschätzt wurden, sind nachstehend aufgeführt. Die tatsächliche Leistungsfähigkeit richtet sich nach der jeweiligen Verwendung. Ein der Last angepasster externer Schutzkreis verlängert die Lebensdauer der Kontakte. Die typische Lebensdauer von Öffnerkontakten beträgt etwa ein Drittel der Lebensdauer eines Schließers bei induktiver und Lampenlast.

Ein externer Schutzkreis erhöht die Lebensdauer der Kontakte.

Tabelle A- 9 Typische Leistungsdaten

Daten für die Auswahl eines Aktors				
Thermischer Dauerstrom		max. 2 A		
Schaltvermögen und Lebensdauer der Kontakte				
	Bei ohmscher Last	Span-nung	Strom	Anzahl der Schaltzyklen (typ.)
		24 V DC	2,0 A	0,1 Millionen
		24 V DC	1,0 A	0,2 Millionen
		24 V DC	0,5 A	1,0 Millionen
		48 V AC	1,5 A	1,5 Millionen
		60 V AC	1,5 A	1,5 Millionen
		120 V AC	2,0 A	1,0 Millionen
		120 V AC	1,0 A	1,5 Millionen
		120 V AC	0,5 A	2,0 Millionen
		230 V AC	2,0 A	1,0 Millionen
		230 V AC	1,0 A	1,5 Millionen
		230 V AC	0,5 A	2,0 Millionen
	Bei induktiver Last (nach IEC 947-5-1 DC13/AC15)	Span-nung	Strom	Anzahl der Schaltzyklen (typ.)
		24 V DC	2,0 A	0,05 Millionen
		24 V DC	1,0 A	0,1 Millionen
		24 V DC	0,5 A	0,5 Millionen
		24 V AC	1,5 A	1,0 Millionen
		48 V AC	1,5 A	1,0 Millionen
		60 V AC	1,5 A	1,0 Millionen
		120 V AC	2,0 A	0,7 Millionen
		120 V AC	1,0 A	1,0 Millionen
		120 V AC	0,5 A	1,5 Millionen
		230 V AC	2,0 A	0,7 Millionen
		230 V AC	1,0 A	1,0 Millionen
		230 V AC	0,5 A	1,5 Millionen
Aktivieren eines Digitaleingangs		Möglich		

Daten für die Auswahl eines Aktors		
Schaltfrequenz		
	Mechanisch	Max. 10 Hz
	Bei ohmscher Last	Max. 1 Hz
	Bei induktiver Last (nach IEC 947-5-1 DC13/AC15)	Max. 0,5 Hz
	Bei Lampenlast	Max. 1 Hz

Speicherung im internen CPU-Speicher

- Lebensdauer von remanenten Daten und Datenprotokolldaten: 10 Jahre
- Remanente Daten bei Spannungsausfall, Schreibzyklusbeständigkeit: 2 Millionen Zyklen
- Datenprotokolldaten, bis zu 2 KB je Datenprotokolleintrag, Schreibzyklusbeständigkeit: 500 Millionen Datenprotokolleinträge

Hinweis

Auswirkung von Datenprotokollen auf den internen CPU-Speicher

Jeder Schreibvorgang eines Datenprotokolls verbraucht mindestens 2 KB Speicher. Wenn Ihr Programm häufig kleinere Mengen von Daten schreibt, werden bei jedem Schreibvorgang mindestens 2 KB Speicher verbraucht. Eine bessere Umsetzung wäre die Ansammlung kleiner Datenelemente in einem Datenbaustein (DB), der dann weniger häufig ins Datenprotokoll geschrieben würde.

Wenn Ihr Programm sehr häufig viele Datenprotokolleinträge schreibt, ziehen Sie die Verwendung einer austauschbaren SD Memory Card in Betracht.

A.2 CPU 1211C

A.2.1 Allgemeine technische Daten und Leistungsmerkmale

Tabelle A- 10 Allgemeine technische Daten

Technische Daten	CPU 1211C AC/DC/Relais	CPU 1211C DC/DC/Relais	CPU 1211C DC/DC/DC
Artikelnummer	6ES7 211-1BE40-0XB0	6ES7 211-1HE40-0XB0	6ES7 211-1AE40-0XB0
Abmessungen B x H x T (mm)	90 x 100 x 75		
Versandgewicht	420 Gramm	380 Gramm	370 Gramm
Leistungsverlust	10 W	8 W	
Verfügbarer Strom (CM-Bus)	max. 750 mA (5 V DC)		
Verfügbarer Strom (24 V DC)	max. 300 mA (Geberversorgung)		
Stromaufnahme digitaler Eingang (24 V DC)	4 mA/Eingang		

Tabelle A- 11 CPU-Merkmale

Technische Daten		Beschreibung
Anwenderspeicher (Siehe "Allgemeine technische Daten" (Seite 1151), "Speicherung im internen CPU-Speicher".)	Arbeitsspeicher	50 KB
	Last	1 MB, intern, erweiterbar bis auf SD-Kartengröße
	Remanent	10 KB
Integrierte digitale E/A		6 Eingänge/4 Ausgänge
Integrierte analoge E/A		2 Eingänge
Größe des Prozessabbilds		1024 Bytes Eingänge (E)/1024 Bytes Ausgänge (A)
Merker (M)		4096 Byte
Temporärer (lokaler) Speicher		• 16 KB für Anlauf und Programmzyklus (einschließlich der zugehörigen FBs und FCs) • 6 KB für jede andere Alarmprioritätsstufe (einschließlich FBs und FCs)
Zusätzliche Signalmodule		Keine
Erweiterung SB, CB, BB		max. 1
Zusätzliche Kommunikationsmodule		max. 3 CM
Schnelle Zähler		Bis zu 6 konfiguriert für die Verwendung beliebiger integrierter oder SB-Eingänge. Siehe Tabelle CPU 1211C: Zuweisungen der HSC-Standardadressen (Seite 471) 100/[1]80 kHz (Ea.0 bis Ea.5)
Impulsausgänge [2]		Bis zu 4 konfiguriert für die Verwendung beliebiger integrierter oder SB-Ausgänge 100 kHz (Aa.0 bis Aa.3)

Technische Daten	Beschreibung
Eingänge für Impulsabgriff	6
Verzögerungsalarme	4 mit Auflösung von 1 ms
Weckalarme	4 mit Auflösung von 1 ms
Flankenalarme	6 steigend und 6 fallend (10 und 10 mit optionalem Signalboard)
Memory Card	SIMATIC Memory Card (optional)
Genauigkeit Echtzeituhr	+/- 60 Sekunden/Monat
Pufferung Echtzeituhr	Typ. 20 Tage/min. 12 Tage bei 40 °C (wartungsfreier Hochleistungskondensator)

1 Die langsamere Geschwindigkeit gilt, wenn der HSC als A/B-Zähler konfiguriert ist.

2 Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge zu verwenden.

Tabelle A- 12 Leistung

Art der Anweisung	Ausführungsgeschwindigkeit
Boolesch	0,08 µs/Operation
Wort übertragen	1,7 µs/Operation
Realzahlenarithmetik	2,3 µs/Operation

A.2.2 Von der CPU 1211C unterstützte Zeiten, Zähler und Codebausteine

Tabelle A- 13 Von der CPU 1211C unterstützte Bausteine, Zeiten und Zähler

Element		Beschreibung
Bausteine	Typ	OB, FB, FC, DB
	Größe	30 KB
	Anzahl	Bis 1024 Bausteine gesamt (OBs + FBs + FCs + DBs)
	Adressbereich für FBs, FCs und DBs	FB und FC: 1 bis 65535 (z. B. FB 1 bis FB 65535) DB: 1 bis 59999
	Schachtelungstiefe	16 aus Zyklus- oder Anlauf-OBs 6 aus beliebigen Alarmereignis-OBs
	Überwachung	Der Zustand von 2 Codebausteinen kann gleichzeitig überwacht werden.
OBs	Programmzyklus	Mehrere
	Anlauf	Mehrere
	Verzögerungsalarm	4 (1 pro Ereignis)
	Weckalarme	4 (1 pro Ereignis)
	Prozessalarme	50 (1 pro Ereignis)
	Zeitfehleralarme	1
	Diagnosefehleralarme	1
	Ziehen oder Stecken von Modulen	1

Element		Beschreibung
	Fehler bei Baugruppenträger oder Station	1
	Tageszeit	Mehrere
	Status	1
	Update	1
	Profil	1
Zeiten	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, 16 Byte pro Zeit
Zähler	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, Größe abhängig von der Zählart • SInt, USInt: 3 Byte • Int, UInt: 6 Byte • DInt, UDInt: 12 Byte

Tabelle A- 14 Kommunikation

Technische Daten	Beschreibung
Schnittstellen	1
Typ	Ethernet
HMI-Gerät	4
Programmiergerät (PG)	1
Anschlüsse	• 8 für die offene Benutzerkommunikation (aktiv oder passiv): TSEND_C, TRCV_C, TCON, TDISCON, TSEND und TRCV • 3 für die S7-Kommunikation über Server-GET/PUT (CPU-zu-CPU) • 8 für die S7-Kommunikation über Client-GET/PUT (CPU-zu-CPU)
Datenraten	10/100 MBit/s
Elektrische Trennung (externes Signal zu PLC-Logik)	Wandler potentialgetrennt, 1500 V AC, nur für kurzfristige Sicherheit
Kabelart	CAT5e geschirmt

Tabelle A- 15 Stromversorgung

Technische Daten		CPU 1211C AC/DC/Relais	CPU 1211C DC/DC/Relais	CPU 1211C DC/DC/DC
Spannungsbereich		85 bis 264 V AC	20,4 V DC bis 28,8 V DC	
Netzfrequenz		47 bis 63 Hz	--	
Eingangs-strom	nur CPU bei max. Last	60 mA bei 120 V AC 30 mA bei 240 V AC	30 mA bei 24 V DC	300 mA bei 24 V DC
	CPU mit allen Erweite-rungsbaugruppen bei max. Last	180 mA bei 120 V AC 90 mA bei 240 V AC	900 mA bei 24 V DC	
Einschaltstrom (max.)		20 A bei 264 V AC	12 A bei 28,8 V DC	
Elektrische Trennung (Eingangsleistung zu Logik)		1500 V AC	Nicht elektrisch getrennt	
Kriechstrom an Erde, AC-Leitung an Funktionserde		max. 0,5 mA	--	
Verzögerungszeit (Spannungsverlust)		20 ms bei 120 V AC 80 ms bei 240 V AC	10 ms bei 24 V DC	
Interne Sicherung, nicht durch Anwender austauschbar		3 A, 250 V, träge		

Tabelle A- 16 Geberversorgung

Technische Daten	CPU 1211C AC/DC/Relais	CPU 1211C DC/DC/Relais	CPU 1211C DC/DC/DC
Spannungsbereich	20,4 bis 28,8 V DC	L+ minus 4 V DC min.	
Nennausgangsstrom (max.)	300 mA (kurzschlussfest)		
Max. Welligkeit/Störströme (<10 MHz)	< 1 V Spitze-Spitze	Wie Eingangsleitung	
Elektrische Trennung (CPU-Logik zu Geberversorgung)	Nicht elektrisch getrennt		

A.2.3 Digitale Eingänge und Ausgänge

Tabelle A- 17 Digitale Eingänge

Technische Daten	CPU 1211C AC/DC/Relais, CPU 1211C DC/DC/Relais und CPU 1211C DC/DC/DC
Anzahl der Eingänge	6
Typ	Stromziehend/stromliefernd (IEC Typ 1, wenn stromziehend)
Nennspannung	24 V DC bei 4 mA, Nennwert
Zulässige Dauerspannung	max. 30 V DC
Stoßspannung	35 V DC für 0,5 s
Signal logisch 1 (min.)	15 V DC bei 2,5 mA
Signal logisch 0 (max.)	5 V DC bei 1 mA
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute
Potenzialgetrennte Gruppen	1
Filterzeiten	µs-Einstellungen: 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0 ms-Einstellungen: 0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0
HSC Eingangstaktfrequenzen (max.) (Pegel logisch 1 = 15 bis 26 V DC)	100/80 kHz (Ea.0 bis Ea.5)
Anzahl gleichzeitig eingeschalteter Eingänge	6 bei 60 °C horizontal, 50 °C vertikal
Leitungslänge (Meter)	500 m geschirmt, 300 m ungeschirmt, 50 m geschirmt für HSC-Eingänge

Tabelle A- 18 Digitale Ausgänge

Technische Daten	CPU 1211C AC/DC/Relais und CPU 1211C DC/DC/Relais	CPU 1211C DC/DC/DC
Ausgänge	4	
Typ	Relais, mechanisch	MOSFET, elektronisch (stromliefernd)
Spannungsbereich	5 bis 30 V DC oder 5 bis 250 V AC	20,4 bis 28,8 V DC
Signal logisch 1 bei max. Strom	--	min. 20 V DC
Signal logisch 0 bei 10 kΩ Last	--	max. 0,1 V DC
Strom (max.)	2,0 A	0,5 A
Lampenlast	30 W DC/200 W AC	5 W
Widerstand bei EIN	max. 0,2 Ω wenn neu	max. 0,6 Ω
Kriechstrom pro Ausgang	--	max. 10 µA
Einschaltstrom	7 A bei geschlossenen Kontakten	8 A für max. 100 ms
Überlastschutz	Nein	
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute (Spule zu Kontakt) Keine (Spule zu Logik)	500 V AC für 1 Minute
Isolationswiderstand	min. 100 MΩ, wenn neu	--
Elektrische Trennung zwischen offenen Kontakten	750 V AC für 1 Minute	--

Technische Daten	CPU 1211C AC/DC/Relais und CPU 1211C DC/DC/Relais	CPU 1211C DC/DC/DC
Potentialgetrennte Gruppen	1	
Induktive Klemmspannung	--	L+ minus 48 V DC, 1 W Verlustleistung
Maximale Schaltfrequenz Relais	1 Hz	--
Schaltverzögerung (Aa.0 bis Aa.3)	max. 10 ms	max. 1,0 µs von Aus nach Ein max. 3,0 µs von Ein nach Aus
Frequenz Impulsgenerator	Nicht empfehlenswert [1]	100 kHz (Aa.0 bis Aa.3)[2], min. 2 Hz
Mechanische Lebensdauer (ohne Last)	10.000.000 Schaltspiele auf/zu	--
Lebensdauer der Kontakte bei Nennlast	100.000 Schaltspiele auf/zu	--
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)	
Anzahl gleichzeitig eingeschalteter Ausgänge	4 bei 60 °C horizontal, 50 °C vertikal	
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt	

[1] Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge zu verwenden.

[2] Je nach Impulsempfänger und Kabel kann ein zusätzlicher Lastwiderstand (bei mindestens 10% des Nennstroms) die Qualität der Impulssignale und die Störfestigkeit verbessern.

A.2.4 Analoge Eingänge

Tabelle A- 19 Analoge Eingänge

Technische Daten	Beschreibung
Anzahl der Eingänge	2
Typ	Spannung (Eintakteingang)
Vollausschlagsbereich	0 bis 10 V
Vollausschlag (Datenwort)	0 bis 27648
Überschwingbereich	10,001 bis 11,759 V
Überschwingbereich (Datenwort)	27649 bis 32511
Überlaufbereich	11,760 bis 11,852 V
Überlaufbereich (Datenwort)	32512 bis 32767
Auflösung	10 Bits
Max. Stehspannung	35 V DC
Glättung	Keine, schwach, mittel oder stark Siehe Tabelle der Schrittantwort (ms) für die analogen Eingänge der CPU (Seite 1169).
Rauschunterdrückung	10, 50 oder 60 Hz
Impedanz	≥100 kΩ
Elektrische Trennung (Feld zu Logik)	Keine
Genauigkeit (25 °C/0 bis 55 °C)	3,0% / 3,5% des Vollausschlags
Leitungslänge (Meter)	100 m, geschirmtes, verdrilltes Leiterpaar

A.2.4.1 Schrittantwort der integrierten analogen Eingänge der CPU

Tabelle A- 20 Schrittantwort (ms), 0 V bis 10 V gemessen bei 95 %

Auswahl der Glättung (Mittelwertbildung aus Abtastwerten)	Unterdrückungsfrequenz (Integrationszeit)		
	60 Hz	50 Hz	10 Hz
Keine (1 Zyklus): Keine Mittelwertbildung	50 ms	50 ms	100 ms
Schwach (4 Zyklen): 4 Abtastwerte	60 ms	70 ms	200 ms
Mittel (16 Zyklen): 16 Abtastwerte	200 ms	240 ms	1150 ms
Stark (32 Zyklen): 32 Abtastwerte	400 ms	480 ms	2300 ms
Abtastzeit	**4,17 ms**	**5 ms**	**25 ms**

A.2.4.2 Abtastzeit der integrierten analogen Ports der CPU

Tabelle A- 21 Abtastzeit der integrierten analogen Eingänge der CPU

Unterdrückungsfrequenz (Auswahl Integrationszeit)	Abtastzeit
60 Hz (16,6 ms)	4,17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

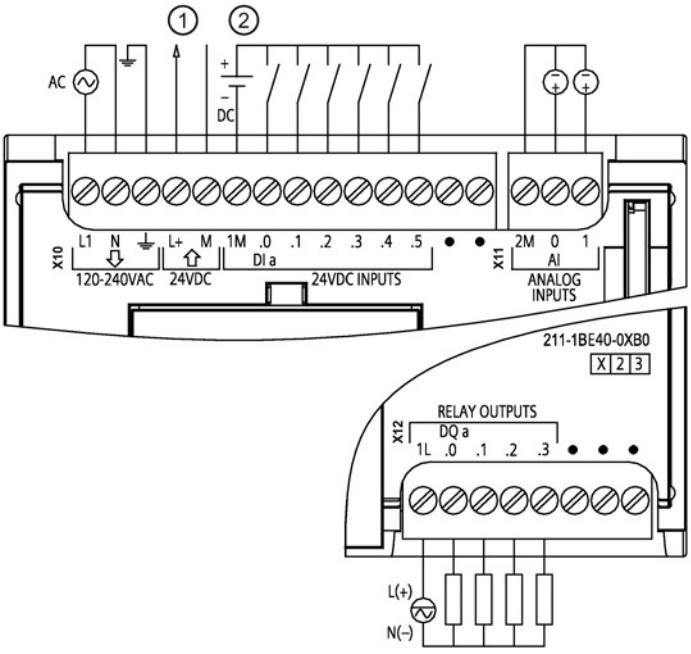
A.2.4.3 Messbereiche der analogen Eingänge für Spannung (CPUs)

Tabelle A- 22 Darstellung Analogeingang für Spannung (CPUs)

System		Messbereich Spannung	
Dezimal	Hexadezimal	0 bis 10 V	
32767	7FFF	11,851 V	Überlauf
32512	7F00		
32511	7EFF	11,759 V	Überschwingbereich
27649	6C01		
27648	6C00	10 V	Bemessungsbereich
20736	5100	7,5 V	
34	22	12 mV	
0	0	0 V	
Negative Werte		Negative Werte werden nicht unterstützt	

A.2.5 Schaltpläne der CPU 1211

Tabelle A- 23 CPU 1211C AC/DC/Relais (6ES7 211-1BE40-0XB0)



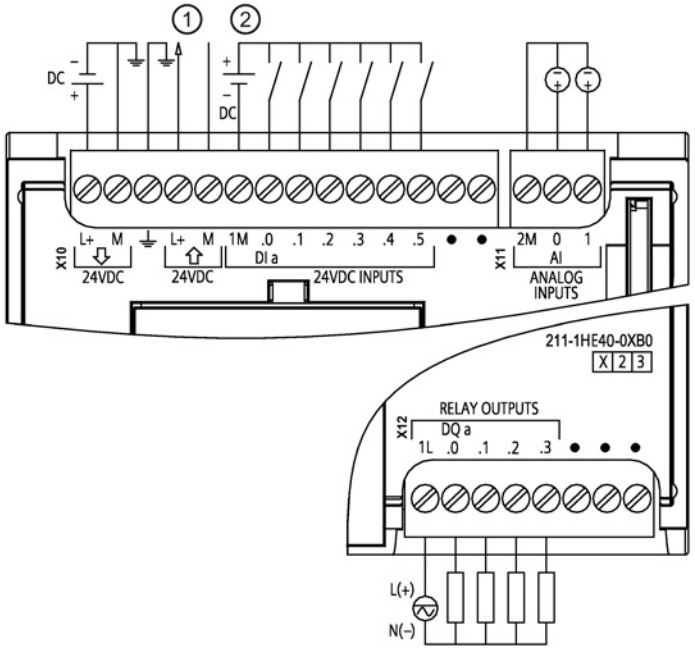
①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Eine der beiden Klemmen L1 oder N (L2) kann an eine Spannungsquelle bis 240 V AC angeschlossen werden. Die Klemme N kann als L2 betrachtet werden und muss nicht geerdet werden. Für die Klemmen L1 und N (L2) ist kein Verpolungsschutz erforderlich.	
Hinweis 3: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 24 Anschlussbelegung für die CPU 1211C AC/DC/Relais (6ES7 211-1BE40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L1 / 120-240 V AC	2 M	1L
2	N / 120-240 V AC	AI 0	DO a.0
3	Funktionserde	AI 1	DO a.1
4	L+ / 24-V-DC-Sensorausgang	--	DO a.2
5	M / 24-V-DC-Sensorausgang	--	DO a.3
6	1M	--	Kein Anschluss

Pin	X10	X11 (vergoldet)	X12
7	DI a.0	--	Kein Anschluss
8	DI a.1	--	Kein Anschluss
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	Kein Anschluss	--	--
14	Kein Anschluss	--	--

Tabelle A- 25 CPU 1211C DC/DC/Relais (6ES7 211-1HE40-0XB0)



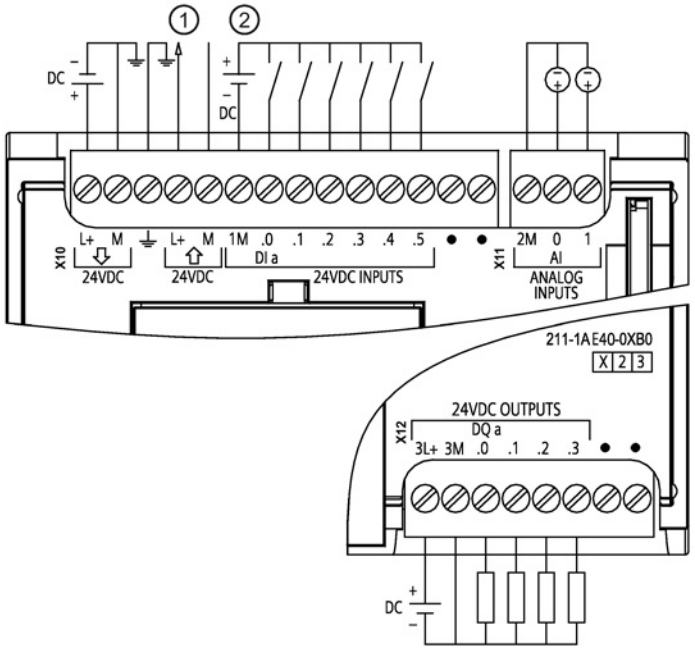
①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 26 Anschlussbelegung für die CPU 1211C DC/DC/Relais (6ES7 211-1HE40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L+ / 24 V DC	2 M	1L
2	M / 24 V DC	AI 0	DO a.0
3	Funktionserde	AI 1	DO a.1
4	L+ / 24-V-DC-Sensorausgang	--	DO a.2
5	M / 24-V-DC-Sensorausgang	--	DO a.3

Pin	X10	X11 (vergoldet)	X12
6	1M	--	Kein Anschluss
7	DI a.0	--	Kein Anschluss
8	DI a.1	--	Kein Anschluss
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	Kein Anschluss	--	--
14	Kein Anschluss	--	--

Tabelle A- 27 CPU 1211C DC/DC/DC (6ES7 211-1AE40-0XB0)



①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 28 Anschlussbelegung für die CPU 1211C DC/DC/DC (6ES7 211-1AE40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L+ / 24 V DC	2 M	3L+
2	M / 24 V DC	AI 0	3M
3	Funktionserde	AI 1	DO a.0
4	L+ / 24-V-DC-Sensorausgang	--	DO a.1

Pin	X10	X11 (vergoldet)	X12
5	M / 24-V-DC-Sensorausgang	--	DO a.2
6	1M	--	DO a.3
7	DI a.0	--	Kein Anschluss
8	DI a.1	--	Kein Anschluss
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	Kein Anschluss	--	--
14	Kein Anschluss	--	--

Hinweis

Nicht verwendete analoge Eingänge sollten kurzgeschlossen werden.

A.3 CPU 1212C

A.3.1 Allgemeine technische Daten und Leistungsmerkmale

Tabelle A- 29 Allgemein

Technische Daten	CPU 1212C AC/DC/Relais	CPU 1212C DC/DC/Relais	CPU 1212C DC/DC/DC
Artikelnummer	6ES7 212-1BE40-0XB0	6ES7 212-1HE40-0XB0	6ES7 212-1AE40-0XB0
Abmessungen B x H x T (mm)	90 x 100 x 75		
Versandgewicht	425 Gramm	385 Gramm	370 Gramm
Leistungsverlust	11 W	9 W	
Verfügbarer Strom (SM- und CM-Bus)	max. 1.000 mA (5 V DC)		
Verfügbarer Strom (24 V DC)	max. 300 mA (Geberversorgung)		
Stromaufnahme digitaler Eingang (24 V DC)	4 mA/Eingang		

Tabelle A- 30 CPU-Merkmale

Technische Daten		Beschreibung
Anwenderspeicher (Siehe "Allgemeine technische Daten" (Seite 1151), "Speicherung im internen CPU-Speicher".)	Arbeitsspeicher	75 KB
	Ladespeicher	1 MB, intern, erweiterbar bis auf SD-Kartengröße
	Remanent	10 KB
Integrierte digitale E/A		8 Eingänge/6 Ausgänge
Integrierte analoge E/A		2 Eingänge
Größe des Prozessabbilds		1024 Bytes Eingänge (E)/1024 Bytes Ausgänge (A)
Merker (M)		4096 Byte
Temporärer (lokaler) Speicher		• 16 KB für Anlauf und Programmzyklus (einschließlich der zugehörigen FBs und FCs) • 6 KB für jede andere Alarmprioritätsstufe (einschließlich FBs und FCs)
Zusätzliche Signalmodule		max. 2 SMs
Erweiterung SB, CB, BB		max. 1
Zusätzliche Kommunikationsmodule		max. 3 CMs

Technische Daten	Beschreibung
Schnelle Zähler	Bis zu 6 konfiguriert für die Verwendung beliebiger integrierter oder SB-Eingänge. Siehe Tabelle, CPU 1212C: Zuweisungen der HSC-Standardadressen (Seite 471) • 100/[1]80 kHz (Ea.0 bis Ea.5) • 30 /[1]20 kHz (Ea.6 bis Ea.7)
Impulsausgänge [2]	Bis zu 4 konfiguriert für die Verwendung beliebiger integrierter oder SB-Ausgänge • 100 kHz (Aa.0 bis Aa.3) • 30 kHz (Aa.4 bis Aa.5)
Eingänge für Impulsabgriff	8
Verzögerungsalarme	4 gesamt mit Auflösung von 1 ms
Weckalarme	4 gesamt mit Auflösung von 1 ms
Flankenalarme	8 steigend und 8 fallend (12 und 12 mit optionalem Signalboard)
Memory Card	SIMATIC Memory Card (optional)
Genauigkeit Echtzeituhr	+/- 60 Sekunden/Monat
Pufferung Echtzeituhr	Typ. 20 Tage/min. 12 Tage bei 40 °C (wartungsfreier Hochleistungskondensator)

[1] Die langsamere Geschwindigkeit gilt, wenn der HSC als A/B-Zähler konfiguriert ist.

[2] Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge zu verwenden.

Tabelle A- 31 Leistung

Art der Anweisung	Ausführungsgeschwindigkeit
Boolesch	0,08 µs/Operation
Wort übertragen	1,7 µs/Operation
Realzahlenarithmetik	2,3 µs/Operation

A.3.2 Von der CPU 1212C unterstützte Zeiten, Zähler und Codebausteine

Tabelle A- 32 Von der CPU 1212C unterstützte Bausteine, Zeiten und Zähler

Element		Beschreibung
Bausteine	Typ	OB, FB, FC, DB
	Größe	50 KB
	Anzahl	Bis 1024 Bausteine gesamt (OBs + FBs + FCs + DBs)
	Adressbereich für FBs, FCs und DBs	FB und FC: 1 bis 65535 (z. B. FB 1 bis FB 65535) DB: 1 bis 59999
	Schachtelungstiefe	16 aus Zyklus- oder Anlauf-OBs 6 aus beliebigen Alarmereignis-OBs
	Überwachung	Der Zustand von 2 Codebausteinen kann gleichzeitig überwacht werden.
OBs	Programmzyklus	Mehrere
	Anlauf	Mehrere
	Verzögerungsalarm	4 (1 pro Ereignis)
	Weckalarme	4 (1 pro Ereignis)
	Prozessalarme	50 (1 pro Ereignis)
	Zeitfehleralarme	1
	Diagnosefehleralarme	1
	Ziehen oder Stecken von Modulen	1
	Fehler bei Baugruppenträger oder Station	1
	Tageszeit	Mehrere
	Status	1
	Update	1
	Profil	1
Zeiten	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, 16 Byte pro Zeit
Zähler	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, Größe abhängig von der Zählart • SInt, USInt: 3 Byte • Int, UInt: 6 Byte • DInt, UDInt: 12 Byte

Tabelle A- 33 Kommunikation

Technische Daten	Beschreibung
Schnittstellen	1
Typ	Ethernet
HMI-Gerät	4
Programmiergerät (PG)	1
Anschlüsse	• 8 für die offene Benutzerkommunikation (aktiv oder passiv): TSEND_C, TRCV_C, TCON, TDISCON, TSEND und TRCV • 3 für die S7-Kommunikation über Server-GET/PUT (CPU-zu-CPU) • 8 für die S7-Kommunikation über Client-GET/PUT (CPU-zu-CPU)
Datenraten	10/100 MBit/s
Elektrische Trennung (externes Signal zu PLC-Logik)	Wandler potentialgetrennt, 1500 V AC, nur für kurzfristige Sicherheit
Kabelart	CAT5e geschirmt

Tabelle A- 34 Stromversorgung

Technische Daten		CPU 1212C AC/DC/Relais	CPU 1212C DC/DC/Relais	CPU 1212C DC/DC/DC
Spannungsbereich		85 bis 264 V AC	20,4 V DC bis 28,8 V DC	
Netzfrequenz		47 bis 63 Hz	--	
Einschaltstrom (max. Last)	Nur CPU	80 mA bei 120 V AC 40 mA bei 240 V AC	400 mA bei 24 V DC	
	CPU mit allen Erweiterungsbaugruppen	240 mA bei 120 V AC 120 mA bei 240 V AC	1200 mA bei 24 V DC	
Einschaltstrom (max.)		20 A bei 264 V AC	12 A bei 28,8 V DC	
Elektrische Trennung (Eingangsleistung zu Logik)		1500 V AC	Nicht elektrisch getrennt	
Kriechstrom an Erde, AC-Leitung an Funktionserde		max. 0,5 mA	--	
Verzögerungszeit (Spannungsverlust)		20 ms bei 120 V AC 80 ms bei 240 V AC	10 ms bei 24 V DC	
Interne Sicherung, nicht durch Anwender austauschbar		3 A, 250 V, träge		

Tabelle A- 35 Geberversorgung

Technische Daten	CPU 1212C AC/DC/Relais	CPU 1212C DC/DC/Relais	CPU 1212C DC/DC/DC
Spannungsbereich	20,4 bis 28,8 V DC	L+ minus 4 V DC min.	
Nennausgangsstrom (max.)	300 mA (kurzschlussfest)		
Max. Welligkeit/Störströme (<10 MHz)	< 1 V Spitze-Spitze	Wie Eingangsleitung	
Elektrische Trennung (CPU-Logik zu Geberversorgung)	Nicht elektrisch getrennt		

A.3.3 Digitale Eingänge und Ausgänge

Tabelle A- 36 Digitale Eingänge

Technische Daten	CPU 1212C AC/DC/Relais, DC/DC/Relais und DC/DC/DC
Anzahl der Eingänge	8
Typ	Stromziehend/stromliefernd (IEC Typ 1, wenn stromziehend)
Nennspannung	24 V DC bei 4 mA, Nennwert
Zulässige Dauerspannung	max. 30 V DC
Stoßspannung	35 V DC für 0,5 s
Signal logisch 1 (min.)	15 V DC bei 2,5 mA
Signal logisch 0 (max.)	5 V DC bei 1 mA
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute
Potentialgetrennte Gruppen	1
Filterzeiten	µs-Einstellungen: 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0 ms-Einstellungen: 0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0
HSC Eingangstaktfrequenzen (max.) (Pegel logisch 1 = 15 bis 26 V DC)	100/80 kHz (Ea.0 bis Ea.5) 30 /20 kHz (Ea.6 bis Ea.7)
Anzahl gleichzeitig eingeschalteter Eingänge	4 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal 8 bei 55 °C horizontal oder 45 °C vertikal
Leitungslänge (Meter)	500 m geschirmt, 300 m ungeschirmt, 50 m geschirmt für HSC-Eingänge

Tabelle A- 37 Digitale Ausgänge

Technische Daten	CPU 1212C AC/DC/Relais und DC/DC/Relais	CPU 1212C DC/DC/DC
Ausgänge	6	
Typ	Relais, mechanisch	MOSFET, elektronisch (stromliefernd)
Spannungsbereich	5 bis 30 V DC oder 5 bis 250 V AC	20,4 bis 28,8 V DC
Signal logisch 1 bei max. Strom	--	min. 20 V DC
Signal logisch 0 bei 10 kΩ Last	--	max. 0,1 V DC
Strom (max.)	2,0 A	0,5 A
Lampenlast	30 W DC/200 W AC	5 W
Widerstand bei EIN	max. 0,2 Ω, wenn neu	max. 0,6 Ω
Kriechstrom pro Ausgang	--	max. 10 µA
Einschaltstrom	7 A bei geschlossenen Kontakten	8 A für max. 100 ms
Überlastschutz	Nein	
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute (Spule zu Kontakt) Keine (Spule zu Logik)	500 V AC für 1 Minute
Isolationswiderstand	min. 100 MΩ, wenn neu	--
Elektrische Trennung zwischen offenen Kontakten	750 V AC für 1 Minute	--
Potentialgetrennte Gruppen	2	1

Technische Daten	CPU 1212C AC/DC/Relais und DC/DC/Relais	CPU 1212C DC/DC/DC
Induktive Klemmspannung	--	L+ minus 48 V DC, 1 W Verlustleistung
Schaltverzögerung (Aa.0 bis Aa.3)	max. 10 ms	max. 1,0 µs von Aus nach Ein max. 3,0 µs von Ein nach Aus
Schaltverzögerung (Aa.4 bis Aa.5)	max. 10 ms	max. 50 µs von Aus nach Ein max. 200 µs von Ein nach Aus
Maximale Schaltfrequenz Relais	1 Hz	--
Frequenz Impulsgenerator	Nicht empfehlenswert [1]	100 kHz (Aa.0 bis Aa.3)[2], 2 Hz min. 20 kHz (Aa.4 bis Aa.5)[2]
Mechanische Lebensdauer (ohne Last)	10.000.000 Schaltspiele auf/zu	--
Lebensdauer der Kontakte bei Nennlast	100.000 Schaltspiele auf/zu	--
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)	Letzter Wert oder Ersatzwert (Voreinstellung 0)
Anzahl gleichzeitig eingeschalteter Ausgänge	3 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal 6 bei 55 °C horizontal oder 45 °C vertikal	
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt	

[1] Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge zu verwenden.

[2] Je nach Impulsempfänger und Kabel kann ein zusätzlicher Lastwiderstand (bei mindestens 10% des Nennstroms) die Qualität der Impulssignale und die Störfestigkeit verbessern.

A.3.4 Analoge Eingänge

Tabelle A- 38 Analoge Eingänge

Technische Daten	Beschreibung
Anzahl der Eingänge	2
Typ	Spannung (Eintakteingang)
Vollausschlagsbereich	0 bis 10 V
Vollausschlag (Datenwort)	0 bis 27648
Überschwingbereich	10,001 bis 11,759 V
Überschwingbereich (Datenwort)	27649 bis 32511
Überlaufbereich	11,760 bis 11,852 V
Überlaufbereich (Datenwort)	32512 bis 32767
Auflösung	10 Bits
Max. Stehspannung	35 V DC
Glättung	Keine, schwach, mittel oder stark Siehe Tabelle der Schrittantwort (ms) für die analogen Eingänge der CPU (Seite 1181).
Rauschunterdrückung	10, 50 oder 60 Hz
Impedanz	≥100 kΩ
Elektrische Trennung (Feld zu Logik)	Keine
Genauigkeit (25 °C/0 bis 55 °C)	3,0% / 3,5% des Vollausschlags
Leitungslänge (Meter)	100 m, geschirmtes, verdrilltes Leiterpaar

A.3.4.1 Schrittantwort der integrierten analogen Eingänge der CPU

Tabelle A- 39 Schrittantwort (ms), 0 V bis 10 V gemessen bei 95 %

Auswahl der Glättung (Mittelwertbildung aus Abtastwerten)	Unterdrückungsfrequenz (Integrationszeit)		
	60 Hz	50 Hz	10 Hz
Keine (1 Zyklus): Keine Mittelwertbildung	50 ms	50 ms	100 ms
Schwach (4 Zyklen): 4 Abtastwerte	60 ms	70 ms	200 ms
Mittel (16 Zyklen): 16 Abtastwerte	200 ms	240 ms	1150 ms
Stark (32 Zyklen): 32 Abtastwerte	400 ms	480 ms	2300 ms
Abtastzeit	**4,17 ms**	**5 ms**	**25 ms**

A.3.4.2 Abtastzeit der integrierten analogen Ports der CPU

Tabelle A- 40 Abtastzeit der integrierten analogen Eingänge der CPU

Unterdrückungsfrequenz (Auswahl Integrationszeit)	Abtastzeit
60 Hz (16,6 ms)	4,17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

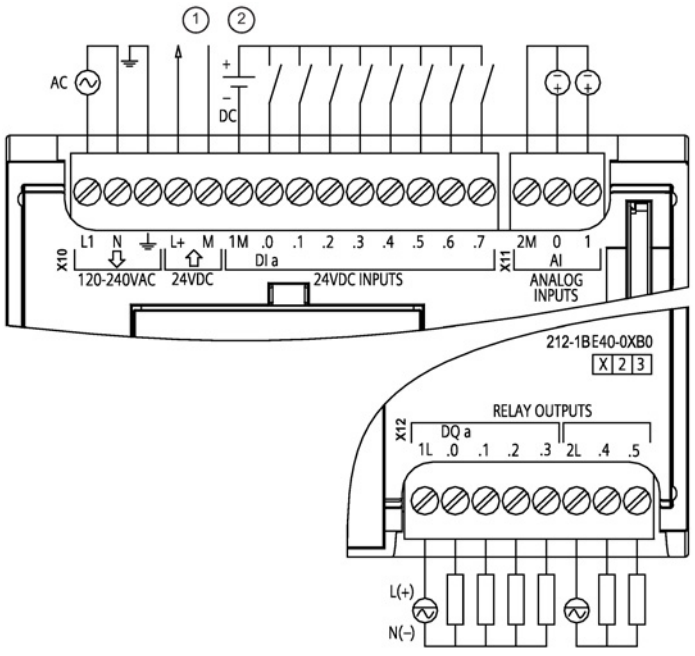
A.3.4.3 Messbereiche der analogen Eingänge für Spannung (CPUs)

Tabelle A- 41 Darstellung Analogeingang für Spannung (CPUs)

System		Messbereich Spannung	
Dezimal	Hexadezimal	0 bis 10 V	
32767	7FFF	11,851 V	Überlauf
32512	7F00		
32511	7EFF	11,759 V	Überschwingbereich
27649	6C01		
27648	6C00	10 V	Bemessungsbereich
20736	5100	7,5 V	
34	22	12 mV	
0	0	0 V	
Negative Werte		Negative Werte werden nicht unterstützt	

A.3.5 Schaltpläne der CPU 1212C

Tabelle A- 42 CPU 1212C AC/DC/Relais (6ES7 212-1BE40-0XB0)



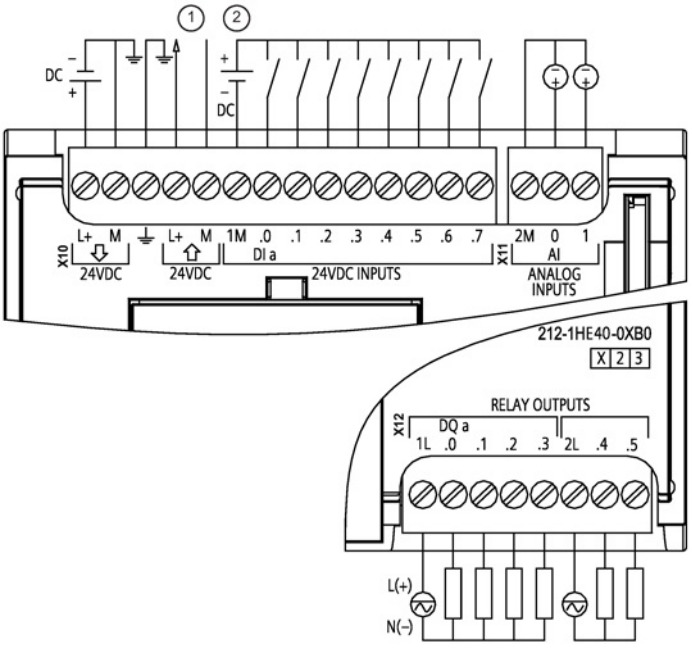
①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Eine der beiden Klemmen L1 oder N (L2) kann an eine Spannungsquelle bis 240 V AC angeschlossen werden. Die Klemme N kann als L2 betrachtet werden und muss nicht geerdet werden. Für die Klemmen L1 und N (L2) ist kein Verpolungsschutz erforderlich.	
Hinweis 3: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 43 Anschlussbelegung für die CPU 1212C AC/DC/Relais (6ES7 212-1BE40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L1 / 120-240 V AC	2 M	1L
2	N / 120-240 V AC	AI 0	DO a.0
3	Funktionserde	AI 1	DO a.1
4	L+ / 24-V-DC-Sensorausgang	--	DO a.2
5	M / 24-V-DC-Sensorausgang	--	DO a.3

Pin	X10	X11 (vergoldet)	X12
6	1M	--	2L
7	DI a.0	--	DO a.4
8	DI a.1	--	DO a.5
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	DI a.6	--	--
14	DI a.7	--	--

Tabelle A- 44 CPU 1212C DC/DC/Relais (6ES7 212-1HE40-0XB0)



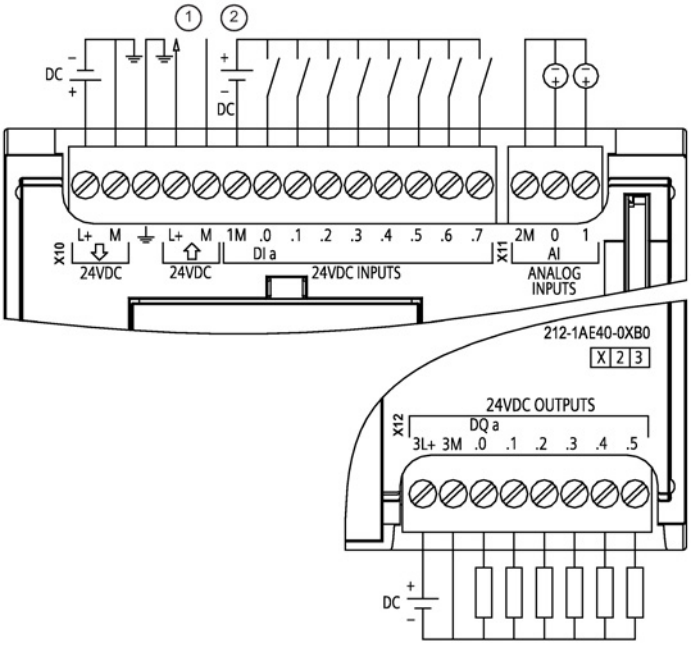
①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 45 Anschlussbelegung für die CPU 1212C DC/DC/Relais (6ES7 212-1HE40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L+ / 24 V DC	2 M	1L
2	M / 24 V DC	AI 0	DO a.0
3	Funktionserde	AI 1	DO a.1
4	L+ / 24-V-DC-Sensorausgang	--	DO a.2

Pin	X10	X11 (vergoldet)	X12
5	M / 24-V-DC-Sensorausgang	--	DO a.3
6	1M	--	2L
7	DI a.0	--	DO a.4
8	DI a.1	--	DO a.5
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	DI a.6	--	--
14	DI a.7	--	--

Tabelle A- 46 CPU 1212C DC/DC/DC (6ES7-212-1AE40-0XB0)



①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 47 Anschlussbelegung für die CPU 1212C DC/DC/DC (6ES7 212-1AE40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L+ / 24 V DC	2 M	3L+
2	M / 24 V DC	AI 0	3M
3	Funktionserde	AI 1	DO a.0

Pin	X10	X11 (vergoldet)	X12
4	L+ / 24-V-DC-Sensorausgang	--	DO a.1
5	M / 24-V-DC-Sensorausgang	--	DO a.2
6	1M	--	DO a.3
7	DI a.0	--	DO a.4
8	DI a.1	--	DO a.5
9	DI a.2	--	--
10	DI a.3	--	--
11	DI a.4	--	--
12	DI a.5	--	--
13	DI a.6	--	--
14	DI a.7	--	--

Hinweis

Nicht verwendete analoge Eingänge sollten kurzgeschlossen werden.

A.4 CPU 1214C

A.4.1 Allgemeine technische Daten und Leistungsmerkmale

Tabelle A- 48 Allgemein

Technische Daten	CPU 1214C AC/DC/Relais	CPU 1214C DC/DC/Relais	CPU 1214C DC/DC/DC
Artikelnummer	6ES7 214-1BG40-0XB0	6ES7 214-1HG40-0XB0	6ES7 214-1AG40-0XB0
Abmessungen (B x H x T) (mm)	110 x 100 x 75		
Versandgewicht	475 Gramm	435 Gramm	415 Gramm
Leistungsverlust	14 W	12 W	
Verfügbarer Strom (SM- und CM-Bus)	max. 1600 mA (5 V DC)		
Verfügbarer Strom (24 V DC)	max. 400 mA (Geberversorgung)		
Stromaufnahme digitaler Eingang (24 V DC)	4 mA/Eingang		

Tabelle A- 49 CPU-Merkmale

Technische Daten		Beschreibung
Anwenderspeicher (Siehe "Allgemeine technische Daten" (Seite 1151), "Speicherung im internen CPU-Speicher".)	Arbeitsspeicher	100 KB
	Last	4 MB intern, erweiterbar bis auf SD-Kartengröße
	Remanent	10 KB
Integrierte digitale E/A		14 Eingänge/10 Ausgänge
Integrierte analoge E/A		2 Eingänge
Größe des Prozessabbilds		1024 Bytes Eingänge (E)/1024 Bytes Ausgänge (A)
Merker (M)		8192 Byte
Temporärer (lokaler) Speicher		• 16 KB für Anlauf und Programmzyklus (einschließlich der zugehörigen FBs und FCs) • 6 KB für jede andere Alarmprioritätsstufe (einschließlich FBs und FCs)
Zusätzliche Signalmodule		max. 8 SMs
Erweiterung SB, CB, BB		max. 1
Zusätzliche Kommunikationsmodule		max. 3 CM

Technische Daten	Beschreibung
Schnelle Zähler	Bis zu 6 konfiguriert für die Verwendung beliebiger integrierter oder SB-Eingänge. Siehe Tabelle CPU1214C: Zuweisungen der HSC-Standardadressen (Seite 471) • 100/[1]80 kHz (Ea.0 bis Ea.5) • 30/[1]20 kHz (Ea.6 bis Eb.5)
Impulsausgänge [2]	Bis zu 4 konfiguriert für die Verwendung beliebiger integrierter oder SB-Ausgänge • 100 kHz (Aa.0 bis Aa.3) • 30 kHz (Aa.4 bis Ab.1)
Eingänge für Impulsabgriff	14
Verzögerungsalarme	4 mit Auflösung von 1 ms
Weckalarme	4 mit Auflösung von 1 ms
Flankenalarme	12 steigend und 12 fallend (16 und 16 mit optionalem Signalboard)
Memory Card	SIMATIC Memory Card (optional)
Genauigkeit Echtzeituhr	+/- 60 Sekunden/Monat
Pufferung Echtzeituhr	Typ. 20 Tage/min. 12 Tage bei 40 °C (wartungsfreier Hochleistungskondensator)

[1] Die langsamere Geschwindigkeit gilt, wenn der HSC als A/B-Zähler konfiguriert ist.

[2] Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge zu verwenden.

Tabelle A- 50 Leistung

Art der Anweisung	Ausführungsgeschwindigkeit
Boolesch	0,08 µs/Operation
Wort übertragen	1,7 µs/Operation
Realzahlenarithmetik	2,3 µs/Operation

A.4.2 Von der CPU 1214C unterstützte Zeiten, Zähler und Codebausteine

Tabelle A- 51 Von der CPU 1214C unterstützte Bausteine, Zeiten und Zähler

Element		Beschreibung
Bausteine	Typ	OB, FB, FC, DB
	Größe	64 KB
	Anzahl	Bis 1024 Bausteine gesamt (OBs + FBs + FCs + DBs)
	Adressbereich für FBs, FCs und DBs	FB und FC: 1 bis 65535 (z. B. FB 1 bis FB 65535) DB: 1 bis 59999
	Schachtelungstiefe	16 aus Zyklus- oder Anlauf-OBs 6 aus beliebigen Alarmereignis-OBs
	Überwachung	Der Zustand von 2 Codebausteinen kann gleichzeitig überwacht werden.
OBs	Programmzyklus	Mehrere

Element		Beschreibung
	Anlauf	Mehrere
	Verzögerungsalarme	4 (1 pro Ereignis)
	Weckalarme	4 (1 pro Ereignis)
	Prozessalarme	50 (1 pro Ereignis)
	Zeitfehleralarme	1
	Diagnosefehleralarme	1
	Ziehen oder Stecken von Modulen	1
	Fehler bei Baugruppenträger oder Station	1
	Tageszeit	Mehrere
	Status	1
	Update	1
	Profil	1
Zeiten	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, 16 Bytes pro Zeit
Zähler	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, Größe abhängig von der Zählart • SInt, USInt: 3 Bytes • Int, UInt: 6 Bytes • DInt, UDInt: 12 Bytes

Tabelle A- 52 Kommunikation

Technische Daten	Beschreibung
Schnittstellen	1
Typ	Ethernet
HMI-Gerät	4
Programmiergerät (PG)	1
Anschlüsse	• 8 für die offene Benutzerkommunikation (aktiv oder passiv): TSEND_C, TRCV_C, TCON, TDISCON, TSEND und TRCV • 3 für die S7-Kommunikation über Server-GET/PUT (CPU-zu-CPU) • 8 für die S7-Kommunikation über Client-GET/PUT (CPU-zu-CPU)
Datenraten	10/100 MBit/s
Elektrische Trennung (externes Signal zu PLC-Logik)	Wandler potentialgetrennt, 1500 V AC, nur für kurzfristige Sicherheit
Kabelart	CAT5e geschirmt

Tabelle A- 53 Stromversorgung

Technische Daten		CPU 1214C AC/DC/Relais	CPU 1214C DC/DC/Relais	CPU 1214C DC/DC/DC
Spannungsbereich		85 bis 264 V AC	20,4 V DC bis 28,8 V DC	
Netzfrequenz		47 ... 63 Hz	--	
Einschalt-strom (max. Last)	Nur CPU	100 mA bei 120 V AC 50 mA bei 240 V AC	500 mA bei 24 V DC	
	CPU mit allen Erweite-rungsbaugruppen	300 mA bei 120 V AC 150 mA bei 240 V AC	1500 mA bei 24 V DC	
Einschaltstrom (max.)		20 A bei 264 V AC	12 A bei 28,8 V DC	
Elektrische Trennung (Eingangsleistung zu Logik)		1500 V AC	Nicht elektrisch getrennt	
Kriechstrom an Erde, AC-Leitung an Funktionserde		max. 0.5 mA	-	
Verzögerungszeit (Spannungsverlust)		20 ms bei 120 V AC 80 ms bei 240 V AC	10 ms bei 24 V DC	
Interne Sicherung, nicht durch Anwender austauschbar		3 A, 250 V, träge		

Tabelle A- 54 Geberversorgung

Technische Daten	CPU 1214C AC/DC/Relais	CPU 1214C DC/DC/Relais	CPU 1214C DC/DC/DC
Spannungsbereich	20,4 bis 28,8 V DC	L+ minus 4 V DC min.	
Nennausgangsstrom (max.)	400 mA (kurzschlussfest)		
Max. Welligkeit/Störströme (<10 MHz)	< 1 V Spitze-Spitze	Wie Eingangsleitung	
Elektrische Trennung (CPU-Logik zu Sensorversorgung)	Nicht elektrisch getrennt		

A.4.3 Digitale Eingänge und Ausgänge

Tabelle A- 55 Digitale Eingänge

Technische Daten	CPU 1214C AC/DC/Relais	CPU 1214C DC/DC/Relais	CPU 1214C DC/DC/DC
Anzahl der Eingänge	14		
Typ	Stromziehend/stromliefernd (IEC Typ 1, wenn stromziehend)		
Nennspannung	24 V DC bei 4 mA, Nennwert		
Zulässige Dauerspannung	max. 30 V DC		
Stoßspannung	35 V DC für 0,5 s		
Signal logisch 1 (min.)	15 V DC bei 2,5 mA		
Signal logisch 0 (max.)	5 V DC bei 1 mA		
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute		
Potenzialgetrennte Gruppen	1		
Filterzeiten	µs-Einstellungen: 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0 ms-Einstellungen: 0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0		
HSC Eingangstaktfrequenzen (max.) (Pegel logisch 1 = 15 bis 26 V DC)	100/80 kHz (Ea.0 bis Ea.5) 30/20 kHz (Ea.6 bis Eb.5)		
Anzahl gleichzeitig eingeschalteter Eingänge	• 7 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 14 bei 55 °C horizontal oder 45 °C vertikal		
Leitungslänge (Meter)	500 m geschirmt, 300 m ungeschirmt, 50 m geschirmt für HSC-Eingänge		

Tabelle A- 56 Digitale Ausgänge

Technische Daten	CPU 1214C AC/DC/Relais und DC/DC/Relais	CPU 1214C DC/DC/DC
Ausgänge	10	
Typ	Relais, mechanisch	MOSFET, elektronisch (stromliefernd)
Spannungsbereich	5 bis 30 V DC oder 5 bis 250 V AC	20,4 bis 28,8 V DC
Signal logisch 1 bei max. Strom	--	min. 20 V DC
Signal logisch 0 bei 10 kΩ Last	--	max. 0,1 V DC
Strom (max.)	2,0 A	0,5 A
Lampenlast	30 W DC/200 W AC	5 W
Widerstand bei EIN	max. 0,2 Ω, wenn neu	max. 0,6 Ω
Kriechstrom pro Ausgang	--	max. 10 µA
Einschaltstrom	7 A bei geschlossenen Kontakten	8 A für max. 100 ms
Überlastschutz	Nein	
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute (Spule zu Kontakt) Keine (Spule zu Logik)	500 V AC für 1 Minute
Isolationswiderstand	min. 100 MΩ, wenn neu	--
Elektrische Trennung zwischen offenen Kontakten	750 V AC für 1 Minute	--

Technische Daten	CPU 1214C AC/DC/Relais und DC/DC/Relais	CPU 1214C DC/DC/DC
Potentialgetrennte Gruppen	2	1
Elektrische Trennung (Gruppe zu Gruppe)	1500 V AC[1]	--
Induktive Klemmspannung	--	L+ minus 48 V DC, 1 W Verlustleistung
Schaltverzögerung (Aa.0 bis Aa.3)	max. 10 ms	max. 1,0 µs von Aus nach Ein max. 3,0 µs von Ein nach Aus
Schaltverzögerung (Aa.4 bis Ab.1)	max. 10 ms	max. 50 µs von Aus nach Ein max. 200 µs von Ein nach Aus
Maximale Schaltfrequenz Relais	1 Hz	--
Frequenz Impulsgenerator	Nicht empfohlen [2]	100 kHz (Aa.0 bis Aa.3)[3], 2 Hz min. 20 kHz (Aa.4 bis Ab.1)[3]
Mechanische Lebensdauer (ohne Last)	10.000.000 Schaltspiele auf/zu	--
Lebensdauer der Kontakte bei Nennlast	100.000 Schaltspiele auf/zu	--
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)	
Anzahl gleichzeitig eingeschalteter Ausgänge	• 5 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 10 bei 55 °C horizontal oder 45 °C vertikal	
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt	

[1] Die elektrische Trennung zwischen Gruppen von Relais trennt die Leitungsspannung von SELV/PELV und trennt unterschiedliche Phasen bis zu 250 V AC von der Leitung zur Erde.

[2] Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge verwenden zu können.

[3] Je nach Impulsempfänger und Kabel kann ein zusätzlicher Lastwiderstand (bei mindestens 10 % des Nennstroms) die Qualität der Impulssignale und die Störfestigkeit verbessern.

A.4.4 Analoge Eingänge

Tabelle A- 57 Analoge Eingänge

Technische Daten	Beschreibung
Anzahl der Eingänge	2
Typ	Spannung (Eintakteingang)
Vollausschlagsbereich	0 bis 10 V
Vollausschlag (Datenwort)	0 bis 27648
Überschwingbereich	10,001 bis 11,759 V
Überschwingbereich (Datenwort)	27649 bis 32511
Überlaufbereich	11,760 bis 11,852 V
Überlaufbereich (Datenwort)	32512 bis 32767
Auflösung	10 Bits
Max. Stehspannung	35 V DC
Glättung	Keine, schwach, mittel oder stark Siehe Tabelle der Schrittantwort (ms) für die analogen Eingänge der CPU (Seite 1193).
Rauschunterdrückung	10, 50 oder 60 Hz
Impedanz	≥100 kΩ
Elektrische Trennung (Feld zu Logik)	Keine
Genauigkeit (25 °C/0 bis 55 °C)	3,0% / 3,5% des Vollausschlags
Leitungslänge (Meter)	100 m, geschirmtes, verdrilltes Leiterpaar

A.4.4.1 Schrittantwort der integrierten analogen Eingänge der CPU

Tabelle A- 58 Schrittantwort (ms), 0 V bis 10 V gemessen bei 95 %

Auswahl der Glättung (Mittelwertbildung aus Abtastwerten)	Unterdrückungsfrequenz (Integrationszeit)		
	60 Hz	50 Hz	10 Hz
Keine (1 Zyklus): Keine Mittelwertbildung	50 ms	50 ms	100 ms
Schwach (4 Zyklen): 4 Abtastwerte	60 ms	70 ms	200 ms
Mittel (16 Zyklen): 16 Abtastwerte	200 ms	240 ms	1150 ms
Stark (32 Zyklen): 32 Abtastwerte	400 ms	480 ms	2300 ms
Abtastzeit	**4,17 ms**	**5 ms**	**25 ms**

A.4.4.2 Abtastzeit der integrierten analogen Ports der CPU

Tabelle A- 59 Abtastzeit der integrierten analogen Eingänge der CPU

Unterdrückungsfrequenz (Auswahl Integrationszeit)	Abtastzeit
60 Hz (16,6 ms)	4,17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

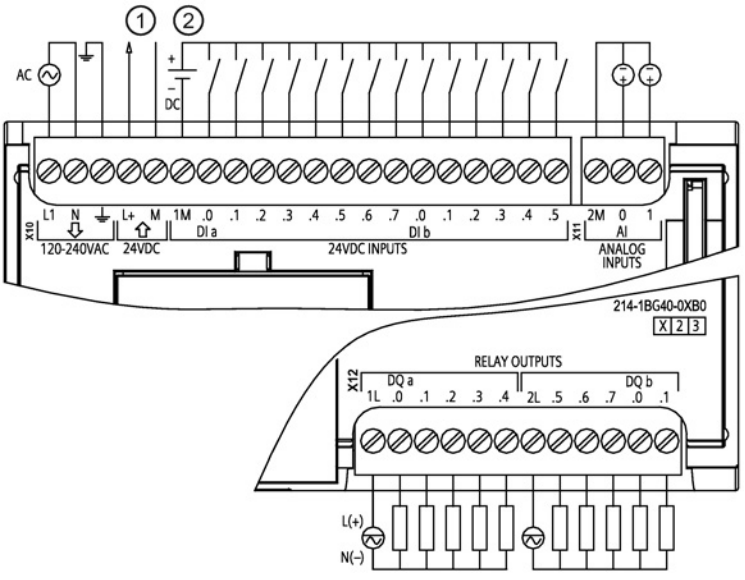
A.4.4.3 Messbereiche der analogen Eingänge für Spannung (CPUs)

Tabelle A- 60 Darstellung Analogeingang für Spannung (CPUs)

System		Messbereich Spannung	
Dezimal	Hexadezimal	0 bis 10 V	
32767	7FFF	11,851 V	Überlauf
32512	7F00		
32511	7EFF	11,759 V	Überschwingbereich
27649	6C01		
27648	6C00	10 V	Bemessungsbereich
20736	5100	7,5 V	
34	22	12 mV	
0	0	0 V	
Negative Werte		Negative Werte werden nicht unterstützt	

A.4.5 Schaltpläne der CPU 1214C

Tabelle A- 61 CPU 1214C AC/DC/Relais (6ES7 214-1BG40-0XB0)



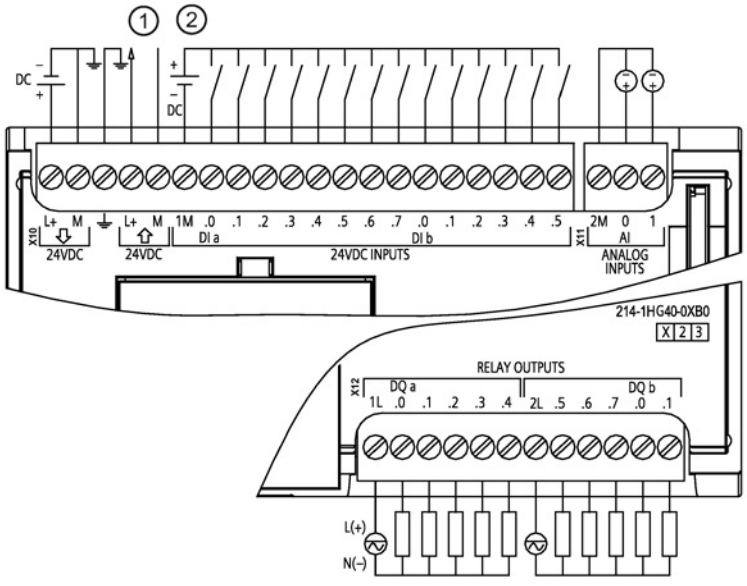
①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Eine der beiden Klemmen L1 oder N (L2) kann an eine Spannungsquelle bis 240 V AC angeschlossen werden. Die Klemme N kann als L2 betrachtet werden und muss nicht geerdet werden. Für die Klemmen L1 und N (L2) ist kein Verpolungsschutz erforderlich.	
Hinweis 3: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 62 Anschlussbelegung für die CPU 1214C AC/DC/Relais (6ES7 214-1BG40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L1 / 120-240 V AC	2 M	1L
2	N / 120-240 V AC	AI 0	DO a.0
3	Funktionserde	AI 1	DO a.1
4	L+ / 24-V-DC-Sensorausgang	--	DO a.2
5	M / 24-V-DC-Sensorausgang	--	DO a.3

Pin	X10	X11 (vergoldet)	X12
6	1M	--	DO a.4
7	DI a.0	--	2L
8	DI a.1	--	DO a.5
9	DI a.2	--	DO a.6
10	DI a.3	--	DO a.7
11	DI a.4	--	DO b.0
12	DI a.5	--	DO b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

Tabelle A- 63 CPU 1214C DC/DC/Relais (6ES7 214-1HG40-0XB0)

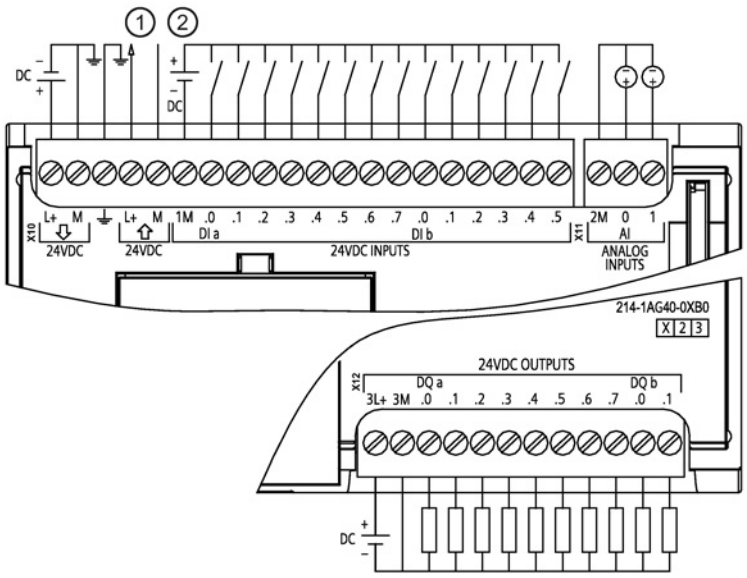


①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 64 Anschlussbelegung für die CPU 1214C DC/DC/Relais (6ES7 214-1HG40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L+ / 24 V DC	2 M	1L
2	M / 24 V DC	AI 0	DO a.0
3	Funktionserde	AI 1	DO a.1
4	L+ / 24-V-DC-Sensorausgang	--	DO a.2
5	M / 24-V-DC-Sensorausgang	--	DO a.3
6	1M	--	DO a.4
7	DI a.0	--	2L
8	DI a.1	--	DO a.5
9	DI a.2	--	DO a.6
10	DI a.3	--	DO a.7
11	DI a.4	--	DO b.0
12	DI a.5	--	DO b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

Tabelle A- 65 CPU 1214C DC/DC/DC (6ES7 214-1AG40-0XB0)



①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 66 Anschlussbelegung für die CPU 1214C DC/DC/DC (6ES7 214-1AG40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L+ / 24 V DC	2 M	3L+
2	M / 24 V DC	AI 0	3M
3	Funktionserde	AI 1	DO a.0
4	L+ / 24-V-DC-Sensorausgang	--	DO a.1
5	M / 24-V-DC-Sensorausgang	--	DO a.2
6	1M	--	DO a.3
7	DI a.0	--	DO a.4
8	DI a.1	--	DO a.5
9	DI a.2	--	DO a.6
10	DI a.3	--	DO a.7
11	DI a.4	--	DO b.0
12	DI a.5	--	DO b.1
13	DI a.6	--	--
14	DI a.7	--	-
15	DI b.0	--	--
16	DI b.1	--	--

Pin	X10	X11 (vergoldet)	X12
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

Hinweis

Nicht verwendete analoge Eingänge sollten kurzgeschlossen werden.

A.5 CPU 1215C

A.5.1 Allgemeine technische Daten und Leistungsmerkmale

Tabelle A- 67 Allgemein

Technische Daten	CPU 1215C AC/DC/Relais	CPU 1215C DC/DC/Relais	CPU 1215C DC/DC/DC
Artikelnummer	6ES7 215-1BG40-0XB0	6ES7 215-1HG40-0XB0	6ES7 215-1AG40-0XB0
Abmessungen (B x H x T) (mm)	130 x 100 x 75		
Versandgewicht	585 Gramm	550 Gramm	520 Gramm
Leistungsverlust	14 W	12 W	
Verfügbarer Strom (SM- und CM-Bus)	max. 1600 mA (5 V DC)		
Verfügbarer Strom (24 V DC)	max. 400 mA (Geberversorgung)		
Stromaufnahme digitaler Eingang (24 V DC)	4 mA/Eingang		

Tabelle A- 68 CPU-Merkmale

Technische Daten		Beschreibung
Anwenderspeicher (Siehe "Allgemeine technische Daten" (Seite 1151), "Speicherung im internen CPU-Speicher".)	Arbeitsspeicher	125 KB
	Ladespeicher	4 MB intern, erweiterbar bis auf SD-Kartengröße
	Remanent	10 KB
Integrierte digitale E/A		14 Eingänge/10 Ausgänge

Technische Daten	Beschreibung
Integrierte analoge E/A	2 Eingänge/2 Ausgänge
Größe des Prozessabbilds	1024 Byte Eingänge (E)/1024 Byte Ausgänge (A)
Merker (M)	8192 Byte
Temporärer (lokaler) Speicher	• 16 KB für Anlauf und Programmzyklus (einschließlich der zugehörigen FBs und FCs) • 6 KB für jede andere Alarmprioritätsstufe (einschließlich FBs und FCs)
Zusätzliche Signalmodule	max. 8 SMs
Erweiterung SB, CB, BB	max. 1
Zusätzliche Kommunikationsmodule	max. 3 CM
Schnelle Zähler	Bis zu 6 konfiguriert für die Verwendung beliebiger integrierter oder SB-Eingänge. Siehe Tabelle CPU 1215C: Zuweisungen der HSC-Standardadressen (Seite 471) • 100/[1]80 kHz (Ea.0 bis Ea.5) • 30/[1]20 kHz (Ea.6 bis Eb.5)
Impulsausgänge [2]	Bis zu 4 konfiguriert für die Verwendung beliebiger integrierter oder SB-Ausgänge • 100 kHz (Aa.0 bis Aa.3) • 30 kHz (Aa.4 bis Ab.1)
Eingänge für Impulsabgriff	14
Verzögerungsalarme	4 mit Auflösung von 1 ms
Weckalarme	4 mit Auflösung von 1 ms
Flankenalarme	12 steigend und 12 fallend (16 und 16 mit optionalem Signalboard)
Memory Card	SIMATIC Memory Card (optional)
Genauigkeit Echtzeituhr	+/- 60 Sekunden/Monat
Pufferung Echtzeituhr	Typ. 20 Tage/min. 12 Tage bei 40 °C (wartungsfreier Hochleistungskondensator)

[1] Die langsamere Geschwindigkeit gilt, wenn der HSC als A/B-Zähler konfiguriert ist.

[2] Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge zu verwenden.

Tabelle A- 69 Leistung

Art der Anweisung	Ausführungsgeschwindigkeit
Boolesch	0,08 µs/Operation
Wort übertragen	1,7 µs/Operation
Realzahlenarithmetik	2,3 µs/Operation

A.5.2 Von der CPU 1215C unterstützte Zeiten, Zähler und Codebausteine

Tabelle A- 70 Von der CPU 1215C unterstützte Bausteine, Zeiten und Zähler

Element		Beschreibung
Bausteine	Typ	OB, FB, FC, DB
	Größe	64 kBytes
	Anzahl	Bis 1024 Bausteine gesamt (OBs + FBs + FCs + DBs)
	Adressbereich für FBs, FCs und DBs	FB und FC: 1 bis 65535 (z. B. FB 1 bis FB 65535) DB: 1 bis 59999
	Schachtelungstiefe	16 aus Zyklus- oder Anlauf-OBs 6 aus beliebigen Alarmereignis-OBs
	Überwachung	Der Zustand von 2 Codebausteinen kann gleichzeitig überwacht werden.
OBs	Programmzyklus	Mehrere
	Anlauf	Mehrere
	Verzögerungsalarme	4 (1 pro Ereignis)
	Weckalarme	4 (1 pro Ereignis)
	Prozessalarme	50 (1 pro Ereignis)
	Zeitfehleralarme	1
	Diagnosefehleralarme	1
	Ziehen oder Stecken von Modulen	1
	Fehler bei Baugruppenträger oder Station	1
	Tageszeit	Mehrere
	Status	1
	Update	1
	Profil	1
Zeiten	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, 16 Bytes pro Zeit
Zähler	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, Größe abhängig von der Zählart • SInt, USInt: 3 Bytes • Int, UInt: 6 Bytes • DInt, UDInt: 12 Bytes

Tabelle A- 71 Kommunikation

Technische Daten	Beschreibung
Schnittstellen	2
Typ	Ethernet
HMI-Gerät	4
Programmiergerät (PG)	1
Anschlüsse	• 8 für die offene Benutzerkommunikation (aktiv oder passiv): TSEND_C, TRCV_C, TCON, TDISCON, TSEND und TRCV • 3 für die S7-Kommunikation über Server-GET/PUT (CPU-zu-CPU) • 8 für die S7-Kommunikation über Client-GET/PUT (CPU-zu-CPU)
Datenraten	10/100 MBit/s
Elektrische Trennung (externes Signal zu PLC-Logik)	Wandler potenzialgetrennt, 1500 V AC, nur für kurzfristige Sicherheit
Kabelart	CAT5e geschirmt

Tabelle A- 72 Stromversorgung

Technische Daten		CPU 1215C AC/DC/Relais	CPU 1215C DC/DC/Relais	CPU 1215C DC/DC/DC
Spannungsbereich		85 bis 264 V AC	20,4 V DC bis 28,8 V DC	
Netzfrequenz		47 bis 63 Hz	--	
Einschalt-strom (max. Last)	Nur CPU	100 mA bei 120 V AC 50 mA bei 240 V AC	500 mA bei 24 V DC	
	CPU mit allen Erweite-rungsbaugruppen	300 mA bei 120 V AC 150 mA bei 240 V AC	1500 mA bei 24 V DC	
Einschaltstrom (max.)		20 A bei 264 V AC	12 A bei 28,8 V DC	
Elektrische Trennung (Eingangsleistung zu Logik)		1500 V AC	Nicht elektrisch getrennt	
Kriechstrom an Erde, AC-Leitung an Funktionserde		max. 0,5 mA	-	
Verzögerungszeit (Spannungsverlust)		20 ms bei 120 V AC 80 ms bei 240 V AC	10 ms bei 24 V DC	
Interne Sicherung, nicht durch Anwender austauschbar		3 A, 250 V, träge		

Tabelle A- 73 Geberversorgung

Technische Daten	CPU 1215C AC/DC/Relais	CPU 1215C DC/DC/Relais	CPU 1215C DC/DC/DC
Spannungsbereich	20,4 bis 28,8 V DC	L+ minus 4 V DC min.	
Nennausgangsstrom (max.)	400 mA (kurzschlussfest)		
Max. Welligkeit/Störströme (<10 MHz)	< 1 V Spitze-Spitze	Wie Eingangsleitung	
Elektrische Trennung (CPU-Logik zu Geberversorgung)	Nicht elektrisch getrennt		

A.5.3 Digitale Eingänge und Ausgänge

Tabelle A- 74 Digitale Eingänge

Technische Daten	CPU 1215C AC/DC/Relais	CPU 1215C DC/DC/Relais	CPU 1215C DC/DC/DC
Anzahl der Eingänge	14		
Typ	Stromziehend/stromliefernd (IEC Typ 1, wenn stromziehend)		
Nennspannung	24 V DC bei 4 mA, Nennwert		
Zulässige Dauerspannung	max. 30 V DC		
Stoßspannung	35 V DC für 0,5 s		
Signal logisch 1 (min.)	15 V DC bei 2,5 mA		
Signal logisch 0 (max.)	5 V DC bei 1 mA		
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute		
Potentialgetrennte Gruppen	1		
Filterzeiten	us-Einstellungen: 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0 ms-Einstellungen: 0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0		
HSC Eingangstaktfrequenzen (max.) (Pegel logisch 1 = 15 bis 26 V DC)	100/80 kHz (Ea.0 bis Ea.5) 30/20 kHz (Ea.6 bis Eb.5)		
Anzahl gleichzeitig eingeschalteter Eingänge	• 7 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 14 bei 55 °C horizontal oder 45 °C vertikal		
Leitungslänge (Meter)	500 m geschirmt, 300 m ungeschirmt, 50 m geschirmt für HSC-Eingänge		

Tabelle A- 75 Digitale Ausgänge

Technische Daten	CPU 1215C AC/DC/Relais und CPU 1215C DC/DC/Relais	CPU 1215C DC/DC/DC
Ausgänge	10	
Typ	Relais, mechanisch	MOSFET, elektronisch (stromliefernd)
Spannungsbereich	5 bis 30 V DC oder 5 bis 250 V AC	20,4 bis 28,8 V DC
Signal logisch 1 bei max. Strom	--	min. 20 V DC
Signal logisch 0 bei 10 kΩ Last	--	max. 0,1 V DC
Strom (max.)	2,0 A	0,5 A
Lampenlast	30 W DC/200 W AC	5 W
Widerstand bei EIN	max. 0,2 Ω wenn neu	max. 0,6 Ω
Kriechstrom pro Ausgang	--	max. 10 µA
Einschaltstrom	7 A bei geschlossenen Kontakten	8 A für max. 100 ms
Überlastschutz	Nein	
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute (Spule zu Kontakt) Keine (Spule zu Logik)	500 V AC für 1 Minute
Isolationswiderstand	min. 100 MΩ, wenn neu	--
Elektrische Trennung zwischen offenen Kontakten	750 V AC für 1 Minute	--

Technische Daten	CPU 1215C AC/DC/Relais und CPU 1215C DC/DC/Relais	CPU 1215C DC/DC/DC
Potenzialgetrennte Gruppen	2	1
Elektrische Trennung (Gruppe zu Gruppe)	1500 V AC[1]	--
Induktive Klemmspannung	--	L+ minus 48 V DC, 1 W Verlustleistung
Schaltverzögerung (Aa.0 bis Aa.3)	max. 10 ms	max. 1,0 µs von Aus nach Ein max. 3,0 µs von Ein nach Aus
Schaltverzögerung (Aa.4 bis Ab.1)	max. 10 ms	max. 50 µs von Aus nach Ein max. 200 µs von Ein nach Aus
Maximale Schaltfrequenz Relais	1 Hz	--
Frequenz Impulsgenerator	Nicht empfohlen [2]	100 kHz (Aa.0 bis Aa.3)[3], 2 Hz min. 20 kHz (Aa.4 bis Ab.1)[3]
Mechanische Lebensdauer (ohne Last)	10.000.000 Schaltspiele auf/zu	--
Lebensdauer der Kontakte bei Nennlast	100.000 Schaltspiele auf/zu	--
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)	
Anzahl gleichzeitig eingeschalteter Ausgänge	• 5 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 10 bei 55 °C horizontal oder 45 °C vertikal	
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt	

[1] Die elektrische Trennung zwischen Gruppen von Relais trennt die Leitungsspannung von SELV/PELV und trennt unterschiedliche Phasen bis zu 250 V AC von der Leitung zur Erde.

[2] Bei CPU-Varianten mit Relaisausgängen müssen Sie ein digitales Signalboard (SB) installieren, um die Impulsausgänge verwenden zu können.

[3] Je nach Impulsempfänger und Kabel kann ein zusätzlicher Lastwiderstand (bei mindestens 10 % des Nennstroms) die Qualität der Impulssignale und die Störfestigkeit verbessern.

A.5.4 Analogeingänge und -ausgänge

Tabelle A- 76 Analoge Eingänge

Technische Daten	Beschreibung
Anzahl der Eingänge	2
Typ	Spannung (Eintakteingang)
Vollausschlagsbereich	0 bis 10 V
Vollausschlag (Datenwort)	0 bis 27648
Überschwingbereich	10,001 bis 11,759 V
Überschwingbereich (Datenwort)	27649 bis 32511
Überlaufbereich	11,760 bis 11,852 V
Überlaufbereich (Datenwort)	32512 bis 32767
Auflösung	10 Bits
Max. Stehspannung	35 V DC
Glättung	Keine, schwach, mittel oder stark Siehe Tabelle der Schrittantwort (ms) für die analogen Eingänge der CPU (Seite 1205).
Rauschunterdrückung	10, 50 oder 60 Hz
Impedanz	≥100 kΩ
Elektrische Trennung (Feld zu Logik)	Keine
Genauigkeit (25 °C/0 bis 55 °C)	3,0% / 3,5% des Vollausschlags
Leitungslänge (Meter)	100 m, geschirmtes, verdrilltes Leiterpaar

A.5.4.1 Schrittantwort der integrierten analogen Eingänge der CPU

Tabelle A- 77 Schrittantwort (ms), 0 V bis 10 V gemessen bei 95 %

Auswahl der Glättung (Mittelwertbildung aus Abtastwerten)	Unterdrückungsfrequenz (Integrationszeit)		
	60 Hz	50 Hz	10 Hz
Keine (1 Zyklus): Keine Mittelwertbildung	50 ms	50 ms	100 ms
Schwach (4 Zyklen): 4 Abtastwerte	60 ms	70 ms	200 ms
Mittel (16 Zyklen): 16 Abtastwerte	200 ms	240 ms	1150 ms
Stark (32 Zyklen): 32 Abtastwerte	400 ms	480 ms	2300 ms
Abtastzeit	**4,17 ms**	**5 ms**	**25 ms**

A.5.4.2 Abtastzeit der integrierten analogen Ports der CPU

Tabelle A- 78 Abtastzeit der integrierten analogen Eingänge der CPU

Unterdrückungsfrequenz (Auswahl Integrationszeit)	Abtastzeit
60 Hz (16,6 ms)	4,17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

A.5.4.3 Messbereiche der analogen Eingänge für Spannung (CPUs)

Tabelle A- 79 Darstellung Analogeingang für Spannung (CPUs)

System		Messbereich Spannung	
Dezimal	Hexadezimal	0 bis 10 V	
32767	7FFF	11,851 V	Überlauf
32512	7F00		
32511	7EFF	11,759 V	Überschwingbereich
27649	6C01		
27648	6C00	10 V	Bemessungsbereich
20736	5100	7,5 V	
34	22	12 mV	
0	0	0 V	
Negative Werte		Negative Werte werden nicht unterstützt	

A.5.4.4 Technische Daten der Analogausgänge

Tabelle A- 80 Analoge Ausgänge

Technische Daten	Beschreibung
Ausgänge	2
Typ	Strom
Vollausschlagsbereich	0 ... 20 mA
Vollausschlag (Datenwort)	0 bis 27648
Überschwingbereich	20,01 bis 23,52 mA
Überschwingbereich (Datenwort)	27649 bis 32511
Überlaufbereich	siehe Fußnote[1]
Datenwort Überlaufbereich	32512 bis 32767
Auflösung	10 Bit
Ausgangsimpedanz	≤ 500 Ω max.
Elektrische Trennung (Feld zu Logik)	Keine
Genauigkeit (25 °C / -20 bis 60 °C)	3,0 % / 3,5 % des Vollausschlags

Technische Daten	Beschreibung
Einschwingzeit	2 ms
Leitungslänge (Meter)	100 m, geschirmtes, verdrilltes Leiterpaar

1 Bei Überlauf verhalten sich die Analogausgänge entsprechend den Eigenschaften der Gerätekonfiguration. Wählen Sie für den Parameter "Reaktion auf CPU-STOP" entweder: "Ersatzwert aufschalten" oder "Letzten Wert halten".

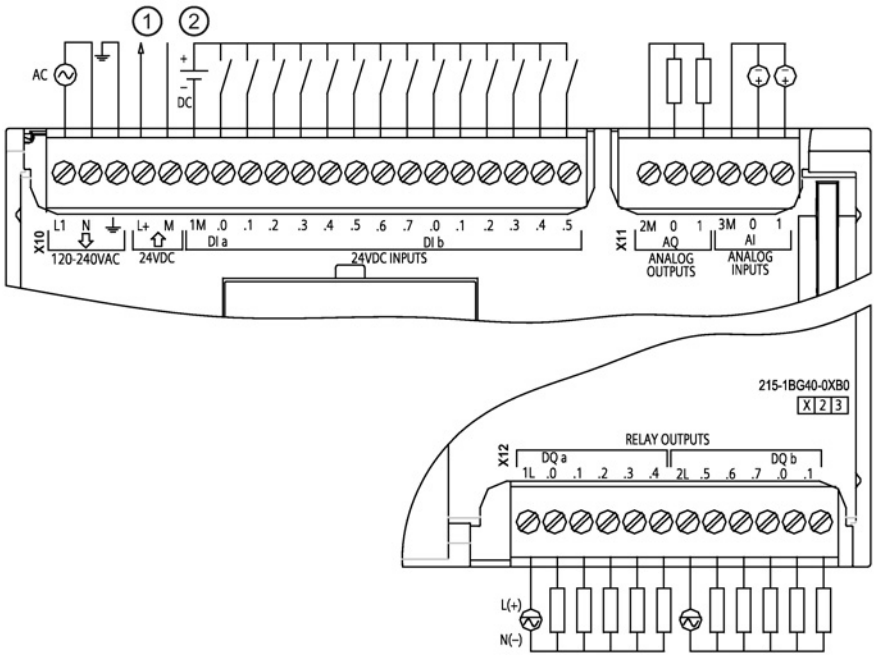
Tabelle A- 81 Darstellung Analogausgang für Strom (CPU 1215C und CPU 1217C)

System		Stromausgangsbereich	
Dezimal	**Hexadezimal**	**0 mA bis 20 mA**	
32767	7FFF	Siehe Hinweis 1	Überlauf
32512	7F00	Siehe Hinweis 1	
32511	7EFF	23,52 mA	Überschwingbereich
27649	6C01		
27648	6C00	20 mA	Bemessungsbereich
20736	5100	15 mA	
34	22	0,0247 mA	
0	0	0 mA	
Negative Werte		Negative Werte werden nicht unterstützt	

1 Bei Überlauf verhalten sich die Analogausgänge entsprechend den eingestellten Eigenschaften der Gerätekonfiguration. Wählen Sie für den Parameter "Reaktion auf CPU-STOP" entweder: "Ersatzwert aufschalten" oder "Letzten Wert halten".

A.5.5 Schaltpläne der CPU 1215C

Tabelle A- 82 CPU 1215C AC/DC/Relais (6ES7 215-1BG40-0XB0)



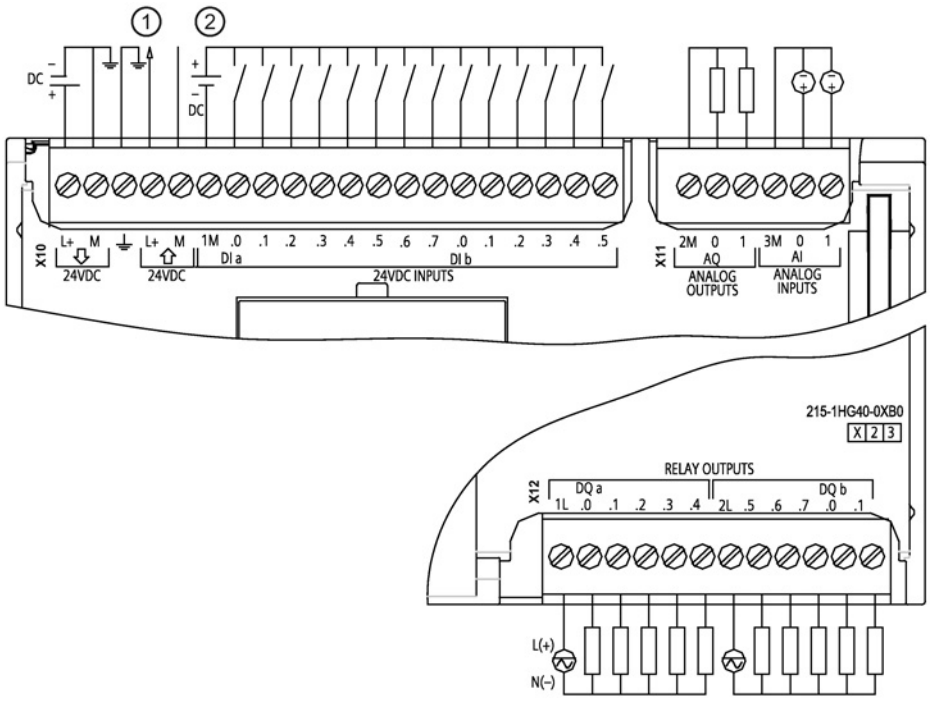
①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Eine der beiden Klemmen L1 oder N (L2) kann an eine Spannungsquelle bis 240 V AC angeschlossen werden. Die Klemme N kann als L2 betrachtet werden und muss nicht geerdet werden. Für die Klemmen L1 und N (L2) ist kein Verpolungsschutz erforderlich.	
Hinweis 3: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 83 Anschlussbelegung für die CPU 1215C AC/DC/Relais (6ES7 215-1BG40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L1 / 120-240 V AC	2 M	1L
2	N / 120-240 V AC	AO 0	DO a.0
3	Funktionserde	AO 1	DO a.1
4	L+ / 24-V-DC-Sensorausgang	3M	DO a.2
5	M / 24-V-DC-Sensorausgang	AI 0	DO a.3
6	1M	AI 1	DO a.4
7	DI a.0	--	2L

Pin	X10	X11 (vergoldet)	X12
8	DI a.1	--	DO a.5
9	DI a.2	--	DO a.6
10	DI a.3	--	DO a.7
11	DI a.4	--	DO b.0
12	DI a.5	--	DO b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

Tabelle A- 84 CPU 1215C DC/DC/Relais (6ES7 215-1HG40-0XB0)



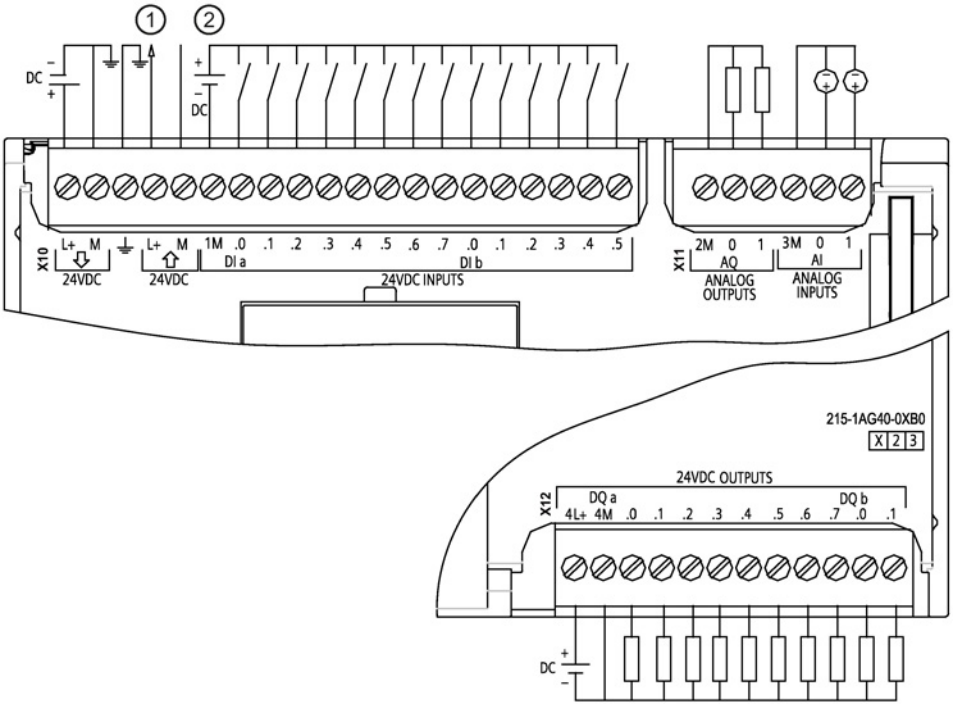
①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 85 Anschlussbelegung für die CPU 1215C DC/DC/Relais (6ES7 215-1HG40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L+ / 24 V DC	2 M	1L
2	M / 24 V DC	AO 0	DO a.0
3	Funktionserde	AO 1	DO a.1
4	L+ / 24-V-DC-Sensorausgang	3M	DO a.2
5	M / 24-V-DC-Sensorausgang	AI 0	DO a.3
6	1M	AI 1	DO a.4
7	DI a.0	--	2L
8	DI a.1	--	DO a.5
9	DI a.2	--	DO a.6
10	DI a.3	--	DO a.7
11	DI a.4	--	DO b.0

Pin	X10	X11 (vergoldet)	X12
12	DI a.5	--	DO b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

Tabelle A- 86 CPU 1215C DC/DC/DC (6ES7 215-1AG40-0XB0)



①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
Hinweis 1: X11-Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 87 Anschlussbelegung für die CPU 1215C DC/DC/DC (6ES7 215-1AG40-0XB0)

Pin	X10	X11 (vergoldet)	X12
1	L1 / 24 V DC	2 M	4L+
2	M / 24 V DC	AO 0	4M
3	Funktionserde	AO 1	DO a.0
4	L+ / 24-V-DC-Sensorausgang	3M	DO a.1
5	M / 24-V-DC-Sensorausgang	AI 0	DO a.2
6	1M	AI 1	DO a.3
7	DI a.0	--	DO a.4
8	DI a.1	--	DO a.5
9	DI a.2	--	DO a.6
10	DI a.3	--	DO a.7
11	DI a.4	--	DO b.0
12	DI a.5	--	DO b.1
13	DI a.6	--	--
14	DI a.7	--	--
15	DI b.0	--	--
16	DI b.1	--	--
17	DI b.2	--	--
18	DI b.3	--	--
19	DI b.4	--	--
20	DI b.5	--	--

Hinweis

Nicht verwendete analoge Eingänge sollten kurzgeschlossen werden.

A.6 CPU 1217C

A.6.1 Allgemeine technische Daten und Leistungsmerkmale

Tabelle A- 88 Allgemein

Technische Daten	CPU 1217C DC/DC/DC
Artikelnummer	6ES7 217-1AG40-0XB0
Abmessungen (B x H x T) (mm)	150 x 100 x 75
Versandgewicht	530 Gramm
Leistungsverlust	12 W
Verfügbarer Strom (SM- und CM-Bus)	max. 1600 mA (5 V DC)
Verfügbarer Strom (24 V DC)	max. 400 mA (Geberversorgung)
Stromaufnahme digitaler Eingang (24 V DC)	4 mA/Eingang

Tabelle A- 89 CPU-Merkmale

Technische Daten		Beschreibung
Anwenderspeicher (Siehe "Allgemeine technische Daten" (Seite 1151)", "Speicherung im internen CPU-Speicher".)	Arbeitsspeicher	150 KB
	Ladespeicher	4 MB, intern, erweiterbar bis auf SD-Kartengröße
	Remanent	10 KB
Integrierte digitale E/A		14 Eingänge/10 Ausgänge
Integrierte analoge E/A		2 Eingänge/2 Ausgänge
Größe des Prozessabbilds		1024 Bytes Eingänge (E)/1024 Bytes Ausgänge (A)
Merker (M)		8192 Byte
Temporärer (lokaler) Speicher		• 16 KB für Anlauf und Programmzyklus (einschließlich der zugehörigen FBs und FCs) • 6 KB für jede andere Alarmprioritätsstufe (einschließlich FBs und FCs)
Zusätzliche Signalmodule		max. 8 SMs
Erweiterung SB, CB, BB		max. 1
Zusätzliche Kommunikationsmodule		max. 3 CM

Technische Daten	Beschreibung
Schnelle Zähler	Bis zu 6 konfiguriert für die Verwendung beliebiger integrierter oder SB-Eingänge (siehe HW-Konfigurationstabelle CPU 1217C Digitaleingang (DI)) (Seite 1217) • 1 MHz (Eb.2 bis Eb.5) • 100/[1]80 kHz (Ea.0 bis Ea.5) • 30/[1]20 kHz (Ea.6 bis Eb.1)
Impulsausgänge	Bis zu 4 konfiguriert für die Verwendung beliebiger integrierter oder SB-Ausgänge (siehe HW-Konfigurationstabelle CPU 1217C Digitalausgang (DO)) (Seite 1217) • 1 MHz (Aa.0 bis Aa.3) • 100 kHz (Aa.4 bis Ab.1)
Eingänge für Impulsabgriff	14
Verzögerungsalarme	4 mit Auflösung von 1 ms
Weckalarme	4 mit Auflösung von 1 ms
Flankenalarme	12 steigend und 12 fallend (16 und 16 mit optionalem Signalboard)
Memory Card	SIMATIC Memory Card (optional)
Genauigkeit Echtzeituhr	+/- 60 Sekunden/Monat
Pufferung Echtzeituhr	Typ. 20 Tage/min. 12 Tage bei 40 °C (wartungsfreier Hochleistungskondensator)

[1] Die langsamere Geschwindigkeit gilt, wenn der HSC als A/B-Zähler konfiguriert ist.

Tabelle A- 90 Leistung

Art der Anweisung	Ausführungsgeschwindigkeit
Boolesch	0,08 µs/Operation
Wort übertragen	1,7 µs/Operation
Realzahlenarithmetik	2,3 µs/Operation

A.6.2 Von der CPU 1217C unterstützte Zeiten, Zähler und Codebausteine

Tabelle A- 91 Von der CPU 1217C unterstützte Bausteine, Zeiten und Zähler

Element		Beschreibung
Bausteine	Typ	OB, FB, FC, DB
	Größe	64 kBytes
	Anzahl	Bis 1024 Bausteine gesamt (OBs + FBs + FCs + DBs)
	Adressbereich für FBs, FCs und DBs	FB und FC: 1 bis 65535 (z. B. FB 1 bis FB 65535) DB: 1 bis 59999
	Schachtelungstiefe	16 aus Zyklus- oder Anlauf-OBs 6 aus beliebigen Alarmereignis-OBs
	Überwachung	Der Zustand von 2 Codebausteinen kann gleichzeitig überwacht werden.
OBs	Programmzyklus	Mehrere
	Anlauf	Mehrere
	Verzögerungsalarme	4 (1 pro Ereignis)
	Weckalarme	4 (1 pro Ereignis)
	Prozessalarme	50 (1 pro Ereignis)
	Zeitfehleralarme	1
	Diagnosefehleralarme	1
	Ziehen oder Stecken von Modulen	1
	Fehler bei Baugruppenträger oder Station	1
	Tageszeit	Mehrere
	Status	1
	Update	1
	Profil	1
Zeiten	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, 16 Bytes pro Zeit
Zähler	Typ	IEC
	Anzahl	Nur durch die Speicherkapazität begrenzt
	Speicherung	Struktur im DB, Größe abhängig von der Zählart • SInt, USInt: 3 Bytes • Int, UInt: 6 Bytes • DInt, UDInt: 12 Bytes

Tabelle A- 92 Kommunikation

Technische Daten	Beschreibung
Schnittstellen	2
Typ	Ethernet
HMI-Gerät	4
Programmiergerät (PG)	1
Anschlüsse	• 8 für die offene Benutzerkommunikation (aktiv oder passiv): TSEND_C, TRCV_C, TCON, TDISCON, TSEND und TRCV • 3 für die S7-Kommunikation über Server-GET/PUT (CPU-zu-CPU) • 8 für die S7-Kommunikation über Client-GET/PUT (CPU-zu-CPU)
Datenraten	10/100 MBit/s
Elektrische Trennung (externes Signal zu PLC-Logik)	Wandler potentialgetrennt, 1500 V AC, nur für kurzfristige Sicherheit
Kabelart	CAT5e geschirmt

Tabelle A- 93 Stromversorgung

Technische Daten		CPU 1217C DC/DC/DC
Spannungsbereich		20,4 V DC bis 28,8 V DC
Netzfrequenz		--
Eingangsstrom (max. Last)	Nur CPU	600 mA bei 24 V DC
	CPU mit allen Erweiterungs-baugruppen	1600 mA bei 24 V DC
Einschaltstrom (max.)		12 A bei 28,8 V DC
Elektrische Trennung (Eingangsleistung zu Logik)		Nicht elektrisch getrennt
Verzögerungszeit (nach Spannungsver-lust)		10 ms bei 24 V DC
Interne Sicherung, nicht durch Anwender austauschbar		3 A, 250 V, träge

Tabelle A- 94 Geberversorgung

Technische Daten	CPU 1217C DC/DC/DC
Spannungsbereich	L+ minus 4 V DC min.
Nennausgangsstrom (max.)	400 mA (kurzschlussfest)
Max. Welligkeit/Störströme (<10 MHz)	Wie Eingangsleitung
Elektrische Trennung (CPU-Logik zu Ge-berversorgung)	Nicht elektrisch getrennt

A.6.3 Digitale Eingänge und Ausgänge

Tabelle A- 95 Digitale Eingänge

Technische Daten	CPU 1217C DC/DC/DC
Anzahl der Eingänge	14: Gesamt: 10: Stromziehend/stromliefernd (IEC Typ 1, wenn stromziehend) 4: Differential (RS422/RS485)
Typ: Stromziehend/stromliefernd (IEC Typ 1 stromziehend)	Ea.0 bis Ea.7, Eb.0 bis Eb.1
Nennspannung	24 V DC bei 4 mA, Nennwert
Zulässige Dauerspannung	max. 30 V DC
Stoßspannung	35 V DC für 0,5 s
Signal logisch 1 (min.)	15 V DC bei 2,5 mA
Signal logisch 0 (max.)	5 V DC bei 1 mA
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute (funktionelle Trennung)
Potentialgetrennte Gruppen	1
Filterzeiten	us-Einstellungen: 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0 ms-Einstellungen: 0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0
HSC Eingangstaktfrequenzen (max.) (Pegel logisch 1 = 15 bis 26 V DC)	100/80 kHz (Ea.0 bis Ea.5) 30/20 kHz (Ea.6 bis Eb.1)
Typ: Differentialeingang (RS422/RS485)	Eb.2 bis Eb.5 (0,2+ 0,2- bis 0,5+ 0,5-)
Gleichtaktspannungsbereich	-7 V bis +12 V, 1 s, 3 V fortlaufender Effektivwert (RS422/RS485-Charakteristik)
Integrierter Abschluss und Vorspannung	390 Ω bis 2M an Eb'-', 390 Ω bis +5 V an Eb'-', (Vorspannung AUS, wenn T/B spannungslos) 220 Ω zwischen Eb'+' und Eb'-'
Eingangsimpedanz Empfänger	100 Ω einschließlich Vorspannung und Abschluss
Differentialempfänger Ansprechgrenze/Sensibilität	min. +/- 0,2 V, 60 mV typische Hysterese (RS422/RS485-Charakteristik)
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute (funktionelle Trennung)
Potentialgetrennte Gruppen	1
Filterzeiten	us-Einstellungen: 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0 ms-Einstellungen: 0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0
HSC Eingangstaktfrequenzen (max.)	Einphasenzähler: 1 MHz (Eb.2 bis Eb.5) A/B-Zähler: 1 MHz (Eb.2 bis Eb.5)
Versatz Kanal-Kanal Differentialeingang	max. 40 ns
Allgemeine technische Daten (alle Digitaleingänge)	
Anzahl gleichzeitig eingeschalteter Eingänge	5 stromziehende/stromliefernde Eingänge (keine benachbarten Punkte) und 4 Differentialeingänge bei 60 °C horizontal oder 50 °C vertikal 14 bei 55 °C horizontal oder 45 °C vertikal
Leitungslänge (Meter)	500 m geschirmt, 300 m ungeschirmt 50 m geschirmt bei HSC-Eingängen (stromziehend/stromliefernd) 50 m geschirmt, verdrillt bei allen Differentialeingängen

Tabelle A- 96 HW-Konfigurationstabelle CPU 1217C Digitaleingang (DI)

Eingang	Typ und Frequenz
DIa.0	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 100 kHz
DIa.1	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 100 kHz
DIa.2	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 100 kHz
DIa.3	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 100 kHz
DIa.4	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 100 kHz
DIa.5	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 100 kHz
DIa.6	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 30 kHz
DIa.7	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 30 kHz
DIb.0	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 30 kHz
DIb.1	Typ: Eingang Typ 1, 24 V, stromziehend/stromliefernd Eingangsfrequenz schneller Zähler: max. 30 kHz
DIb.2+ .2-	Typ: Differentialeingang RS422/RS485 Eingangsfrequenz schneller Zähler: max. 1 MHz
DIb.3+ .3-	Typ: Differentialeingang RS422/RS485 Eingangsfrequenz schneller Zähler: max. 1 MHz
DIb.4+ .4-	Typ: Differentialeingang RS422/RS485 Eingangsfrequenz schneller Zähler: max. 1 MHz
DIb.5+ .5-	Typ: Differentialeingang RS422/RS485 Eingangsfrequenz schneller Zähler: max. 1 MHz

Tabelle A- 97 Digitale Ausgänge

Technische Daten	**CPU 1217C DC/DC/DC**
Ausgänge	10 gesamt 6: MOSFET, elektronisch (stromliefernd) 4: Differential (RS422/RS485)
Typ: MOSFET, elektronisch (stromliefernder Ausgang)	Aa.4 bis Ab.1
Spannungsbereich	20,4 bis 28,8 V DC
Signal logisch 1 bei max. Strom	min. 20 V DC
Signal logisch 0 bei 10 kΩ Last	max. 0,1 V DC
Strom (max.)	0,5 A
Lampenlast	5 W
Widerstand bei EIN	max. 0,6 Ω
Kriechstrom pro Ausgang	max. 10 µA
Einschaltstrom	8 A für max. 100 ms
Überlastschutz	Nein
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute (funktionelle Trennung)
Potentialgetrennte Gruppen	1
Induktive Klemmspannung	L+ minus 48 V DC, 1 W Verlustleistung
Schaltverzögerung (Aa.0 bis Aa.3)	max. 1,0 µs von Aus nach Ein max. 3,0 µs von Ein nach Aus
Schaltverzögerung (Aa.4 bis Ab.1)	max. 50 µs von Aus nach Ein max. 200 µs von Ein nach Aus
Maximale Schaltfrequenz Relais	--
Frequenz Impulsgenerator	max. 100 kHz (Aa.4 bis Ab.1)[1], min. 2 Hz
Typ: Differentialausgang (RS422/RS485)	Aa.0 bis Aa.3 (.0+ 0- bis .3+ .3-)
Gleichtaktspannungsbereich	-7 V bis +12 V, 1 s, 3 V fortlaufender Effektivwert (RS422/RS485-Charakteristik)
Differentialausgangsspannung Sender	min. 2 V bei RL = 100 Ω, min. 1,5 V bei RL = 54 Ω (RS422/RS485-Charakteristik)
Integrierter Abschluss	100 Ω zwischen Aa'+' und Aa'-'
Ausgangsimpedanz Treiber	100 Ω einschließlich Abschluss
Potenzialtrennung	500 V AC, 1 Minute (funktionelle Trennung)
Potentialgetrennte Gruppen	1
Schaltverzögerung (DAa.0 bis DAa.3)	max. 100 ns
Versatz Kanal-Kanal Differentialausgang	max. 40 ns
Frequenz Impulsgenerator	1 MHz (Aa.0 bis Aa.3), min. 2 Hz
Allgemeine technische Daten (alle Digitalausgänge)	
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)

Technische Daten	CPU 1217C DC/DC/DC
Anzahl gleichzeitig eingeschalteter Ausgänge	3 Ausgänge MOSFET, elektronisch (stromziehend) (keine benachbarten Punkte) und 4 Differentialausgänge bei 60 °C horizontal oder 50 °C vertikal 10 bei 55 °C horizontal oder 45 °C vertikal
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt

[1] Je nach Impulsempfänger und Kabel kann ein zusätzlicher Lastwiderstand (bei mindestens 10% des Nennstroms) die Qualität der Impulssignale und die Störfestigkeit verbessern.

Tabelle A- 98 HW-Konfigurationstabelle CPU 1217C Digitalausgang (DO)

Ausgang	Typ und Frequenz
DQa.0+ .0-	Typ: Differentialausgang RS422/RS485 Ausgangsfrequenz Impulsgenerator: max. 1 MHz, min. 2 Hz
DQa.1+ .1-	Typ: Differentialausgang RS422/RS485 Ausgangsfrequenz Impulsgenerator: max. 1 MHz, min. 2 Hz
DQa.2+ .2-	Typ: Differentialausgang RS422/RS485 Ausgangsfrequenz Impulsgenerator: max. 1 MHz, min. 2 Hz
DQa.3+ .3-	Typ: Differentialausgang RS422/RS485 Ausgangsfrequenz Impulsgenerator: max. 1 MHz, min. 2 Hz
DQa.4	Typ: 24-V-stromliefernder Ausgang Ausgangsfrequenz Impulsgenerator: max. 100 kHz, min. 2 Hz
DQa.5	Typ: 24-V-stromliefernder Ausgang Ausgangsfrequenz Impulsgenerator: max. 100 kHz, min. 2 Hz
DQa.6	Typ: 24-V-stromliefernder Ausgang Ausgangsfrequenz Impulsgenerator: max. 100 kHz, min. 2 Hz
DQa.7	Typ: 24-V-stromliefernder Ausgang Ausgangsfrequenz Impulsgenerator: max. 100 kHz, min. 2 Hz
DQb.0	Typ: 24-V-stromliefernder Ausgang Ausgangsfrequenz Impulsgenerator: max. 100 kHz, min. 2 Hz
DQb.1	Typ: 24-V-stromliefernder Ausgang Ausgangsfrequenz Impulsgenerator: max. 100 kHz, min. 2 Hz

A.6.4 Analogeingänge und -ausgänge

A.6.4.1 Technische Daten der analogen Eingänge

Tabelle A- 99 Analoge Eingänge

Technische Daten	Beschreibung
Anzahl der Eingänge	2
Typ	Spannung (Eintakteingang)
Vollausschlagsbereich	0 bis 10 V
Vollausschlag (Datenwort)	0 bis 27648
Überschwingbereich	10,001 bis 11,759 V
Überschwingbereich (Datenwort)	27649 bis 32511
Überlaufbereich	11,760 bis 11,852 V
Überlaufbereich (Datenwort)	32512 bis 32767
Auflösung	10 Bits
Max. Stehspannung	35 V DC
Glättung	Keine, schwach, mittel oder stark Siehe Tabelle der Schrittantwort (ms) für die analogen Eingänge der CPU (Seite 1221).
Rauschunterdrückung	10, 50 oder 60 Hz
Impedanz	≥100 kΩ
Elektrische Trennung (Feld zu Logik)	Keine
Genauigkeit (25 °C/0 bis 55 °C)	3,0% / 3,5% des Vollausschlags
Leitungslänge (Meter)	100 m, geschirmtes, verdrilltes Leiterpaar

A.6.4.2 Schrittantwort der integrierten analogen Eingänge der CPU

Tabelle A- 100 Schrittantwort (ms), 0 V bis 10 V gemessen bei 95 %

Auswahl der Glättung (Mittelwertbildung aus Abtastwerten)	Unterdrückungsfrequenz (Integrationszeit)		
	60 Hz	50 Hz	10 Hz
Keine (1 Zyklus): Keine Mittelwertbildung	50 ms	50 ms	100 ms
Schwach (4 Zyklen): 4 Abtastwerte	60 ms	70 ms	200 ms
Mittel (16 Zyklen): 16 Abtastwerte	200 ms	240 ms	1150 ms
Stark (32 Zyklen): 32 Abtastwerte	400 ms	480 ms	2300 ms
Abtastzeit	**4,17 ms**	**5 ms**	**25 ms**

A.6.4.3 Abtastzeit der integrierten analogen Ports der CPU

Tabelle A- 101 Abtastzeit der integrierten analogen Eingänge der CPU

Unterdrückungsfrequenz (Auswahl Integrationszeit)	Abtastzeit
60 Hz (16,6 ms)	4,17 ms
50 Hz (20 ms)	5 ms
10 Hz (100 ms)	25 ms

A.6.4.4 Messbereiche der analogen Eingänge für Spannung (CPUs)

Tabelle A- 102 Darstellung Analogeingang für Spannung (CPUs)

System		Messbereich Spannung	
Dezimal	Hexadezimal	0 bis 10 V	
32767	7FFF	11,851 V	Überlauf
32512	7F00		
32511	7EFF	11,759 V	Überschwingbereich
27649	6C01		
27648	6C00	10 V	Bemessungsbereich
20736	5100	7,5 V	
34	22	12 mV	
0	0	0 V	
Negative Werte		Negative Werte werden nicht unterstützt	

A.6.4.5 Technische Daten der Analogausgänge

Tabelle A- 103 Analoge Ausgänge

Technische Daten	Beschreibung
Ausgänge	2
Typ	Strom
Vollausschlagsbereich	0 ... 20 mA
Vollausschlag (Datenwort)	0 bis 27648
Überschwingbereich	20,01 bis 23,52 mA
Überschwingbereich (Datenwort)	27649 bis 32511
Überlaufbereich	siehe Fußnote[1]
Datenwort Überlaufbereich	32512 bis 32767
Auflösung	10 Bit
Ausgangsimpedanz	≤ 500 Ω max.
Elektrische Trennung (Feld zu Logik)	Keine
Genauigkeit (25 °C / -20 bis 60 °C)	3,0 % / 3,5 % des Vollausschlags

Technische Daten	Beschreibung
Einschwingzeit	2 ms
Leitungslänge (Meter)	100 m, geschirmtes, verdrilltes Leiterpaar

[1] Bei Überlauf verhalten sich die Analogausgänge entsprechend den Eigenschaften der Gerätekonfiguration. Wählen Sie für den Parameter "Reaktion auf CPU-STOP" entweder: "Ersatzwert aufschalten" oder "Letzten Wert halten".

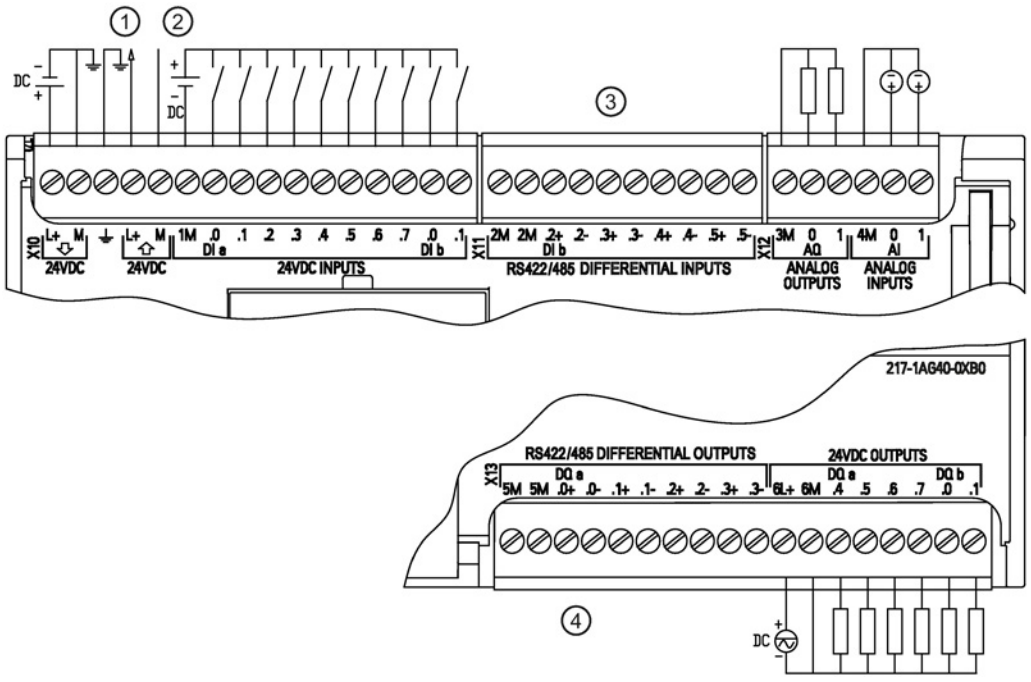
Tabelle A- 104 Darstellung Analogausgang für Strom (CPU 1215C und CPU 1217C)

System		Stromausgangsbereich	
Dezimal	Hexadezimal	0 mA bis 20 mA	
32767	7FFF	Siehe Hinweis 1	Überlauf
32512	7F00	Siehe Hinweis 1	
32511	7EFF	23,52 mA	Überschwingbereich
27649	6C01		
27648	6C00	20 mA	Bemessungsbereich
20736	5100	15 mA	
34	22	0,0247 mA	
0	0	0 mA	
Negative Werte		Negative Werte werden nicht unterstützt	

[1] Bei Überlauf verhalten sich die Analogausgänge entsprechend den eingestellten Eigenschaften der Gerätekonfiguration. Wählen Sie für den Parameter "Reaktion auf CPU-STOP" entweder: "Ersatzwert aufschalten" oder "Letzten Wert halten".

A.6.5 Schaltpläne der CPU 1217C

Tabelle A- 105 CPU 1217C DC/DC/DC (6ES7 217-1AG40-0XB0)



①	24-V-DC-Geberspannung Um zusätzliche Störfestigkeit zu erreichen, schließen Sie "M" an Masse an, auch wenn Sie keine Geberversorgung verwenden.
②	Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.
③	Siehe CPU 1217C Differentialeingang (DI) Detail und Anwendungsbeispiel (Seite 1226).
④	Siehe CPU 1217C Differentialausgang (DO) Detail und Anwendungsbeispiel (Seite 1227).
Hinweis 1: X12-Steckverbinder müssen vergoldet sein. In Anhang C, Ersatzteile (Seite 1345) finden Sie die Artikelnummer.	
Hinweis 2: Weitere Informationen zum Ethernet-Port der CPU finden Sie unter Gerätekonfiguration (Seite 151).	

Tabelle A- 106 Anschlussbelegung für die CPU 1217C DC/DC/DC (6ES7 217-1AG40-0XB0)

Pin	X10	X11	X12 (vergoldet)	X13
1	L+ / 24 V DC	2M	3M	5M
2	M / 24 V DC	2M	AO 0	5M
3	Funktionserde	DI b.2+	AO 1	DO a.0+
4	L+ / 24-V-DC-Sensorausgang	DI b.2-	4M	DO a.0-
5	M / 24-V-DC-Sensorausgang	DI b.3+	AI 0	DO a.1+
6	1M	DI b.3-	AI 1	DO a.1-
7	DI a.0	DI b.4+	--	DO a.2+

Pin	X10	X11	X12 (vergoldet)	X13
8	DI a.1	DI b.4-	--	DO a.2-
9	DI a.2	DI b.5+	--	DO a.3+
10	DI a.3	DI b.5-	--	DO a.3-
11	DI a.4	--	--	6L+
12	DI a.5	--	--	6M
13	DI a.6	--	--	DO a.4
14	DI a.7	--	--	DO a.5
15	DI b.0	--	--	DO a.6
16	DI b.1	--	--	DO a.7
17	--	--	--	DO b.0
18	--	--	--	DO b.1

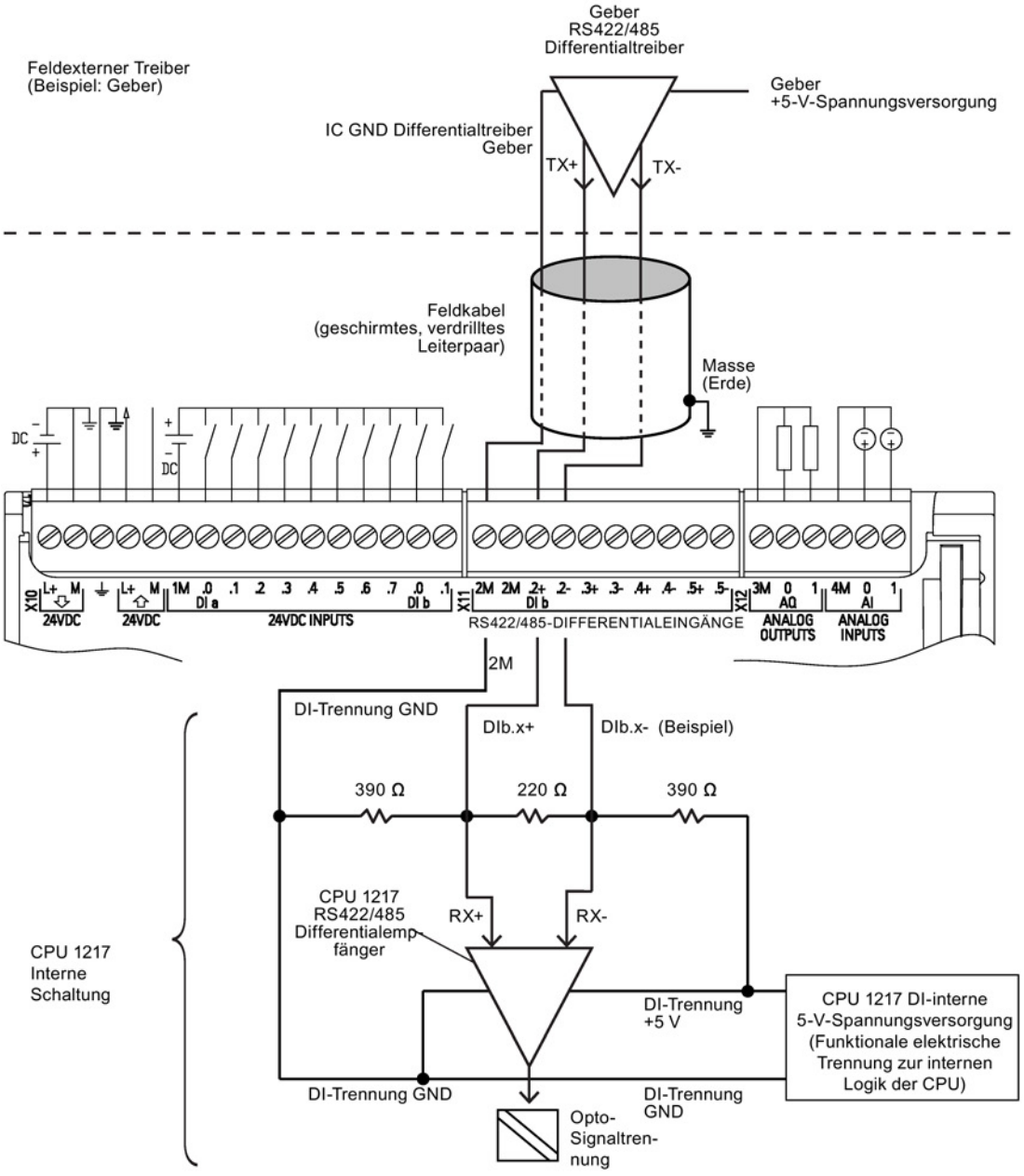
Hinweis

Nicht verwendete analoge Eingänge sollten kurzgeschlossen werden.

Siehe auch

Analogeingänge und -ausgänge (Seite 1205)

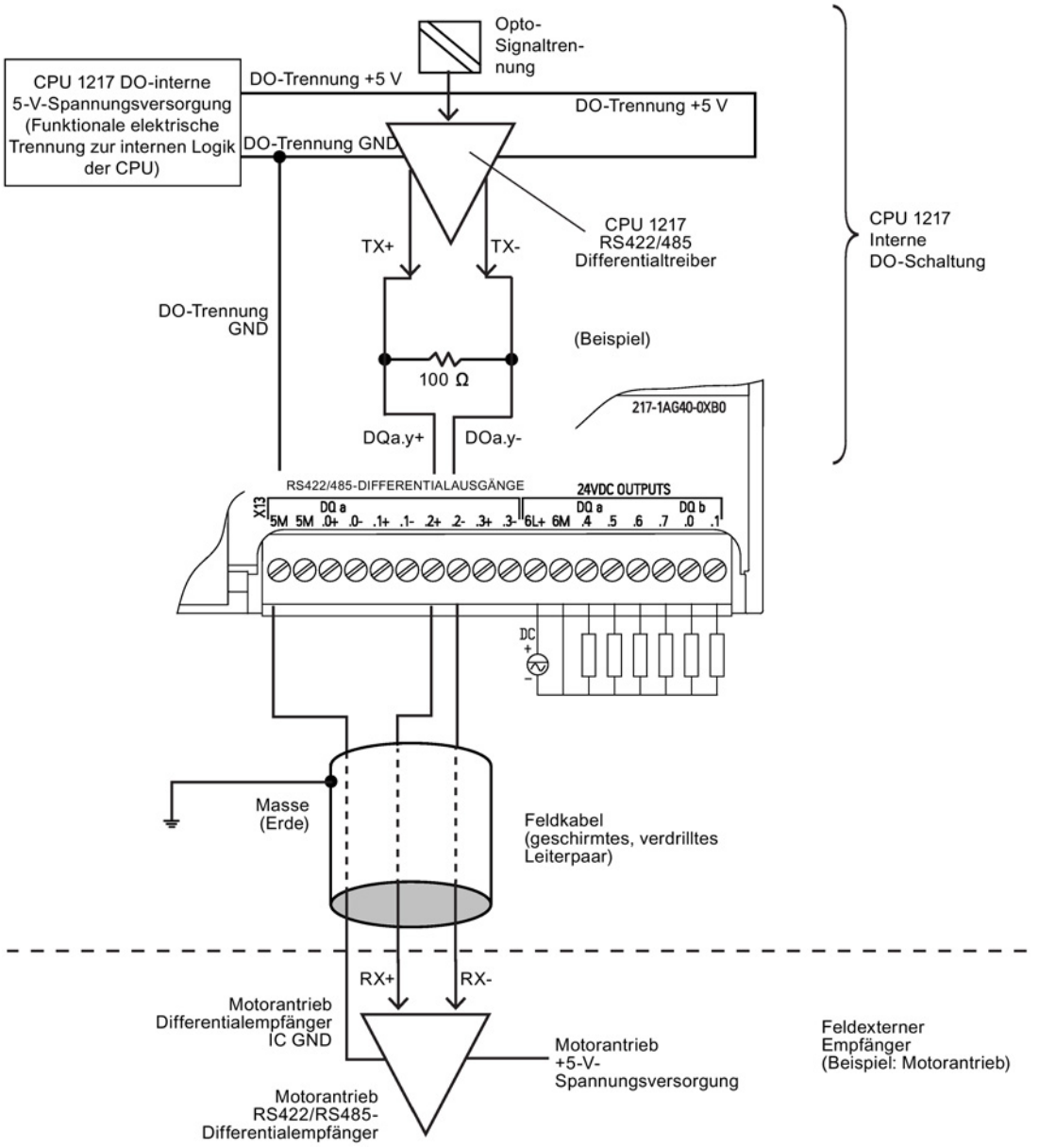
A.6.6 CPU 1217C Differentialeingang (DI) Detail und Anwendungsbeispiel



Hin-
weise

- Bei jedem digitalen Differentialeingang ist die Vorspannung AUS, wenn die Schraubklemmen im Klemmenblock offen sind.
- Integrierte(r) DI Abschluss und Vorspannung = 100 Ω Äquivalentimpedanz.
- Der Gleichtaktspannungsbereich wird durch den integrierten DI-Abschluss und Vorspannungswiderstände begrenzt. Einzelheiten finden Sie in den technischen Daten.

A.6.7 CPU 1217C Differentialausgang (DO) Detail und Anwendungsbeispiel



Hin-
weis

- Der Gleichtaktspannungsbereich wird durch den integrierten DO-Abschlusswiderstand begrenzt. Einzelheiten finden Sie in den technischen Daten.

A.7 Digitale Signalmodule (SM)

A.7.1 Technische Daten für das digitale Eingangsmodul SM 1221

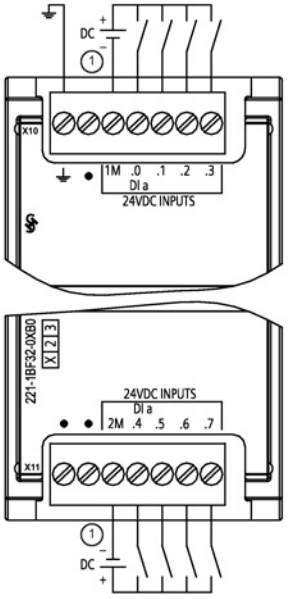
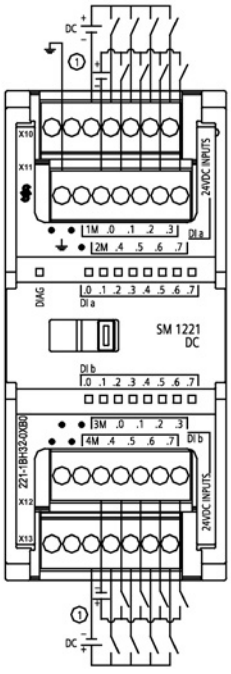
Tabelle A- 107 Allgemeine technische Daten

Modell	SM 1221 DI 8 x 24 V DC	SM 1221 DI 16 x 24 V DC
Artikelnummer	6ES7 221-1BF32-0XB0	6ES7 221-1BH32-0XB0
Abmessungen (B x H x T) (mm)	45 x 100 x 75	
Gewicht	170 Gramm	210 Gramm
Leistungsverlust	1,5 W	2,5 W
Stromaufnahme (SM-Bus)	105 mA	130 mA
Stromaufnahme (24 V DC)	4 mA / Eingang	

Tabelle A- 108 Digitale Eingänge

Modell	SM 1221 DI 8 x 24 V DC	SM 1221 DI 16 x 24 V DC
Anzahl der Eingänge	8	16
Typ	Stromziehend/stromliefernd (IEC Typ 1, wenn stromziehend)	
Nennspannung	24 V DC bei 4 mA, Nennwert	
Zulässige Dauerspannung	max. 30 V DC	
Stoßspannung	35 V DC für 0,5 s	
Signal logisch 1 (min.)	15 V DC bei 2,5 mA	
Signal logisch 0 (max.)	5 V DC bei 1 mA	
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute	
Potentialgetrennte Gruppen	2	4
Filterzeiten	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 und 12,8 ms (wählbar in Gruppen zu je 4)	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 und 12,8 ms (wählbar in Gruppen zu je 4)
Anzahl gleichzeitig eingeschalteter Eingänge	8	16
Leitungslänge (Meter)	500 m geschirmt, 300 m ungeschirmt	

Tabelle A- 109 Schaltpläne der Digitaleingangs-SMs

SM 1221 DI 8 x 24 VDC (6ES7 221-1BF32-0XB0)	SM 1221 DI 16 x 24 VDC (6ES7 221-1BH32-0XB0)
	

① Bei stromziehenden Eingängen "-" an "M" anschließen (gezeigt). Bei stromliefernden Eingängen "+" an "M" anschließen.

Tabelle A- 110 Anschlussbelegung für das SM 1221 DI 8 x 24 VDC (6ES7 221-1BF32-0XB0)

Pin	X10	X11
1	Funktionserde	Kein Anschluss
2	Kein Anschluss	Kein Anschluss
3	1M	2M
4	DI a.0	DI a.4
5	DI a.1	DI a.5
6	DI a.2	DI a.6
7	DI a.3	DI a.7

Tabelle A- 111 Anschlussbelegung für das SM 1221 DI 16 x 24 VDC (6ES7 221-1BH32-0XB0)

Pin	X10	X11	X12	X13
1	Kein Anschluss	Funktionserde	Kein Anschluss	Kein Anschluss
2	Kein Anschluss	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	1M	2M	3 M	4 M
4	DI a.0	DI a.4	DI b.0	DI b.4
5	DI a.1	DI a.5	DI b.1	DI b.5
6	DI a.2	DI a.6	DI b.2	DI b.6
7	DI a.3	DI a.7	DI b.3	DI b.7

A.7.2 Technische Daten für das digitale Ausgangsmodul SM 1222 mit 8 Ausgängen

Tabelle A- 112 Allgemeine technische Daten

Modell	SM 1222 DO 8 x Relais	SM 1222 DO 8 x Relais (Umschaltung)	SM 1222 DO 8 x 24 V DC
Artikelnummer	6ES7 222-1HF32-0XB0	6ES7 222-1XF32-0XB0	6ES7 222-1BF32-0XB0
Abmessungen B x H x T (mm)	45 x 100 x 75	70 x 100 x 75	45 x 100 x 75
Gewicht	190 Gramm	310 Gramm	180 Gramm
Leistungsverlust	4,5 W	5 W	1,5 W
Stromaufnahme (SM-Bus)	120 mA	140 mA	120 mA
Stromaufnahme (24 V DC)	11 mA / Relaisspule	16,7 mA / Relaisspule	50 mA

Tabelle A- 113 Digitale Ausgänge

Modell	SM 1222 DO 8 x Relais	SM 1222 DO8 RLS, Umschaltung	SM 1222 DO 8 x 24 V DC
Ausgänge	8		
Typ	Relais, mechanisch	Relaiswechselkontakt	MOSFET, elektronisch (stromliefernd)
Spannungsbereich	5 bis 30 V DC oder 5 bis 250 V AC		20,4 bis 28,8 V DC
Signal logisch 1 bei max. Strom	--		min. 20 V DC
Signal logisch 0 bei 10 kΩ Last	--		max. 0,1 V DC
Strom (max.)	2,0 A		0,5 A
Lampenlast	30 W DC/200 W AC		5 W
Kontaktwiderstand bei EIN	max. 0,2 Ω wenn neu		max. 0,6 Ω
Kriechstrom pro Ausgang	--		max. 10 µA
Einschaltstrom	7 A bei geschlossenen Kontakten		8 A für max. 100 ms
Überlastschutz	Nein		
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute (Spule-Kontakt) Keine (Spule-Logik)	1500 V AC für 1 Minute (Spule zu Kontakt)	500 V AC für 1 Minute

Modell	SM 1222 DO 8 x Relais	SM 1222 DO8 RLS, Umschaltung	SM 1222 DO 8 x 24 V DC
Isolationswiderstand	min. 100 MΩ, wenn neu		--
Elektrische Trennung zwischen offenen Kontakten	750 V AC für 1 Minute		--
Potenzialgetrennte Gruppen	2	8	1
Strom je Leiter (max.)	10 A	2 A	4 A
Induktive Klemmspannung	--		L+ minus 48 V, 1 W Verlustleistung
Schaltverzögerung	max. 10 ms		max. 50 µs von Aus nach Ein max. 200 µs von Ein nach Aus
Maximale Schaltfrequenz Relais	1 Hz		--
Mechanische Lebensdauer (ohne Last)	10.000.000 Schaltspiele auf/zu		--
Lebensdauer der Kontakte bei Nennlast (Schließerkontakt)	100.000 Schaltspiele auf/zu		--
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)		
Anzahl gleichzeitig eingeschalteter Ausgänge	8	• 4 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 8 bei 55 °C horizontal oder 45 °C vertikal	8
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt		

A.7.3 Technische Daten für das digitale Ausgangsmodul SM 1222 mit 16 Ausgängen

Tabelle A- 114 Allgemeine technische Daten

Modell	SM 1222 DO 16 x Relais	SM 1222 DO 16 x 24 V DC
Artikelnummer	6ES7 222-1HH32-0XB0	6ES7 222-1BH32-0XB0
Abmessungen (B x H x T) (mm)	45 x 100 x 75	
Gewicht	260 Gramm	220 Gramm
Leistungsverlust	8,5 W	2,5 W
Stromaufnahme (SM-Bus)	135 mA	140 mA
Stromaufnahme (24 V DC)	11 mA / Relaisspule	100 mA

Tabelle A- 115 Digitale Ausgänge

Modell	SM1222 DO 16 x Relais	SM1222 DO 16 x 24 V DC
Ausgänge	16	
Typ	Relais, mechanisch	MOSFET, elektronisch (stromliefernd)
Spannungsbereich	5 bis 30 V DC oder 5 bis 250 V AC	20,4 bis 28,8 V DC
Signal logisch 1 bei max. Strom	-	min. 20 V DC
Signal logisch 0 bei 10 kΩ Last	-	max. 0,1 V DC
Strom (max.)	2,0 A	0,5 A
Lampenlast	30 W DC/200 W AC	5 W
Kontaktwiderstand bei EIN	max. 0,2 Ω wenn neu	max. 0,6 Ω
Kriechstrom pro Ausgang	--	max. 10 µA
Einschaltstrom	7 A bei geschlossenen Kontakten	8 A für max. 100 ms
Überlastschutz	Nein	
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute (Spule-Kontakt) Keine (Spule-Logik)	500 V AC für 1 Minute
Isolationswiderstand	min. 100 MΩ, wenn neu	-
Elektrische Trennung zwischen offenen Kontakten	750 V AC für 1 Minute	-
Potenzialgetrennte Gruppen	4	1
Strom je Leiter (max.)	10 A	8 A
Induktive Klemmspannung	-	L+ minus 48 V, 1 W Verlustleistung
Schaltverzögerung	max. 10 ms	max. 50 µs von Aus nach Ein max. 200 µs von Ein nach Aus
Maximale Schaltfrequenz Relais	1 Hz	-
Mechanische Lebensdauer (ohne Last)	10.000.000 Schaltspiele auf/zu	-
Lebensdauer der Kontakte bei Nennlast (Schließerkontakt)	100.000 Schaltspiele auf/zu	-
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)	Letzter Wert oder Ersatzwert (Voreinstellung 0)

Modell	SM1222 DO 16 x Relais	SM1222 DO 16 x 24 V DC
Anzahl gleichzeitig eingeschalteter Ausgänge	• 8 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 16 bei 55 °C horizontal oder 45 °C vertikal	16
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt	

Tabelle A- 116 Schaltpläne der Digitalausgangs-SMs mit 8 Ausgängen

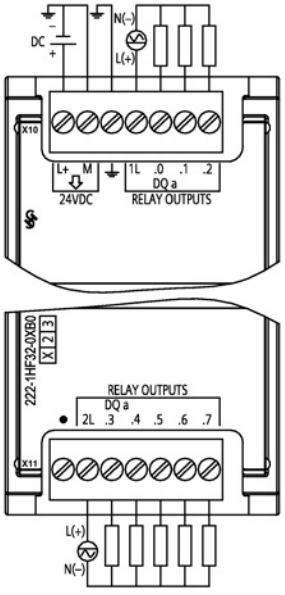
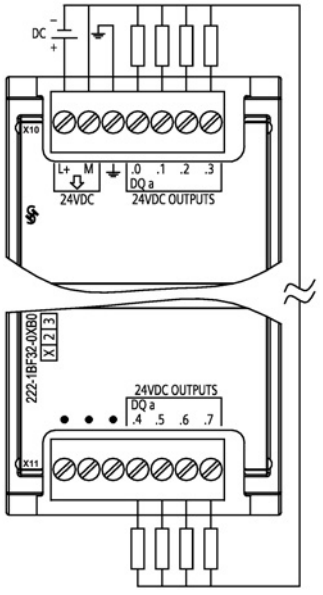
SM 1222 DO 8 x Relais (6ES7 222-1HF32-0XB0)	SM 1222 DO 8 x 24 VDC (6ES7 222-1BF32-0XB0)
	

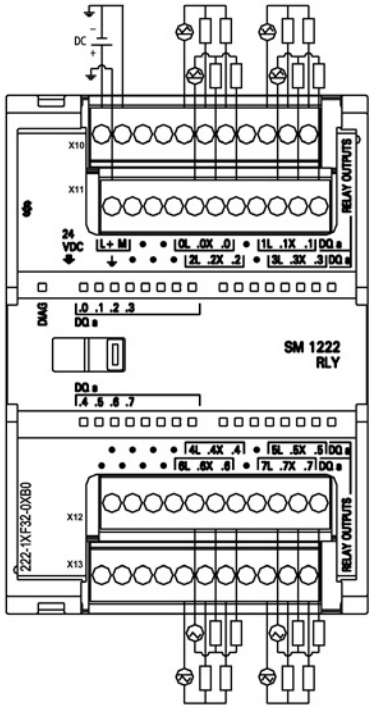
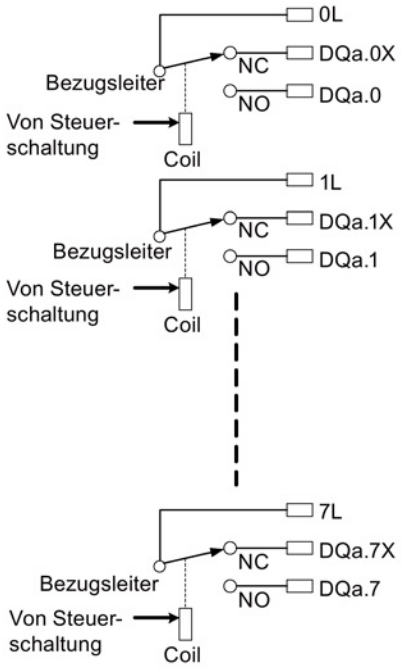
Tabelle A- 117 Anschlussbelegung für das SM 1222 DQ 8 x Relais (6ES7 222-1HF32-0XB0)

Pin	X10	X11
1	L+ / 24 V DC	Kein Anschluss
2	M / 24 V DC	2L
3	Funktionserde	DO a.3
4	1L	DO a.4
5	DO a.0	DO a.5
6	DO a.1	DO a.6
7	DO a.2	DO a.7

Tabelle A- 118 Anschlussbelegung für das SM 1222 DO 8 x 24 VDC (6ES7 222-1BF32-0XB0)

Pin	X10	X11
1	L+ / 24 V DC	Kein Anschluss
2	M / 24 V DC	Kein Anschluss
3	Funktionserde	Kein Anschluss
4	DO a.0	DO a.4
5	DO a.1	DO a.5
6	DO a.2	DO a.6
7	DO a.2	DO a.7

Tabelle A- 119 Schaltplan für das Umschaltungs-Digitalausgangs-SM mit 8 Relaisausgängen

SM 1222 DO 8 x Relais, Umschaltung (6ES7 222-1XF32-0XB0)	
	

Ein Umschaltrelaisausgang steuert zwei Stromkreise über eine gemeinsame Klemme: einen Öffner- und einen Schließerkontakt. Am Beispiel von Ausgang "0" dargestellt, ist bei ausgeschaltetem Ausgang die gemeinsame Spannung (0L) mit dem Öffnerkontakt (.0X) verbunden und vom Schließerkontakt (.0) getrennt. Ist der Ausgang eingeschaltet, wird die Spannung (0L) vom Öffnerkontakt (.0X) getrennt und mit dem Schließerkontakt (.0) verbunden.

Tabelle A- 120 Anschlussbelegung für das SM 1222 DO 8 x Relais, Umschaltung (6ES7 222-1XF32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Funktionserde	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Kein Anschluss	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	Kein Anschluss	Kein Anschluss	Kein Anschluss	Kein Anschluss
5	0L	2L	4L	6L
6	DO a.0X	DO a.2X	DO a.4X	DO a.6X
7	DO a.0	DO a.2	DO a.4	DO a.6
8	Kein Anschluss	Kein Anschluss	Kein Anschluss	Kein Anschluss
9	1L	3L	5L	7L
10	DO a.1X	DO a.3X	DO a.5X	DO a.7X
11	DO a.1	DO a.3	DO a.5	DO a.7

Tabelle A- 121 Schaltpläne der Digitalausgangs-SMs mit 16 Ausgängen

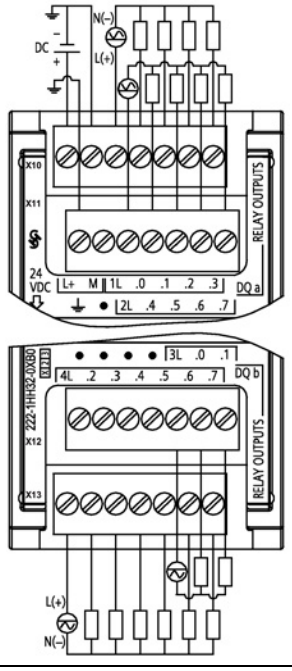
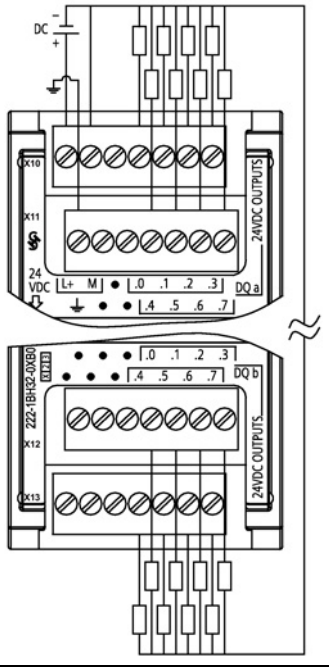
SM 1222 DO 16 x Relais (6ES7 222-1HH32-0XB0)	SM 1222 DO 16 x 24 VDC (6ES7 222-1BH32-0XB0)
	

Tabelle A- 122 Anschlussbelegung für das SM 1222 DO 16 x Relais (6ES7 222-1HH32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Funktionserde	Kein Anschluss	4L
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	DO b.2
3	1L	2L	Kein Anschluss	DO b.3
4	DO a.0	DO a.4	Kein Anschluss	DO b.4
5	DO a.1	DO a.5	3L	DO b.5
6	DO a.2	DO a.6	DO b.0	DO b.6
7	DO a.3	DO a.7	DO b.1	DO b.7

Tabelle A- 123 Anschlussbelegung für das SM 1222 DO 16 x 24 VDC (6ES7 222-1BH32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Funktionserde	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Kein Anschluss	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	DO a.0	DO a.4	DO b.0	DO b.4
5	DO a.1	DO a.5	DO b.1	DO b.5
6	DO a.2	DO a.6	DO b.2	DO b.6
7	DO a.3	DO a.7	DO b.3	DO b.7

A.7.4 Technische Daten des digitalen Ein-/Ausgangsmoduls SM 1223 (V DC)

Tabelle A- 124 Allgemeine technische Daten

Modell	**SM 1223 DI 8 x 24 V DC, DO 8 x Relais**	**SM 1223 DI 16 x 24 V DC, DO 16 x Relais**	**SM 1223 DI 8 x 24 V DC, DO 8 x 24 V DC**	**SM 1223 DI 16 x 24 V DC, DO 16 x 24 V DC**
Artikelnummer	6ES7 223-1PH32-0XB0	6ES7 223-1PL32-0XB0	6ES7 223-1BH32-0XB0	6ES7 223-1BL32-0XB0
Abmessungen B x H x T (mm)	45 x 100 x 75	70 x 100 x 75	45 x 100 x 75	70 x 100 x 75
Gewicht	230 Gramm	350 Gramm	210 Gramm	310 Gramm
Leistungsverlust	5,5 W	10 W	2,5 W	4,5 W
Stromaufnahme (SM-Bus)	145 mA	180 mA	145 mA	185 mA
Stromaufnahme (24 V DC)	4 mA / Eingang 11 mA / Relaisspule		150 mA	200 mA

Tabelle A- 125 Digitale Eingänge

Modell	**SM 1223 DI 8 x 24 V DC, DO 8 x Relais**	**SM 1223 DI 16 x 24 V DC, DO 16 x Relais**	**SM 1223 DI 8 x 24 V DC, DO 8 x 24 V DC**	**SM 1223 DI 16 x 24 V DC, DO 16 x 24 V DC**
Anzahl der Eingänge	8	16	8	16
Typ	Stromziehend/stromliefernd (IEC Typ 1, wenn stromziehend)			
Nennspannung	24 V DC bei 4 mA, Nennwert			
Zulässige Dauerspannung	max. 30 V DC			
Stoßspannung	35 V DC für 0,5 s			
Signal logisch 1 (min.)	15 V DC bei 2,5 mA			
Signal logisch 0 (max.)	5 V DC bei 1 mA			
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute			
Potenzialgetrennte Gruppen	2			
Filterzeiten	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 und 12,8 ms wählbar in Gruppen zu je 4			
Anzahl gleichzeitig eingeschalteter Eingänge	8	• 8 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 16 bei 55 °C horizontal oder 45 °C vertikal	8	16
Leitungslänge (Meter)	500 m geschirmt, 300 m ungeschirmt			

Tabelle A- 126 Digitale Ausgänge

Modell	SM 1223 DI 8 x 24 V DC, DO 8 x Relais	SM 1223 DI 16 x 24 V DC, DO 16 x Relais	SM 1223 DI 8 x 24 V DC, DO 8 x 24 V DC	SM 1223 DI 16 x 24 V DC, DO 16 x 24 V DC
Ausgänge	8	16	8	16
Typ	Relais, mechanisch		MOSFET, elektronisch (stromliefernd)	
Spannungsbereich	5 bis 30 V DC oder 5 bis 250 V AC		20,4 bis 28,8 V DC	
Signal logisch 1 bei max. Strom	--		min. 20 V DC	
Signal logisch 0 bei 10 kΩ Last	--		max. 0,1 V DC	
Strom (max.)	2,0 A		0,5 A	
Lampenlast	30 W DC/200 W AC		5 W	
Kontaktwiderstand bei EIN	max. 0,2 Ω wenn neu		max. 0,6 Ω	
Kriechstrom pro Ausgang	--		max. 10 µA	
Einschaltstrom	7 A bei geschlossenen Kontakten		8 A für max. 100 ms	
Überlastschutz	Nein			
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute (Spule-Kontakt) Keine (Spule-Logik)		500 V AC für 1 Minute	
Isolationswiderstand	min. 100 MΩ, wenn neu		--	
Elektrische Trennung zwischen offenen Kontakten	750 V AC für 1 Minute		--	
Potenzialgetrennte Gruppen	2	4	1	
Strom je Leiter	10 A	8 A	4 A	8 A
Induktive Klemmspannung	--		L+ minus 48 V, 1 W Verlustleistung	
Schaltverzögerung	max. 10 ms		max. 50 µs von Aus nach Ein max. 200 µs von Ein nach Aus	
Maximale Schaltfrequenz Relais	1 Hz		--	
Mechanische Lebensdauer (ohne Last)	10.000.000 Schaltspiele auf/zu		--	
Lebensdauer der Kontakte bei Nennlast (Schließerkontakt)	100.000 Schaltspiele auf/zu		--	
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)			
Anzahl gleichzeitig eingeschalteter Ausgänge	8	• 8 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 16 bei 55 °C horizontal oder 45 °C vertikal	8	16
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt			

Tabelle A- 127 Schaltpläne der digitalen Ein-/Ausgangsmodule (V DC/Relais)

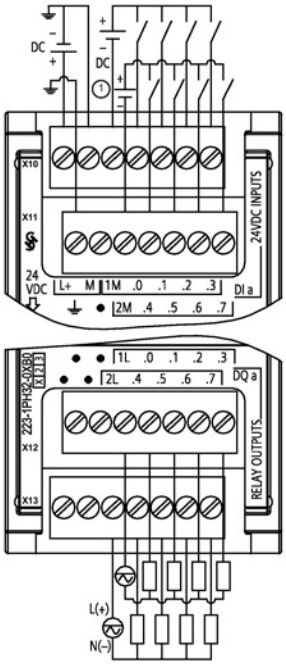
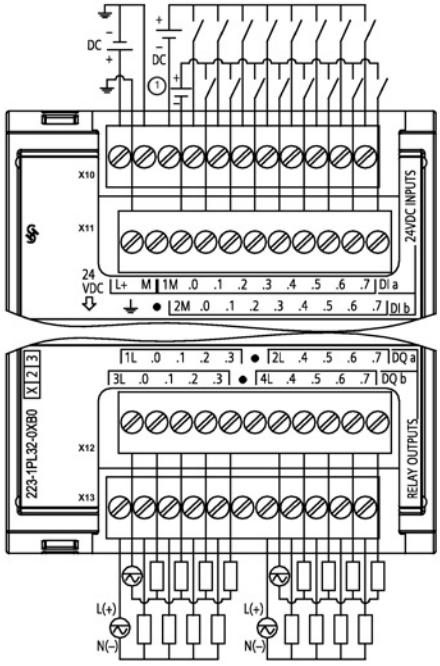
SM 1223 DI 8 x 24 VDC, DO 8 x Relais (6ES7 223-1PH32-0XB0)	SM 1223 DI 16 x 24 V DC, DO 16 x Relais (6ES7 223-1PL32-0XB0)	Hinweise
		① Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.

Tabelle A- 128 Anschlussbelegung für das SM 1223 DI 8 x 24 V DC, DO 8 x Relais (6ES7 223-1PH32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Funktionserde	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	1M	2M	1L	2L
4	DI a.0	DI a.4	DO a.0	DO a.4
5	DI a.1	DI a.5	DO a.1	DO a.5
6	DI a.2	DI a.6	DO a.2	DO a.6
7	DI a.3	DI a.7	DO a.3	DO a.7

Tabelle A- 129 Anschlussbelegung für das SM 1223 DI 16 x 24 V DC, DO 16 x Relais (6ES7 223-1PL32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Funktionserde	1L	3L
2	M / 24 V DC	Kein Anschluss	DO a.0	DO b.0
3	1M	2M	DO a.1	DO b.1
4	DI a.0	DI b.0	DO a.2	DO b.2
5	DI a.1	DI b.1	DO a.3	DO b.3

Pin	X10	X11	X12	X13
6	DI a.2	DI b.2	Kein Anschluss	Kein Anschluss
7	DI a.3	DI b.3	2L	4L
8	DI a.4	DI b.4	DO a.4	DO b.4
9	DI a.5	DI b.5	DO a.5	DO b.5
10	DI a.6	DI b.6	DO a.6	DO b.6
11	DI a.7	DI b.7	DO a.7	DO b.7

Tabelle A- 130 Schaltpläne der digitalen Ein-/Ausgangsmodule (V DC)

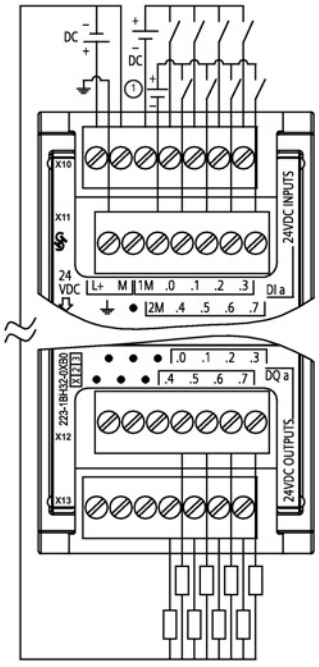
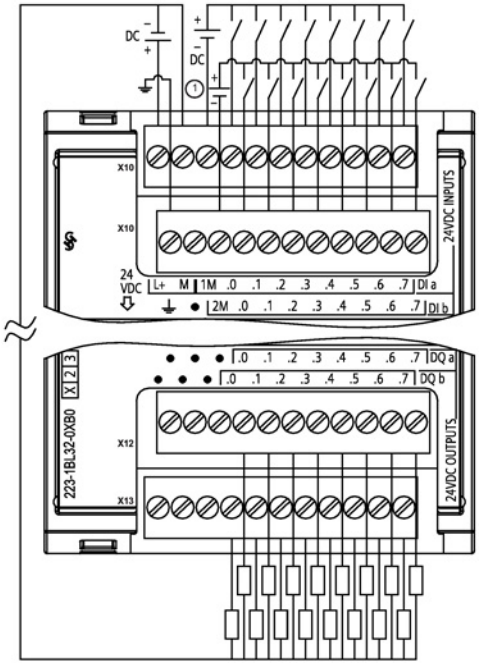
SM 1223 DI 8 x 24 VDC, DO 8 x 24 VDC (6ES7 223-1BH32-0XB0)	SM 1223 DI 16 x 24 V DC, DO 16 x 24 V DC (6ES7 223-1BL32-0XB0)	Hinweise
		① Bei stromziehenden Eingängen "-" an "M" anschließen (abgebildet). Bei stromliefernden Eingängen "+" an "M" anschließen.

Tabelle A- 131 Anschlussbelegung für das SM 1223 DI 8 x 24 V DC, DO 8 x 24 V DC (6ES7 223-1BH32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Funktionserde	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	1M	2M	Kein Anschluss	Kein Anschluss
4	DI a.0	DI a.4	DO a.0	DO a.4
5	DI a.1	DI a.5	DO a.1	DO a.5
6	DI a.2	DI a.6	DO a.2	DO a.6
7	DI a.3	DI a.7	DO a.3	DO a.7

Tabelle A- 132 Anschlussbelegung für das SM 1223 DI 16 x 24 V DC, DO 16 x 24 V DC (6ES7 223-1BL32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Funktionserde	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	1M	2M	Kein Anschluss	Kein Anschluss
4	DI a.0	DI b.0	DO a.0	DO b.0
5	DI a.1	DI b.1	DO a.1	DO b.1
6	DI a.2	DI b.2	DO a.2	DO b.2
7	DI a.3	DI b.3	DO a.3	DO b.3
8	DI a.4	DI b.4	DO a.4	DO b.4
9	DI a.5	DI b.5	DO a.5	DO b.5
10	DI a.6	DI b.6	DO a.6	DO b.6
11	DI a.7	DI b.7	DO a.7	DO b.7

A.7.5 Technische Daten des digitalen Ein-/Ausgangsmoduls SM 1223 (AC)

Tabelle A- 133 Allgemeine technische Daten

Modell	SM 1223 DI 8 x120/230 V AC / DO 8 x Relais
Artikelnummer	6ES7 223-1QH32-0XB0
Abmessungen B x H x T (mm)	45 x 100 x 75 mm
Gewicht	190 Gramm
Leistungsverlust	7,5 W
Stromaufnahme (SM-Bus)	120 mA
Stromaufnahme (24 V DC)	11 mA je Ausgang, wenn eingeschaltet

Tabelle A- 134 Digitale Eingänge

Modell	SM 1223 DI 8 x 120/230 V AC / DO 8 x Relais
Anzahl der Eingänge	8
Typ	IEC Typ 1
Nennspannung	120 V AC bei 6 mA, 230 V AC bei 9 mA
Zulässige Dauerspannung	264 V AC
Stoßspannung	--
Signal logisch 1 (min.)	79 V AC bei 2,5 mA
Signal logisch 0 (max.)	20 V AC bei 1 mA
Kriechstrom (max.)	1 mA
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute
Potenzialgetrennte Gruppen[1]	4
Eingabeverzögerungszeiten	Typisch: 0,2 bis 12,8 ms, vom Anwender einstellbar Maximum: -

Modell	SM 1223 DI 8 x 120/230 V AC / DO 8 x Relais
Anschluss 2Draht-Näherungssensor (Bero) (max.)	1 mA
Leitungslänge	Nicht geschirmt: 300 Meter Geschirmt: 500 Meter
Anzahl gleichzeitig eingeschalteter Eingänge	8

[1] Kanäle in einer Gruppe müssen die gleiche Phase haben.

Tabelle A- 135 Digitale Ausgänge

Modell	SM 1223 DI 8 x 120/230 V AC / DO 8 x Relais
Anzahl der Ausgänge	8
Typ	Relais, mechanisch
Spannungsbereich	5 bis 30 V DC oder 5 bis 250 V AC
Signal logisch 1 bei max. Strom	--
Signal logisch 0 bei 10 kΩ Last	--
Strom (max.)	2,0 A
Lampenlast	30 W DC/200 W AC
Kontaktwiderstand bei EIN	max. 0,2 Ω wenn neu
Kriechstrom pro Ausgang	--
Einschaltstrom	7 A bei geschlossenen Kontakten
Überlastschutz	Nein
Elektrische Trennung (Feld zu Logik)	1500 V AC für 1 Minute (Spule-Kontakt) Keine (Spule-Logik)
Isolationswiderstand	min. 100 MΩ, wenn neu
Elektrische Trennung zwischen offenen Kontakten	750 V AC für 1 Minute
Potenzialgetrennte Gruppen	2
Strom je Leiter (max.)	10 A
Induktive Klemmspannung	--
Schaltverzögerung (max.)	10 ms
Maximale Schaltfrequenz Relais	1 Hz
Mechanische Lebensdauer (ohne Last)	10.000.000 Schaltspiele auf/zu
Lebensdauer der Kontakte bei Nennlast	1.000.000 Schaltspiele auf/zu
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)
Anzahl gleichzeitig eingeschalteter Ausgänge	• 4 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 8 bei 55 °C horizontal oder 45 °C vertikal
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt

Tabelle A- 136 SM 1223 DI 8 x 120/230 V AC, DO 8 x Relais (6ES7 223-1QH32-0XB0)

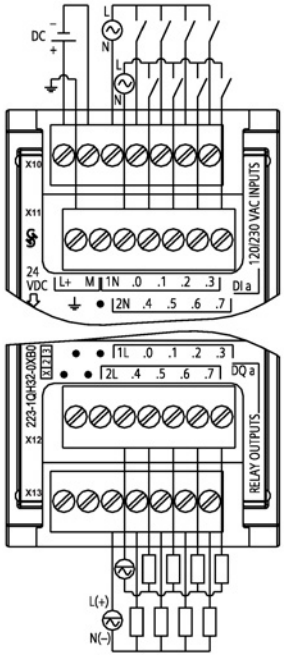
	

Tabelle A- 137 Anschlussbelegung für das SM 1223 DI 8 x 120/240 VAC, DO 8 x Relais (6ES7 223-1QH32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Funktionserde	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	1N	2N	1L	2L
4	DI a.0	DI a.4	DO a.0	DO a.4
5	DI a.1	DI a.5	DO a.1	DO a.5
6	DI a.2	DI a.6	DO a.2	DO a.6
7	DI a.3	DI a.7	DO a.3	DO a.7

A.8 Analoge Signalmodule (SM)

A.8.1 Technische Daten des SM 1231 Analogeingabemoduls

Tabelle A- 138 Allgemeine technische Daten

Modell	SM 1231 AI 4 x 13 Bit	SM 1231 AI 8 x 13 Bit	SM 1231 AI 4 x 16 Bit
Artikelnummer	6ES7 231-4HD32-0XB0	6ES7 231-4HF32-0XB0	6ES7 231-5ND30-0XB0
Abmessungen B x H x T (mm)	45 x 100 x 75		
Gewicht	180 Gramm		
Leistungsverlust	2,2 W	2,3 W	2,0 W
Stromaufnahme (SM-Bus)	80 mA	90 mA	80 mA
Stromaufnahme (24 V DC)	45 mA		65 mA

Tabelle A- 139 Analoge Eingänge

Modell	SM 1231 AI 4 x 13 Bit	SM 1231 AI 8 x 13 Bit	SM 1231 AI 4 x 16 Bit
Anzahl der Eingänge	4	8	4
Typ	Spannung oder Strom (differential): wählbar in Gruppen zu je 2		Spannung oder Strom (differential)
Bereich	±10 V, ±5 V, ±2,5 V, 0 bis 20 mA oder 4 mA bis 20 mA		±10 V, ±5 V, ±2,5 V, ±1,25 V, 0 bis 20 mA oder 4 mA bis 20 mA
Vollausschlag (Datenwort)	-27648 bis 27648 Spannung / 0 bis 27648 Strom		
Überschwing-/Unterschwingbereich (Datenwort) Siehe Abschnitt über Analogeingangsbereiche für Spannung und Strom (Seite 1254).	Spannung: 32511 bis 27649 / -27649 bis -32512 Strom: 32511 bis 27649 / 0 bis -4864		
Überlauf/Unterlauf (Datenwort) Siehe Abschnitt über Eingangsbereiche für Spannung und Strom (Seite 1254).	Spannung: 32767 bis 32512 / -32513 bis -32768 Strom 0 bis 20 mA: 32767 bis 32512 / -4865 bis -32768 Strom 4 bis 20 mA: 32767 bis 32512 (Werte unter -4864 weisen auf Drahtbruch hin)		
Auflösung1	12 Bit + Vorzeichenbit		15 Bit + Vorzeichenbit
Maximale Stehspannung/-strom	±35 V / ±40 mA		
Glättung	Keine, schwach, mittel oder stark Siehe Abschnitt zu Schrittantwortzeiten (Seite 1253).		
Rauschunterdrückung	400, 60, 50 oder 10 Hz Siehe Abschnitt zu Abtastraten (Seite 1253).		
Eingangsimpedanz	≥ 9 MΩ (Spannung) / 280 Ω (Strom)		≥ 1 MΩ (Spannung) / < 315 Ω, > 280 Ω (Strom)

Modell	SM 1231 AI 4 x 13 Bit	SM 1231 AI 8 x 13 Bit	SM 1231 AI 4 x 16 Bit
Elektrische Trennung Feld zu Logik Logik zu 24 V DC Feld zu 24 V DC Kanal zu Kanal	Keine		 500 V AC 500 V AC 500 V AC Keine
Genauigkeit (25 °C / -20 bis 60 °C)	±0,1% / ±0,2% des Vollausschlags		±0,1% / ±0,3% des Vollausschlags
Messprinzip	Istwertumwandlung		
Gleichtaktunterdrückung	40 dB, Nennwert bei 60 Hz		
Betriebssignalbereich[1]	Signal- plus Gleichtaktspannung muss kleiner als +12 V und größer als -12 V sein		
Leitungslänge (Meter)	100 m, verdrillt und geschirmt		

[1] Spannungen außerhalb des Betriebsbereichs, die an einem Kanal angelegt werden, können an anderen Kanälen Störungen verursachen.

Tabelle A- 140 Diagnose

Modell	SM 1231 AI 4 x 13 Bit	SM 1231 AI 8 x 13 Bit	SM 1231 AI 4 x 16 Bit
Überlauf/Unterlauf	Ja		
24-V-DC-Niederspannung	Ja		
Drahtbruch	nur im Bereich 4 bis 20 mA (wenn der Eingang unter -4164 ist; 1,185 mA)		

Tabelle A- 141 Schaltpläne der Analogeingangs-SMs

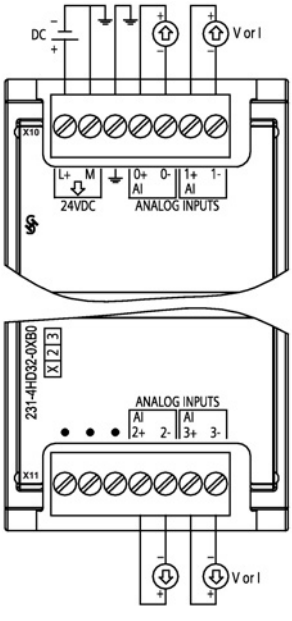
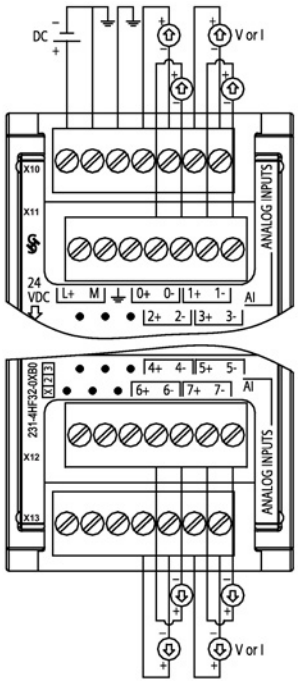
SM 1231 AI 4 x 13 Bit (6ES7 231-4HD32-0XB0)	SM 1231 AI 8 x 13 Bit (6ES7 231-4HF32-0XB0)
	
Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	

Tabelle A- 142 Anschlussbelegung für das SM 1231 AI 4 x 13 Bit (6ES7 231-4HD32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss
2	M / 24 V DC	Kein Anschluss
3	Funktionserde	Kein Anschluss
4	AI 0+	AI 2+
5	AI 0-	AI 2-
6	AI 1+	AI 3+
7	AI 1-	AI 3-

Tabelle A- 143 Anschlussbelegung für das SM 1231 AI 8 x 13 Bit (6ES7 231-4HF32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)	X12 (vergoldet)	X13 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Funktionserde	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	AI 0+	AI 2+	AI 4+	AI 6+
5	AI 0-	AI 2-	AI 4-	AI 6-
6	AI 1+	AI 3+	AI 5+	AI 7+
7	AI 1-	AI 3-	AI 5-	AI 7-

Tabelle A- 144 Schaltplan des Analogeingangs-SMs

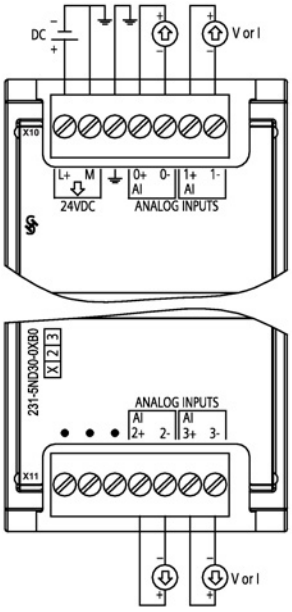
SM 1231 AI 4 x 16 Bit (6ES7 231-5ND30-0XB0)	
	
Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	

Tabelle A- 145 Anschlussbelegung für das SM 1231 AI 4 x 16 Bit (6ES7 231-5ND30-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss
2	M / 24 V DC	Kein Anschluss
3	Funktionserde	Kein Anschluss
4	AI 0+	AI 2+
5	AI 0-	AI 2-
6	AI 1+	AI 3+
7	AI 1-	AI 3-

Hinweis

Nicht verwendete Spannungseingangskanäle sollten kurzgeschlossen werden.

Nicht verwendete Stromeingangskanäle sollten auf den Bereich 0 bis 20 mA gesetzt werden und/oder das Melden von Drahtbruch sollte deaktiviert werden.

Für den Strommodus konfigurierte Eingänge führen nur dann Schleifenstrom, wenn das Modul eingeschaltet und konfiguriert ist.

Stromeingangskanäle sind nur dann betriebsfähig, wenn der Sender mit externer Spannung versorgt wird.

A.8.2 Technische Daten des SM 1232 Analogausgabemoduls

Tabelle A- 146 Allgemeine technische Daten

Technische Daten	SM 1232 AO 2 x 14 Bit	SM 1232 AO 4 x 14 Bit
Artikelnummer	6ES7 232-4HB32-0XB0	6ES7 232-4HD32-0XB0
Abmessungen (B x H x T) (mm)	45 x 100 x 75	
Gewicht	180 Gramm	
Leistungsverlust	1,8 W	2,0 W
Stromaufnahme (SM-Bus)	80 mA	
Stromaufnahme (24 V DC)	45 mA (ohne Last)	

Tabelle A- 147 Analoge Ausgänge

Technische Daten	SM 1232 AO 2 x 14 Bit	SM 1232 AO 4 x 14 Bit
Ausgänge	2	4
Typ	Spannung oder Strom	
Bereich	±10 V, 0 bis 20 mA oder 4 mA bis 20 mA	
Auflösung	Spannung: 14 Bit Strom: 13 Bits	
Vollausschlag (Datenwort)	Spannung: -27.648 bis 27.648, Strom: 0 bis 27.648 Siehe Abschnitt über Ausgangsbereiche für Spannung und Strom (Seite 1255).	
Genauigkeit (25 °C / -20 bis 60 °C)	±0,3 % / ±0,6 % des Vollausschlags	
Ausregelzeit (95 % des neuen Werts)	Spannung: 300 µs (R), 750 µs (1 uF) Strom: 600 µs (1 mH), 2 ms (10 mH)	
Lastimpedanz	Spannung: ≥ 1000 Ω Strom: ≤ 600 Ω	
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)	
Elektrische Trennung (Feld zu Logik)	Keine	
Leitungslänge (Meter)	100 m, verdrillt und geschirmt	

Tabelle A- 148 Diagnose

Technische Daten	SM 1232 AO 2 x 14 Bit	SM 1232 AO 4 x 14 Bit
Überlauf/Unterlauf	Ja	
Erdschluss (nur Spannungsmodus)	Ja	
Drahtbruch (nur Strommodus)	Ja	
24-V-DC-Niederspannung	Ja	

Tabelle A- 149 Schaltpläne der Analogausgangs-SMs

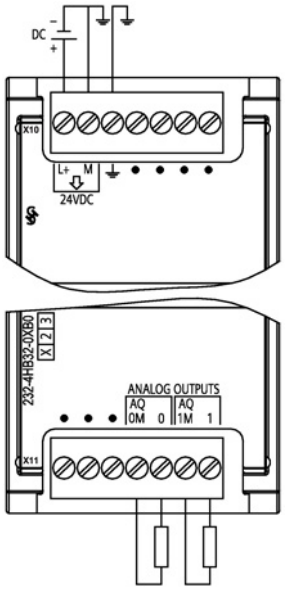
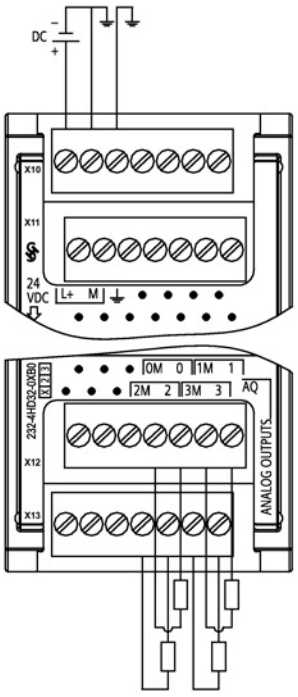
SM 1232 AO 2 x 14 Bit (6ES7 232-4HB32-0XB0)	SM 1232 AO 4 x 14 Bit (6ES7 232-4HD32-0XB0)
	
Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	

Tabelle A- 150 Anschlussbelegung für das SM 1232 AO 2 x 14 Bit (6ES7 232-4HB32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss
2	M / 24 V DC	Kein Anschluss
3	Funktionserde	Kein Anschluss
4	Kein Anschluss	AO 0M
5	Kein Anschluss	AO 0
6	Kein Anschluss	AO 1M
7	Kein Anschluss	AO 1

Tabelle A- 151 Anschlussbelegung für das SM 1232 AO 4 x 14 Bit (6ES7 232-4HD32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)	X12 (vergoldet)	X13 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Funktionserde	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	Kein Anschluss	Kein Anschluss	AO 0M	AO 2M
5	Kein Anschluss	Kein Anschluss	AO 0	AO 2

Pin	X10 (vergoldet)	X11 (vergoldet)	X12 (vergoldet)	X13 (vergoldet)
6	Kein Anschluss	Kein Anschluss	AO 1M	AO 3M
7	Kein Anschluss	Kein Anschluss	AO 1	AIO 3

A.8.3 Technische Daten des SM 1234 Analogein-/Analogausgabemoduls

Tabelle A- 152 Allgemeine technische Daten

Technische Daten	SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit
Artikelnummer	6ES7 234-4HE32-0XB0
Abmessungen (B x H x T) (mm)	45 x 100 x 75
Gewicht	220 Gramm
Leistungsverlust	2,4 W
Stromaufnahme (SM-Bus)	80 mA
Stromaufnahme (24 V DC)	60 mA (ohne Last)

Tabelle A- 153 Analoge Eingänge

Modell	SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit
Anzahl der Eingänge	4
Typ	Spannung oder Strom (differential): wählbar in Gruppen zu je 2
Bereich	±10 V, ±5 V, ±2,5 V, 0 bis 20 mA oder 4 mA bis 20 mA
Vollausschlag (Datenwort)	-27648 bis 27648
Überschwing-/Unterschwingbereich (Datenwort)	Spannung: 32511 bis 27649 / -27649 bis -32512 Strom: 32511 bis 27649 / 0 bis -4864 Siehe Abschnitt über Eingangsbereiche für Spannung und Strom (Seite 1254).
Überlauf/Unterlauf (Datenwort)	Spannung: 32767 bis 32512 / -32513 bis -32768 Strom: 32767 bis 32512 / -4865 bis -32768 Siehe Abschnitt über Eingangsbereiche für Spannung und Strom (Seite 1254).
Auflösung	12 Bit + Vorzeichenbit
Maximale Stehspannung/-strom	±35 V / ±40 mA
Glättung	Keine, schwach, mittel oder stark Siehe Abschnitt zu Schrittantwortzeiten (Seite 1253).
Rauschunterdrückung	400, 60, 50 oder 10 Hz Siehe Abschnitt zu Abtastraten (Seite 1253).
Eingangsimpedanz	≥ 9 MΩ (Spannung) / 280 Ω (Strom)
Elektrische Trennung (Feld zu Logik)	Keine
Genauigkeit (25 °C / -20 bis 60 °C)	±0,1% / ±0,2% des Vollausschlags
Analog-Digital-Umsetzzeit	625 µs (400 Hz Unterdrückung)
Gleichtaktunterdrückung	40 dB, Nennwert bei 60 Hz

Modell	SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit
Betriebssignalbereich[1]	Signal- plus Gleichtaktspannung muss kleiner als +12 V und größer als -12 V sein
Leitungslänge (Meter)	100 m, verdrillt und geschirmt

[1] Spannungen außerhalb des Betriebsbereichs, die an einem Kanal angelegt werden, können an anderen Kanälen Störungen verursachen.

Tabelle A- 154 Analoge Ausgänge

Technische Daten	SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit
Ausgänge	2
Typ	Spannung oder Strom
Bereich	±10 V oder 0 bis 20 mA
Auflösung	Spannung: 14 Bits, Strom: 13 Bits
Vollausschlag (Datenwort)	Spannung: -27648 bis 27648, Strom: 0 bis 27648 Siehe Abschnitt über Ausgangsbereiche für Spannung und Strom (Seite 1255).
Genauigkeit (25 °C / -20 bis 60 °C)	±0,3 % / ±0,6 % des Vollausschlags
Ausregelzeit (95 % des neuen Werts)	Spannung: 300 µs (R), 750 µs (1 uF) Strom: 600 µs (1 mH), 2 ms (10 mH)
Lastimpedanz	Spannung: ≥ 1000 Ω Strom: ≤ 600 Ω
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)
Elektrische Trennung (Feld zu Logik)	Keine
Leitungslänge (Meter)	100 m, verdrillt und geschirmt

Tabelle A- 155 Diagnose

Modell	SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit
Überlauf/Unterlauf	Ja
Erdschluss (nur Spannungsmodus)	Ja an Ausgängen
Drahtbruch (nur Strommodus)	Ja an Ausgängen
24-V-DC-Niederspannung	Ja

Tabelle A- 156 Schaltpläne des analogen Ein-/Ausgangs-SMs

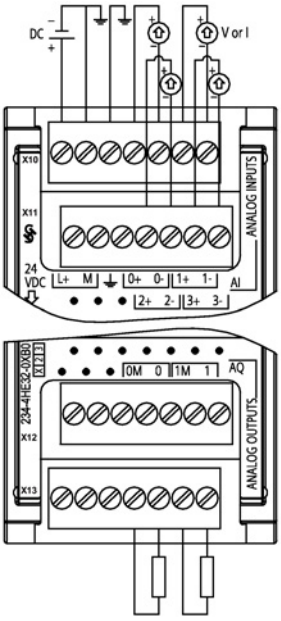
SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit (6ES7 234-4HE32-0XB0)	
	
Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	

Tabelle A- 157 Anschlussbelegung für das SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit (6ES7 234-4HE32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)	X12 (vergoldet)	X13 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Funktionserde	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	AI 0+	AI 2+	Kein Anschluss	AO 0M
5	AI 0-	AI 2-	Kein Anschluss	AO 0
6	AI 1+	AI 3+	Kein Anschluss	AO 1M
7	AI 1-	AI 3-	Kein Anschluss	AO 1

Hinweis

Nicht verwendete Spannungseingangskanäle sollten kurzgeschlossen werden.

Nicht verwendete Stromeingangskanäle sollten auf den Bereich 0 bis 20 mA gesetzt werden und/oder das Melden von Drahtbruch sollte deaktiviert werden.

Für den Strommodus konfigurierte Eingänge führen nur dann Schleifenstrom, wenn das Modul eingeschaltet und konfiguriert ist.

Stromeingangskanäle sind nur dann betriebsfähig, wenn der Sender mit externer Spannung versorgt wird.

A.8.4 Schrittantwort der analogen Eingänge

Tabelle A- 158 Schrittantwort (ms), 0 bis Vollausschlag, gemessen bei 95 %

Auswahl der Glättung (Mittelwertbildung aus Abtastwerten)	Rauschminderung/Unterdrückungsfrequenz (Auswahl Integrationszeit)			
	400 Hz (2,5 ms)	60 Hz (16,6 ms)	50 Hz (20 ms)	10 Hz (100 ms)
Keine (1 Zyklus): Keine Mittelwertbildung	4 ms	18 ms	22 ms	100 ms
Schwach (4 Zyklen): 4 Abtastwerte	9 ms	52 ms	63 ms	320 ms
Mittel (16 Zyklen): 16 Abtastwerte	32 ms	203 ms	241 ms	1200 ms
Stark (32 Zyklen): 32 Abtastwerte	61 ms	400 ms	483 ms	2410 ms
Abtastzeit				
• 4 AI x 13 Bit	• 0,625 ms	• 4,17 ms	• 5 ms	• 25 ms
• 8 AI x 13 Bit	• 1,25 ms	• 4,17 ms	• 5 ms	• 25 ms
• 4 AI4 x 16 Bit	• 0,417 ms	• 0,397 ms	• 0,400 ms	• 0,400 ms

A.8.5 Abtastzeit und Aktualisierungszeiten der Analogeingänge

Tabelle A- 159 Abtastzeit und Aktualisierungszeit

Unterdrückungsfrequenz (Integrationszeit)	Abtastzeit	Aktualisierungszeit des Moduls für alle Kanäle	
		4-kanaliges SM	8-kanaliges SM
400 Hz (2,5 ms)	• 4-kanaliges SM: 0,625 ms • 8-kanaliges SM: 1,250 ms	0,625 ms	1,250 ms
60 Hz (16,6 ms)	4,170 ms	4,17 ms	4,17 ms
50 Hz (20 ms)	5,000 ms	5 ms	5 ms
10 Hz (100 ms)	25,000 ms	25 ms	25 ms

A.8.6 Messbereiche der analogen Eingänge für Spannung und Strom (SB und SM)

Tabelle A- 160 Darstellung Analogeingang für Spannung (SB und SM)

System		Messbereich Spannung				
Dezimal	**Hexadezimal**	**±10 V**	**±5 V**	**±2,5 V**	**±1,25 V**	
32767	7FFF[1]	11,851 V	5,926 V	2,963 V	1,481 V	Überlauf
32512	7F00					
32511	7EFF	11,759 V	5,879 V	2,940 V	1,470 V	Überschwingbereich
27649	6C01					
27648	6C00	10 V	5 V	2,5 V	1,250 V	Bemessungsbereich
20736	5100	7,5 V	3,75 V	1,875 V	0,938 V	
1	1	361,7 µV	180,8 µV	90,4 µV	45,2 µV	
0	0	0 V	0 V	0 V	0 V	
-1	FFFF					
-20736	AF00	-7,5 V	-3,75 V	-1,875 V	-0,938 V	
-27648	9400	-10 V	-5 V	-2,5 V	-1,250 V	
-27649	93FF					Unterschwingbereich
-32512	8100	-11,759 V	-5,879 V	-2,940 V	-1,470 V	
-32513	80FF					Unterlauf
-32768	8000	-11,851 V	-5,926 V	-2,963 V	-1,481 V	

[1] 7FFF kann aus einem der folgenden Gründe zurückgegeben werden: Überlauf (wie in dieser Tabelle angemerkt), bevor gültige Werte verfügbar sind (beispielsweise direkt nach dem Einschalten), oder wenn ein Drahtbruch erkannt wird.

Tabelle A- 161 Darstellung Analogeingang für Strom (SB und SM)

System		Messbereich Strom		
Dezimal	**Hexadezimal**	**0 mA bis 20 mA**	**4 mA bis 20 mA**	
32767	7FFF	23,70 mA	22,96 mA	Überlauf
32512	7F00			
32511	7EFF	23,52 mA	22,81 mA	Überschwingbereich
27649	6C01			
27648	6C00	20 mA	20 mA	Nennbereich
20736	5100	15 mA	16 mA	
1	1	723,4 nA	4 mA + 578,7 nA	
0	0	0 mA	4 mA	
-1	FFFF			Unterschwingbereich
-4864	ED00	-3,52 mA	1,185 mA	
-4865	ECFF			Unterlauf
-32768	8000			

A.8.7 Messbereiche der analogen Ausgänge für Spannung und Strom (SB und SM)

Tabelle A- 162 Darstellung Analogausgang für Spannung (SB und SM)

System		Spannungsausgangsbereich	
Dezimal	**Hexadezimal**	**±10 V**	
32767	7FFF	Siehe Hinweis 1	Überlauf
32512	7F00	Siehe Hinweis 1	
32511	7EFF	11,76 V	Überschwingbereich
27649	6C01		
27648	6C00	10 V	Bemessungsbereich
20736	5100	7,5 V	
1	1	361,7 µV	
0	0	0 V	
-1	FFFF	-361,7 µV	
-20736	AF00	-7,5 V	
-27648	9400	-10 V	
-27649	93FF		Unterschwingbereich
-32512	8100	-11,76 V	
-32513	80FF	Siehe Hinweis 1	Unterlauf
-32768	8000	Siehe Hinweis 1	

[1] Bei Überlauf oder Unterlauf nehmen die Analogausgänge den Ersatzwert für den Betriebszustand STOP an.

Tabelle A- 163 Darstellung Analogausgang für Strom (SB und SM)

System		Stromausgangsbereich		
Dezimal	**Hexadezimal**	**0 mA bis 20 mA**	**4 mA bis 20 mA**	
32767	7FFF	Siehe Hinweis 1	Siehe Hinweis 1	Überlauf
32512	7F00	Siehe Hinweis 1	Siehe Hinweis 1	
32511	7EFF	23,52 mA	22,81 mA	Überschwingbereich
27649	6C01			
27648	6C00	20 mA	20 mA	Bemessungsbereich
20736	5100	15 mA	16 mA	
1	1	723,4 nA	4 mA + 578,7 nA	
0	0	0 mA	4 mA	
-1	FFFF		4 mA bis 578,7 nA	Unterschwingbereich
-6912	E500		0 mA	
-6913	E4FF			Nicht möglich. Ausgangswert auf 0 mA begrenzt.
-32512	8100			
-32513	80FF	Siehe Hinweis 1	Siehe Hinweis 1	Unterlauf
-32768	8000	Siehe Hinweis 1	Siehe Hinweis 1	

[1] Bei Überlauf oder Unterlauf nehmen die Analogausgänge den Ersatzwert für den Betriebszustand STOP an.

A.9 Thermoelement- und RTD-Signalmodule (SMs)

A.9.1 SM 1231 Thermoelement

Tabelle A- 164 Allgemeine technische Daten

Modell	SM 1231 AI 4 x 16 Bit TC	SM 1231 AI 8 x 16 Bit TC
Artikelnummer	6ES7 231-5QD32-0XB0	6ES7 231-5QF32-0XB0
Abmessungen B x H x T (mm)	45 x 100 x 75	
Gewicht	180 Gramm	190 Gramm
Leistungsverlust	1,5 W	
Stromaufnahme (SM-Bus)	80 mA	
Stromaufnahme (24 V DC) [1]	40 mA	

[1] 20,4 bis 28,8 V DC (Klasse 2, leistungsbegrenzt oder Geberspannung aus SPS)

Tabelle A- 165 Analoge Eingänge

Modell		SM 1231 AI 4 x 16 Bit TC	SM 1231 AI 8 x 16 Bit TC
Anzahl der Eingänge		4	8
Bereich Nennbereich (Datenwort) Überlauf/Unterlauf (Datenwort) Überlauf/Unterlauf (Datenwort)		Siehe Thermoelement-Auswahltabelle (Seite 1260).	
Auflösung	Temperaturbereich	0,1 °C / 0,1 °F	
	Spannung	15 Bit plus Vorzeichen	
Max. Stehspannung		±35 V	
Rauschunterdrückung		85 dB für die gewählte Filtereinstellung (10 Hz, 50 Hz, 60 Hz oder 400 Hz)	
Gleichtaktunterdrückung		> 120 dB bei 120 V AC	
Impedanz		≥ 10 MΩ	
Elektrische Trennung	Feld zu Logik	500 V AC	
	Feld zu 24 V DC	500 V AC	
	24 V DC zu Logik	500 V AC	
Kanal zu Kanal		120 V AC	
Genauigkeit		Siehe Thermoelement-Auswahltabelle (Seite 1260).	
Wiederholgenauigkeit		±0,05 % Vollausschlag	
Messprinzip		Integrierend	
Aktualisierungszeit Modul		Siehe Auswahltabelle Rauschminderung (Seite 1260).	
Fehler kalte Verbindungsstelle		±1,5 °C	
Leitungslänge (Meter)		Max. 100 m zum Geber	
Leitungswiderstand		max. 100 Ω	

Tabelle A- 166 Diagnose

Modell	SM 1231 AI 4 x 16 Bit TC	SM 1231 AI 8 x 16 Bit TC
Überlauf/Unterlauf[1]	Ja	
Drahtbruch (nur Strommodus) [2]	Ja	
24-V-DC-Niederspannung[1]	Ja	

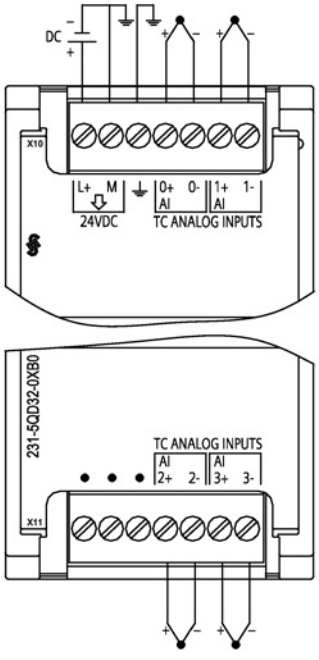
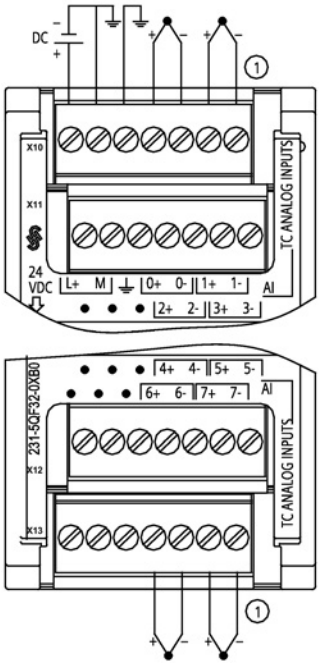
[1] Die Informationen der Diagnosealarme "Überlauf", "Unterlauf" und "Niederspannung" werden auch dann mit den Analogdatenwerten gemeldet, wenn die entsprechenden Alarme bei der Modulprojektierung deaktiviert werden.

[2] Wenn der Drahtbruchalarm deaktiviert ist und in der Geberverdrahtung ein frei liegender Draht vorliegt, kann das Modul zufällige Werte melden.

Das analoge Thermoelement-Signalmodul SM 1231 TC misst den Wert der an die Moduleingänge angeschlossenen Spannung. Als Temperaturmessart sind entweder "Thermoelement" oder "Spannung" möglich.

- "Thermoelement": Der Messwert wird in Grad, multipliziert mit zehn, ausgegeben (Beispiel: 25,3 Grad werden als Dezimalwert 253 dargestellt).
- "Spannung": Der Messbereichsendwert im Nennbereich beträgt 27648 dezimal.

Tabelle A- 167 Schaltpläne der Thermoelement-SMs

SM 1231 AI 4 x TC 16 Bit (6ES7 231-5QD32-0XB0)	SM 1231 AI 8 x TC Bit (6ES7 231-5QF32-0XB0)
	
Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	

① Die Anschlüsse von TC 2, 3, 4 und 5 werden zur besseren Übersichtlichkeit weggelassen.

Tabelle A- 168 Anschlussbelegung für das SM 1231 AI 4 x TC 16 Bit (6ES7 231-5QD32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss
2	M / 24 V DC	Kein Anschluss
3	Funktionserde	Kein Anschluss
4	AI 0+ /TC	AI 2+ /TC
5	AI 0- /TC	AI 2- /TC
6	AI 1+ /TC	AI 3+ /TC
7	AI 1- /TC	AI 3- /TC

Tabelle A- 169 Anschlussbelegung für das SM 1231 AI 8 x TC Bit (6ES7 231-5QF32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)	X12 (vergoldet)	X13 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Funktionserde	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	AI 0+ /TC	AI 2+ /TC	AI 4 I- /TC	AI 6 I- /TC
5	AI 0- /TC	AI 2- /TC	AI 4 I+ /TC	AI 6 I+ /TC
6	AI 1+ /TC	AI 3+ /TC	AI 5 M- /TC	AI 7 M- /TC
7	AI 1- /TC	AI 3- /TC	AI 5 M+ /TC	AI 7 M+ /TC

Hinweis

Nicht verwendete analoge Eingänge sollten kurzgeschlossen werden.

Die vom Thermoelement nicht verwendeten Kanäle können deaktiviert werden. Es tritt kein Fehler auf, wenn ein nicht verwendeter Kanal deaktiviert wird.

A.9.1.1 Grundlegende Funktionsweise eines Thermoelements

Thermoelemente entstehen, wenn zwei unterschiedliche Metalle elektrisch miteinander verbunden werden. Dadurch wird eine Spannung erzeugt, die proportional zu der Temperatur der Verbindungsstelle ist. Es handelt sich um eine geringe Spannung. Ein Mikrovolt kann viele Grade darstellen. Grundlage für die Temperaturmessung mit Thermoelementen sind das Messen der Spannung eines Thermoelements, das Kompensieren von zusätzlichen Verbindungsstellen und das Linearisieren der Ergebnisse.

Wenn Sie ein Thermoelement an das SM 1231 Thermoelementmodul anschließen, werden die beiden Leitungen der unterschiedlichen Metalle am Signalanschluss des Moduls angeschlossen. Die Stelle, an der die beiden unterschiedlichen Leitungen miteinander verbunden werden, bildet den Sensor des Thermoelements.

Zwei weitere Thermoelemente entstehen an der Stelle, an der die unterschiedlichen Leitungen an den Signalanschluss angeschlossen werden. Die Temperatur des Anschlusses erzeugt eine Spannung, die zu der Spannung des Thermoelementsensors addiert wird. Wird diese Spannung nicht ausgeglichen, weicht die ausgegebene Temperatur von der Temperatur des Sensors ab.

Durch die Kompensation der kalten Verbindungsstelle wird das Thermoelement am Anschluss ausgeglichen. Tabellen für Thermoelemente basieren auf einer Bezugstemperatur an der Verbindungsstelle, üblicherweise Null Grad Celsius. Durch die Kompensation der kalten Verbindungsstelle wird der Anschluss auf Null Grad Celsius kompensiert. Die Spannung, die durch das Thermoelement des Anschlusses addiert wird, wird durch die Kompensation der kalten Verbindungsstelle korrigiert. Die Temperatur des Moduls wird innen gemessen und dann in einen Wert umgewandelt, der zur Sensorumsetzung addiert wird. Die korrigierte Sensorumsetzung wird dann mittels der Thermoelement-Tabellen linearisiert.

Für die optimale Funktionsweise der Kompensation der kalten Verbindungsstelle muss sich das Thermoelementmodul in einer thermisch stabilen Umgebung befinden. Ein langsame Veränderung (weniger als 0,1 °C/Minute) der Temperatur in der Umgebung des Moduls wird innerhalb der Modulspezifikation korrekt ausgeglichen. Auch Luftbewegungen am Modul verursachen Fehler bei der Kompensation der kalten Verbindungsstelle.

Ist eine bessere Kompensation von Fehlern an der kalten Verbindungsstelle erforderlich, kann eine externe isothermische Klemmenleiste verwendet werden. Mit dem Thermoelementmodul kann eine auf 0 °C bezogene oder eine auf 50 °C bezogene Klemmenleiste eingesetzt werden.

A.9.1.2 Auswahltabellen für das SM 1231 Thermoelement

Die Bereiche und Genauigkeit der verschiedenen vom SM 1231 Thermoelement-Signalmodul unterstützten Thermoelementtypen entnehmen Sie bitte der unten stehenden Tabelle.

Tabelle A- 170 SM 1231 Thermoelement-Auswahltabelle

Typ	Minimum unterer Bereich[1]	Unterer Grenzwert Nennbereich	Oberer Grenzwert Nennbereich	Maximum oberer Bereich[2]	Genauigkeit Normalbereich [3, 4] bei 25 °C	Genauigkeit Normalbereich [1, 2] bei -20 °C bis 60 °C
J	-210,0 °C	-150,0 °C	1.200,0 °C	1.450,0 °C	±0,3 °C	±0,6 °C
K	-270,0 °C	-200,0 °C	1.372,0 °C	1.622,0 °C	±0,4 °C	±1,0 °C
T	-270,0 °C	-200,0 °C	400,0 °C	540,0 °C	±0,5 °C	±1,0 °C
E	-270,0 °C	-200,0 °C	1.000,0 °C	1.200,0 °C	±0,3 °C	±0,6 °C
R & S	-50,0 °C	100,0 °C	1.768,0 °C	2.019,0 °C	±1,0 °C	±2,5 °C
B	0,0 °C	200,0 °C	800,0 °C	--	±2,0 °C	±2,5 °C
	--	800,0 °C	1.820,0 °C	1.820,0 °C	±1,0 °C	±2,3 °C
N	-270,0 °C	-200,0 °C	1.300,0 °C	1.550,0 °C	±1,0 °C	±1,6 °C
C	0,0 °C	100,0 °C	2.315,0 °C	2.500,0 °C	±0,7 °C	±2,7 °C
TXK/XK(L)	-200,0 °C	-150,0 °C	800,0 °C	1.050,0 °C	±0,6 °C	±1,2 °C
Spannung	-32512	-27648 -80 mV	27648 80 mV	32511	±0.05%	±0.1%

1 Die Thermoelementwerte unterhalb des Minimums für den unteren Bereich werden als -32768 angegeben.

2 Die Thermoelementwerte oberhalb des Minimums für den oberen Bereich werden als 32767 angegeben.

3 Der interne Fehler an der kalten Verbindungsstelle beträgt ±1,5 °C für alle Bereiche. Dieser Wert ist zum in dieser Tabelle aufgeführten Fehler zu addieren. Das Modul benötigt eine Aufwärmzeit von mindestens 30 Minuten, bis die hier genannten Spezifikationen erfüllt werden.

4 Bei Vorhandensein von abgestrahlter Funkfrequenz zwischen 970 MHz und 990 MHz kann sich die Genauigkeit des SM 1231 AI 4 x 16 Bit TC verschlechtern.

Hinweis

Thermoelementkanal

Für jeden Kanal am Thermoelement-Signalmodul kann ein unterschiedlicher Thermoelementtyp konfiguriert werden (einstellbar in der Software während der Modulkonfiguration).

Tabelle A- 171 Rauschminderung und Aktualisierungszeiten für das SM 1231 Thermoelement

Auswahl Unterdrückungsfrequenz	Integrationszeit	Aktualisierungszeit 4-Kanal-Modul(Sekunden)	Aktualisierungszeit 8-Kanal-Modul(Sekunden)
400 Hz (2,5 ms)	10 ms [1]	0.143	0.285
60 Hz (16,6 ms)	16,67 ms	0.223	0.445
50 Hz (20 ms)	20 ms	0.263	0.525
10 Hz (100 ms)	100 ms	1.225	2.450

[1] Um die Auflösung und Messgenauigkeit des Moduls bei Auswahl der 400-Hz-Unterdrückung aufrecht zu erhalten, beträgt die Integrationszeit 10 ms. Mit dieser Auswahl erfolgt auch die Rauschunterdrückung bei 100 Hz und 200 Hz.

Für die Messung von Thermoelementen wird eine Integrationszeit von 100 ms empfohlen. Niedriger eingestellte Integrationszeiten führen zu einem höheren Wiederholgenauigkeitsfehler der Temperaturmessungen.

Hinweis

Nach dem Einschalten führt das Modul die interne Kalibrierung für den A/D-Wandler durch. In diesem Zeitraum meldet das Modul auf jedem Kanal den Wert 32767, bis für den jeweiligen Kanal gültige Daten vorliegen. Diese Initialisierungszeit muss im Anwenderprogramm ggf. berücksichtigt werden. Weil sich die Konfiguration des Moduls auf die Dauer der Initialisierungszeit auswirken kann, prüfen Sie das Verhalten des Moduls in Ihrer Konfiguration. Ggf. können Sie Logik in Ihr Anwenderprogramm aufnehmen, um die Initialisierungszeit des Moduls unterzubringen.

Darstellung der Analogwerte für ein Thermoelement vom Typ J

Die Analogwerte für ein Thermoelement vom Typ J finden Sie in der folgenden Tabelle.

Tabelle A- 172 Darstellung der Analogwerte für ein Thermoelement vom Typ J

Typ J in °C	Einheiten		Typ J in °F	Einheiten		Bereich
	Dezimal	Hexadezimal		Dezimal	Hexadezimal	
> 1450.0	32767	7FFF	> 2642.0	32767	7FFF	Überlauf
1450.0 : 1200.1	14500 : 12001	38A4 : 2EE1	2642.0 : 2192.2	26420 : 21922	6734 : 55A2	Oberhalb des Bereichs
1200.0 : -150.0	12000 : -1500	2EE0 : FA24	2192.0 : -238.0	21920 : -2380	55A0 : F6B4	Bemes-sungsbe-reich
< -150.0	-32768	8000	< -238.0	-32768	8000	Unterlauf[1]

[1] Fehlerhafte Verdrahtung (z. B. Verpolung oder offene Eingänge) oder ein Sensorfehler im negativen Bereich (z. B. falscher Typ des Thermoelements) können bewirken, dass das Thermoelementmodul Unterlauf meldet.

A.9.2 SM 1231 RTD

Technische Daten des SM 1231 RTD

Tabelle A- 173 Allgemeine technische Daten

Technische Daten	SM 1231 AI 4 x RTD x 16 Bit	SM 1231 AI 8 x RTD x 16 Bit
Artikelnummer	6ES7 231-5PD32-0XB0	6ES7 231-5PF32-0XB0
Abmessungen B x H x T (mm)	45 x 100 x 75	70 x 100 x 75
Gewicht	220 Gramm	270 Gramm
Leistungsverlust	1,5 W	
Stromaufnahme (SM-Bus)	80 mA	90 mA
Stromaufnahme (24 V DC) [1]	40 mA	

[1] 20,4 bis 28,8 V DC (Klasse 2, leistungsbegrenzt oder Geberspannung aus CPU)

Tabelle A- 174 Analoge Eingänge

Technische Daten		SM 1231 AI 4 x RTD x 16 Bit	SM 1231 AI 8 x RTD x16 Bit
Anzahl der Eingänge		4	8
Typ		Modulreferenz RTD und Ω	
Bereich Nennbereich (Datenwort) Überschwing-/Unterschwingbereich (Datenwort) Überlauf/Unterlauf (Datenwort)		Siehe Auswahltabelle RTD-Geber (Seite 1265).	
Auflösung	Temperatur	0,1 °C / 0,1 °F	
	Widerstand	15 Bit plus Vorzeichen	
Max. Stehspannung		±35 V	
Rauschunterdrückung		85 dB für die gewählte Rauschminderung (10 Hz, 50 Hz, 60 Hz oder 400 Hz)	
Gleichtaktunterdrückung		>120 dB	
Impedanz		≥ 10 MΩ	
Elektrische Trennung	Feldseite zu Logik	500 V AC	
	Feld zu 2 V DC	500 V AC	
	24 V DC zu Logik	500 V AC	
Trennung Kanäle untereinander		Keine	
Genauigkeit		Siehe Auswahltabelle RTD-Geber (Seite 1265).	
Wiederholgenauigkeit		±0,05 % Vollausschlag	
Maximale Verlustleistung Geber		0,5 mW	
Messprinzip		Integrierend	
Aktualisierungszeit Modul		Siehe Auswahltabelle Rauschminderung (Seite 1265).	
Leitungslänge (Meter)		Max. 100 m zum Geber	
Leitungswiderstand		20 Ω, 2,7 Ω für max. 10 Ω RTD	

Tabelle A- 175 Diagnose

Technische Daten	SM 1231 AI 4 x RTD x 16 Bit	SM 1231 AI 8 x RTD x16 Bit
Überlauf/Unterlauf [1,2]	Ja	
Drahtbruch[3]	Ja	
24-V-DC-Niederspannung[1]	Ja	

1 Die Informationen der Diagnosealarme "Überlauf", "Unterlauf" und "Niederspannung" werden auch dann mit den Analogdatenwerten gemeldet, wenn die entsprechenden Alarme bei der Modulprojektierung deaktiviert werden.

2 Für Widerstandsmessbereiche ist die Prüfung auf Unterlauf grundsätzlich nicht aktiviert.

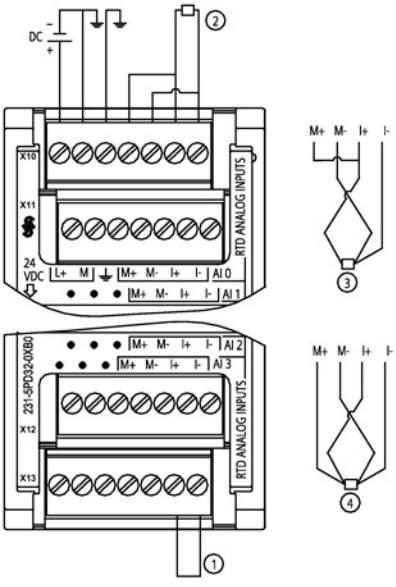
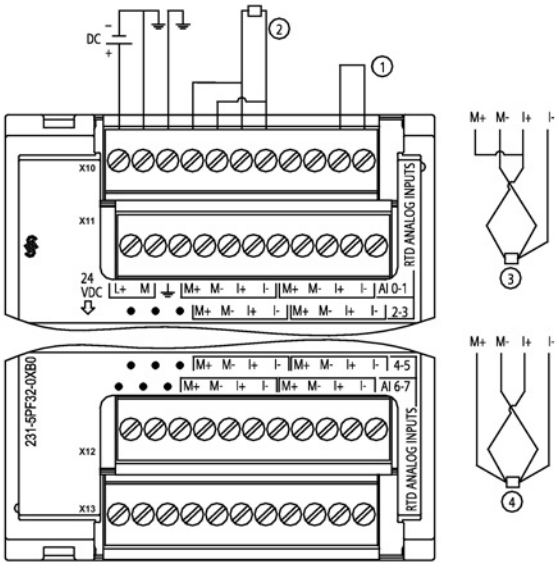
3 Wenn der Drahtbruchalarm deaktiviert ist und in der Geberverdrahtung ein frei liegender Draht vorliegt, kann das Modul zufällige Werte melden.

Das analoge Signalmodul SM 1231 RTD misst den Wert des an die Moduleingänge angeschlossenen Widerstands. Als Messart kann entweder "Widerstand" oder "Thermischer Widerstand" ausgewählt werden.

- "Widerstand": Der Messbereichsendwert im Nennbereich beträgt 27648 dezimal.
- "Thermischer Widerstand": Der Messwert wird in Grad, multipliziert mit zehn, ausgegeben (Beispiel: 25,3 Grad werden als Dezimalwert 253 dargestellt). Die Werte des klimatischen Bereichs werden in Grad, multipliziert mit Hundert, ausgegeben (Beispiel: 25,34 Grad werden als Dezimalwert 2534 dargestellt).

Das Signalmodul SM 1231 RTD unterstützt Messungen über 2-Leiter-, 3-Leiter- und 4-Leiter-Anschlüsse zum Geberwiderstand.

Tabelle A- 176 Schaltpläne der RTD-SMs

SM 1231 RTD 4 x 16 Bit (6ES7 231-5PD32-0XB0)	SM 1231 RTD 8 x 16 Bit (6ES7 231-5PF32-0XB0)
	

① Nicht belegte RTD-Eingänge zurückschleifen

② 2-Draht-RTD ③ 3-Draht-RTD ④ 4-Draht-RTD

HINWEIS: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.

Tabelle A- 177 Anschlussbelegung für das SM 1231 RTD 4 x 16 Bit (6ES7 231-5PD32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)	X12 (vergoldet)	X13 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Funktionserde	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	AI 0 M+ /RTD	AI 1 M+ /RTD	AI 2 M+ /RTD	AI 3 M+ /RTD
5	AI 0 M- /RTD	AI 1 M- /RTD	AI 2 M- /RTD	AI 3 M- /RTD
6	AI 0 I+ /RTD	AI 1 I+ /RTD	AI 2 I+ /RTD	AI 3 I+ /RTD
7	AI 0 I- /RTD	AI 1 I- /RTD	AI 2 I- /RTD	AI 3 I- /RTD

Tabelle A- 178 Anschlussbelegung für das SM 1231 RTD 8 x 16 Bit (6ES7 231-5PF32-0XB0)

Pin	X10 (vergoldet)	X11 (vergoldet)	X12 (vergoldet)	X13 (vergoldet)
1	L+ / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Funktionserde	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	AI 0 M+ /RTD	AI 2 M+ /RTD	AI 4 M+ /RTD	AI 6 M+ /RTD
5	AI 0 M- /RTD	AI 2 M- /RTD	AI 4 M- /RTD	AI 6 M- /RTD
6	AI 0 I+ /RTD	AI 2 I+ /RTD	AI 4 I+ /RTD	AI 6 I+ /RTD
7	AI 0 I- /RTD	AI 2 I- /RTD	AI 4 I- /RTD	AI 6 I- /RTD
8	AI 1 M+ /RTD	AI 3 M+ /RTD	AI 5 M+ /RTD	A7 M+ /RTD
9	AI 1 M- /RTD	AI 3 M- /RTD	AI 5 M- /RTD	AI 7 M- /RTD
10	AI 1 I+ /RTD	AI 3 I+ /RTD	AI 5 I+ /RTD	AI 7 I+ /RTD
11	AI 1 I- /RTD	AI 3 I- /RTD	AI 5 I- /RTD	AI 7 I- /RTD

Hinweis

Die vom RTD nicht verwendeten Kanäle können deaktiviert werden. Es tritt kein Fehler auf, wenn ein nicht verwendeter Kanal deaktiviert wird.

Das RTD-Modul benötigt eine kontinuierliche Stromschleife, um die zusätzliche Stabilisierungszeit zu beseitigen, die bei einem nicht verwendeten Kanal, der nicht deaktiviert ist, automatisch anfällt. Aus Konsistenzgründen muss ein Widerstand an das RTD-Modul angeschlossen werden (z. B. der 2-Draht-RTD-Anschluss).

A.9.2.1 Auswahltabellen für das SM 1231 RTD

Tabelle A- 179 Bereiche und Genauigkeit für die verschiedenen Geber, die von den RTD-Modulen unterstützt werden

Temperaturkoeffizient	RTD-Typ	Minimum unterer Bereich[1]	Unterer Grenzwert Nennbereich	Oberer Grenzwert Nennbereich	Maximum oberer Bereich[2]	Genauigkeit Normalbereich bei 25 °C	Genauigkeit Normalbereich -20 °C bis 60 °C
Pt 0,003850 ITS90 DIN EN 60751	Pt 100 Klima	-145,00 °C	-120,00 °C	145,00 °C	155,00 °C	±0,20 °C	±0,40 °C
	Pt 10	-243,0 °C	-200,0 °C	850,0 °C	1.000,0 °C	±1,0 °C	±2,0 °C
	Pt 50	-243,0 °C	-200,0 °C	850,0 °C	1.000,0 °C	±0,5 °C	±1,0 °C
	Pt 100						
	Pt 200						
	Pt 500						
	Pt 1000						
Pt 0,003902 Pt 0,003916 Pt 0,003920	Pt 100	-243,0 °C	-200,0 °C	850,0 °C	1.000,0 °C	±0,5 °C	±1,0 °C
	Pt 200	-243,0 °C	-200,0 °C	850,0 °C	1.000,0 °C	±0,5 °C	±1,0 °C
	Pt 500						
	Pt 1000						
Pt 0,003910	Pt 10	-273,2 °C	-240,0 °C	1.100,0 °C	1.295 °C	±1,0 °C	±2,0 °C
	Pt 50	-273,2 °C	-240,0 °C	1.100,0 °C	1.295 °C	±0,8 °C	±1,6 °C
	Pt 100						
	Pt 500						
Ni 0,006720 Ni 0,006180	Ni 100	-105,0 °C	-60,0 °C	250,0 °C	295,0 °C	±0,5 °C	±1,0 °C
	Ni 120						
	Ni 200						
	Ni 500						
	Ni 1000						
LG-Ni 0,005000	LG-Ni 1000	-105,0 °C	-60,0 °C	250,0 °C	295,0 °C	±0,5 °C	±1,0 °C
Ni 0,006170	Ni 100	-105,0 °C	-60,0 °C	180,0 °C	212,4 °C	±0,5 °C	±1,0 °C
Cu 0,004270	Cu 10	-240,0 °C	-200,0 °C	260,0 °C	312,0 °C	±1,0 °C	±2,0 °C
Cu 0,004260	Cu 10	-60,0 °C	-50,0 °C	200,0 °C	240,0 °C	±1,0 °C	±2,0 °C
	Cu 50	-60,0 °C	-50,0 °C	200,0 °C	240,0 °C	±0,6 °C	±1,2 °C
	Cu 100						
Cu 0,004280	Cu 10	-240,0 °C	-200,0 °C	200,0 °C	240,0 °C	±1,0 °C	±2,0 °C
	Cu 50	-240,0 °C	-200,0 °C	200,0 °C	240,0 °C	±0,7 °C	±1,4 °C
	Cu 100						

1 Die RTD-Werte unterhalb des Minimums für den unteren Bereich werden als -32768 ausgegeben.

2 RTD-Werte oberhalb des Maximums für den oberen Bereich werden als +32767 ausgegeben.

Tabelle A- 180 Widerstand

Bereich	Minimum unterer Bereich	Unterer Grenzwert Nennbereich	Oberer Grenzwert Nennbereich	Maximum oberer Bereich[1]	Genauigkeit Normalbereich bei 25 °C	Genauigkeit Normalbereich -20 °C bis 60 °C
150 Ω	nicht zutreffend	0 (0 Ω)	27648 (150 Ω)	176.383 Ω	±0.05%	±0.1%
300 Ω	nicht zutreffend	0 (0 Ω)	27648 (300 Ω)	352.767 Ω	±0.05%	±0.1%
600 Ω	nicht zutreffend	0 (0 Ω)	27648 (600 Ω)	705.534 Ω	±0.05%	±0.1%

[1] Die Widerstandswerte oberhalb des Minimums für den oberen Bereich werden als +32767 ausgegeben.

Hinweis

Für alle aktivierte Kanäle ohne angeschlossenen Geber meldet das Modul 32767. Wenn außerdem die Prüfung auf offene Leitungen aktiviert ist, blinken am Modul die entsprechenden roten LED.

Wenn RTD-Bereiche von 500 Ω und 1000 Ω mit anderen Widerständen niederen Werts verwendet werden, kann sich der Fehler auf den zweifachen spezifizierten Fehler erhöhen.

Optimale Genauigkeit für die 10 Ω-RTD-Bereiche ermöglichen 4-Leiter-Anschlüsse.

Der Widerstand der Anschlussleitungen im 2-Leiter-Modus verursacht einen Fehler der Gebermessung. Die Messgenauigkeit ist daher nicht mehr gewährleistet.

Tabelle A- 181 Rauschminderung und Aktualisierungszeiten für die RTD-Module

Auswahl Unterdrückungsfrequenz	Integrationszeit	Aktualisierungszeit (Sekunden)	
		4-Kanal-Modul	8-Kanal-Modul
400 Hz (2,5 ms)	10 ms [1]	4-/2-Draht: 0.142 3-Draht: 0.285	4-/2-Draht: 0.285 3-Draht: 0.525
60 Hz (16,6 ms)	16,67 ms	4-/2-Draht: 0.222 3-Draht: 0.445	4-/2-Draht: 0.445 3-Draht: 0.845
50 Hz (20 ms)	20 ms	4-/2-Draht: 0.262 3-Draht: .505	4-/2-Draht: 0.524 3-Draht: 1.015
10 Hz (100 ms)	100 ms	4-/2-Draht: 1.222 3-Draht: 2.445	4-/2-Draht: 2.425 3-Draht: 4.845

[1] Um die Auflösung und Messgenauigkeit des Moduls bei Auswahl des 400-Hz-Filters aufrecht zu erhalten, beträgt die Integrationszeit 10 ms. Mit dieser Auswahl erfolgt auch die Rauschunterdrückung bei 100 Hz und 200 Hz.

Hinweis

Nach dem Einschalten führt das Modul die interne Kalibrierung für den A/D-Wandler durch. In diesem Zeitraum meldet das Modul auf jedem Kanal den Wert 32767, bis für den jeweiligen Kanal gültige Daten vorliegen. Diese Initialisierungszeit muss im Anwenderprogramm ggf. berücksichtigt werden. Weil sich die Konfiguration des Moduls auf die Dauer der Initialisierungszeit auswirken kann, prüfen Sie das Verhalten des Moduls in Ihrer Konfiguration. Ggf. können Sie Logik in Ihr Anwenderprogramm aufnehmen, um die Initialisierungszeit des Moduls unterzubringen.

Darstellung der Analogwerte für RTDs

Die folgenden Tabellen zeigen den digitalisierten Messwert für die Sensoren mit RTD-Standardtemperaturbereich.

Tabelle A- 182 Darstellung von Analogwerten für die Widerstandsthermometer PT 100, 200, 500, 1000 und PT 10, 50, 100, 500 GOST (0,003850) Standard

Pt x00 Standard in °C (1 Ziffer = 0,1 °C)	Einheiten		Pt x00 Standard in °F (1 Ziffer = 0,1 °F)	Einheiten		Bereich
	Dezimal	Hexadezimal		Dezimal	Hexadezimal	
> 1000.0	32767	7FFF	> 1832.0	32767	7FFF	Überlauf
1000.0 : 850.1	10000 : 8501	2710 : 2135	1832.0 : 1562.1	18320 : 15621	4790 : 3D05	Oberhalb des Bereichs
850.0 : -200.0	8500 : -2000	2134 : F830	1562.0 : -328.0	15620 : -3280	3D04 : F330	Bemessungsbereich
-200.1 : -243.0	-2001 : -2430	F82F : F682	-328.1 : -405.4	-3281 : -4054	F32F : F02A	Unterhalb des Bereichs
< -243.0	-32768	8000	< -405.4	-32768	8000	Unterlauf

A.10 Technologiemodule

A.10.1 SM 1278 4xIO-Link-Master SM

Tabelle A- 183 Allgemeine technische Daten

Technische Daten		Signalmodul SM 1278 4xIO-Link-Master
Artikelnummer		6ES7 278-4BD32-0XB0
Abmessungen B x H x T (mm)		45 x 100 x 75
Gewicht		150 Gramm
Allgemeine Informationen		
	I&M-Daten	Ja, IM0 bis IM3
Versorgungsspannung		
	Nennspannung (DC)	24 V DC
	Unterer Grenzwert gültiger Bereich (DC)	19,2 V; 20,5 V bei Verwendung von IO-Link (weil die Versorgungsspannung für IO-Link-Geräte auf dem Master mindestens 20 V betragen muss)
	Oberer Grenzwert gültiger Bereich (DC)	28,8 V DC
	Verpolschutz	Ja
Eingangsstrom		
	Stromaufnahme	65 mA, ohne Last
Drehgeberversorgung		
	Anzahl der Ausgänge	4
	Ausgangsstrom, Nennwert	200 mA
Verlustleistung		
	Verlustleistung, typ.	1 W, ohne Portlast
Digitaleingänge/-ausgänge		
	Leitungslänge (Meter)	Max. 20 m, ungeschirmt
SDLC		
	Leitungslänge (Meter)	Max. 20 m, ungeschirmt
IO-Link		
	Anzahl Ports	4
	Anzahl Ports, die gleichzeitig gesteuert werden können	4
	IO-Link-Protokoll 1.0	Ja
	IO-Link-Protokoll 1.1	Ja
Betriebszustand		
	IO-Link	Ja
	DI	Ja
	DO	Ja, max. 100 mA
Anschluss von IO-Link-Geräten		
	Porttyp A	Ja

Technische Daten			Signalmodul SM 1278 4xIO-Link-Master
		Übertragungsgeschwindigkeit	4,8 kBd (COM1)
			38,4 kBd (COM2)
			230,4 kBd (COM3)
	Min. Zykluszeit		2 ms, dynamisch, abhängig von der Länge der Anwender-daten
	Größe der Prozessdaten, Eingang je Port		max. 32 Byte
	Größe der Prozessdaten, Eingang je Modul		32 Byte
	Größe der Prozessdaten, Ausgang je Port		max. 32 Bytes
	Größe der Prozessdaten, Ausgang je Modul		32 Bytes
	Speichergröße für Geräteparameter		2 KB
	Max. Kabellänge ungeschirmt (Meter)		20 m
Alarme/Diagnose/Statusinformationen			
	Statusanzeige		Ja
Alarme			
	Diagnosealarm		Ja, die Portdiagnose ist nur im IO-Link-Modus verfügbar
Diagnosealarme			
	Diagnose		
		Überwachung der Versorgungsspannung	Ja
		Kurzschluss	Ja
Diagnoseanzeige-LED			
	Überwachung der Versorgungsspannung		Ja, blinkende rote DIAG-LED
	Kanalstatusanzeige		Ja, je Kanal eine grüne LED für den Kanalstatus Qn (SIO-Modus) und PORT-Status Cn (IO-Link-Modus)
	Für Kanaldiagnose		Ja, rote Fn-LED
	Für Moduldiagnose		Ja, grüne/rote DIAG-LED
Potentialtrennung			
	Potentialtrennung Kanäle		
		Zwischen den Kanälen	Nein
		Zwischen den Kanälen und dem Rückwandbus	Ja
Zulässige Potentialdifferenz			
	Zwischen den verschiedenen Stromkreisen		75 V DC / 60 V AC (Basisisolierung)
Isolierung			
	Isolierung geprüft mit		707 V DC (Typprüfung)
Umgebungsbedingungen			
	Betriebstemperatur		
		Min.	-20 °C
		Max.	60 °C
		Horizontaler Einbau, min.	-20 °C
		Horizontaler Einbau, max.	60 °C
		Vertikaler Einbau, min.	-20 °C
		Vertikaler Einbau, max.	50 °C

Übersicht über die Reaktionszeit

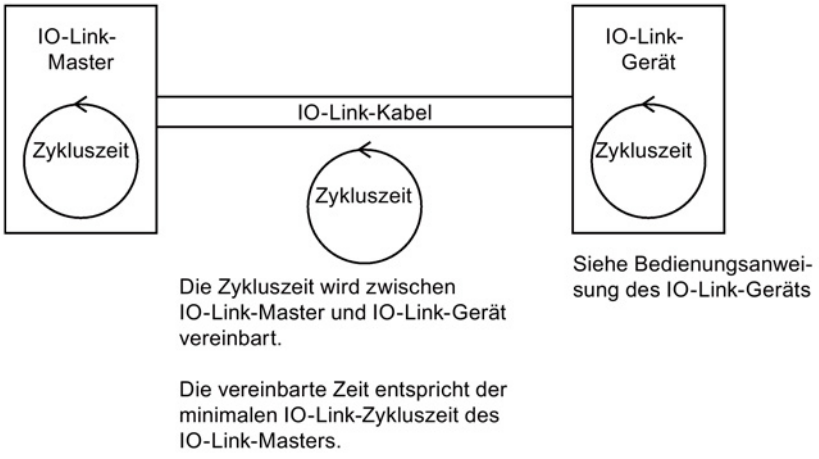


Tabelle A- 184 Schaltplan für das SM 1278 IO-Link-Master

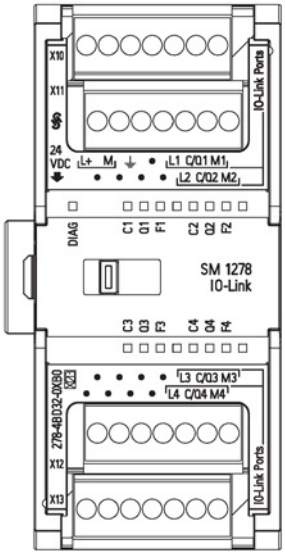
SM 1278 IO-Link-Master (6ES7 278-4BD32-0XB0)	
	

Tabelle A- 185 Anschlussbelegung für das SM 1278 IO-Link-Master (6ES7 278-4BD32-0XB0)

Pin	X10	X11	X12	X13
1	L+ / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
2	M / 24 V DC	Kein Anschluss	Kein Anschluss	Kein Anschluss
3	Funktionserde	Kein Anschluss	Kein Anschluss	Kein Anschluss
4	Kein Anschluss	Kein Anschluss	Kein Anschluss	Kein Anschluss
5	L_1	L_2	L_3	L_4

Pin	X10	X11	X12	X13
6	C/O_1	C/OL_2	C/O_3	C/OL_4
7	ML_1	ML_2	M_3	ML_4

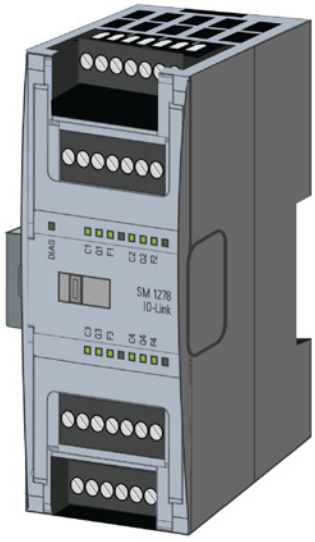
A.10.1.1 SM 1278 4xIO-Link-Master - Überblick

Der SM 1278 4xIO-Link-Master ist ein Modul mit 4 Ports, das als Signalmodul und als Kommunikationsmodul fungiert. Jeder Port kann im IO-Link-Modus als einzelner 24 VDC-Digitaleingang oder als 24 VDC-Digitalausgang betrieben werden.

Der IO-Link-Master programmiert die azyklische Kommunikation mit dem IO-Link-Gerät über den Funktionsbaustein (FB) für den IOL_CALL-Aufruf in Ihrem S7-1200-Steuerungsprogramm in STEP 7. Der Funktionsbaustein IOL_CALL zeigt den von Ihrem Programm verwendeten IO-Link-Master sowie die vom Master für den Datenaustausch verwendeten Ports an.

Auf der Siemens Industry-Website für Online-Support (http://support.automation.siemens.com) finden Sie ausführliche Informationen zum Arbeiten mit dem Funktionsbaustein IOL_CALL. Geben Sie im Suchfeld der Website "IO-Link" ein, um Informationen über IO-Link-Produkte und deren Einsatz aufzurufen.

Ansicht des Moduls



Eigenschaften

Technische Merkmale

- IO-Link-Master nach IO-Link-Spezifikation V1.1 (siehe Webseite des IO-Link-Konsortiums (http://io-link.com/en/index.php) mit weiteren Informationen)
- Serielles Kommunikationsmodul mit vier Schnittstellen (Kanälen)
- Datenübertragungsrate COM1 (4.8 kbaud), COM2 (38.4 kbaud), COM3 (230.4 kbaud)
- SIO-Modus (Standard IO-Modus)
- Anschluss von bis zu vier IO-Link-Geräten (3-Leiter-Anschluss) oder vier Standard-Aktoren oder Standard-Gebern
- Programmierbare Diagnosefunktion für jede Schnittstelle

Unterstützte Funktionen

- Identifikationsdaten für I&M (Instalnlation und Instandhaltung)
- Firmware-Update
- IO-Link-Parametrierung mit dem Schnittstellenkonfigurationstool S7-PCT, STEP 7 Professional und einer S7-1200 CPU V4.0 oder höher

IO-Link ist eine Punkt-zu-Punkt-Verbindung zwischen einem Master und einem Gerät. Als Geräte können sowohl konventionelle als auch intelligente Sensoren/Aktoren über ungeschirmte Standardkabel in der bewährten 3-Leiter-Technik am IO-Link angeschlossen werden. IO-Link ist abwärts kompatibel mit konventionellen digitalen Sensoren und Aktoren. Stromkreis und Datenkanal sind in bewährter 24 VDC-Technik ausgeführt.

Weitere Informationen über die SIMATIC IO-Link-Technologie finden Sie im "IO-Link Systemhandbuch" auf der Siemens Industry-Website für den Online-Support (http://support.automation.siemens.com).

Hinweis

IO-Link Parameterdaten

Bei einem Austausch des SM 4xIO-Link-Mastermoduls werden die Parameterdaten nicht automatisch zugewiesen.

⚠ VORSICHT

Ziehen und Stecken

Wird das SM 4xIO-Link-Mastermodul bei eingeschalteter Last gesteckt, können dadurch gefährliche Zustände in Ihrer Anlage entstehen.

Das S7-1200-Automatisierungssystem kann beschädigt werden.

Ziehen und stecken Sie das SM 4xIO-Link-Mastermodul nur, wenn die Last ausgeschaltet ist.

Auswirkungen des Rücksetzens auf die Werkseinstellungen

Mit der Funktion "Rücksetzen auf Werkseinstellungen" können Sie die mit S7-PCT am Lieferzustand vorgenommene Parametrierung wiederherstellen.

Nach dem "Rücksetzen auf Werkseinstellungen" wird das SM 1278 4xIO-Link-Modul wie folgt parametriert:

- Die Ports sind im DE-Modus
- Die Ports sind den relativen Adressen 0.0 bis 0.3 zugeordnet
- Der PortQualifier ist deaktiviert
- Die Wartungsdaten 1 bis 3 sind gelöscht

Hinweis

Beim Rücksetzen auf die Werkseinstellungen werden die Geräteparameter gelöscht und der Lieferzustand wiederhergestellt.

Wenn Sie ein SM 1278 4xIO-Link-Signalmodul entfernen, setzen Sie es vor dem Einlagern auf die Werkseinstellungen zurück.

Vorgehensweise

Gehen Sie zum "Rücksetzen auf die Werkseinstellungen" so vor, wie in der S7-PCT Online-Hilfe unter "Masterkonfiguration > Register 'Befehle' beschrieben.

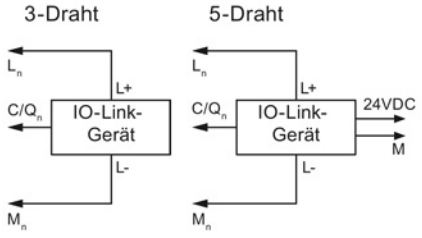
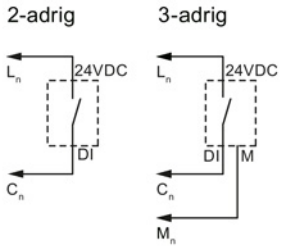
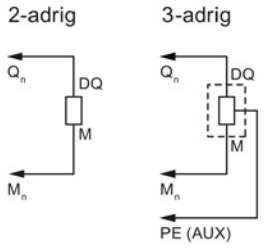
A.10.1.2 Anschluss

Näheres zur Anschlussbelegung finden Sie in der Tabelle Anschlussbelegung für SM 1278 I/O-Link-Master (6ES 278-4BD32-0XB0). (Seite 1268)

Die folgende Tabelle zeigt die Klemmenbelegung für den SM 1278 4xIO-Link-Master:

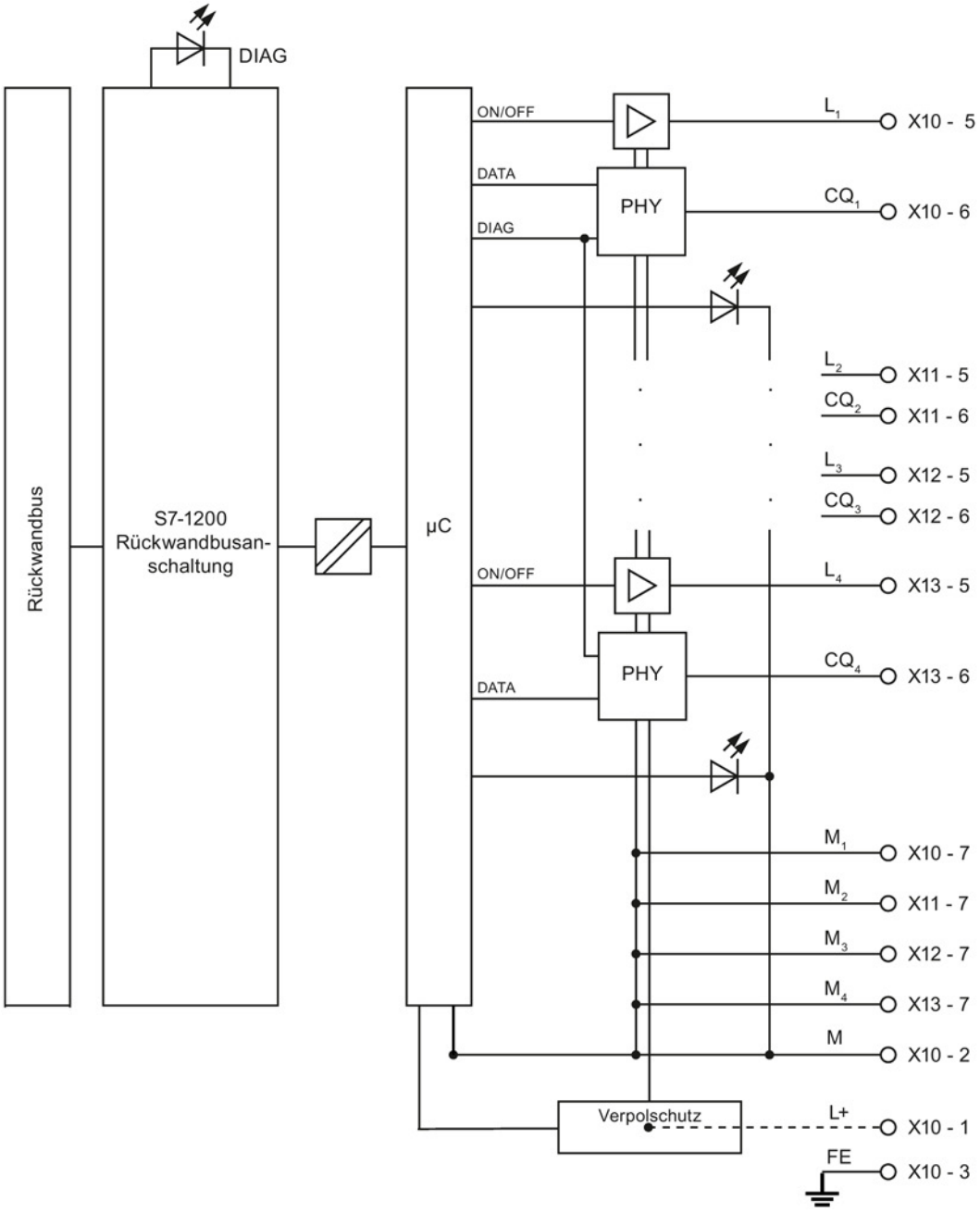
Pin	X10	X11	X12	X13	Hinweise	BaseUnits
7	M_1	M_2	M_3	M_4	• M_n: Masse zu Slave • C/O_n: SDLC, DI oder DO • L_n: 24 V DC zum Slave • M: Masse • L+: 24 V DC zum Master • RES: reserviert; darf nicht belegt werden	A1
6	C/O_1	C/O_2	C/O_3	C/O_4		
5	L_1	L_2	L_3	L_4		
4	RES	RES	RES	RES		
3	⏚ (Funktionserde)	RES	RES	RES		
2	M	RES	RES	RES		
1	L+	RES	RES	RES		

Die folgende Tabelle zeigt Anschlussbeispiele; dabei ist n = Schnittstellennummer.

IO-Link-Betrieb	Betriebsart DI	Betriebsart DO
		

Hinweis

Angeschlossene Sensoren müssen die über Anschluss L_n des Master-Moduls bereitgestellte Spannungsversorgung nutzen.


DIAG
Rückwandbus
S7-1200
Rückwandbusan-
schaltung
µC
ON/OFF
DATA
DIAG
PHY
L1
CQ1
X10 - 5
X10 - 6
L2
CQ2
L3
CQ3
X11 - 5
X11 - 6
X12 - 5
X12 - 6
ON/OFF
DATA
PHY
L4
CQ4
X13 - 5
X13 - 6
M1
M2
M3
M4
M
X10 - 7
X11 - 7
X12 - 7
X13 - 7
X10 - 2
Verpolschutz
L+
FE
X10 - 1
X10 - 3

A.10.1.3 Parameter/Adressbereich

SM 1278 4xIO-Link-Master konfigurieren

Zur Integration des Moduls benötigen Sie das Siemens-Engineeringtool TIA Portal V13 oder höher. Ferner benötigen Sie S7-PCT V3.2 oder höher für die IO-Link-Integration.

Zur Inbetriebnahme benötigen Sie ein Engineeringtool sowie S7-PCT V3.2 oder höher für die Parametrierung.

Die folgende Tabelle zeigt die Parameter für den SM 1278 4xIO-Link-Master:

Parameter	Wertebereich	Voreinstellung	Konfiguration in RUN	Effizienzbereich
Diagnoseschnittstelle 1	• Deaktivieren • Aktivieren	Deaktivieren	Ja	Schnittstelle (Kanal)
Diagnoseschnittstelle 2	• Deaktivieren • Aktivieren	Deaktivieren	Ja	Schnittstelle (Kanal)
Diagnoseschnittstelle 3	• Deaktivieren • Aktivieren	Deaktivieren	Ja	Schnittstelle (Kanal)
Diagnoseschnittstelle 4	• Deaktivieren • Aktivieren	Deaktivieren	Ja	Schnittstelle (Kanal)

Parameter Diagnose für Schnittstelle 1 bis 4 aktivieren

Mit diesem Parameter kann die Diagnose für bestimmte Schnittstellen der vier IO-Link-Schnittstellen aktiviert werden.

Die Schnittstellenzuordnungen sind wie folgt:

Schnittstelle 1 → Kanal 1

Schnittstelle 2 → Kanal 2

Schnittstelle 3 → Kanal 3

Schnittstelle 4 → Kanal 4

Die maximale Größe der Ein- und Ausgangsadressen des SM 4xIO-Link Master beträgt jeweils 32 Byte. Mit dem S7-PCT-Schnittstellenkonfigurationstool können Sie Adressbereiche zuweisen.

Parameter-Datensatz

Parametrieren im Anwenderprogramm

Sie können das Gerät während der Laufzeit konfigurieren.

Parameter während der Laufzeit ändern

Die Modulparameter sind in Datensatz 128 enthalten. Mit der Anweisung WRREC können Sie die änderbaren Parameter an das Modul übertragen..

Nach dem Rücksetzen (Aus- und Einschalten) der CPU überschreibt die CPU die Parameter, die während der Parametrierung mit Anweisung WRREC in das Modul übertragen wurden.

Anweisung für die Parametrierung

Für die Parametrierung des E/A-Moduls im Anwenderprogramm steht die folgende Anweisung zur Verfügung:

Anweisung	Anwendung
SFB 53 WRREC	Übertragung der änderbaren Parameter in das Modul.

Fehlermeldung

Bei einem Fehler wird der folgende Rückgabewert gemeldet:

Fehlercode	Bedeutung
$80B1_H$	Fehlerhafte Datenlänge
$80E0_H$	Fehlerhafte Header-Information
$80E1_H$	Parameterfehler

Datensatz-Struktur

Die folgende Tabelle zeigt die IO-Link-Parameter:

Offset	Bezeichnung	Typ	Voreinstel-lung	Beschreibung
0	Version	1 Byte	0x02	Zeigt die Struktur von Datensatz 0x02 für den IO-Link-Master nach IO-Link V1.1
1	Parameterlänge	1 Byte	0x02	Parameterlänge (2 Byte + 2 Header)
IO-Link-Startparameter				
2	Schnittstellendi-agnose (Schnitt-stelle1 1 bis n)	1 Byte	0x00	Aktivieren der Diagnose für Schnittstelle 1 bis n
3	IOL-Eigenschaften	1 Byte	0x00	Moduleigenschaften

Die folgende Tabelle zeigt den Versionsdatensatz:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserviert		Hauptversion (00)		Nebenversion (0010)			

Die folgende Tabelle zeigt den Datensatz für die Schnittstellendiagnose:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserviert				EN_Port4	EN_Port3	EN_Port2	EN_Port1

EN_Portx:

0 = Diagnose deaktiviert

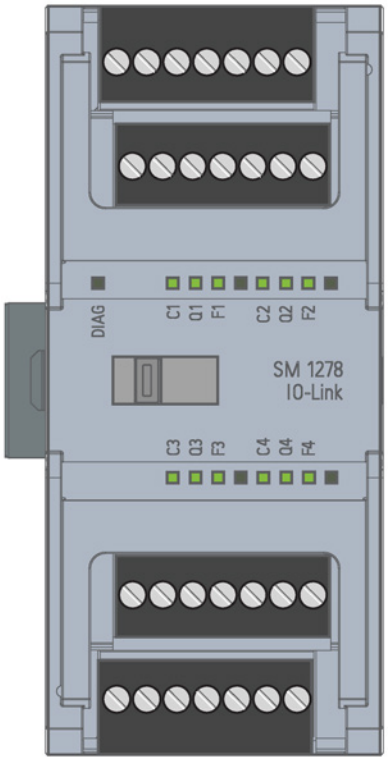
1 = Diagnose aktiviert

Die folgende Tabelle zeigt den Datensatz mit den IOL-Eigenschaften:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reserviert							

A.10.1.4 Alarm-, Fehler- und Systemmeldungen

LED-Anzeige



Bedeutung der LED-Anzeigen

Die folgende Tabelle erläutert die Bedeutung der Zustands- und Fehleranzeigen. Abhilfemaßnahmen bei Diagnosealarmen finden Sie im Kapitel "Diagnosemeldungen".

LED DIAG

DIAG	Bedeutung
Aus	Rückwandbus-Versorgung der S7-1200 nicht OK
Blinkt	Modul ist nicht konfiguriert
Ein	Modul parametriert und keine Moduldiagnose
Blinkt	Modul parametriert und Moduldiagnose ODER L+ nicht angeschlossen

LED Port-Status

Gültig für einen IO-Link-Port im IO-Link-Portmodus.

COM/1 ... COM/4	Bedeutung
Aus	Port deaktiviert
Blinkt	Port aktiviert, Gerät nicht angeschlossen oder Port nicht an das konfigurierte Gerät angeschlossen
Ein	Port aktiviert, Gerät angeschlossen

LED Kanalstatus

Gültig für einen IO-Link-Port in DI/O-Modus.

DI/O1 ... DI/O4	Bedeutung
Aus	Prozesssignal = 0
Ein	Prozesssignal = 1

LED Portfehler

F1...F4	Bedeutung
Aus	Kein Fehler
Ein	Fehler

Modulfehler werden nur im IO-Link-Modus als Diagnose (Modulstatus) angezeigt.

Diagnose-alarm	Fehler-code (dezimal)	STATUS (W#16#...)	Bedeutung (IO-Link-Fehlercode)	IO-Link-Master	IO-Link-Gerät
Kurzschluss	1	1804	Kurzschluss an den Prozesskabeln am IO-Link-Gerät	X	
		7710	Kurzschluss am IO-Gerät		X
Unterspan-nung	2	5111 5112	Versorgungsspannung zu gering		X
Überspan-nung	3	5110	Versorgungsspannung zu hoch		X
Überhitzung	5	1805	Temperatur am Master überschritten	X	
		4000 4210	Temperatur am Gerät überschritten		X
Drahtbruch	6	1800	• Kein IO-Link-Gerät angeschlos-sen • Drahtbruch in der Signalleitung zum IO-Link-Gerät • IO-Link-Gerät kann wegen eines anderen Fehlers nicht kommuni-zieren	X	
Überlauf	7	8C10 8C20	Prozessvariablenbereich überschrit-ten		X
		8C20	Messbereich überschritten		
Unterlauf	8	8C30	Prozessvariablenbereich zu gering		X
Fehler	9	---	Alle hier nicht aufgeführten IO-Link-Fehlercodes werden diesem PROFIBUS DP-Fehler zugeordnet.		X
Parametrie-rungsfehler	16	1882 1883	IO-Link-Master konnte nicht konfigu-riert werden	X	
		1802	Falsches Gerät		
		1886	Speicherfehler		
		6320 6321 6350	Gerät wurde nicht richtig konfiguriert		X
Versor-gungsspan-	17	1806	L+-Versorgungsspannung für das Gerät fehlt	X	

Diagnose-alarm	Fehler-code (dezimal)	STATUS (W#16#...)	Bedeutung (IO-Link-Fehlercode)	IO-Link-Master	IO-Link-Gerät
nung fehlt		1807	L+-Versorgungsspannung für das Gerät zu gering (< 20 V)		
Defekte Sicherung	18	5101	Sicherung am Gerät ist defekt		X
Sicherheits-abschaltung	25	1880	Schwerer Fehler (Master muss ersetzt werden)	X	
Externe Störung	26	1809 180A 180B 180C 180D	Fehler im Datenspeicher	X	
		1808	Am IO-Link-Gerät stehen mehr als 6 Fehler gleichzeitig an		

A.11 Digitale Signalboards (SBs)

A.11.1 Technische Daten des SB 1221 200 kHz Digitaleingabe

Tabelle A- 186 Allgemeine technische Daten

Technische Daten	SB 1221 DI 4 x 24 V DC, 200 kHz	SB 1221 DI 4 x 5 V DC, 200 kHz
Artikelnummer	6ES7 221-3BD30-0XB0	6ES7 221-3AD30-0XB0
Abmessungen (B x H x T) (mm)	38 x 62 x 21	
Gewicht	35 Gramm	
Leistungsverlust	1,5 W	1,0 W
Stromaufnahme (SM-Bus)	40 mA	
Stromaufnahme (24 V DC)	7 mA / Eingang + 20 mA	15 mA / Eingang + 15 mA

Tabelle A- 187 Digitale Eingänge

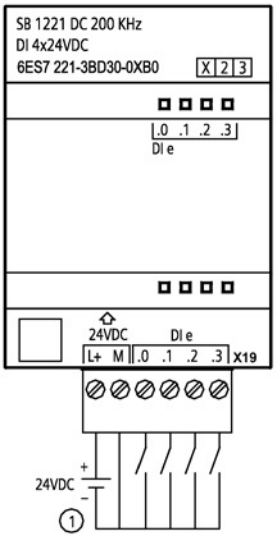
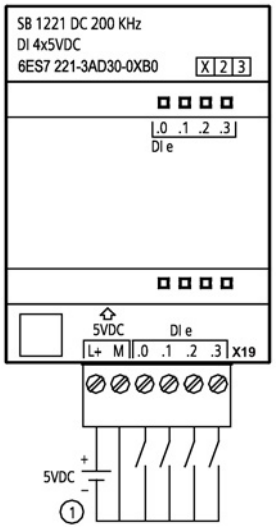
Technische Daten		SB 1221 DI 4 x 24 V DC, 200 kHz	SB 1221 DI 4 x 5 V DC, 200 kHz
Anzahl der Eingänge		4	
Typ		Stromliefernd	
Nennspannung		24 V DC bei 7 mA, Nennwert	5 V DC bei 15 mA, Nennwert
Zulässige Dauerspannung		28,8 V DC	6 V DC
Stoßspannung		35 V DC für 0,5 s	6 V
Signal logisch 1 (min.)		L+ minus 10 V DC bei 2,9 mA	L+ minus 2,0 V DC bei 5,1 mA
Signal logisch 0 (max.)		L+ minus 5 V DC bei 1,4 mA	L+ minus 1,0 V DC bei 2,2 mA
HSC Eingangstaktfrequenzen (max.)		Einphasenzähler: 200 kHz A/B-Zähler: 160 kHz	
Elektrische Trennung (Feld zu Logik)		500 V AC für 1 Minute	
Potentialgetrennte Gruppen		1	
Filterzeiten	µs-Einstellungen:	0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0,12.8, 20.0	
	ms-Einstellungen	0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0	
Anzahl gleichzeitig eingeschalteter Eingänge		• 2 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 4 bei 55 °C horizontal oder 45 °C vertikal	4
Leitungslänge (Meter)		50 m, geschirmtes, verdrilltes Leiterpaar	

Hinweis

Beim Umschalten von Frequenzen über 20 kHz ist es wichtig, dass die Digitaleingänge Rechtecksignale empfangen. Sie haben folgende Möglichkeiten, die Qualität des Eingangssignals zu verbessern:

- Verkürzen Sie die Leitung auf die Mindestlänge.
- Verwenden Sie statt eines nur stromziehenden Treibers einen stromziehenden/stromliefernden (P-M-schaltenden) Treiber.
- Tauschen Sie das Leitungskabel gegen ein höherwertiges Kabel aus.
- Verringern Sie die Spannung der Schaltkreise/Bauteile von 24 V auf 5 V.
- Fügen Sie am Eingang eine externe Last hinzu.

Tabelle A- 188 Schaltpläne der Digitaleingangs-SBs (200 kHz)

SB 1221 DI 4 x 24 VDC, 200 kHz (6ES7 221-3BD30-0XB0)	SB 1221 DI 4 x 5 VDC, 200 kHz (6ES7 221-3AD30-0XB0)
	

① Unterstützt nur stromliefernde Eingänge

Tabelle A- 189 Anschlussbelegung für das SB 1221 DI 4 x 24 VDC, 200 kHz (6ES7 221-3BD30-0XB0)

Pin	X19
1	L+ / 24 V DC
2	M / 24 V DC
3	DI e.0
4	DI e.1
5	DI e.2
6	DI e.3

Tabelle A- 190 Anschlussbelegung für das SB 1221 DI 4 x 5 VDC, 200 kHz (6ES7 221-3AD30-0XB0)

Pin	X19
1	L+ / 5 V DC
2	M / 5 V DC
3	DI e.0
4	DI e.1
5	DI e.2
6	DI e.3

A.11.2 Technische Daten des SB 1222 200 kHz Digitalausgabe

Tabelle A- 191 Allgemeine technische Daten

Technische Daten	SB 1222 DO 4 x 24 V DC, 200 kHz	SB 1222 DO 4 x 5 V DC, 200 kHz
Artikelnummer	6ES7 222-1BD30-0XB0	6ES7 222-1AD30-0XB0
Abmessungen (B x H x T) (mm)	38 x 62 x 21	
Gewicht	35 Gramm	
Leistungsverlust	0,5 W	
Stromaufnahme (SM-Bus)	35 mA	
Stromaufnahme (24 V DC)	15 mA	

Tabelle A- 192 Digitale Ausgänge

Technische Daten	SB 1222 DO 4 x 24 V DC, 200 kHz	SB 1222 DO 4 x 5 V DC, 200 kHz
Ausgänge	4	
Ausgangstyp	MOSFET, elektronisch (stromziehend/stromliefernd)[1]	
Spannungsbereich	20,4 bis 28,8 V DC	4,25 bis 6,0 V DC
Signal logisch 1 bei max. Strom	L+ minus 1,5 V	L+ minus 0,7 V
Signal logisch 0 bei max. Strom	max. 1,0 V DC	max. 0,2 V DC
Strom (max.)	0,1 A	
Lampenlast	--	
Kontaktwiderstand bei EIN	max. 11 Ω	max. 7 Ω
Widerstand bei AUS	max. 6 Ω	max. 0,2 Ω
Kriechstrom pro Ausgang	--	
Frequenz Impulsgenerator	max. 200 kHz, min. 2 Hz	
Einschaltstrom	0,11 A	
Überlastschutz	Nein	
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute	

Technische Daten	SB 1222 DO 4 x 24 V DC, 200 kHz	SB 1222 DO 4 x 5 V DC, 200 kHz
Potentialgetrennte Gruppen	1	
Ströme je Leiter	0,4 A	
Induktive Klemmspannung	Nein	
Schaltverzögerung	1,5 µs + 300 ns steigend 1,5 µs + 300 ns fallend	200 ns + 300 ns steigend 200 ns + 300 ns fallend
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)	
Anzahl gleichzeitig eingeschalteter Ausgänge	• 2 (keine benachbarten Punkte) bei 60 °C horizontal oder 50 °C vertikal • 4 bei 55 °C horizontal oder 45 °C vertikal	4
Leitungslänge (Meter)	50 m, geschirmtes, verdrilltes Leiterpaar	

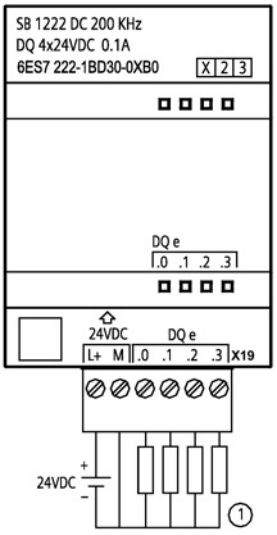
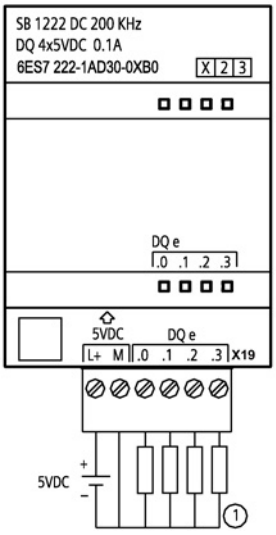
[1] Weil sowohl stromliefernde als auch stromziehende Konfigurationen von denselben Schaltungen unterstützt werden, ist der aktive Zustand einer stromliefernden Last das Gegenteil von dem einer stromziehenden Last. Ein stromliefernder Ausgang zeigt positive Logik (Q-Bit und LED sind EIN, wenn die Last Stromfluss hat), während ein stromziehender Ausgang negative Logik aufweist (Q-Bit und LED sind AUS, wenn die Last Stromfluss hat). Wenn das Modul ohne Anwenderprogramm gesteckt ist, beträgt der Standardwert für dieses Modul 0 V, was bedeutet, dass eine stromziehende Last eingeschaltet wird.

Hinweis

Beim Umschalten von Frequenzen über 20 kHz ist es wichtig, dass die Digitaleingänge Rechtecksignale empfangen. Sie haben folgende Möglichkeiten, die Qualität des Eingangssignals zu verbessern:

- Verkürzen Sie die Leitung auf die Mindestlänge.
- Verwenden Sie statt eines nur stromziehenden Treibers einen stromziehenden/stromliefernden (P-M-schaltenden) Treiber.
- Tauschen Sie das Leitungskabel gegen ein höherwertiges Kabel aus.
- Verringern Sie die Spannung der Schaltkreise/Bauteile von 24 V auf 5 V.
- Fügen Sie am Eingang eine externe Last hinzu.

Tabelle A- 193 Schaltpläne der Digitalausgangs-SBs (200 kHz)

SB 1222 DO 4 x 24 VDC, 200 kHz (6ES7 222-1BD30-0XB0)	SB 1222 DO 4 x 5 VDC, 200 kHz (6ES7 222-1AD30-0XB0)
	

① Bei stromliefernden Ausgängen "Load" an "-" anschließen (s. Abbildung). Bei stromziehenden Ausgängen "Load" an "+" anschließen. Weil sowohl stromliefernde als auch stromziehende Konfigurationen von denselben Schaltungen unterstützt werden, ist der aktive Zustand einer stromliefernden Last das Gegenteil von dem einer stromziehenden Last. Ein stromliefernder Ausgang zeigt positive Logik (Q-Bit und LED sind EIN, wenn die Last Stromfluss hat), während ein stromziehender Ausgang negative Logik aufweist (Q-Bit und LED sind AUS, wenn die Last Stromfluss hat). Wenn das Modul ohne Anwenderprogramm gesteckt ist, beträgt der Standardwert für dieses Modul 0 V, was bedeutet, dass eine stromziehende Last eingeschaltet wird.

Tabelle A- 194 Anschlussbelegung für das SB 1222 DO 4 x 24 VDC, 200 kHz (6ES7 222-1BD30-0XB0)

Pin	X19
1	L+ / 24 V DC
2	M / 24 V DC
3	DO e.0
4	DO e.1
5	DO e.2
6	DO e.3

Tabelle A- 195 Anschlussbelegung für das SB 1222 DO 4 x 5 VDC, 200 kHz (6ES7 222-1AD30-0XB0)

Pin	X19
1	L+ / 5 V DC
2	M / 5 V DC
3	DO e.0
4	DO e.1
5	DO e.2
6	DO e.3

A.11.3 Technische Daten des SB 1223 200 kHz Digitalein-/Digitalausgabe

Tabelle A- 196 Allgemeine technische Daten

Technische Daten	SB 1223 DI 2 x 24 VDC / DO 2 x 24 VDC, 200 kHz	SB 1223 DI 2 x 5 VDC / DO 2 x 5 VDC, 200 kHz
Artikelnummer	6ES7 223-3BD30-0XB0	6ES7 223-3AD30-0XB0
Abmessungen (B x H x T) (mm)	38 x 62 x 21	
Gewicht	35 Gramm	
Leistungsverlust	1,0 W	0,5 W
Stromaufnahme (SM-Bus)	35 mA	
Stromaufnahme (24 V DC)	7 mA / Eingang + 30 mA	15 mA / Eingang + 15 mA

Tabelle A- 197 Digitale Eingänge

Technische Daten		SB 1223 DI 2 x 24 VDC / DO 2 x 24 VDC, 200 kHz	SB 1223 DI 2 x 5 VDC / DO 2 x 5 VDC, 200 kHz
Anzahl der Eingänge		2	
Typ		Stromliefernd	
Nennspannung		24 V DC bei 7 mA, Nennwert	5 V DC bei 15 mA, Nennwert
Zulässige Dauerspannung		28,8 V DC	6 V DC
Stoßspannung		35 V DC für 0,5 s	6 V
Signal logisch 1 (min.)		L+ minus 10 V DC bei 2,9 mA	L+ minus 2,0 V DC bei 5,1 mA
Signal logisch 0 (max.)		L+ minus 5 V DC bei 1,4 mA	L+ minus 1,0 V DC bei 2,2 mA
HSC Eingangstaktfrequenzen (max.)		Einphasenzähler: 200 kHz A/B-Zähler: 160 kHz	
Elektrische Trennung (Feld zu Logik)		500 V AC für 1 Minute	
Potentialgetrennte Gruppen		1 (keine Trennung gegen Ausgang)	
Filterzeiten	µs-Einstellungen:	0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0	
	ms-Einstellungen	0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0	

Technische Daten	SB 1223 DI 2 x 24 VDC / DO 2 x 24 VDC, 200 kHz	SB 1223 DI 2 x 5 VDC / DO 2 x 5 VDC, 200 kHz
Anzahl gleichzeitig eingeschalteter Eingänge	2	
Leitungslänge (Meter)	50 m, geschirmtes, verdrilltes Leiterpaar	

Tabelle A- 198 Digitale Ausgänge

Technische Daten	SB 1223 DI 2 x 24 VDC / DO 2 x 24 VDC, 200 kHz	SB 1223 DI 2 x 5 VDC / DO 2 x 5 VDC, 200 kHz
Ausgänge	2	
Ausgangstyp	MOSFET, elektronisch (stromziehend/stromliefernd)[1]	
Spannungsbereich	20,4 bis 28,8 V DC	4,25 bis 6,0 V DC
Nennwert	24 V DC	5 V DC
Signal logisch 1 bei max. Strom	L+ minus 1,5 V	L+ minus 0,7 V
Signal logisch 0 bei max. Strom	max. 1,0 V DC	max. 0,2 V DC
Strom (max.)	0,1 A	
Lampenlast	--	
Kontaktwiderstand bei EIN	max. 11 Ω	max. 7 Ω
Widerstand bei AUS	max. 6 Ω	max. 0,2 Ω
Kriechstrom pro Ausgang	--	
Frequenz Impulsgenerator	max. 200 kHz, min. 2 Hz	
Einschaltstrom	0,11 A	
Überlastschutz	Nein	
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute	
Potentialgetrennte Gruppen	1 (keine Trennung gegen Eingang)	
Ströme je Leiter	0,2 A	
Induktive Klemmspannung	Nein	
Schaltverzögerung	1,5 µs + 300 ns steigend 1,5 µs + 300 ns fallend	200 ns + 300 ns steigend 200 ns + 300 ns fallend
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)	
Anzahl gleichzeitig eingeschalteter Ausgänge	2	
Leitungslänge (Meter)	50 m, geschirmtes, verdrilltes Leiterpaar	

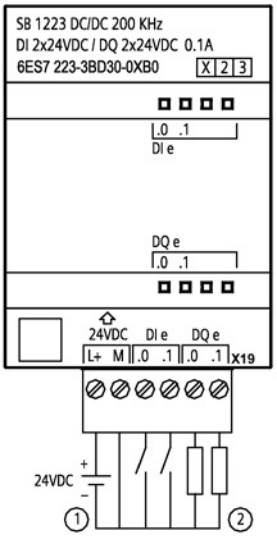
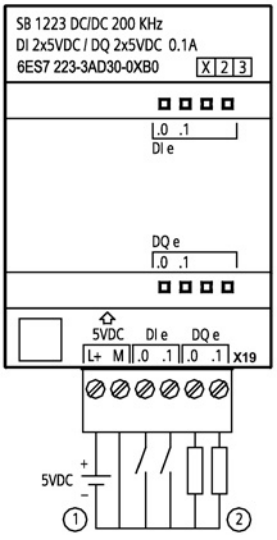
[1] Weil sowohl stromliefernde als auch stromziehende Konfigurationen von denselben Schaltungen unterstützt werden, ist der aktive Zustand einer stromliefernden Last das Gegenteil von dem einer stromziehenden Last. Ein stromliefernder Ausgang zeigt positive Logik (Q-Bit und LED sind EIN, wenn die Last Stromfluss hat), während ein stromziehender Ausgang negative Logik aufweist (Q-Bit und LED sind AUS, wenn die Last Stromfluss hat). Wenn das Modul ohne Anwenderprogramm gesteckt ist, beträgt der Standardwert für dieses Modul 0 V, was bedeutet, dass eine stromziehende Last eingeschaltet wird.

Hinweis

Beim Umschalten von Frequenzen über 20 kHz ist es wichtig, dass die Digitaleingänge Rechtecksignale empfangen. Sie haben folgende Möglichkeiten, die Qualität des Eingangssignals zu verbessern:

- Verkürzen Sie die Leitung auf die Mindestlänge.
- Verwenden Sie statt eines nur stromziehenden Treibers einen stromziehenden/stromliefernden (P-M-schaltenden) Treiber.
- Tauschen Sie das Leitungskabel gegen ein höherwertiges Kabel aus.
- Verringern Sie die Spannung der Schaltkreise/Bauteile von 24 V auf 5 V.
- Fügen Sie am Eingang eine externe Last hinzu.

Tabelle A- 199 Schaltpläne der digitalen Eingangs-/Ausgangs-SBs (200 kHz)

SB 1223 DI 2 x 24 VDC/DO 2 x 24 VDC, 200 kHz (6ES7 223-3BD30-0XB0)	SB 1223 DI 2 x 5 VDC / DO 2 x 5 VDC, 200 kHz (6ES7 223-3AD30-0XB0)
	

① Unterstützt nur stromliefernde Eingänge

② Bei stromliefernden Ausgängen "Load" an "-" anschließen (s. Abbildung). Bei stromziehenden Ausgängen "Load" an "+" anschließen. [1] Weil sowohl stromliefernde als auch stromziehende Konfigurationen von denselben Schaltungen unterstützt werden, ist der aktive Zustand einer stromliefernden Last das Gegenteil von dem einer stromziehenden Last. Ein stromliefernder Ausgang zeigt positive Logik (Q-Bit und LED sind EIN, wenn die Last Stromfluss hat), während ein stromziehender Ausgang negative Logik aufweist (Q-Bit und LED sind AUS, wenn die Last Stromfluss hat). Wenn das Modul ohne Anwenderprogramm gesteckt ist, beträgt der Standardwert für dieses Modul 0 V, was bedeutet, dass eine stromziehende Last eingeschaltet wird.

Tabelle A- 200 Anschlussbelegung für das SB 1223 DI 2 x 24 VDC/DO 2 x 24 VDC, 200 kHz (6ES7 223-3BD30-0XB0)

Pin	X19
1	L+ / 24 V DC
2	M / 24 V DC
3	DI e.0
4	DI e.1
5	DO e.0
6	DO e.1

Tabelle A- 201 Anschlussbelegung für das SB 1223 DI 2 x 5 VDC/DO 2 x 5 VDC, 200 kHz (6ES7 223-3AD30-0XB0)

Pin	X19
1	L+ / 5 V DC
2	M / 5 V DC
3	DI e.0
4	DI e.1
5	DO e.0
6	DO e.1

A.11.4 Technische Daten SB 1223 2 x 24 V DC Eingang / 2 x 24 V DC Ausgang

Tabelle A- 202 Allgemeine technische Daten

Technische Daten	SB 1223 DI 2 x 24 VDC, DO 2 x 24 V DC
Artikelnummer	6ES7 223-0BD30-0XB0
Abmessungen B x H x T (mm)	38 x 62 x 21
Gewicht	40 Gramm
Leistungsverlust	1,0 W
Stromaufnahme (SM-Bus)	50 mA
Stromaufnahme (24 V DC)	4 mA / Eingang

Tabelle A- 203 Digitale Eingänge

Technische Daten	SB 1223 DI 2 x 24 VDC, DO 2 x 24 V DC
Anzahl der Eingänge	2
Typ	IEC Typ 1 stromziehend
Nennspannung	24 V DC bei 4 mA, Nennwert
Zulässige Dauerspannung	max. 30 V DC
Stoßspannung	35 V DC für 0,5 s

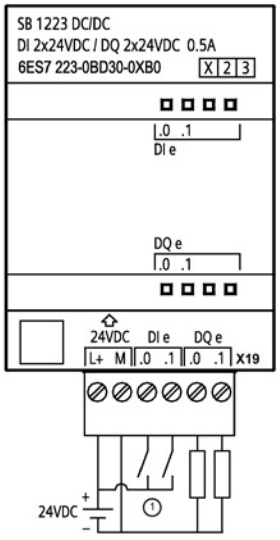
Technische Daten		SB 1223 DI 2 x 24 VDC, DO 2 x 24 V DC
Signal logisch 1 (min.)		15 V DC bei 2,5 mA
Signal logisch 0 (max.)		5 V DC bei 1 mA
HSC Eingangstaktfrequenzen (max.)		Einphasenzähler: 30 kHz (15 bis 26 V DC) A/B-Zähler: 20 kHz (15 bis 26 V DC)
Elektrische Trennung (Feld zu Logik)		500 V AC für 1 Minute
Potentialgetrennte Gruppen		1
Filterzeiten	µs-Einstellungen:	0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4,10.0, 12.8, 20.0
	ms-Einstellungen	0.05, 0.1, 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 10.0, 12.8, 20.0
Anzahl gleichzeitig eingeschalteter Eingänge		2
Leitungslänge (Meter)		500 geschirmt, 300 ungeschirmt

Tabelle A- 204 Digitale Ausgänge

Technische Daten	SB 1223 DI 2 x 24 VDC, DO 2 x 24 V DC
Ausgänge	2
Ausgangstyp	MOSFET, elektronisch (stromliefernd)
Spannungsbereich	20,4 bis 28,8 V DC
Signal logisch 1 bei max. Strom	min. 20 V DC
Signal logisch 0 bei 10 kΩ Last	max. 0,1 V DC
Strom (max.)	0,5 A
Lampenlast	5 W
Kontaktwiderstand bei EIN	max. 0,6 Ω
Kriechstrom pro Ausgang	max. 10 µA
Frequenz Impulsfolgeausgang (PTO)	max. 20 kHz, min. 2 Hz [1]
Einschaltstrom	5 A für max. 100 ms
Überlastschutz	Nein
Elektrische Trennung (Feld zu Logik)	500 V AC für 1 Minute
Potentialgetrennte Gruppen	1
Ströme je Leiter	1 A
Induktive Klemmspannung	L+ minus 48 V, 1 W Verlustleistung
Schaltverzögerung	max. 2 µs von Aus nach Ein max. 10 µs von Ein nach Aus
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)
Anzahl gleichzeitig eingeschalteter Ausgänge	2
Leitungslänge (Meter)	500 m geschirmt, 150 m ungeschirmt

[1] Je nach Impulsempfänger und Kabel kann ein zusätzlicher Lastwiderstand (bei mindestens 10% des Nennstroms) die Qualität der Impulssignale und die Störfestigkeit verbessern.

Tabelle A- 205 Schaltplan des digitalen Eingangs-/Ausgangs-SBs

SB 1223 DI 2 x 24 V DC, DO 2 x 24 V DC (6ES7 223-0BD30-0XB0)	
	

① Unterstützt nur stromziehende Eingänge

Tabelle A- 206 Anschlussbelegung für das SB 1223 DI 2 x 24 VDC/DO 2 x 24 VDC (6ES7 223-0BD30-0XB0)

Pin	X19
1	L+ / 24 V DC
2	M / 24 V DC
3	DI e.0
4	DI e.1
5	DO e.0
6	DO e.1

A.12 Analoge Signalboards (SBs)

A.12.1 Technische Daten des SB 1231 1 Analogeingang

Hinweis

Um diesen SB nutzen zu können, benötigen Sie eine CPU mit Firmware ab V2.0.

Tabelle A- 207 Allgemeine technische Daten

Technische Daten	SB 1231 AI 1 x 12 Bit
Artikelnummer	6ES7 231-4HA30-0XB0
Abmessungen B x H x T (mm)	38 x 62 x 21
Gewicht	35 Gramm
Leistungsverlust	0,4 W
Stromaufnahme (SM-Bus)	55 mA
Stromaufnahme (24 V DC)	Keine

Tabelle A- 208 Analoge Eingänge

Technische Daten	SB 1231 AI 1x12 Bit
Anzahl der Eingänge	1
Typ	Spannung oder Strom (differential)
Bereich	±10 V, ±5 V, ±2,5 oder 0 bis 20 mA
Auflösung	11 Bit + Vorzeichenbit
Vollausschlag (Datenwort)	-27648 bis 27648
Überbereich/Unterbereich (Datenwort)	Spannung: 32511 bis 27649 / -27649 bis -32512 Strom: 32511 bis 27649 / 0 bis -4864 (siehe Darstellung Analogeingang für Spannung und Darstellung Analogeingang für Strom (Seite 1297)).
Überlauf/Unterlauf (Datenwort)	Spannung: 32767 bis 32512 / -32513 bis -32768 Strom: 32767 bis 32512 / -4865 bis -32768 (siehe Darstellung Analogeingang für Spannung und Darstellung Analogeingang für Strom (Seite 1297)).
Maximale Stehspannung/-strom	±35 V / ±40 mA
Glättung	Keine, schwach, mittel oder stark (siehe Antwortzeiten Analogeingang für Schrittantwortzeit (Seite 1297)).
Rauschunterdrückung	400, 60, 50 oder 10 Hz (siehe Ansprechzeiten Analogeingang für Abtastraten (Seite 1297)).
Genauigkeit (25 °C / -20 bis 60 °C)	±0,3% / ±0,6% des Vollausschlags

Technische Daten	SB 1231 AI 1x12 Bit
Eingangsimpedanz Differential Gleichtakt	 Spannung: 220 kΩ; Strom: 250 Ω Spannung: 55 kΩ; Strom: 55 kΩ
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)
Messprinzip	Istwertumwandlung
Gleichtaktunterdrückung	40 dB, Nennwert bei 60 Hz
Betriebssignalbereich	Signal- plus Gleichtaktspannung muss kleiner als +35 V und größer als -35 V sein
Elektrische Trennung (Feld zu Logik)	Keine
Leitungslänge (Meter)	100 m, verdrillt und geschirmt

Tabelle A- 209 Diagnose

Technische Daten	SB 1231 AI 1 x 12 Bit
Überlauf/Unterlauf	Ja
24-V-DC-Niederspannung	Nein

Tabelle A- 210 Schaltplan des Analogeingangs-SBs

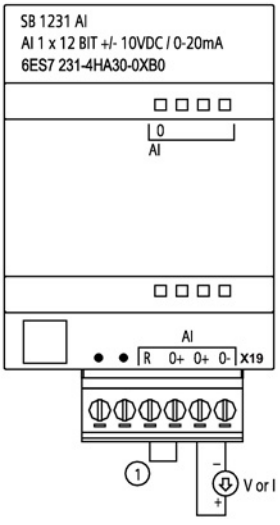
SB 1231 AI x 12 Bit (6ES7 231-4HA30-0XB0)	
	① "R" und "0+" für Strom anschließen. Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.

Tabelle A- 211 Anschlussbelegung für das SB 1231 AI x 12 Bit (6ES7 231-4HA30-0XB0)

Pin	X19 (vergoldet)
1	Kein Anschluss
2	Kein Anschluss
3	AI R
4	AI 0+
5	AI 0+
6	AI 0-

A.12.2 Technische Daten des SB 1232 1 Analogausgabe

Tabelle A- 212 Allgemeine technische Daten

Technische Daten	SB 1232 AO 1 x 12 Bit
Artikelnummer	6ES7 232-4HA30-0XB0
Abmessungen (B x H x T) (mm)	38 x 62 x 21
Gewicht	40 Gramm
Leistungsverlust	1,5 W
Stromaufnahme (SM-Bus)	15 mA
Stromaufnahme (24 V DC)	40 mA (ohne Last)

Tabelle A- 213 Analoge Ausgänge

Technische Daten	SB 1232 AO 1 x 12 Bit
Ausgänge	1
Typ	Spannung oder Strom
Bereich	±10 V oder 0 bis 20 mA
Auflösung	Spannung: 12 Bits Strom: 11 Bits
Vollausschlag (Datenwort) Siehe Abschnitt der Ausgangsbereiche für Spannung und Strom (Seite 1298).	Spannung: -27648 bis 27648 Strom: 0 bis 27648
Genauigkeit (25 °C / -20 bis 60 °C)	±0,5% / ±1% des Vollausschlags
Ausregelzeit (95 % des neuen Werts)	Spannung: 300 µs (R), 750 µs (1 uF) Strom: 600 µs (1 mH), 2 ms (10 mH)
Lastimpedanz	Spannung: ≥ 1000 Ω Strom: ≤ 600 Ω
Verhalten bei Wechsel von RUN nach STOP	Letzter Wert oder Ersatzwert (Voreinstellung 0)
Elektrische Trennung (Feld zu Logik)	Keine
Leitungslänge (Meter)	100 m, verdrillt und geschirmt

Tabelle A- 214 Diagnose

Technische Daten	SB 1232 AO 1 x 12 Bit
Überlauf/Unterlauf	Ja
Erdschluss (nur Spannungsmodus)	Ja
Drahtbruch (nur Strommodus)	Ja

Tabelle A- 215 Schaltplan für das SB 1232 AO 1 x 12 Bit

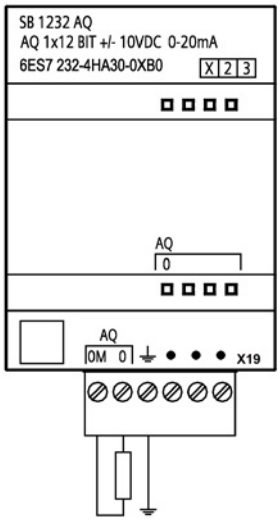
SB 1232 AO 1 x 12 Bit (6ES7 232-4HA30-0XB0)	
	
Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	

Tabelle A- 216 Anschlussbelegung für das SB 1232 AO 1 x 12 Bit (6ES7 232-4HA30-0XB0)

Pin	X19 (vergoldet)
1	AO 0M
2	AO 0
3	Funktionserde
4	Kein Anschluss
5	Kein Anschluss
6	Kein Anschluss

A.12.3 Messbereiche der analogen Eingänge und Ausgänge

A.12.3.1 Schrittantwort der analogen Eingänge

Tabelle A- 217 Schrittantwort (ms), 0 V bis 10 V gemessen bei 95 %

Auswahl der Glättung (Mittelwertbildung aus Abtastwerten)	Auswahl Integrationszeit			
	400 Hz (2,5 ms)	60 Hz (16,6 ms)	50 Hz (20 ms)	10 Hz (100 ms)
Keine (1 Zyklus): Keine Mittelwertbildung	4,5 ms	18,7 ms	22,0 ms	102 ms
Schwach (4 Zyklen): 4 Abtastwerte	10,6 ms	59,3 ms	70,8 ms	346 ms
Mittel (16 Zyklen): 16 Abtastwerte	33,0 ms	208 ms	250 ms	1240 ms
Stark (32 Zyklen): 32 Abtastwerte	63,0 ms	408 ms	490 ms	2440 ms
Abtastzeit	**0,156 ms**	**1,042 ms**	**1,250 ms**	**6,250 ms**

A.12.3.2 Abtastzeit und Aktualisierungszeiten der Analogeingänge

Tabelle A- 218 Abtastzeit und Aktualisierungszeit

Auswahl	Abtastzeit	Aktualisierungszeit SB
400 Hz (2,5 ms)	0,156 ms	0,156 ms
60 Hz (16,6 ms)	1,042 ms	1,042 ms
50 Hz (20 ms)	1,250 ms	1,25 ms
10 Hz (100 ms)	6,250 ms	6,25 ms

A.12.3.3 Messbereiche der analogen Eingänge für Spannung und Strom (SB und SM)

Tabelle A- 219 Darstellung Analogeingang für Spannung (SB und SM)

System		Messbereich Spannung				
Dezimal	Hexadezimal	±10 V	±5 V	±2,5 V	±1,25 V	
32767	7FFF[1]	11,851 V	5,926 V	2,963 V	1,481 V	Überlauf
32512	7F00					
32511	7EFF	11,759 V	5,879 V	2,940 V	1,470 V	Überschwingbereich
27649	6C01					
27648	6C00	10 V	5 V	2,5 V	1,250 V	Bemessungsbereich
20736	5100	7,5 V	3,75 V	1,875 V	0,938 V	
1	1	361,7 µV	180,8 µV	90,4 µV	45,2 µV	
0	0	0 V	0 V	0 V	0 V	
-1	FFFF					
-20736	AF00	-7,5 V	-3,75 V	-1,875 V	-0,938 V	
-27648	9400	-10 V	-5 V	-2,5 V	-1,250 V	

System		Messbereich Spannung				
Dezimal	**Hexadezimal**	**±10 V**	**±5 V**	**±2,5 V**	**±1,25 V**	
-27649	93FF					Unterschwingbereich
-32512	8100	-11,759 V	-5,879 V	-2,940 V	-1,470 V	
-32513	80FF					Unterlauf
-32768	8000	-11,851 V	-5,926 V	-2,963 V	-1,481 V	

[1] 7FFF kann aus einem der folgenden Gründe zurückgegeben werden: Überlauf (wie in dieser Tabelle angemerkt), bevor gültige Werte verfügbar sind (beispielsweise direkt nach dem Einschalten), oder wenn ein Drahtbruch erkannt wird.

Tabelle A- 220 Darstellung Analogeingang für Strom (SB und SM)

System		Messbereich Strom		
Dezimal	**Hexadezimal**	**0 mA bis 20 mA**	**4 mA bis 20 mA**	
32767	7FFF	23,70 mA	22,96 mA	Überlauf
32512	7F00			
32511	7EFF	23,52 mA	22,81 mA	Überschwingbereich
27649	6C01			
27648	6C00	20 mA	20 mA	Nennbereich
20736	5100	15 mA	16 mA	
1	1	723,4 nA	4 mA + 578,7 nA	
0	0	0 mA	4 mA	
-1	FFFF			Unterschwingbereich
-4864	ED00	-3,52 mA	1,185 mA	
-4865	ECFF			Unterlauf
-32768	8000			

A.12.3.4 Messbereiche der analogen Ausgänge für Spannung und Strom (SB und SM)

Tabelle A- 221 Darstellung Analogausgang für Spannung (SB und SM)

System		Spannungsausgangsbereich	
Dezimal	**Hexadezimal**	**±10 V**	
32767	7FFF	Siehe Hinweis 1	Überlauf
32512	7F00	Siehe Hinweis 1	
32511	7EFF	11,76 V	Überschwingbereich
27649	6C01		
27648	6C00	10 V	Bemessungsbereich
20736	5100	7,5 V	
1	1	361,7 µV	
0	0	0 V	
-1	FFFF	-361,7 µV	
-20736	AF00	-7,5 V	
-27648	9400	-10 V	

System		Spannungsausgangsbereich	
Dezimal	**Hexadezimal**	**±10 V**	
-27649	93FF		Unterschwingbereich
-32512	8100	-11,76 V	
-32513	80FF	Siehe Hinweis 1	Unterlauf
-32768	8000	Siehe Hinweis 1	

[1] Bei Überlauf oder Unterlauf nehmen die Analogausgänge den Ersatzwert für den Betriebszustand STOP an.

Tabelle A- 222 Darstellung Analogausgang für Strom (SB und SM)

System		Stromausgangsbereich		
Dezimal	**Hexadezimal**	**0 mA bis 20 mA**	**4 mA bis 20 mA**	
32767	7FFF	Siehe Hinweis 1	Siehe Hinweis 1	Überlauf
32512	7F00	Siehe Hinweis 1	Siehe Hinweis 1	
32511	7EFF	23,52 mA	22,81 mA	Überschwingbereich
27649	6C01			
27648	6C00	20 mA	20 mA	Bemessungsbereich
20736	5100	15 mA	16 mA	
1	1	723,4 nA	4 mA + 578,7 nA	
0	0	0 mA	4 mA	
-1	FFFF		4 mA bis 578,7 nA	Unterschwingbereich
-6912	E500		0 mA	
-6913	E4FF			Nicht möglich. Ausgangswert auf 0 mA begrenzt.
-32512	8100			
-32513	80FF	Siehe Hinweis 1	Siehe Hinweis 1	Unterlauf
-32768	8000	Siehe Hinweis 1	Siehe Hinweis 1	

[1] Bei Überlauf oder Unterlauf nehmen die Analogausgänge den Ersatzwert für den Betriebszustand STOP an.

A.12.4 Thermoelement-Signalboards (SBs)

A.12.4.1 Technische Daten des SB 1231 1 Analogeingang Thermoelement

Hinweis

Um diesen SB nutzen zu können, benötigen Sie eine CPU mit Firmware ab V2.0.

Tabelle A- 223 Allgemeine technische Daten

Technische Daten	SB 1231 AI 1 x 16 Bit Thermoelement
Artikelnummer	6ES7 231-5QA30-0XB0
Abmessungen B x H x T (mm)	38 x 62 x 21
Gewicht	35 Gramm
Leistungsverlust	0,5 W
Stromaufnahme (SM-Bus)	5 mA
Stromaufnahme (24 V DC)	20 mA

Tabelle A- 224 Analoge Eingänge

Technische Daten		SB 1231 AI 1x16 Bit Thermoelement
Anzahl der Eingänge		1
Typ		Potentialfrei, TC und mV
Bereich • Nennbereich (Datenwort) • Überlauf/Unterlauf (Datenwort) • Überlauf/Unterlauf (Datenwort)		Siehe Filterauswahltabelle Thermoelement (Seite 1301).
Auflösung	Temperaturbereich	0,1 °C/0,1 °F
	Spannung	15 Bit plus Vorzeichen
Max. Stehspannung		±35 V
Rauschunterdrückung		85 dB für die gewählte Filtereinstellung (10 Hz, 50 Hz, 60 Hz, 400 Hz)
Gleichtaktunterdrückung		> 120 dB bei 120 V AC
Impedanz		≥ 10 MΩ
Genauigkeit		Siehe Thermoelement-Auswahltabelle (Seite 1301).
Wiederholgenauigkeit		±0,05 % Vollausschlag
Messprinzip		Integrierend
Aktualisierungszeit Modul		Siehe Filterauswahltabelle Thermoelement (Seite 1301).
Fehler kalte Verbindungsstelle		±1,5 °C
Elektrische Trennung (Feld zu Logik)		500 V AC

Technische Daten	SB 1231 AI 1x16 Bit Thermoelement
Leitungslänge (Meter)	Max. 100 m zum Geber
Leitungswiderstand	max. 100 Ω

Tabelle A- 225 Diagnose

Technische Daten	SB 1231 AI 1 x 16 Bit Thermoelement
Überlauf/Unterlauf [1]	Ja
Drahtbruch[2]	Ja

[1] Die Informationen der Diagnosealarme "Überlauf" und "Unterlauf" werden auch dann mit den Analogdatenwerten gemeldet, wenn die entsprechenden Alarme bei der Modulprojektierung deaktiviert werden.

[2] Wenn der Drahtbruchalarm deaktiviert ist und in der Geberverdrahtung ein frei liegender Draht vorliegt, kann das Modul zufällige Werte melden.

Das analoge Thermoelement-Signalmodul SM 1231 TC misst den Wert der an die Moduleingänge angeschlossenen Spannung.

Das analoge Thermoelement-Signalboard SB 1231 misst den Wert der an die Signalboardeingänge angeschlossenen Spannung. Als Temperaturmessart sind entweder "Thermoelement" oder "Spannung" möglich.

- "Thermoelement": Der Messwert wird in Grad, multipliziert mit zehn, ausgegeben (Beispiel: 25,3 Grad werden als Dezimalwert 253 dargestellt).
- "Spannung": Der Messbereichsendwert im Nennbereich beträgt 27648 dezimal.

A.12.4.2 Grundlegende Funktionsweise eines Thermoelements

Thermoelemente entstehen, wenn zwei unterschiedliche Metalle elektrisch miteinander verbunden werden. Dadurch wird eine Spannung erzeugt, die proportional zu der Temperatur der Verbindungsstelle ist. Es handelt sich um eine geringe Spannung. Ein Mikrovolt kann viele Grade darstellen. Grundlage für die Temperaturmessung mit Thermoelementen sind das Messen der Spannung eines Thermoelements, das Kompensieren von zusätzlichen Verbindungsstellen und das Linearisieren der Ergebnisse.

Wenn Sie ein Thermoelement an das SM 1231 Thermoelementmodul anschließen, werden die beiden Leitungen der unterschiedlichen Metalle am Signalanschluss des Moduls angeschlossen. Die Stelle, an der die beiden unterschiedlichen Leitungen miteinander verbunden werden, bildet den Sensor des Thermoelements.

Zwei weitere Thermoelemente entstehen an der Stelle, an der die unterschiedlichen Leitungen an den Signalanschluss angeschlossen werden. Die Temperatur des Anschlusses erzeugt eine Spannung, die zu der Spannung des Thermoelementsensors addiert wird. Wird diese Spannung nicht ausgeglichen, weicht die ausgegebene Temperatur von der Temperatur des Sensors ab.

Durch die Kompensation der kalten Verbindungsstelle wird das Thermoelement am Anschluss ausgeglichen. Tabellen für Thermoelemente basieren auf einer Bezugstemperatur an der Verbindungsstelle, üblicherweise Null Grad Celsius. Durch die Kompensation der kalten Verbindungsstelle wird der Anschluss auf Null Grad Celsius kompensiert. Die Spannung, die durch das Thermoelement des Anschlusses addiert wird, wird durch die Kompensation der kalten Verbindungsstelle korrigiert. Die Temperatur des Moduls wird innen gemessen und dann in einen Wert umgewandelt, der zur Sensorumsetzung addiert wird. Die korrigierte Sensorumsetzung wird dann mittels der Thermoelement-Tabellen linearisiert.

Für die optimale Funktionsweise der Kompensation der kalten Verbindungsstelle muss sich das Thermoelementmodul in einer thermisch stabilen Umgebung befinden. Ein langsame Veränderung (weniger als 0,1 °C/Minute) der Temperatur in der Umgebung des Moduls wird innerhalb der Modulspezifikation korrekt ausgeglichen. Auch Luftbewegungen am Modul verursachen Fehler bei der Kompensation der kalten Verbindungsstelle.

Ist eine bessere Kompensation von Fehlern an der kalten Verbindungsstelle erforderlich, kann eine externe isothermische Klemmenleiste verwendet werden. Mit dem Thermoelementmodul kann eine auf 0 °C bezogene oder eine auf 50 °C bezogene Klemmenleiste eingesetzt werden.

Auswahltabelle für das SB 1231 Thermoelement

Die Bereiche und Genauigkeit der verschiedenen vom SB 1231 Thermoelement-Signalboard unterstützten Thermoelementtypen entnehmen Sie bitte der unten stehenden Tabelle.

Tabelle A- 226 SB 1231 Thermoelement-Auswahltabelle

Thermoelementtyp	Minimum unterer Bereich[1]	Unterer Grenzwert Nennbereich	Oberer Grenzwert Nennbereich	Maximum oberer Bereich[2]	Genauigkeit Normalbereich [3] bei 25 °C	Genauigkeit Normalbereich [3] bei -20 °C bis 60 °C
J	-210,0 °C	-150,0 °C	1.200,0 °C	1.450,0 °C	±0,3 °C	±0,6 °C
K	-270,0 °C	-200,0 °C	1.372,0 °C	1.622,0 °C	±0,4 °C	±1,0 °C
T	-270,0 °C	-200,0 °C	400,0 °C	540,0 °C	±0,5 °C	±1,0 °C
E	-270,0 °C	-200,0 °C	1.000,0 °C	1.200,0 °C	±0,3 °C	±0,6 °C
R & S	-50,0 °C	100,0 °C	1.768,0 °C	2.019,0 °C	±1,0 °C	±2,5 °C
B	0,0 °C	200,0 °C	800,0 °C	--	±2,0 °C	±2,5 °C
	--	800,0 °C	1.820,0 °C	1.820,0 °C	±1,0 °C	±2,3 °C
N	-270,0 °C	0,0 °C	1.300,0 °C	1.550,0 °C	±1,0 °C	±1,6 °C
C	0,0 °C	100,0 °C	2.315,0 °C	2.500,0 °C	±0,7 °C	±2,7 °C
TXK/XK(L)	-200,0 °C	-150,0 °C	800,0 °C	1.050,0 °C	±0,6 °C	±1,2 °C
Spannung	-32511	-27648 -80 mV	27648 80 mV	32511	±0.05%	±0.1%

1 Die Thermoelementwerte unterhalb des Minimums für den unteren Bereich werden als -32768 ausgegeben.

2 Die Thermoelementwerte oberhalb des Minimums für den oberen Bereich werden als 32767 ausgegeben.

3 Der interne Fehler an der kalten Verbindungsstelle beträgt ±1,5 °C für alle Bereiche. Dieser Wert ist zum in dieser Tabelle aufgeführten Fehler zu addieren. Das Signalboard benötigt eine Aufwärmzeit von mindestens 30 Minuten, bis die hier genannten Spezifikationen erfüllt werden.

Tabelle A- 227 Filterauswahltabelle für das SB 1231 Thermoelement

Unterdrückungsfrequenz (Hz)	Integrationszeit (ms)	Aktualisierungszeit Signalboard (Sekunden)
10	100	0.306
50	20	0.066
60	16.67	0.056
400[1]	10	0.036

[1] Um die Auflösung und Messgenauigkeit des Moduls bei Auswahl der 400-Hz-Unterdrückung aufrecht zu erhalten, beträgt die Integrationszeit 10 ms. Mit dieser Auswahl erfolgt auch die Rauschunterdrückung bei 100 Hz und 200 Hz.

Für die Messung von Thermoelementen wird eine Integrationszeit von 100 ms empfohlen. Niedriger eingestellte Integrationszeiten führen zu einem höheren Wiederholgenauigkeitsfehler der Temperaturmessungen.

Hinweis

Nach dem Einschalten führt das Modul die interne Kalibrierung für den A/D-Wandler durch. In diesem Zeitraum meldet das Modul auf jedem Kanal den Wert 32767, bis für den jeweiligen Kanal gültige Daten vorliegen. Diese Initialisierungszeit muss im Anwenderprogramm ggf. berücksichtigt werden.

Tabelle A- 228 Schaltplan des SB 1231 AI 1 x 16 Thermoelement

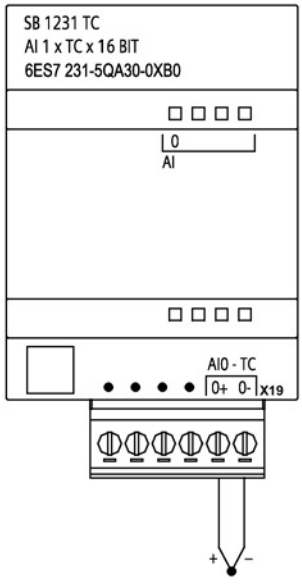
SB 1231 AI 1 x 16 Bit Thermoelement (6ES7 231-5QA30-0XB0)	
	
Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.	

Tabelle A- 229 Anschlussbelegungen für das SB 1231 AI 1 x 16 Bit Thermoelement (6ES7 231-5QA30-0XB0)

Pin	X19 (vergoldet)
1	Kein Anschluss
2	Kein Anschluss
3	Kein Anschluss
4	Kein Anschluss
5	AI 0- /TC
6	AI 0+ /TC

A.12.5 RTD-Signalboards (SBs)

A.12.5.1 Technische Daten des SB 1231 1 Analogeingang RTD

Hinweis

Um diesen SB nutzen zu können, benötigen Sie eine CPU mit Firmware ab V2.0.

Tabelle A- 230 Allgemeine technische Daten

Technische Daten	SB 1231 AI 1 x 16 Bit RTD
Artikelnummer	6ES7 231-5PA30-0XB0
Abmessungen B x H x T (mm)	38 x 62 x 2
Gewicht	35 Gramm
Leistungsverlust	0,7 W
Stromaufnahme (SM-Bus)	5 mA
Stromaufnahme (24 V DC)	25 mA

Tabelle A- 231 Analoge Eingänge

Technische Daten	SB 1231 AI 1 x 16 Bit RTD
Anzahl der Eingänge	1
Typ	Modulreferenz RTD und Ohm
Bereich • Nennbereich (Datenwort) • Überlauf/Unterlauf (Datenwort) • Überlauf/Unterlauf (Datenwort)	Siehe Auswahltabellen (Seite 1307).

Technische Daten		SB 1231 AI 1 x 16 Bit RTD
Auflösung	Temperatur	0,1 °C / 0,1 °F
	Spannung	15 Bit plus Vorzeichen
Max. Stehspannung		±35 V
Rauschunterdrückung		85 dB (10 Hz, 50 Hz, 60 Hz, 400 Hz)
Gleichtaktunterdrückung		> 120 dB
Impedanz		≥ 10 MΩ
Genauigkeit		Siehe Auswahltabellen (Seite 1307).
Wiederholgenauigkeit		±0,05 % Vollausschlag
Maximale Verlustleistung Geber		0.5 m W
Messprinzip		Integrierend
Aktualisierungszeit Modul		Siehe Auswahltabelle (Seite 1307).
Elektrische Trennung (Feld zu Logik)		500 V AC
Leitungslänge (Meter)		Max. 100 m zum Geber
Leitungswiderstand		20 Ω, 2,7 für max. 10 Ω RTD

Tabelle A- 232 Diagnose

Technische Daten	SB 1231 AI 1 x 16 Bit RTD
Überlauf/Unterlauf[1, 2]	Ja
Drahtbruch[3]	Ja

1 Die Informationen der Diagnosealarme "Überlauf" und "Unterlauf" werden auch dann mit den Analogdatenwerten gemeldet, wenn die entsprechenden Alarme bei der Modulprojektierung deaktiviert werden.

2 Für Widerstandsmessbereiche ist die Prüfung auf Unterlauf grundsätzlich nicht aktiviert.

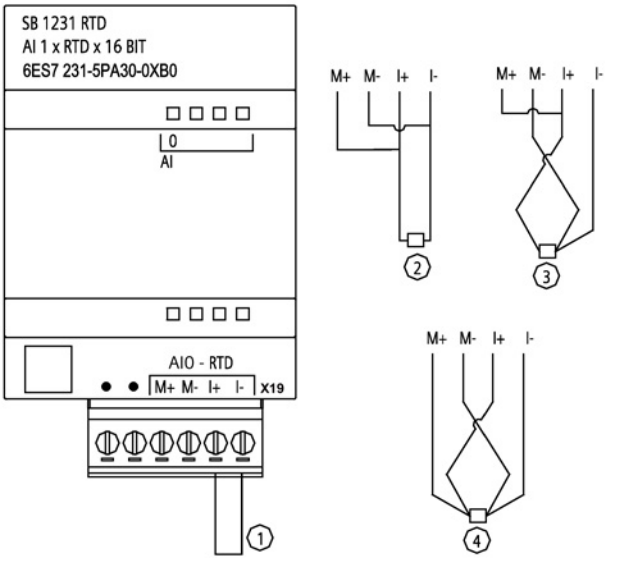
3 Wenn der Drahtbruchalarm deaktiviert ist und in der Geberverdrahtung ein frei liegender Draht vorliegt, kann das Modul zufällige Werte melden.

Das analoge Thermoelement-Signalboard SM 1231 RTD misst den Wert des an die Signalboardeingänge angeschlossenen Widerstands. Als Messart kann entweder "Widerstand" oder "Thermischer Widerstand" ausgewählt werden.

- "Widerstand": Der Messbereichsendwert im Nennbereich beträgt 27648 dezimal.
- "Thermischer Widerstand": Der Messwert wird in Grad, multipliziert mit zehn, ausgegeben (Beispiel: 25,3 Grad werden als Dezimalwert 253 dargestellt). Die Werte des klimatischen Bereichs werden in Grad, multipliziert mit Hundert, ausgegeben (Beispiel: 25,34 Grad werden als Dezimalwert 2534 dargestellt).

Das Signalboard SB 1231 RTD unterstützt Messungen über 2-Leiter-, 3-Leiter- und 4-Leiter-Anschlüsse zum Geberwiderstand.

Tabelle A- 233 Schaltplan des SB 1231 AI 1 x 16 Bit RTD

SB 1231 AI 1 x 16 Bit RTD (6ES7 231-5PA30-0XB0)	
	

① Nicht belegten RTD-Eingang zurückschleifen

① 2Leiter-RTD

① 3Leiter-RTD

① 4Leiter-RTD

Hinweis: Die Steckverbinder müssen vergoldet sein. Im Anhang C, Ersatzteile, finden Sie die Artikelnummer.

Tabelle A- 234 Anschlussbelegungen für das SB 1231 AI 1 x 16 Bit RTD (6ES7 231-5PA30-0XB0)

Pin	X19 (vergoldet)
1	Kein Anschluss
2	Kein Anschluss
3	AI 0 M+ /RTD
4	AI 0 M- /RTD
5	AI 0 I+ /RTD
6	AI 0 I- /RTD

A.12.5.2 Auswahltabellen für das SB 1231 RTD

Tabelle A- 235 Bereiche und Genauigkeit für die verschiedenen Geber, die von den RTD-Modulen unterstützt werden

Temperaturkoeffizient	RTD-Typ	Minimum unterer Bereich[1]	Unterer Grenzwert Nennbereich	Oberer Grenzwert Nennbereich	Maximum oberer Bereich[2]	Genauigkeit Normalbereich bei 25 °C	Genauigkeit Normalbereich -20 °C bis 60 °C
Pt 0,003850 ITS90 DIN EN 60751	Pt 100 Klima	-145,00 °C	-120,00 °C	-145,00 °C	-155,00 °C	±0,20 °C	±0,40 °C
	Pt 10	-243,0 °C	-200,0 °C	850,0 °C	1.000,0 °C	±1,0 °C	±2,0 °C
	Pt 50	-243,0 °C	-200,0 °C	850,0 °C	1.000,0 °C	±0,5 °C	±1,0 °C
	Pt 100						
	Pt 200						
	Pt 500						
	Pt 1000						
Pt 0,003902 Pt 0,003916 Pt 0,003920	Pt 100	-243,0 °C	-200,0 °C	850,0 °C	1.000,0 °C	±0,5 °C	±1,0 °C
	Pt 200						
	Pt 500						
	Pt 1000						
Pt 0,003910	Pt 10	-273,2 °C	-240,0 °C	1.100,0 °C	1.295 °C	±1,0 °C	±2,0 °C
	Pt 50	-273,2 °C	-240,0 °C	1.100,0 °C	1.295 °C	±0,8 °C	±1,6 °C
	Pt 100						
	Pt 500						
Ni 0,006720 Ni 0,006180	Ni 100	-105,0 °C	-60,0 °C	250,0 °C	295,0 °C	±0,5 °C	±1,0 °C
	Ni 120						
	Ni 200						
	Ni 500						
	Ni 1000						
LG-Ni 0,005000	LG-Ni 1000	-105,0 °C	-60,0 °C	250,0 °C	295,0 °C	±0,5 °C	±1,0 °C
Ni 0,006170	Ni 100	-105,0 °C	-60,0 °C	180,0 °C	212,4 °C	±0,5 °C	±1,0 °C
Cu 0,004270	Cu 10	-240,0 °C	-200,0 °C	260,0 °C	312,0 °C	±1.0 °	±2,0 °C
Cu 0,004260	Cu 10	-60,0 °C	-50,0 °C	200,0 °C	240,0 °C	±1,0 °C	±2,0 °C
	Cu 50	-60,0 °C	-50,0 °C	200,0 °C	240,0 °C	±0,6 °C	±1,2 °C
	Cu 100						
Cu 0,004280	Cu 10	-240,0 °C	-200,0 °C	200,0 °C	240,0 °C	±1,0 °C	±2,0 °C
	Cu 50	-240,0 °C	-200,0 °C	200,0 °C	240,0 °C	±0,7 °C	±1,4 °C
	Cu 100						

1 Die RTD-Werte unterhalb des Minimums für den unteren Bereich werden als -32768 ausgegeben.

2 Die RTD-Werte oberhalb des Maximums für den oberen Bereich werden als +32768 ausgegeben.

Tabelle A- 236 Beständigkeit gegen:

Bereich	Minimum unterer Bereich	Unterer Grenzwert Nennbereich	Oberer Grenzwert Nennbereich	Maximum oberer Bereich[1]	Genauigkeit Normalbereich bei 25 °C	Genauigkeit Normalbereich -20 °C bis 60 °C
150 Ω	nicht zutreffend	0 (0 Ω)	27648 (150 Ω)	176,383 Ω	±0.05%	±0.1%
300 Ω	nicht zutreffend	0 (0 Ω)	27648 (300 Ω)	352,767 Ω	±0.05%	±0.1%
600 Ω	nicht zutreffend	0 (0 Ω)	27648 (600 Ω)	705,534 Ω	±0.05%	±0.1%

[1] Die Widerstandswerte oberhalb des Maximums für den oberen Bereich werden als 32767 ausgegeben.

Hinweis

Für alle aktivierte Kanäle ohne angeschlossenen Geber meldet das Modul 32767. Wenn außerdem die Prüfung auf offene Leitungen aktiviert ist, blinken am Modul die entsprechenden roten LED.

Optimale Genauigkeit für die 10 Ω-RTD-Bereiche ermöglichen 4-Leiter-Anschlüsse.

Der Widerstand der Anschlussleitungen im 2-Leiter-Modus verursacht einen Fehler der Gebermessung. Die Messgenauigkeit ist daher nicht mehr gewährleistet.

Tabelle A- 237 Rauschminderung und Aktualisierungszeiten für die RTD-Module

Auswahl Unterdrückungsfrequenz	Integrationszeit	1-kanaliges 4/2-Draht-Modul Aktualisierungszeit (Sekunden)	1-kanaliges 3-Draht-Modul Aktualisierungszeit (Sekunden)
400 Hz (2,5 ms)	10 ms [1]	0.036	0.071
60 Hz (16,6 ms)	16,67 ms	0.056	0.111
50 Hz (20 ms)	20 ms	0.066	1.086
10 Hz (100 ms)	100 ms	0.306	0.611

[1] Um die Auflösung und Messgenauigkeit des Moduls bei Auswahl des 400-Hz-Filters aufrecht zu erhalten, beträgt die Integrationszeit 10 ms. Mit dieser Auswahl erfolgt auch die Rauschunterdrückung bei 100 Hz und 200 Hz.

Hinweis

Nach dem Einschalten führt das Modul die interne Kalibrierung für den A/D-Wandler durch. In diesem Zeitraum meldet das Modul auf jedem Kanal den Wert 32767, bis für den jeweiligen Kanal gültige Daten vorliegen. Diese Initialisierungszeit muss im Anwenderprogramm ggf. berücksichtigt werden. Weil sich die Konfiguration des Moduls auf die Dauer der Initialisierungszeit auswirken kann, prüfen Sie das Verhalten des Moduls in Ihrer Konfiguration. Ggf. können Sie Logik in Ihr Anwenderprogramm aufnehmen, um die Initialisierungszeit des Moduls unterzubringen.

A.13 BB 1297 Batterieboard

BB 1297 Batterieboard

Das S7-1200 BB 1297 Batterieboard dient der langfristigen Pufferung der Echtzeituhr. Es kann in den Steckplatz des Signalboards der S7-1200 CPU (ab Firmware 3.0) gesteckt werden. Sie müssen das BB 1297 in die Gerätekonfiguration aufnehmen und die Hardwarekonfiguration in die CPU laden, damit das BB funktionsfähig ist.

Die Batterie (Typ CR1025) ist nicht im Lieferumfang des BB 1297 enthalten und ist separat zu erwerben.

Hinweis

Das BB 1297 ist mechanisch für die CPUs mit Firmware ab Version 3.0 ausgelegt.

Verwenden Sie das BB 1297 nicht mit CPUs früherer Versionen, weil sich der Stecker des BB 1297 nicht in die CPU stecken lässt.

⚠ WARNUNG

Die Installation einer nicht spezifizierten Batterie im BB 1297 oder der Anschluss einer nicht spezifizierten Batterie an den Schaltkreis auf andere Weise kann zu Brand oder Bauteilbeschädigung und unvorhersehbarem Betrieb von Maschinen führen.

Brand oder unvorhersehbarer Betrieb von Maschinen können tödliche oder schwere Verletzungen oder Sachschäden verursachen.

Verwenden Sie für die Pufferung der Echtzeituhr nur die spezifizierte Batterie CR1025.

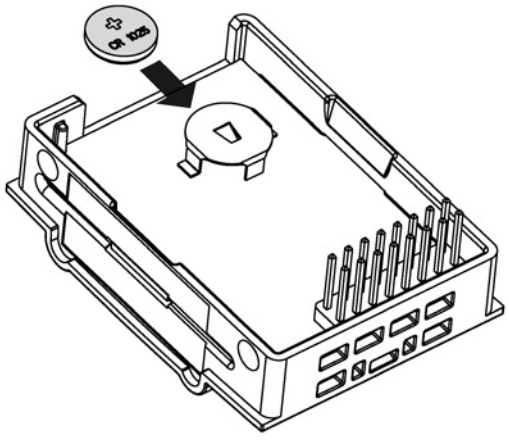
Tabelle A- 238 Allgemeine technische Daten

Technische Daten	BB 1297 Batterieboard
Artikelnummer	6ES7 297-0AX30-0XA0
Abmessungen B x H x T (mm)	38 x 62 x 21
Gewicht	28 Gramm
Leistungsverlust	0,5 W
Stromaufnahme (SM-Bus)	11 mA
Stromaufnahme (24 V DC)	Keine

Batterie (nicht enthalten)	BB 1297 Batterieboard
Netzausfallüberbrückung	Ca. 1 Jahr
Batterietyp	CR1025 Siehe Einsetzen oder Austauschen der Batterie des Batterieboards BB 1297 (Seite 62)
Nennspannung	3 V
Nennkapazität	Mindestens 30 mAH

Diagnose	BB 1297 Batterieboard
Kritischer Batteriezustand	< 2,5 V
Batteriediagnose	Anzeige bei geringer Spannung: • Geringe Batteriespannung bewirkt, dass die MAINT-LED der CPU ständig bernsteinfarben leuchtet. • Diagnosepuffereignis: 16#06:2700 "Submodulwartung erforderlich: mindestens eine Batterie ist erschöpft (BATTF)"
Batteriezustand	Batteriezustandsbit 0 = Batterie OK 1 = Batterie schwach
Aktualisierung des Batteriezustands	Der Batteriezustand wird beim Anlauf und dann einmal täglich aktualisiert, wenn sich die CPU im Betriebszustand RUN befindet.

Tabelle A- 239 Einbauabbildung für das Batterieboard BB 1297

Batterieboard BB 1297 (6ES7 297-0AX30-0XA0)	
	

A.14 Kommunikationsschnittstellen

A.14.1 PROFIBUS

A.14.1.1 CM 1242-5 PROFIBUS DP-SLAVE

Tabelle A- 240 Technische Daten des CM 1242-5

Technische Daten	
Artikelnummer	6GK7 242-5DX30-0XE0
Schnittstellen	
Anschluss an PROFIBUS	9-polige Sub-D-Buchse
Maximale Stromaufnahme an der PROFIBUS-Schnittstelle beim Anschluss von Netzkomponenten (beispielsweise optische Netzkomponenten)	15 mA bei 5 V (nur für den Busabschluss) *)
Zulässige Umgebungsbedingungen	
Umgebungstemperatur • während der Lagerung • während des Transports • während des Betriebs bei vertikalem Einbau (Hutschiene horizontal) • während des Betriebs bei horizontalem Einbau (Hutschiene vertikal)	 • -40 °C bis 70 °C • -40 °C bis 70 °C • 0 °C bis 55 °C • 0 °C bis 45 °C
Relative Luftfeuchtigkeit bei 25 °C im Betrieb, ohne Kondensation, max.	95 %
Schutzart	IP20
Spannungsversorgung, Stromaufnahme und Verlustleistung	
Art der Spannungsversorgung	DC
Spannungsversorgung aus dem Rückwandbus	5 V
Stromaufnahme (typisch)	150 mA
Effektive Verlustleistung (typisch)	0,75 W
Potentialtrennung • PROFIBUS-Schnittstelle zur Masse • PROFIBUS-Schnittstelle zur internen Schaltung	710 V DC für 1 Minute
Abmessungen und Gewichte	
• Breite • Höhe • Tiefe	• 30 mm • 100 mm • 75 mm

Technische Daten	
Gewicht	
• Nettogewicht	• 115 g
• Gewicht mit Verpackung	• 152 g

*)Die Strombelastung durch einen externen Verbraucher, der zwischen VP (Pin 6) und DGND (Pin 5) angeschlossen wird, darf für den Busabschluss maximal 15 mA betragen (kurzschlussfest).

A.14.1.2 Anschlussbelegung der Sub-D-Buchse des CM 1242-5

PROFIBUS-Schnittstelle

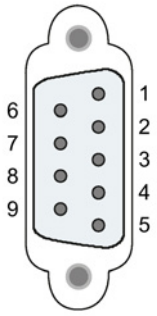


Tabelle A- 241 Anschlussbelegung der Sub-D-Buchse

Pin	Beschreibung	Pin	Beschreibung
1	- nicht belegt -	6	P5V2: Spannungsversorgung +5V
2	- nicht belegt -	7	- nicht belegt -
3	RxD/TxD-P: Datenleitung B	8	RxD/TxD-N: Datenleitung A
4	RTS	9	- nicht belegt -
5	M5V2: Datenbezugspotential (Masse DGND)	Gehäuse	Masseanschluss

A.14.1.3 CM 1243-5 PROFIBUS DP-Master

Tabelle A- 242 Technische Daten des CM 1243-5

Technische Daten	
Artikelnummer	6GK7 243-5DX30-0XE0
Schnittstellen	
Anschluss an PROFIBUS	9-polige Sub-D-Buchse
Maximale Stromaufnahme an der PROFIBUS-Schnittstelle beim Anschluss von Netzkomponenten (beispielsweise optische Netzkomponenten)	15 mA bei 5 V (nur für den Busabschluss) *)
Zulässige Umgebungsbedingungen	
Umgebungstemperatur • während der Lagerung • während des Transports • während des Betriebs bei vertikalem Einbau (Hutschiene horizontal) • während des Betriebs bei horizontalem Einbau (Hutschiene vertikal)	 • -40 °C bis 70 °C • -40 °C bis 70 °C • 0 °C bis 55 °C • 0 °C bis 45 °C
Relative Luftfeuchtigkeit bei 25 °C im Betrieb, ohne Kondensation, max.	95 %
Schutzart	IP20
Spannungsversorgung, Stromaufnahme und Verlustleistung	
Art der Spannungsversorgung	DC
Spannungsversorgung / extern • minimal • maximal	24 V • 19,2 V • 28,8 V
Stromaufnahme (typisch) • aus 24 V DC • aus dem S7-1200 Rückwandbus	 • 100 mA • 0 mA
Effektive Verlustleistung (typisch) • aus 24 V DC • aus dem S7-1200 Rückwandbus	 • 2,4 W • 0 W
Spannungsversorgung 24 V DC / extern • Min. Leitungsquerschnitt • Max. Leitungsquerschnitt • Anzugsmoment der Schraubklemmen	 • min.: 0,14 mm^2 (AWG 25) • max.: 1,5 mm^2 (AWG 15) • 0,45 Nm (4 lb-in)
Potentialtrennung • PROFIBUS-Schnittstelle zur Masse • PROFIBUS-Schnittstelle zur internen Schaltung	710 V DC für 1 Minute

Technische Daten	
Abmessungen und Gewichte	
• Breite • Höhe • Tiefe	• 30 mm • 100 mm • 75 mm
Gewicht • Nettogewicht • Gewicht mit Verpackung	 • 134 g • 171 g

*)Die Strombelastung durch einen externen Verbraucher, der zwischen VP (Pin 6) und DGND (Pin 5) angeschlossen wird, darf für den Busabschluss maximal 15 mA betragen (kurzschlussfest).

Hinweis

Das CM 1243- (PROFIBUS-Mastermodul) muss über die 24-V-DC-Geberversorgung der CPU gespeist werden.

Hinweis

Das CM 1243-5 (PROFIBUS-Mastermodul) muss über die 24-V-DC-Geberversorgung der CPU gespeist werden.

A.14.1.4 Anschlussbelegung der Sub-D-Buchse des CM 1243-5

PROFIBUS-Schnittstelle

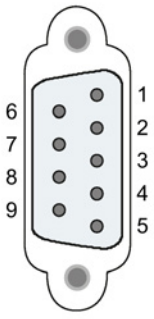


Tabelle A- 243 Anschlussbelegung der Sub-D-Buchse

Pin	Beschreibung	Pin	Beschreibung
1	- nicht belegt -	6	VP: Spannungsversorgung +5 V nur für Busabschlusswiderstände; nicht für die Versorgung externer Geräte
2	- nicht belegt -	7	- nicht belegt -
3	RxD/TxD-P: Datenleitung B	8	RxD/TxD-N: Datenleitung A
4	CNTR-P: RTS	9	- nicht belegt -
5	DGND: Masse für Datensignale und VP	Gehäuse	Masseanschluss

PROFIBUS-Kabel

Hinweis

Auflegen der Schirmung des PROFIBUS-Kabels

Der Schirm des PROFIBUS-Kabels muss aufgelegt werden.

Isolieren Sie hierzu das PROFIBUS-Kabel am Ende ein Stück ab und verbinden Sie den Schirm mit der Funktionserde.

A.14.2 CP 1242-7

Hinweis

Der CP 1242-7 hat keine Zulassung für Anwendungen für das Seewesen

Der CP 1242-7 hat keine Zulassung für das Seewesen.

Hinweis

Um diese Module nutzen zu können, benötigen Sie eine CPU mit Firmware ab V2.0.

A.14.2.1 CP 1242-7 GPRS

Tabelle A- 244 Technische Daten des CP 1242-7 GPRS V2

Technische Daten	
Artikelnummer	6GK7 242-7KX3-0XE0
Funkschnittstelle	
Antennenanschluss	SMA-Buchse
Nennimpedanz	50 Ohm
Funkverbindung	
Maximale Sendeleistung	• GSM 850, Class 4: +33 dBm ±2dBm • GSM 900, Class 4: +33 dBm ±2dBm • GSM 1800, Class 1: +30 dBm ±2dBm • GSM 1900, Class 1: +30 dBm ±2dBm
GPRS	Multislot-Klasse 10 Endgeräteklasse B Codierungsschema 1...4 (GMSK)
SMS	Betriebsmodus abgehend: MO Dienst: Punkt zu Punkt
Zulässige Umgebungsbedingungen	
Umgebungstemperatur • während der Lagerung • während des Transports • während des Betriebs bei vertikalem Einbau (Hutschiene horizontal) • während des Betriebs bei horizontalem Einbau (Hutschiene vertikal)	 • -40 °C bis 70 °C • -40 °C bis 70 °C • 0 °C bis 55 °C • 0 °C bis 45 °C
Relative Luftfeuchtigkeit bei 25 °C im Betrieb, ohne Kondensation, max.	95 %
Schutzart	IP20
Spannungsversorgung, Stromaufnahme und Verlustleistung	
Art der Spannungsversorgung	DC
Spannungsversorgung / extern • minimal • maximal	24 V • 19,2 V • 28,8 V
Stromaufnahme (typisch) • aus 24 V DC • aus dem S7-1200 Rückwandbus	 • 100 mA • 0 mA

Technische Daten	
Effektive Verlustleistung (typisch) • aus 24 V DC • aus dem S7-1200 Rückwandbus	 • 2,4 W • 0 W
Stromversorgung 24 V DC • Min. Leitungsquerschnitt • Max. Leitungsquerschnitt • Anzugsmoment der Schraubklemmen	 • min.: 0,14 mm^2 (AWG 25) • max.: 1,5 mm^2 (AWG 15) • 0,45 Nm (4 lb-in)
Potentialtrennung Netzteil gegen interne Schaltung	710 V DC für 1 Minute
Abmessungen und Gewichte	
• Breite • Höhe • Tiefe	• 30 mm • 100 mm • 75 mm
Gewicht • Nettogewicht • Gewicht mit Verpackung	 • 133 g • 170 g

A.14.2.2 GSM/GPRS-Antenne ANT794-4MR

Technische Daten der GSM/GPRS-Antenne ANT794-4MR

ANT794-4MR	
Artikelnummer	6NH9860-1AA00
Mobilfunknetze	GSM/GPRS
Frequenzbereiche	• 824 bis 960 MHz (GSM 850, 900) • 1710 bis 1880 MHz (GSM 1800, 900) • 1900 bis 2200 MHz (GSM / UMTS)
Charakteristik	omnidirektional
Antennengewinn	0 dB
Impedanz	50 Ohm
Stehwellenverhältnis (SWR)	< 2,0
Max. Leistung	20 W
Polarität	linear vertikal
Stecker	SMA
Länge Antennenkabel	5 m
Außenmaterial	Hart-PVC, UV-beständig
Schutzart	IP20

ANT794-4MR	
Zulässige Umgebungsbedingungen • Betriebstemperatur • Transport-/Lagertemperatur • Relative Luftfeuchtigkeit	 • -40 °C bis +70 °C • -40 °C bis +70 °C • 100 %
Außenmaterial	Hart-PVC, UV-beständig
Konstruktiver Aufbau	Antenne mit 5 m fest verbundenem Kabel und SMA-Stecker
Maße (T x H) in mm	25 x 193
Gewicht • Antenne mit Kabel • Montageteile	 • 310 g • 54 g
Montage	Mit mitgeliefertem Haltewinkel

A.14.2.3 Flachantenne ANT794-3M

Technische Daten der Flachantenne ANT794-3M

ANT794-3M		
Artikelnummer	6NH9870-1AA00	
Mobilfunknetze	**GSM 900**	**GSM 1800/1900**
Frequenzbereiche	890 - 960 MHz	1710 - 1990 MHz
Stehwellenverhältnis (VSWR)	≤ 2:1	≤ 1,5:1
Rückflussdämpfung (Tx)	≈ 10 dB	≈ 14 dB
Antennengewinn	0 dB	
Impedanz	50 Ohm	
Max. Leistung	10 W	
Antennenkabel	HF-Kabel RG 174 (fest angeschlossen) mit SMA-Stecker	
Kabellänge	1,2 m	
Schutzart	IP64	
Zulässiger Temperaturbereich	-40 ℃ bis +75 ℃	
Entflammbarkeit	UL 94 V2	
Außenmaterial	ABS Polylac PA-765, lichtgrau (RAL 7035)	
Maße (B x L x H) in mm	70,5 x 146,5 x 20,5	
Gewicht	130 g	

A.14.3 CM 1243-2 AS-i-Master

A.14.3.1 Technische Daten des AS-i Master CM 1243-2

Tabelle A- 245 Technische Daten des AS-i Master CM 1243-2

Technische Daten	
Artikelnummer	3RK7243-2AA30-0XB0
Firmwareversion	V1.0
Datum	01.12.2011
Schnittstellen	
Maximale Stromaufnahme aus dem S7-1200 Rückwandbus aus der AS-i Leitung	 max. 250 mA, Versorgungsspannung S7-1200 Kommunikationsbus 5 V DC max. 100 mA
Maximale Strombelastbarkeit zwischen ASI+/ASI-Klemmen	8 A
Anschlussbelegung	Siehe Kapitel Elektrischer Anschluss des AS-i-Masters (Seite 1320)
Leiterquerschnitt	0,2 mm² (AWG 24) ... 3,3 mm² (AWG 12)
Anzugsmoment ASI-Stecker	0,56 Nm
Zulässige Umgebungsbedingungen	
Umgebungstemperatur während der Lagerung während des Transports während des Betriebs bei vertikalem Einbau (Standard-Hutschiene horizontal) während des Betriebs bei horizontalem Einbau (Standard-Hutschiene vertikal)	 -40 °C ... 70 °C -40 °C ... 70 °C 0 °C ... 55 °C 0 °C ... 45 °C
Relative Luftfeuchtigkeit bei 25 °C im Betrieb, ohne Kondensation, max.	95 %
Schutzart	IP20
Spannungsversorgung, Stromaufnahme, Verlustleistung	
Art der Spannungsversorgung	DC
Stromaufnahme (typisch) aus dem S7-1200 Rückwandbus	 200 mA
Gesamtverlustleistung (typisch): • aus dem S7-1200 Rückwandbus • aus der AS-i Leitung	 1 W 2,4 W

Technische Daten	
Abmessungen und Gewichte	
Breite	30 mm
Höhe	100 mm
Tiefe	75 mm
Gewicht	
Nettogewicht	122 g
Gewicht mit Verpackung	159 g

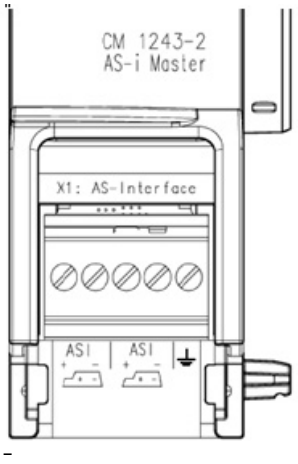
A.14.3.2 Elektrischer Anschluss des AS-i-Masters

Spannungsversorgung des AS-i-Masters CM 1243-2

Der AS-i-Master CM 1243-2 wird über den Kommunikationsbus der S7-1200 mit Spannung versorgt. Somit kann auch bei Ausfall der AS-i-Versorgungsspannung eine Diagnosemeldung an die S7-1200 gesendet werden. Der Anschluss an den Kommunikationsbus befindet sich rechts am AS-i-Master CM 1243-2.

Anschlussklemmen AS-Interface

Die abnehmbare Klemme zum Anschluss der AS-i-Leitung befindet sich hinter der unteren Abdeckklappe auf der Vorderseite des AS-i-Masters CM 1243-2.



Bei Verwendung der AS-i-Profilleitung ist die richtige Polarität der Anschlussleitung durch das Symbol ersichtlich

Informationen zum Ausbau und Wiedereinbau des Klemmenblocks finden Sie im Kapitel "Einbau" (Seite 67).

Hinweis

Maximale Belastbarkeit der Anschlusskontakte

Die Belastbarkeit der Anschlusskontakte beträgt max. 8 A. Falls dieser Wert auf der AS-i-Leitung überschritten wird, darf der AS-i-Master CM 1243-2 nicht in die AS-i-Leitung "eingeschleift" werden, sondern muss über eine Stichleitung angeschlossen werden (nur ein Anschlusspaar des AS-i-Masters CM 1243-2 belegt).
Bitte beachten Sie zudem, dass bei Stromführung durch den AS-i-Master und Strömen über 4 Ampère die verwendeten Leitungen für eine Einsatztemperatur von mindestens 75 °C geeignet sind.
Weitere Hinweise zum Anschluss der AS-i-Leitung finden Sie im Kapitel "Montage, Anschluss und Inbetriebnahme der Baugruppen" im Handbuch "AS-i Master CM 1243-2 und AS-i Datenentkopplungsmodul DCM 1271 für SIMATIC S7-1200".

Anschlussbelegung

Beschriftung	Bedeutung
ASI+	AS-i-Anschluss – positive Polarität
ASI–	AS-i-Anschluss – negative Polarität
	Funktionserde

A.14.4 RS232, RS422 und RS485

A.14.4.1 Technische Daten des CM 1241 RS485

Hinweis

Um dieses CB nutzen zu können, benötigen Sie eine CPU mit Firmware ab V2.0.

Tabelle A- 246 Allgemeine technische Daten

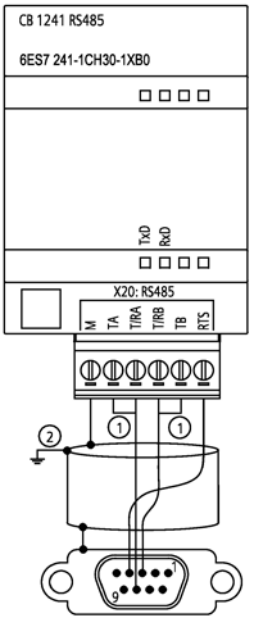
Technische Daten	CB 1241 RS485
Artikelnummer	6ES7 241-1CH30-1XB0
Abmessungen B x H x T (mm)	38 x 62 x 21
Gewicht	40 Gramm

Tabelle A- 247 Sender und Empfänger

Technische Daten	CB 1241 RS485
Typ	RS485 (2-Leiter-Halbduplex)
Gleichtaktspannungsbereich	-7 V bis +12 V, 1 s, 3 V fortlaufender Effektivwert
Differentialausgangsspannung Sender	min. 2 V bei R_L = 100 Ω min. 1,5 V bei R_L = 54 Ω
Abschluss und Bias	10 k zu +5 V an B, RS485 Pin 3 10 k zu GND an A, RS485 Pin 4
Optionaler Abschluss	Kurzschluss Pin TB mit Pin T/RB, effektive Abschlussimpedanz beträgt 127 Ω, Anschluss an RS485 Pin 3 Kurzschluss Pin TA mit Pin T/RA, effektive Abschlussimpedanz beträgt 127 Ω, Anschluss an RS485 Pin 4
Eingangsimpedanz Empfänger	min. 5,4 kΩ einschließlich Abschluss
Ansprechgrenze/Sensibilität Empfänger	min. +/- 0,2 V, 60 mV typ. Hysterese
Potentialtrennung RS485-Signal zu Masse RS485-Signal zu CPU-Logik	500 V AC für 1 Minute
Leitungslänge (geschirmt)	max. 1000 m
Baudrate	300 Baud, 600 Baud, 1,2 kBit/s, 2,4 kBit/s, 4,8 kBit/s, 9,6 kBit/s (Standard), 19,2 kBit/s, 38,4 kBit/s, 57,6 kBit/s, 76,8 kBit/s, 115,2 kBit/s,
Parität	Keine Parität (Standard), gerade, ungerade, Mark (Paritätsbit immer auf 1), Space (Paritätsbit immer auf 0)
Anzahl Stoppbits	1 (Standard), 2
Flusskontrolle	Nicht unterstützt
Wartezeit	0 bis 65535 ms

Tabelle A- 248 Stromversorgung

Technische Daten	CB 1241 RS485
Verlustleistung	1,5 W
Max. Stromaufnahme (SM-Bus)	50 mA
Max. Stromaufnahme (24 V DC)	80 mA

CB 1241 RS485 (6ES7 241-1CH30-1XB0)	
	
① "TA" und TB" wie gezeigt anschließen, um das Netzwerk abzuschließen. (Nur die Endgeräte im RS485-Netz abschließen.)	
② Verwenden Sie geschirmte, verdrillte Leiterpaare und schließen Sie den Kabelschirm an Erde an.	

Sie schließen nur die zwei Enden des RS485-Netzes ab. Die Geräte zwischen den beiden Endgeräten werden nicht abgeschlossen. Weitere Informationen finden Sie unter "Abschließen eines RS485-Busanschlusssteckers" (Seite 898).

Tabelle A- 249 Anschlussbelegungen für das CB 1241 RS485 (6ES7 241-1CH30-1XB0)

Pin	9-poliger Steckverbinder	X20
1	RS485 / logische Erde	--
2	RS485 / frei	--
3	RS485 / TxD+	3 - T/RB
4	RS485 / RTS	1 - RTS
5	RS485 / logische Erde	--
6	RS485 / 5-V-Leistung	--
7	RS485 / frei	--
8	RS485 / TxD-	4 - T/RA

Pin	9-poliger Steckverbinder	X20
9	RS485 / frei	--
Hül-se		7 - M

A.14.4.2 Technische Daten des CM 1241 RS232

Tabelle A- 250 Allgemeine technische Daten

Technische Daten	CM 1241 RS232
Artikelnummer	6ES7 241-1AH32-0XB0
Abmessungen (mm)	30 x 100 x 75
Gewicht	150 Gramm

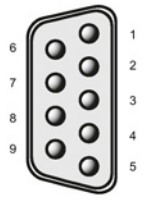
Tabelle A- 251 Sender und Empfänger

Technische Daten	CM 1241 RS232
Typ	RS232 (Vollduplex)
Ausgangsspannung Sender	min. +/- 5 V bei R_L = 3 kΩ
Ausgangsspannung Sender	max. +/- 15 V DC
Eingangsimpedanz Empfänger	min. 3 kΩ
Ansprechgrenze/Sensibilität Empfänger	min. 0,8 V Low, max. 2,4 V High, typ. Hysterese 0,5 V
Empfänger-Eingangsspannung	max. +/- 30 V DC
Potentialtrennung RS-232-Signal zu Masse RS-232-Signal zu CPU-Logik	500 V AC für 1 Minute
Leitungslänge (geschirmt)	max. 10 m
Baudrate	300 Baud, 600 Baud, 1,2 kBit/s, 2,4 kBit/s, 4,8 kBit/s, 9,6 kBit/s (Standard), 19,2 kBit/s, 38,4 kBit/s, 57,6 kBit/s, 76,8 kBit/s, 115,2 kBit/s,
Parität	Keine Parität (Standard), gerade, ungerade, Mark (Paritätsbit immer auf 1), Space (Paritätsbit immer auf 0)
Anzahl Stoppbits	1 (Standard), 2
Flusskontrolle	Hardware, Software
Wartezeit	0 bis 65535 ms

Tabelle A- 252 Stromversorgung

Technische Daten	CM 1241 RS232
Verlustleistung	1 W
aus +5 V DC	200 mA

Tabelle A- 253 RS232-Steckverbinder (Stecker)

Pin	Beschreibung	Steckverbinder (Stecker)	Pin	Beschreibung
1 DCD	Datenträgererkennung: Eingang		6 DSR	Datensatz bereit: Eingang
2 RxD	Daten von DCE empfangen: Eingang		7 RTS	Sendeanforderung: Ausgang
3 TxD	Daten an DCE gesendet: Ausgang		8 CTS	Bereit zum Senden: Eingang
4 DTR	Datenterminal bereit: Ausgang		9 RI	Rufanzeige (nicht verwendet)
5 GND	Logikmasse		SHELL	Erdungsanschluss

A.14.4.3 Technische Daten des CM 1241 RS422/485

Technische Daten des CM 1241 RS422/485

Tabelle A- 254 Allgemeine technische Daten

Technische Daten	CM 1241 RS422/485
Artikelnummer	6ES7 241-1CH32-0XB0
Abmessungen B x H x T (mm)	30 x 100 x 75
Gewicht	155 Gramm

Tabelle A- 255 Sender und Empfänger

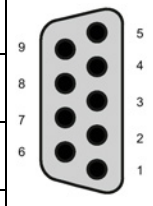
Technische Daten	CM 1241 RS422/485
Typ	RS422 oder RS485, 9-polige Sub-D-Buchse
Gleichtaktspannungsbereich	-7 V bis +12 V, 1 s, 3 V fortlaufender Effektivwert
Differentialausgangsspannung Sender	min. 2 V bei R_L = 100 Ω min. 1,5 V bei R_L = 54 Ω
Abschluss und Bias	10 kΩ zu +5 V an B, PROFIBUS Pin 3 10 kΩ zu GND an A, PROFIBUS Pin 8 Interne Bias-Optionen vorhanden oder kein interner Bias. In allen Fällen ist der externe Abschluss erforderlich, siehe Abschließen eines RS485-Busanschlusssteckers (Seite 898) und Konfigurieren von RS422 und RS485 im Systemhandbuch S7-1200 Automatisierungssystem (Seite 946).
Eingangsimpedanz Empfänger	min. 5,4 kΩ einschließlich Abschluss
Ansprechgrenze/Sensibilität Empfänger	min. +/- 0,2 V, 60 mV typ. Hysterese
Potentialtrennung RS485-Signal zu Masse RS485-Signal zu CPU-Logik	500 V AC für 1 Minute
Leitungslänge (geschirmt)	max. 1000 m (abhängig von der Baudrate)
Baudrate	300 Baud, 600 Baud, 1,2 kBit/s, 2,4 kBit/s, 4,8 kBit/s, 9,6 kBit/s (Standard), 19,2 kBit/s, 38,4 kBit/s, 57,6 kBit/s, 76,8 kBit/s, 115,2 kBit/s

Technische Daten	CM 1241 RS422/485
Parität	Keine Parität (Standard), gerade, ungerade, Mark (Paritätsbit immer auf 1), Space (Paritätsbit immer auf 0)
Anzahl Stoppbits	1 (Standard), 2
Flusskontrolle	XON/XOFF wird im RS422-Modus unterstützt
Wartezeit	0 bis 65535 ms

Tabelle A- 256 Stromversorgung

Technische Daten	CM 1241 RS422/485
Verlustleistung	1,1 W
aus +5 V DC	220 mA

Tabelle A- 257 RS485- oder RS422-Steckverbinder (Buchse)

Pin	Beschreibung	Steckverbinder (Buchse)	Pin	Beschreibung
1	Logik- oder Kommunikationsmasse		6 PWR	+5 V mit 100 Ohm Reihenwiderstand: Ausgang
2 TxD+ [1]	Angeschlossen für RS422 Nicht belegt für RS485: Ausgang		7	Nicht angeschlossen
3 TxD+	Signal B (RxD/TxD+): Eingang/Ausgang		8 TXD-	Signal A (RxD/TxD-): Eingang/Ausgang
4 RTS [2]	Ausgang Sendeanforderung/RTS (TTL-Pegel)		9 TXD- [1]	Angeschlossen für RS422 Nicht belegt für RS485: Ausgang
5 GND	Logik- oder Kommunikationsmasse		SHELL	Erdungsanschluss

[1] Die Pins 2 und 9 werden lediglich als Sendesignale bei RS422 verwendet.

[2] RTS ist ein Signal auf TTL-Ebene und kann dazu genutzt werden, ein weiteres Halbduplex-Gerät basierend auf diesem Signal zu steuern. Es ist beim Senden aktiv und ansonsten inaktiv.

A.15 TeleService (TS-Adapter und TS-Adaptermodul)

Die folgenden Handbücher enthalten die technischen Daten des TS-Adapters IE Basic und des TS-Adaptermoduls:

- Industrie-Software Engineering Tools
  Modularer TS-Adapter
- Industrie-Software Engineering Tools
  TS-Adapter IE Basic

Weitere Informationen zu diesem Produkt und die Produktdokumentation finden Sie auf der Produktkatalog-Website für den TS-Adapter (https://eb.automation.siemens.com/mall/de/de/Catalog/Search?searchTerm=TS%20Adapter%20IE%20basic&tab=).

A.16 SIMATIC Memory Cards

Artikelnummer	Kapazität
6ES7 954-8LP01-0AA0	2 GB
6ES7 954-8LL02-0AA0	256 MB
6ES7 954-8LF02-0AA0	24 MB
6ES7 954-8LE02-0AA0	12 MB
6ES7 954-8LC02-0AA0	4 MB

A.17 Eingangssimulatoren

Tabelle A- 258 Allgemeine technische Daten

Technische Daten	Simulator mit 8 Anschluss-klemmen	Simulator mit 14 An-schlussklemmen	CPU 1217C Simulator
Artikelnummer	6ES7 274-1XF30-0XA0	6ES7 274-1XH30-0XA0	6ES7 274-1XK30-0XA0
Abmessungen (B x H x T) (mm)	43 x 35 x 23	67 x 35 x 23	93 x 40 x 23
Gewicht	20 Gramm	30 Gramm	43 Gramm
Ein- und Ausgänge	8	14	14
Verwendung mit	CPU 1211C, CPU 1212C	CPU 1214C, CPU 1215C	CPU 1217C

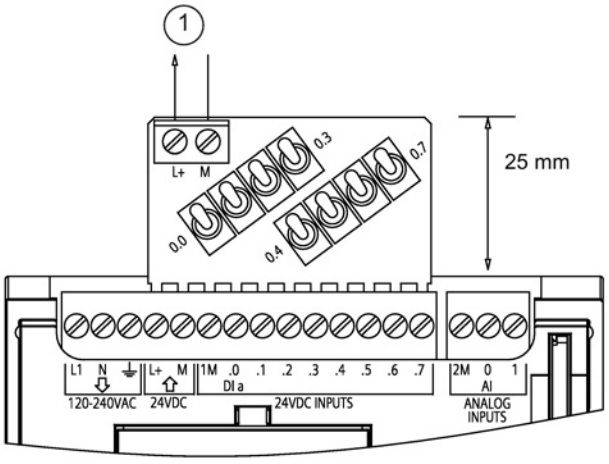
⚠WARNUNG

Sichere Verwendung von Eingangssimulatoren

Diese Eingangssimulatoren sind nicht geeignet für die Nutzung in explosionsgefährdeten Betriebsstätten der Gefahrengruppen Klasse I DIV 2 und Klasse I Zone 2. Die Schalter können bei Einsatz an Standorten der Klasse I DIV2 bzw. Klasse I Zone 2 zu Funkenbildung/Explosion führen. Nicht zugelassene Verwendung kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

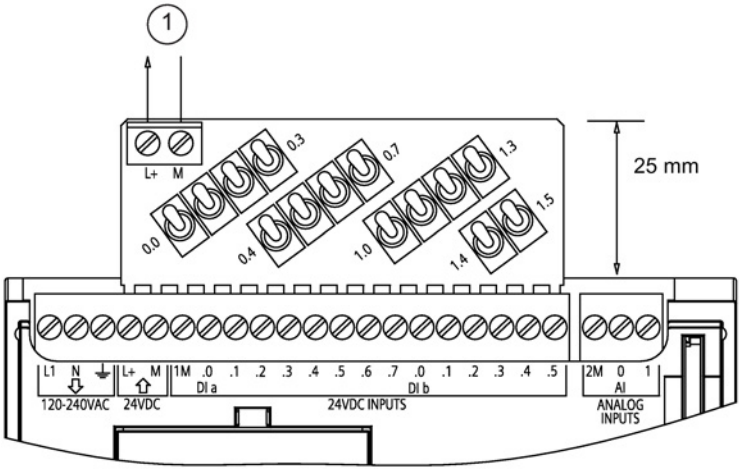
Verwenden Sie diese Eingangssimulatoren nur in Nicht-Ex-Bereichen. Verwenden Sie sie nicht in explosionsgefährdeten Betriebsstätten der Gefahrengruppen Klasse I DIV 2 und Klasse I Zone 2.

Simulator mit 8 Positionen (6ES7 274-1XF30-0XA0)



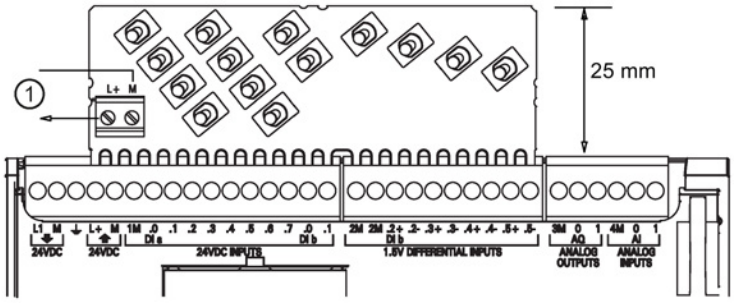
① 24-V-DC-Geberspannung

Simulator mit 14 Positionen (6ES7 274-1XF30-0XA0)



① 24-V-DC-Geberspannung

CPU 1217C Simulator (6ES7 274-1XK30-0XA0)



① 24-V-DC-Geberspannung

A.18 S7-1200 Potentiometermodul

Das S7-1200 Potentiometermodul gehört zum Zubehör für die S7-1200 CPU. Jedes Potentiometer erzeugt eine Ausgangsspannung proportional zur Position des Potentiometers, um jeden der beiden Analogeingänge der CPU mit 0 V DC bis 10 V DC zu speisen. So installieren Sie das Potentiometer:

1. Stecken Sie die "Finger" der Leiterplatte in einen beliebigen Analogeingangsklemmenblock einer S7-1200 CPU und schließen Sie eine externe DC-Spannungsversorgung an den 2-fachen Anschluss am Potentiometermodul an.
2. Mit einem kleinen Schraubendreher stellen Sie das Potentiometer ein: drehen Sie rechts herum, wenn Sie den Spannungsausgang erhöhen möchten, drehen Sie links herum, um den Spannungsausgang zu verringern.

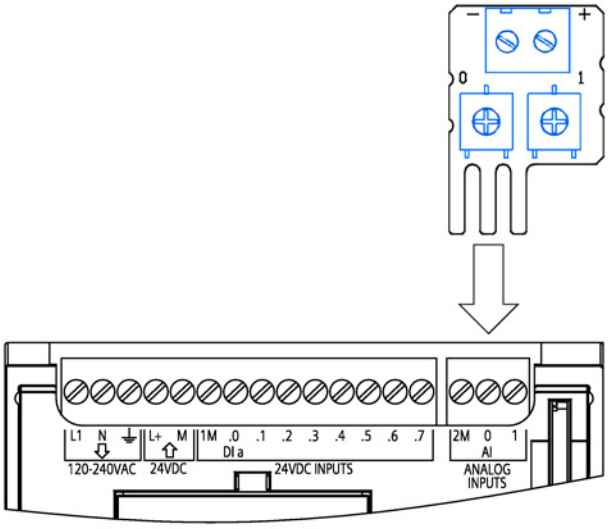
Hinweis

Befolgen Sie beim Arbeiten mit dem S7-1200 Potentiometermodul die EGB-Schutzmaßnahmen.

Technische Daten	S71200 Potentiometermodul
Artikelnummer	6ES7 274-1XA30-0XA0
Verwendung mit	Allen S71200 CPUs
Anzahl der Potentiometer	2
Abmessungen (B x H x T) (mm)	20 x 33 x 14
Gewicht	26 Gramm
Vom Anwender bereitgestellter Spannungseingang am 2-fachen Anschluss[1] (Klasse 2, leistungsbegrenzt oder Geberspannung aus PLC)	16,4 V DC bis 28,8 V DC
Leitungslänge (Meter)/Typ	< 30 m, geschirmtes, verdrilltes Leiterpaar

Technische Daten	S71200 Potentiometermodul
Eingangsstromaufnahme bei	max. 10 mA
Potentiometer-Spannungsausgabe an S7-1200 CPU-Analogeingänge[1]	min. 0 V DC bis 10,5 V DC
Potenzialtrennung	Nicht elektrisch getrennt
Umgebungstemperaturbereich	-20 °C bis 60 °C

[1] Die Stabilität der Ausgangsspannung des Potentiometermoduls hängt von der Qualität des vom Anwender bereitgestellten Spannungseingangs am 2-fachen Anschluss ab - kann als Analogeingangsspannung betrachtet werden.

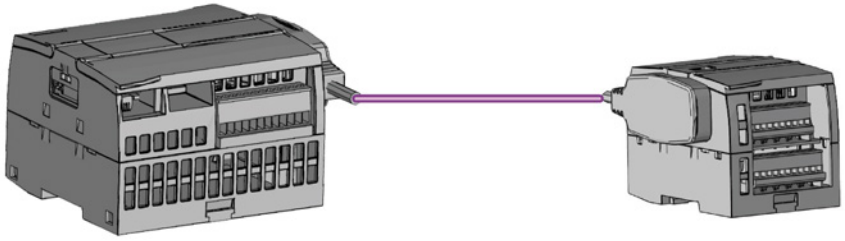


A.19 Steckleitung für Erweiterungsmodule

Tabelle A- 259 Erweiterungskabel

Technische Daten	
Artikelnummer	6ES7 290-6AA30-0XA0
Leitungslänge	2 m
Gewicht	200 g

Weitere Informationen zum Einbauen und Ausbauen des S7-1200 Erweiterungskabels finden Sie im Abschnitt "Einbau" (Seite 68).



A.20 Zugehörige Produkte

A.20.1 PM 1207 Stromversorgungsmodul

Das PM 1207 ist ein Stromversorgungsmodul für die SIMATIC S7-1200. Es bietet die folgenden Leistungsmerkmale:

- 120/230-V-AC-Eingang, 24-V-DC/2,5-A-Ausgang
- Artikelnummer 6ESP 332-1SH71-4AA0

Weitere Informationen zu diesem Produkt und die Produktdokumentation finden Sie auf der Website mit dem Produktkatalog zum PM 1207 (https://eb.automation.siemens.com/mall/de/de/Catalog/Product/6AG1332-1SH71-4AA0).

A.20.2 CSM 1277 Compact Switch Module

Das CSM1277 ist ein Compact Switch Module für Industrial Ethernet. Es kann zur Multiplikation der Ethernet-Schnittstelle der S7-1200 eingesetzt werden, um die gleichzeitige Kommunikation mit Bedienpanels, Programmiergeräten oder anderen Steuerungen zu ermöglichen. Das Modul bietet die folgenden Leistungsmerkmale:

- 4 x RJ45-Buchsen für den Anschluss an Industrial Ethernet
- 3-polige Klemmenleiste für den Anschluss der externen 24-V-DC-Versorgung von oben
- LEDs für Diagnose- und Statusanzeige von Industrial Ethernet-Anschlüssen
- Artikelnummer 6GK7 277-1AA00-0AA0

Weitere Informationen zu diesem Produkt und die Produktdokumentation finden Sie auf der Website mit dem Produktkatalog zum CSM 1277 (https://eb.automation.siemens.com/mall/de/de/Catalog/Search?searchTerm=csm%201277&tab=).

A.20.3 CM CANopen-Modul

Das CM CANopen -Modul ist ein steckbares Modul zwischen dem SIMATIC S7-1200 PLC und einem Gerät, auf dem CANopen ausgeführt wird. Das CM CANopen -Modul kann als Master oder als Slave konfiguriert werden. Es gibt zwei CM CANopen modules: das CANopen-Modul (Artikelnummer 021620-B) und das für höhere Belastung ausgelegte CANopen (Ruggedized) -Modul (Artikelnummer 021730-B).

Das CANopen-Modul bietet die folgenden Leistungsmerkmale:

- Anschlussmöglichkeit von 3 Modulen je CPU
- Anschluss von max. 16 CANopen-Slaveknoten
- 256 Byte Eingang und 256 Byte Ausgang je Modul
- 3 LEDs liefern Diagnoseinformationen zu Modul, Netzwerk und E/A-Zustand
- Unterstützt die Speicherung der CANopen-Netzwerkkonfiguration im PLC
- Das Modul ist in den Hardwarekatalog der Konfigurationssuite im TIA Portal integrierbar
- CANopen-Konfiguration über CANopen Configuration Studio (im Lieferumfang enthalten) oder jedes andere externe CANopen -Konfigurationswerkzeug
- Erfüllt die CANopen -Kommunikationsprofile CiA 301 Rev. 4.2 und CiA 302 Rev. 4.1
- Unterstützt transparentes CAN 2.0A für die Verarbeitung benutzerdefinierter Protokolle
- Vorgefertigte Funktionsbausteine für jede PLC-Programmierung im TIA Portal verfügbar
- Im Lieferumfang der CM CANopen -Module enthalten: Sub-D-Steckverbinder mit Schraubklemmen für Subnetze, CM CANopen Configuration Studio CD und USB-Konfigurationskabel.

Weitere Informationen zu diesem Produkt und die Produktdokumentation finden Sie auf der Website mit dem Produktkatalog zum CM CANopen.

A.20.4 Kommunikationsmodul RF120C

Über das Kommunikationsmodul RF120C können RFID- und Codelesesysteme von Siemens mühelos und direkt mit der S7-1200 verbunden werden. Das Lesegerät wird über eine Punkt-zu-Punkt-Verbindung an das RF120C angeschlossen. An eine S7-1200 können links von der CPU bis zu drei Kommunikationsmodule angeschlossen werden. Das Kommunikationsmodul RF120C wird über das TIA Portal konfiguriert. Die Artikelnummer für das Kommunikationsmodul RF120C lautet 6GT2002-0LA00.

Weitere Informationen zu diesem Produkt und die Produktdokumentation finden Sie auf der Webseite mit dem Produktkatalog zum RF120C.

Berechnen der Leistungsbilanz

B

Die CPU besitzt eine interne Spannungsversorgung, die neben der CPU selbst die Erweiterungsmodule und andere 24VDCVerbraucher versorgt.

Es gibt vier Arten von Erweiterungsmodulen:

- Signalmodule (SM) werden an der rechten Seite der CPU angeschlossen. Für jede CPU gibt es eine maximal anschließbare Anzahl Signalmodule, unabhängig von der Leistungsbilanz.
  - An die CPU 1214C, CPU 1215C und CPU 1217C können 8 Signalmodule angeschlossen werden.
  - An die CPU 1212C können 2 Signalmodule angeschlossen werden.
  - An die CPU 1211C können keine Signalmodule angeschlossen werden.
- Kommunikationsmodule (CM) werden an der linken Seite der CPU angeschlossen. An jede CPU können maximal 3 Kommunikationsmodule angeschlossen werden, unabhängig von der Leistungsbilanz.
- Signalboards (SB), Kommunikationsboards (CB) und Batterieboards (BB) werden oben auf der CPU angeschlossen. An jede CPU kann maximal 1 Signalboard, Kommunikationsboard oder Batterieboard angeschlossen werden.

Mit Hilfe der folgenden Informationen können Sie berechnen, wieviel Leistung die CPU für Ihre Konfiguration zur Verfügung stellen kann.

Jede CPU liefert Gleichspannung von 5 V und 24 V:

- Die CPU liefert 5VGleichspannung für eventuell angeschlossene Erweiterungsmodule. Ist der Leistungsbedarf der Erweiterungsmodule an 5VGleichspannung höher, als die interne Spannungsversorgung liefern kann, müssen Sie auf Erweiterungsmodule verzichten, bis die Leistungsbilanz der S7200 wieder eingehalten wird.
- Jede CPU besitzt eine 24VDCGeberversorgung, die die integrierten Eingänge und die Relaisspulen der Erweiterungsmodule mit 24VGleichspannung versorgt. Ist der Leistungsbedarf der CPU an 24VGleichspannung höher, als die interne Spannungsversorgung liefern kann, können Sie zusätzlich eine externe 24VDC-Versorgung anschließen, die die Erweiterungsmodule speist. Sie müssen die 24VDC-Versorgung manuell an die Eingänge und die Relaisspulen anschließen.

⚠WARNUNG

Wenn Sie parallel zur DCGeberversorgung eine externe 24VDCSpannungsquelle anschließen, kann es sein, dass die beiden Spannungsquellen sich beim Aufbauen der geeigneten Ausgangsspannung gegenseitig beeinträchtigen.

Als Folge kann sich die Lebensdauer verkürzen bzw. eine oder beide Spannungsquellen können sofort ausfallen. Unvorhersehbarer Betrieb kann zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Die DC-Geberversorgung der CPU und eine externe Spannungsquelle müssen die Spannung an unterschiedlichen Punkten liefern. Dabei darf maximal eine Verbindung zwischen den beiden Spannungsquellen bestehen.

Einige der 24-V-DC-Eingangsports des PLC-Systems sind miteinander verbunden, wobei ein logischer Bezugsleiter mehrere M-Klemmen verbindet. Der 24-V-DC-Versorgungseingang der CPU, der Relaisspuleneingang des SM und ein nicht potenzialgetrennter analoger Versorgungseingang sind Beispiele für Stromkreise, die miteinander verbunden sind, sofern sie in den Datenblättern als nicht potenzialgetrennt angegeben sind. Alle nicht potentialgetrennten M-Klemmen müssen an dasselbe externe Bezugspotential angeschlossen werden.

⚠WARNUNG

Wenn Sie nicht potentialgetrennte M-Klemmen an verschiedene Bezugspotentiale anschließen, verursacht dies unbeabsichtigten Stromfluss, der zu Beschädigung oder unvorhersehbarem Betrieb des Zielsystems und angeschlossener Geräte führen kann.

Beschädigung oder unvorhersehbarer Betrieb können zu tödlichen oder schweren Verletzungen und/oder Sachschaden führen.

Schließen Sie stets alle nicht potentialgetrennten M-Klemmen in einem PLC-System an dasselbe Bezugspotential an.

Informationen zu den Leistungsbilanzen der CPUs und dem Leistungsbedarf der Signalmodule finden Sie in den technischen Daten (Seite 1151).

Hinweis

Wird die Leistungsbilanz der CPU überschritten, können Sie evtl. nicht die maximale Anzahl Module an Ihre CPU anschließen.

Beispiel für eine Leistungsbilanz

Im folgenden Beispiel wird der Leistungsbedarf berechnet für eine Konfiguration mit einer CPU 1214C AC/DC/Relais, einem SB 1223 2 x 24 V DC Eingang/ 2 x 24 V DC Ausgang, einem CM 1241, drei SM 1223 8 DC-Eingänge/8 Relaisausgänge und einem SM 1221 8 DC Eingang. Dieses Beispiel umfasst insgesamt 48 Eingänge und 36 Ausgänge.

Hinweis

Die CPU hat den für die internen Relaisspulen benötigten Leistungsbedarf bereits zugeordnet. Sie brauchen den Leistungsbedarf der internen Relaisspulen nicht in die Berechnung der Leistungsbilanz aufzunehmen.

Die CPU in diesem Beispiel liefert genügend 5-V-DC-Spannung für die Signalmodule, doch die Geberversorgung liefert nicht genügend 24-V-DC-Spannung für alle Eingänge und Erweiterungsrelais. Die Ein- und Ausgänge benötigen 456 mA und die CPU liefert nur 400 mA. Dieser Aufbau benötigt zusätzlich mindestens 56 mA an 24-V-DC-Spannung, um alle 24-V-DC-Eingänge und -Ausgänge zu versorgen.

Tabelle B- 1 Beispiel für eine Leistungsbilanz

Leistungsbilanz der CPU	**5 V DC**	**24 V DC**
CPU 1214C AC/DC/Relais	1600 mA	400 mA
Minus		
Systemvoraussetzungen	**5 V DC**	**24 V DC**
CPU 1214C, 14 Eingänge	-	14 * 4 mA = 56 mA
1 SB 1223 2 x 24 VDC Eingang/ 2 x 24 VDC Ausgang	50 mA	2 * 4 mA = 8 mA
1 CM 1241 RS422/485, 5 V	220 mA	
3 SM 1223, 5-V-Spannung	3 * 145 mA = 435 mA	-
1 SM 1221, 5-V-Spannung	1 * 105 mA = 105 mA	-
3 SM 1223, je 8 Eingänge	-	3 * 8 * 4 mA = 96 mA
3 SM 1223, je 8 Relaisausgänge	-	3 * 8 * 11 mA = 264 mA
1 SM 1221, je 8 Eingänge	-	8 * 4 mA = 32 mA
Gesamtbedarf	810 mA	456 mA
Gleich		
Leistungsbilanz	**5 V DC**	**24 V DC**
GesamtLeistungsbilanz	790 mA	(56 mA)

Formular zum Berechnen der Leistungsbilanz

Mit Hilfe dieser Tabelle können Sie berechnen, wieviel Leistung die S7-1200 CPU für Ihre Konfiguration liefern kann. Die technischen Daten (Seite 1151) bieten Informationen zu der verfügbaren Leistung der einzelnen Ausführungen der CPUs und zum Leistungsbedarf Ihrer Signalmodule.

Tabelle B- 2 Berechnen der Leistungsbilanz

Leistungsbilanz der CPU	**5 V DC**	**24 V DC**
Minus		
Systemvoraussetzungen	**5 V DC**	**24 V DC**
Gesamtbedarf		
Gleich		
Leistungsbilanz	**5 V DC**	**24 V DC**
GesamtLeistungsbilanz		

Bestellinformationen

C

C.1 CPU-Module

Tabelle C- 1 S71200 CPUs

CPU-Varianten		Artikelnummer
CPU 1211C	CPU 1211C DC/DC/DC	6ES7 211-1AE40-0XB0
	CPU 1211C AC/DC/Relais	6ES7 211-1BE40-0XB0
	CPU 1211C DC/DC/Relais	6ES7 211-1HE40-0XB0
CPU 1212C	CPU 1212C DC/DC/DC	6ES7 212-1AE40-0XB0
	CPU 1212C AC/DC/Relais	6ES7 212-1BE40-0XB0
	CPU 1212C DC/DC/Relais	6ES7 212-1HE40-0XB0
CPU 1214C	CPU 1214C DC/DC/DC	6ES7 214-1AG40-0XB0
	CPU 1214C AC/DC/Relais	6ES7 214-1BG40-0XB0
	CPU 1214C DC/DC/Relais	6ES7 214-1HG40-0XB0
CPU 1215C	CPU 1215C DC/DC/DC	6ES7 215-1AG40-0XB0
	CPU 1215C AC/DC/Relais	6ES7 215-1BG40-0XB0
	CPU 1215C DC/DC/Relais	6ES7 215-1HG40-0XB0
CPU 1217C	CPU 1217C DC/DC/DC	6ES7 217-1AG40-0XB0

C.2 Signalmodule (SMs), Signalboards (SBs) und Batterieboards (BBs)

Tabelle C- 2 Signalmodule (SMs)

Signalmodule		Artikelnummer
Digitaleingang	SM 1221 8 x 24-V-DC-Eingang (stromziehend/stromliefernd)	6ES7 221-1BF32-0XB0
	SM 1221 16 x 24-V-DC-Eingang (stromziehend/stromliefernd)	6ES7 221-1BH32-0XB0
Digitalausgang	SM 1222 8 x 24-V-DC-Ausgang (stromliefernd)	6ES7 222-1BF32-0XB0
	SM 1222 16 x 24-V-DC-Ausgang (stromliefernd)	6ES7 222-1BH32-0XB0
	SM 1222 8 x Relaisausgang	6ES7 222-1HF32-0XB0
	SM 1222 8 x Relaisausgang (Umschaltung)	6ES7 222-1XF32-0XB0
	SM 1222 16 x Relaisausgang	6ES7 222-1HH32-0XB0
Digitaleingänge/-ausgänge	SM 1223 8 x 24-V-DC-Eingang (stromziehend/stromliefernd) / 8 x 24-V-DC-Ausgang (stromliefernd)	6ES7 223-1BH32-0XB0
	SM 1223 16 x 24-V-DC-Eingang (stromziehend/stromliefernd) / 16 x 24-V-DC-Ausgang (stromliefernd)	6ES7 223-1BL32-0XB0
	SM 1223 8 x 24-V-DC-Eingang (stromziehend/stromliefernd) / 8 x Relaisausgang	6ES7 223-1PH32--0XB0
	SM 1223 16 x 24-V-DC-Eingang (stromziehend/stromliefernd) / 16 x Relaisausgang	6ES7 223-1PL32-0XB0
	SM 1223 8 x 120/230-V-AC-Eingang (stromziehend/stromliefernd) / 8 x Relaisausgang	6ES7 223-1QH32-0XB0
Analogeingang	SM 1231 4 x Analogeingang	6ES7 231-4HD32-0XB0
	SM 1231 8 x Analogeingang	6ES7 231-4HF32-0XB0
	SM 1231 4 x Analogeingang x 16 Bit (hochfunktionell)	6ES7 231-5ND32-0XB0
Analogausgang	SM 1232 2 x Analogausgang	6ES7 232-4HB32-0XB0
	SM 1232 4 x Analogausgang	6ES7 232-4HD32-0XB0
Analogeingänge/-ausgänge	SM 1234 4 x Analogeingang / 2 x Analogausgang	6ES7 234-4HE32-0XB0
RTD und Thermoelement	SM 1231 TC 4 x 16 Bit	6ES7 231-5QD32-0XB0
	SM 1231 TC 8 x 16 Bit	6ES7 231-5QF32-0XB0
	SM 1231 RTD 4 x 16 Bit	6ES7 231-5PD32-0XB0
	SM 1231 RTD 8 x 16 Bit	6ES7 231-5PF32-0XB0
Technologiemodule	SM 1278 4xIO-Link-Master	6ES7 278-4BD32-0XB0

Tabelle C- 3 Signalboards (SBs) und Batterieboards (BBs)

Signal- und Batterieboards		Artikelnummer
Digitaleingang	SB 1221 200 kHz, 4 x 24-V-DC-Eingang (stromliefernd)	6ES7 221-3BD30-0XB0
	SB 1221 200 kHz, 4 x 5-V-DC-Eingang (stromliefernd)	6ES7 221-3AD30-0XB0
Digitalausgang	SB 1222 200 kHz, 4 x 24-V-DC-Ausgang (stromziehend/stromliefernd)	6ES7 222-1BD30-0XB0
	SB 1222 200 kHz, 4 x 5-V-DC-Ausgang (stromziehend/stromliefernd)	6ES7 222-1AD30-0XB0
Digitaleingänge/-ausgänge	SB 1223 2 x 24-V-DC-Eingang (stromziehend) / 2 x 24-V-DC-Ausgang (stromliefernd)	6ES7 223-0BD30-0XB0
	SB 1223 200 kHz, 2 x 24-V-DC-Eingang (stromliefernd) / 2 x 24-V-DC-Ausgang (stromziehend/stromliefernd)	6ES7 223-3BD30-0XB0
	SB 1223 200 kHz, 2 x 5-V-DC-Eingang (stromliefernd) / 2 x 5-V-DC-Ausgang (stromziehend/stromliefernd)	6ES7 223-3AD30-0XB0
Analog	SB 1232 1 Analogausgang	6ES7 232-4HA30-0XB0
	SB 1231 1 Analogeingang	6ES7 231-4HA30-0XB0
	SB 1231 1 Analogeingang Thermoelement	6ES7 231-5QA30-0XB0
	SB 1231 1 Analogeingang RTD	6ES7 231-5PA30-0XB0
Batterie	BB 1297 Batterieboard (Batterietyp CR1025 nicht enthalten)	6ES7 297-0AX30-0XA0

C.3 Kommunikation

Tabelle C- 4 Kommunikationsmodul (CM)

Kommunikationsmodul (CM)			Artikelnummer
RS232, RS422 und RS485	CM 1241 RS232	RS232	6ES7 241-1AH32-0XB0
	CM 1241 RS422/485	RS422/485	6ES7 241-1CH32-0XB0
PROFIBUS	CM 1243-5	PROFIBUS-Master	6GK7 243-5DX30-0XE0
	CM 1242-5	PROFIBUS-Slave	6GK7 242-5DX30-0XE0
AS-i-Master	CM 1243-2	AS-i-Master	3RK7 243-2AA30-0XB0
RF120C CM	RF120C	RFID-Leser	6GT2 002-0LA00

Tabelle C- 5 Kommunikationsboard (CB)

Kommunikationsboard (CB)			Artikelnummer
RS485	CB 1241 RS485	RS485	6ES7 241-1CH30-1XB0

Tabelle C- 6 Kommunikationsprozessor (CP)

CP	Schnittstelle	Artikelnummer
CP 1242-7 GPRS V2	GPRS	6GK7 242-7KX31-0XE0
CP 1243-7 LTE-EU	LTE	6GK7 243-7KX30-0XE0
CP 1243-1 DNP3	IE-Schnittstelle	6GK7 243-1JX30-0XE0
CP 1243-1 IEC	IE-Schnittstelle	6GK7 243-1PX30-0XE0
CP 1243-1	IE-Schnittstelle	6GK7 243-1BX30-0XE0
CP 1243-1 PCC	IE-Schnittstelle	6GK7 243-1HX30-0XE0
CP 1243-8 ST7	IE- und serielle Schnittstelle	6GK7 243-8RX30-0XE0

Tabelle C- 7 TeleService

TS-Adapter	Artikelnummer
TS-Adapter IE Basic	6ES7 972-0EB00-0XA0
TS Adapter IE Advanced	6ES7 972-0EA00-0XA0
TS-Modul GSM	6GK7 972-0MG00-0XA0
TS-Modul RS232	6ES7 792-0MS00-0XA0
TS-Modul Modem	6ES7 972-0MM00-0XA0
TS-Modul ISDN	6ES7 972-0MD00-0XA0

Tabelle C- 8 Zubehör

Zubehör			Artikelnummer
Antenne	ANT794-4MR	GSM/GPRS-Antenne	6NH9 860-1AA00
	ANT794-3M	Flachantenne	6NH9 870-1AA00

Tabelle C- 9 Steckverbinder

Art des Steckverbinders		Artikelnummer
RS485	Schraubklemmenanschluss, 35-Grad-Kabelabgang	6ES7 972-0BA42-0XA0
	FastConnect-Anschluss, 35-Grad-Kabelabgang	6ES7 972-0BA60-0XA0

C.4 Fehlersichere CPUs und Signalmodule

Tabelle C- 10 Fehlersichere CPUs

Fehlersichere CPU-Modelle		Artikelnummer
CPU 1214FC	CPU 1214FC DC/DC/DC	6ES7 214-1AF40-0XB0
	CPU 1214FC DC/DC/Relais	6ES7 214-1HF40-0XB0
CPU 1215FC	CPU 1215FC DC/DC/DC	6ES7 215-1AF40-0XB0
	CPU 1215FC DC/DC/Relais	6ES7 215-1HF40-0XB0

Tabelle C- 11 Fehlersichere Signalmodule

Signalmodule mit funktionaler Sicherheit		Artikelnummer
Digitaleingang	SM 1226 F-DI 16 x 24 VDC	6ES7 226-6BA32-0XB0
Digitalausgang	SM 1226 F-DQ 4 x 24 VDC	6ES7 226-6DA32-0XB0
	SM 1226 F-DQ 2 x Relay	6ES7 226-6RA32-0XB0

C.5 Sonstige Module

Tabelle C- 12 Zugehörige Produkte

Beschreibung		Artikelnummer
Stromversorgung	PM 1207 Stromversorgung	6EP1 332-1SH71-4AA0
Ethernet-Switch	CSM 1277 Ethernet-Switch - 4 Ports	6GK7 277-1AA10-0AA0
CM CANopen	CANopen für SIMATIC S7-1200	021620-B
	CANopen (Ruggedized) für SIMATIC S7-1200	021730-B
RF120C	Kommunikationsmodul RF120C	6GT2002-0LA00

C.6 Memory Cards

Tabelle C- 13 Memory Cards

SIMATIC Memory Cards	Artikelnummer
SIMATIC MC 2 GB	6ES7 954-8LP01-0AA0
SIMATIC MC 256 MB	6ES7 954-8LL02-0AA0
SIMATIC MC 24 MB	6ES7 954-8LF02-0AA0
SIMATIC MC 12 MB	6ES7 954-8LE02-0AA0
SIMATIC MC 4 MB	6ES7 954-8LC02-0AA0

C.7 Grundlegende HMI-Geräte

Tabelle C- 14 HMI-Geräte

Grundlegende HMI-Panels	Artikelnummer
KTP400 Basic (Mono, PN)	6AV2 123-2DB03-0AX0
KTP700 Basic	6AV2 123-2GB03-0AX0
KTP700 Basic DP	6AV2 123-2GA03-0AX0
KTP900 Basic	6AV2 123-2JB03-0AX0
KTP1200 Basic	6AV2 123-2MB03-0AX0
KTP1200 Basic DP	6AV2 123-2MA03-0AX0

C.8 Ersatzteile und sonstige Hardware

Tabelle C- 15 Erweiterungskabel, Simulatoren und Endhalter

Beschreibung		Artikelnummer
Steckleitung für Erweiterungsmodule	Steckleitung für Erweiterungsmodule, 2 m	6ES7 290-6AA30-0XA0
E/A-Simulator	Simulator (1211C/1212C - 8 E/A)	6ES7 274-1XF30-0XA0
	Simulator (1214C/1215C - 14 E/A)	6ES7 274-1XH30-0XA0
	Simulator CPU 1217C	6ES7 274-1XK30-0XA0
Potentiometermodul	S71200 Potentiometermodul	6ES7 274-1XA30-0XA0
Ethernet-Zugentlastung	Zugentlastung ein RJ45-Anschluss	6ES7 290-3AA30-0XA0
	Zugentlastung zwei RJ45-Anschlüsse	6ES7 290-3AB30-0XA0
Ersatzabdeckklappe	CPU 1211C/1212C	6ES7 291-1AA30-0XA0
	CPU 1214C	6ES7 291-1AB30-0XA0
	CPU 1215C	6ES7 291-1AC30-0XA0
	CPU 1217C	6ES7 291-1AD30-0XA0
	Signalmodul (SM), 45 mm	6ES7 291-1BA30-0XA0
	Signalmodul (SM), 70 mm	6ES7 291-1BB30-0XA0
	Kommunikationmodul (zur Verwendung mit Modulen 6ES7 2xx-xxx32-0XB0 und 6ES7 2xx-xxx30-0XB0)	6ES7 291-1CC30-0XA0
Endhalter	Endhalter Thermoplastik, 10 mm	8WA1808
	Endhalter Stahl, 10,3 mm	8WA1805

Tabelle C- 16 S7-1200 CPU ab V4 - Ersatzklemmenblöcke

Sie haben eine S7-1200 CPU ab V4.0 (Artikelnummer)	Dann verwenden Sie diese Ersatzklemmenblöcke (4/Packung)	
	Artikelnummer Klemmenblöcke	Beschreibung der Klemmenblöcke
CPU 1211C DC/DC/DC (6ES7 211-1AE40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AH30-0XA0	8-polig, verzinnt
	6ES7 292-1AP30-0XA0	14-polig, verzinnt
CPU 1211C DC/DC/Relais (6ES7 211-1HE40-0XB0)	6ES7 292-1AH30-0XA0	3-polig, vergoldet
	6ES7 292-1AH40-0XA0	8-polig, verzinnt, codiert
	6ES7 292-1AP30-0XA0	14-polig, verzinnt
CPU 1211C AC/DC/Relais (6ES7 211-1BE40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AH40-0XA0	8-polig, verzinnt, codiert
	6ES7 292-1AP40-0XA0	14-polig, verzinnt, codiert
CPU 1212C DC/DC/DC (6ES7212-1AE40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AH30-0XA0	8-polig, verzinnt
	6ES7 292-1AP30-0XA0	14-polig, verzinnt
CPU 1212C DC/DC/Relais (6ES7 212-1HE40-0XB0)	6ES7 292-1AH30-0XA0	3-polig, vergoldet
	6ES7292-1AH40-0XA0	8-polig, verzinnt, codiert
	6ES7 292-1AP30-0XA0	14-polig, verzinnt
CPU 1212C AC/DC/Relais (6ES7 212-1BE40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AH40-0XA0	8-polig, verzinnt, codiert
	6ES7 292-1AP40-0XA0	14-polig, verzinnt, codiert
CPU 1214C DC/DC/DC (6ES7 214-1AG40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AM30-0XA0	12-polig, verzinnt
	6ES7 292-1AV30-0XA0	20-polig, verzinnt
CPU 1214C DC/DC/Relais (6ES7 214-1HG40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AM40-0XA0	12-polig, verzinnt, codiert
	6ES7 292-1AV30-0XA0	20-polig, verzinnt
CPU 1214C AC/DC/Relais (6ES7 214-1BG40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AM40-0XA0	12-polig, verzinnt, codiert
	6ES7 292-1AV40-0XA0	20-polig, verzinnt, codiert
CPU 1215C DC/DC/DC (6ES7 215-1AG40-0XB0)	6ES7 292-1BF30-0XB0	6-polig, vergoldet
CPU 1215C DC/DC/Relais (6ES7 215-1HG40-0XB0)	6ES7 292-1BF30-0XB0	6-polig, vergoldet
	6ES7 292-1AM40-0XA0	12-polig, verzinnt, codiert
	6ES7 292-1AV30-0XA0	20-polig, verzinnt
CPU 1215C AC/DC/Relais (6ES7 215-1BG40-0XB0)	6ES7 292-1BF30-0XB0	6-polig, vergoldet
	6ES7 292-1AM40-0XA0	12-polig, verzinnt, codiert
	6ES7 292-1AV40-0XA0	20-polig, verzinnt, codiert
CPU 1217C DC/DC/DC (6ES7 217-1AG40-0XB0)	6ES7 292-1BF30-0XB0	6-polig, vergoldet
	6ES7 292-1AK30-0XA0	10-polig, verzinnt
	6ES7 292-1AR30-0XA0	16-polig, verzinnt
	6ES7 292-1AT30-0XA0	18-polig, verzinnt

Tabelle C- 17 S7-1200 SMs ab V4.0 - Ersatzklemmenblöcke

Sie haben ein S7-1200 SM ab V4.0 (Artikelnummer)	Dann verwenden Sie diese Ersatzklemmenblöcke (4/Packung)	
	Artikelnummer Klemmenblöcke	Beschreibung der Klemmenblöcke
SM1221 DI 8xDC (6ES7 221-1BF32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 8xDC (6ES7 222-1BF32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 8xRelais (6ES7 222-1HF32-0XB0)	6ES7 292-1AG40-0XA1	7-polig, verzinnt, links codiert
SM1231 AI 4x13 Bit (6ES7 231-4HD32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1232 AO 2x14 Bit (6ES7 232-4HB32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1231 AI 4xTC (6ES7 231-5QD32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1231 AI 4x16 Bit (6ES7 231-5ND32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1221 DI 16xDC (6ES7 221-1BH32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 16xDC (6ES7 222-1BH32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 16xRelais (6ES7 222-1HH32-0XB0)	6ES7 292-1AG40-0XA0	7-polig, verzinnt, rechts codiert
SM1223 DI 8xDC/DO 8xDC (6ES7 223-1BH32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1223 8xDC/8xRelais (6ES7 223-1PH32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
	6ES7 292-1AG40-0XA0	7-polig, verzinnt, rechts codiert
SM1223 8xAC/8xRelais (6ES7 223-1QH32-0XB0)	6ES7 292-1AG40-0XA0	7-polig, verzinnt, rechts codiert
SM1234 AI 4 / AO 2 (6ES7 234-4HE32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1231 AI 8x13 BIT (6ES7 231-4HF32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1232 AO 4x14 Bit (6ES7 232-4HD32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1231 AI 4xRTD (6ES7 231-5PD32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1231 AI 8xTC (6ES7 231-5QF32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM 1278 IO-LINK (6ES7 278-4BD32 0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 8xRelais (Umschaltung) (6ES7 222-1XF32-0XB0)	6ES7 292-1AL30-0XA0	11-polig, verzinnt
SM1223 DI 16xDC/DO 16xDC (6ES7 223-1BL32-0XB0)	6ES7 292-1AL30-0XA0	11-polig, verzinnt
	6ES7 292-1AL40-0XA0	11-polig, verzinnt, codiert
SM1231 AI 8xRTD (6ES7 231-5PF32-0XB0)	6ES7 231-5PF32-0XB0	11-polig, vergoldet

Tabelle C- 18 Fehlersichere CPUs - Ersatzklemmenblöcke

Wenn Sie eine fehlersichere CPU haben (Artikelnummer)	Dann verwenden Sie diese Ersatzklemmenblöcke (4/Packung)	
	Artikelnummer Klemmenblöcke	Beschreibung der Klemmenblöcke
CPU 1214FC DC/DC/DC (6ES7 214-1AF40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AM30-0XA0	12-polig, verzinnt
	6ES7 292-1AV30-0XA0	20-polig, verzinnt
CPU 1214FC DC/DC/Relais (6ES7 214-1HF40-0XB0)	6ES7 292-1BC30-0XA0	3-polig, vergoldet
	6ES7 292-1AM40-0XA0	12-polig, verzinnt, codiert
	6ES7 292-1AV30-0XA0	20-polig, verzinnt
CPU 1215FC DC/DC/DC (6ES7 215 1AF40 0XB0)	6ES7 292-1BF30-0XB0	6-polig, vergoldet
	6ES7 292-1AM30-0XA0	12-polig, verzinnt
	6ES7 292-1AV30-0XA0	20-polig, verzinnt
CPU 1215FC DC/DC/Relais (6ES7 215 1HF40 0XB0)	6ES7 292-1BF30-0XB01	6-polig, vergoldet
	6ES7 292-1AM40-0XA0	2-polig, verzinnt, codiert
	6ES7 292-1AV30-0XA0	20-polig, verzinnt

Tabelle C- 19 Fehlersichere Signalmodule - Ersatzklemmenblöcke

Wenn Sie ein fehlersicheres Signalmodul haben (Artikelnummer)	Dann verwenden Sie diese Ersatzklemmenblöcke (4/Packung)	
	Artikelnummer Klemmenblöcke	Beschreibung der Klemmenblöcke
SM 1226 F-DI (6ES7 226-6BA32-0XB0)	6ES7 292-1AL30-0XA0	11-polig, verzinnt
SM 1226 F-DO (6ES7 226-6DA32-0XB0)	6ES7 292-1AL30-0XA0	11-polig, verzinnt
SM 1226 F-Relais (6ES7 226-6RA32-0XB0)	6ES7 292-1AL40-0XA0	11-polig, verzinnt, codiert

C.9 Programmiersoftware

Tabelle C- 20 Programmiersoftware

SIMATIC-Software		Artikelnummer
Programmiersoftware	STEP 7 Basic V13	6ES7 822-0AA01-0YA0
	STEP 7 Professional V13	6ES7 822-1AA01-0YA5
Visualisierungssoftware	WinCC Basic V12 SP1	6AV2100-0AA01-0AA0
	WinCC Comfort V12 SP1	6AV2101-0AA01-0AA5
	WinCC Advanced V12 SP1	6AV2102-0AA01-0AA5
	WinCC Professional 512 PowerTags V12 SP1	6AV2103-0DA01-0AA5
	WinCC Professional 4096 PowerTags V12 SP1	6AV2103-0HA01-0AA5
	WinCC Professional max. PowerTags V12 SP1	6AV2103-0XA01-0AA5

Geräteaustausch und Ersatzteilkompatibilität

D

Sie können Ihre V3.0-CPU durch eine V4.1-CPU ersetzen (Seite 168) und Ihr bestehendes, für die V3.0-CPU entwickeltes STEP 7-Projekt verwenden. Eine V3.0-CPU kann nicht mit einem Firmware-Update auf eine V4.1-CPU hochgerüstet werden; die Hardware muss ausgetauscht werden. Wenn Sie eine V3.0-CPU durch eine V4.1-CPU austauschen, können Sie gleichzeitig nach Firmware-Updates (Seite 146) für Ihre Signal- und Kommunikationsmodule suchen und diese installieren.

Hinweis

Kein Gerätetausch möglich von V4.1 nach V3.0

Sie können eine V3.0-CPU durch eine V4.1-CPU ersetzen, doch Sie können nach dem Laden der Konfiguration eine V4.1-CPU nicht durch eine V3.0-CPU ersetzen. Wenn Sie Ihr bestehendes STEP 7 V3.0-Projekt anzeigen oder anderweitig nutzen möchten, legen Sie vor dem Gerätetausch ein Archiv Ihres STEP 7 V3.0-Projekts an.

Sofern Sie die ausgetauschte Gerätekonfiguration noch nicht geladen haben, können Sie sie rückgängig machen. Nach dem Laden können Sie den Austausch von V3.0 durch V4.1 nicht mehr rückgängig machen.

Beachten Sie dabei einige Konfigurations- und Betriebsänderungen zwischen den beiden CPU-Versionen:

Organisationsbausteine

Bei Version 4.1 können Sie konfigurieren, ob die OB-Ausführung unterbrechbar oder nicht unterbrechbar (Seite 100) sein soll. In Projekten von früheren V3.0-CPUs legt STEP 7 für alle OBs standardmäßig die nicht unterbrechbare Ausführung fest.

STEP 7 legt für alle OB-Prioritäten (Seite 100) die Werte aus dem STEP 7-Projekt der V3.0-CPU fest.

Nachfolgend können Sie bei Bedarf die Unterbrechbarkeits- und Prioritätseinstellungen ändern.

Die Anlaufinformationen des Diagnosefehler-OB (Seite 95) beziehen sich auf das Submodul als Ganzes, wenn kein Diagnoseereignis ansteht.

CPU-Passwortschutz

STEP 7 legt für die V4.1-CPU als Passwortschutzstufe (Seite 205) die gleiche Stufe fest, die in der V3.0-CPU eingestellt war, und weist das Passwort der Version 3.0 dem Passwort für "Vollzugriff (kein Schutz)" der V4.1-CPU zu:

Schutzstufe in der Version 3.0	Zugriffsstufe in der Version V4.1
Kein Schutz	Vollzugriff (kein Schutz)
Schreibschutz	Lesezugriff
Schreib-/Leseschutz	HMI-Zugriff

Beachten Sie, dass die V4.1-Zugriffsstufe "Kein Zugriff (kompletter Schutz)" in der Version 3.0 nicht vorhanden war.

Webserver

Wenn Sie in Ihrem V3.0-Projekt benutzerdefinierte Webseiten verwenden, speichern Sie diese im Installationsverzeichnis Ihres Projekts in Unterverzeichnis "UserFiles\Webserver", bevor Sie das Upgrade Ihres Projekts starten. Sind die benutzerdefinierten Seiten in diesem Verzeichnis abgelegt, werden beim Speichern des STEP 7-Projekts auch die benutzerdefinierten Webseiten gespeichert.

Wenn Sie eine V3.0-CPU durch eine V4.1-CPU austauschen, sind Ihre Webserver-Projekteinstellungen (Seite 819) zum Aktivieren des Webservers und der HTTPS-Einstellungen die gleichen Einstellungen wie in der Version 3.0. Sie können dann Benutzer, Rechte, Passwörter (Seite 821) und Sprachen (Seite 819) wie für die Arbeit mit dem Webserver benötigt ändern. Wenn Sie keine Benutzer mit zusätzlichen Rechten konfigurieren, sind Sie auf das begrenzt, was auf den Standard-Webseiten (Seite 827) angezeigt wird. Die S7-1200 V4.1-CPU unterstützt den bisher vorkonfigurierten Benutzer "admin" und das zugehörige Passwort nicht.

Die Webserverseite für das Datenprotokoll der S7-1200 V3.0 bot bisher die Funktion "Herunterladen und Löschen". Die Dateibrowser-Seite (Seite 844) des Webservers der Version 4.1, über die Sie auf Datenprotokolle zugreifen, bietet diese Funktion nicht mehr an. Stattdessen bietet der Webserver die Möglichkeit zum Herunterladen, Umbenennen und Löschen von Datenprotokolldateien.

Inkompatibilität der Übertragungskarte

Sie können eine Übertragungskarte (Seite 137) der Version 3.0 nicht dazu verwenden, ein Programm der Version 3.0 in eine CPU der Version 4.1 zu übertragen. Sie müssen das V3.0-Projekt in STEP 7 öffnen, das Gerät in eine V4.1-CPU ändern (Seite 168) und das STEP 7-Projekt dann in Ihre V4.1-CPU laden. Nachdem Sie Ihr Projekt in ein Projekt der Version 4.1 geändert haben, können Sie eine V4.1-Übertragungskarte für nachfolgende Programmübertragungen einrichten.

GET/PUT-Kommunikation

In der Version 3.0 war die GET/PUT-Kommunikation standardmäßig aktiviert. Wenn Sie Ihre V3.0-CPU durch eine V4.1-CPU ersetzen (Seite 168), wird im Informationsbereich zur Kompatibilität eine Meldung angezeigt, dass GET/PUT aktiviert ist.

Unterstützung der Bewegungssteuerung

S7-1200 V4.1-CPUs unterstützen die Bewegungssteuerungbibliotheken V1.0 und V2.0 nicht. Wenn Sie für ein STEP 7-Projekt mit Bewegungssteuerungbibliotheken V1.0 oder V2.0 einen Gerätetausch durchführen, werden durch den Gerätetausch die Bewegungssteuerungsbibliotheken V1.0 und V2.0 zum Zeitpunkt des Übersetzens durch kompatible Bewegungssteuerungsanweisungen (Seite 589) der Version 3.0 ersetzt.

Wenn Sie für ein STEP 7-Projekt mit zwei verschiedenen Versionen von Bewegungssteuerungsanweisungen (V3.0 und V5.0) einen Gerätetausch von einer V3.0-CPU auf eine V4.1-CPU durchführen, werden durch den Gerätetausch zum Zeitpunkt des Übersetzens kompatible Bewegungssteuerungsanweisungen (Seite 589) der Version 5.0 ersetzt.

Bei einem Gerätetausch von einer V3.0-CPU auf eine V4.1-CPU wird die Version des Technologieobjekts (TO) für die Bewegungssteuerung nicht automatisch von V3.0 in V5.0 geändert. Wenn Sie auf die späteren Versionen aufrüsten möchten, müssen Sie im Anweisungsverzeichnis die erforderliche Version der S7-1200 Bewegungssteuerung für Ihr Projekt auswählen. Dies wird in der nachstehenden Tabelle gezeigt:

CPU-Version	Zulässige Bewegungssteuerungsversionen
V4.1 (Bewegungssteuerung V5.0)	V5.0 oder V4.0 oder V3.0
V4.0 (Bewegungssteuerung V4.0)	V4.0 oder V3.0
V3.0 (Bewegungssteuerung V3.0)	V3.0

Die TO-Struktur ist bei den Bewegungssteuerungsversionen V3.0 und V5.0 unterschiedlich. Alle zugehörigen Bausteine ändern sich ebenfalls. Bausteinschnittstellen, Beobachtungstabellen und die Trace-Funktion werden auf die neue Struktur der Bewegungssteuerung V5.0 aktualisiert. Die Unterschiede in den Achsenparametern der Bewegungssteuerung zwischen der CPU V3.0 und der CPU V4.1 werden in den folgenden zwei Tabellen aufgeführt:

CPU V3.0 (Bewegungssteuerung V3.0)	CPU V4.1 (Bewegungssteuerung V5.0)
Config.General.LengthUnit	Units.LengthUnit
Config.Mechanics.PulsesPerDriveRevolution	Actor.DriveParameter.PulsesPerDriveRevolution
Config.Mechanics.LeadScrew	Mechanics.LeadScrew
Config.Mechanics.InverseDirection	Actor.InverseDirection
Config.DynamicLimits.MinVelocity	DynamicLimits.MinVelocity
Config.DynamicLimits.MaxVelocity	DynamicLimits.MaxVelocity
Config.DynamicDefaults.Acceleration	DynamicDefaults.Acceleration
Config.DynamicDefaults.Deceleration	DynamicDefaults.Deceleration
Config.DynamicDefaults.EmergencyDeceleration	DynamicDefaults.EmergencyDeceleration
Config.DynamicDefaults.Jerk	DynamicDefaults.Jerk
Config.PositionLimits_SW.Active	PositionLimitsSW.Active
Config.PositionLimits_SW.MinPosition	PositionLimitsSW.MinPosition
Config.PositionLimits_SW.MaxPosition	PositionLimitsSW.MaxPosition
Config.PositionLimits_HW.Active	PositionLimitsHW.Active
Config.PositionLimits_HW.MinSwitchedLevel	PositionLimitsHW.MinSwitchLevel

CPU V3.0 (Bewegungssteuerung V3.0)	CPU V4.1 (Bewegungssteuerung V5.0)
Config.PositionLimits_HW.MaxSwitchedLevel	PositionLimitsHW.MaxSwitchLevel
Config.Homing.AutoReversal	Homing.AutoReversal
Config.Homing.Direction	Homing.ApproachDirection
Config.Homing.SideActiveHoming	Sensor[1].ActiveHoming.SideInput
Config.Homing.SidePassiveHoming	Sensor[1].PassiveHoming.SideInput
Config.Homing.Offset	Sensor[1].ActiveHoming.HomePositionOffset
Config.Homing.FastVelocity	Homing.ApproachVelocity
Config.Homing.SlowVelocity	Homing.ReferencingVelocity
MotionStatus.Position	Position
MotionStatus.Velocity	Velocity
MotionStatus.Distance	StatusPositioning.Distance
MotionStatus.TargetPosition	StatusPositioning.TargetPosition
StatusBits.SpeedCommand	StatusBits.VelocityCommand
StatusBits.Homing	StatusBits.HomingCommand

Der einzige "commandtable"-Parameter, der umbenannt wrid, ist das Array mit den Befehlen:

V3.0	V4.1
Config.Command[]	Command[]

Hinweis: Das Array "Command[]" ist in Version 3.0 ein UDT vom Typ "TO_CmdTab_Config_Command" und in Version 4.1 ein UDT vom Typ "TO_Struct_Command".

Geänderte Anweisungen

Bei den folgenden Anweisungen ändern sich Parameter oder das Verhalten:

- RDREC und WRREC (Seite 357)
- CONV (Seite 278)

HMI-Panel-Kommunikation

Wenn Sie ein oder mehrere HMI-Panels (Seite 31) an Ihre S7-1200 CPU der Version 3.0 angeschlossen hatten, hängt die Kommunikation mit der S7-1200 CPU der Version 4.1 vom verwendeten Kommunikationstyp und der Firmware-Version des HMI-Panels ab. Übersetzen Sie Ihr Projekt erneut und laden Sie es in die CPU und HMI und/oder aktualisieren Sie Ihre HMI-Firmware.

Voraussetzung für das Neuübersetzen von Programmbausteinen

Nach dem Austauschen einer V3.0-CPU durch eine V4.1-CPU müssen Sie alle Programmbausteine neu übersetzen, damit Sie sie in die CPU der Version 4.1 laden können. Außerdem müssen Sie bei Bausteinen mit Knowhow-Schutz (Seite 208) oder an eine PLC-Seriennummer gebundenem Kopierschutz (Seite 209) den Schutz aufheben, damit Sie die Bausteine übersetzen und laden können. (An eine Memory Card gebundenen Kopierschutz brauchen Sie jedoch nicht aufzuheben.) Nach dem erfolgreichen Übersetzen können Sie den Knowhow-Schutz und/oder den an eine PLC-Seriennummer gebundenen Kopierschutz erneut konfigurieren. Sollte Ihr Projekt von einem OEM (Original Equipment Manufacturer) bereitgestellte Bausteine mit Knowhow-Schutz enthalten, müssen Sie sich an den OEM wenden, damit er Ihnen diese Bausteine für die Version 4.1 zur Verfügung stellt.

Im Allgemeinen empfiehlt Siemens, nach dem Gerätetausch die Hardwarekonfiguration und Software in STEP 7 neu zu übersetzen und in alle Geräte Ihres Projekts zu laden. Berichtigen Sie alle Fehler, die beim Übersetzen des Projekts auftauchen und wiederholen Sie die Übersetzung, bis keine Fehler mehr auftreten. Dann können Sie das Projekt in die V4.1-CPU laden.

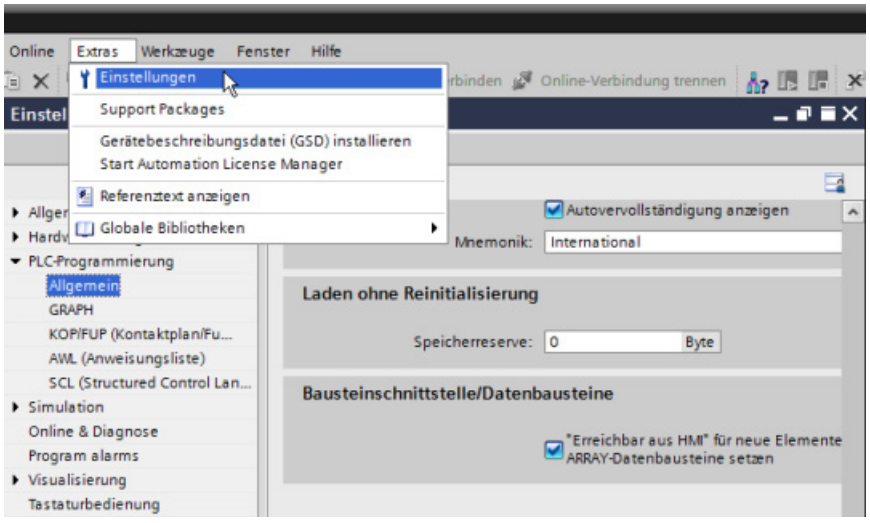
S7-1200 Projekte der Version 3.0 passen möglicherweise nicht in S7-1200 CPUs der Version 4.1

S7-1200 V4.0 ergänzt jeden DB um einen Reservebereich von 100 Bytes, um das Laden ohne Reinitialisierung zu unterstützen.

Sie können den 100-Byte-Reservebereich vor dem Laden eines V3.0-Projekts in eine CPU V4.1 aus den DBs löschen.

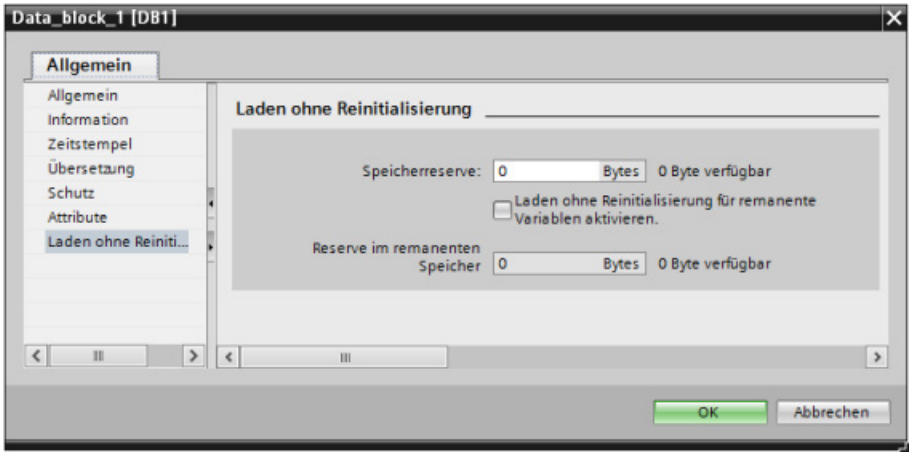
Um den 100-Byte-Reservebereich zu löschen, gehen Sie vor dem Gerätetausch wie folgt vor:

1. Wählen Sie im Hauptmenü im TIA-Portal den Menübefehl "Optionen" > "Einstellungen".
2. Öffnen Sie in der Projektnavigation den Knoten "PLC-Programmierung" > "Allgemein".
3. Legen Sie im Bereich "Laden ohne Reinitialisierung" eine Speicherreserve von 0 Bytes fest.



Wenn Sie den Gerätetausch bereits durchgeführt haben, müssen Sie die 100-Byte-Reserve in jedem Baustein einzeln löschen:

1. Klicken Sie in der Projektnavigation im Ordner der Programmbausteine mit der rechten Maustaste auf einen Datenbaustein und wählen Sie im Kontextmenü "Eigenschaften".
2. Wählen Sie im Eigenschaftsdialog des Datenbausteins den Knoten "Laden ohne Reinitialisierung".
3. Legen Sie eine Speicherreserve von 0 Bytes fest.
4. Wiederholen Sie diesen Vorgang für jeden Datenbaustein in Ihrem Projekt.



D.1 Ersatzklemmenblöcke S7-1200 V3.0 und V4.0

Tabelle D- 1 S7-1200 CPU bis V3.0 - Ersatzklemmenblöcke

Sie haben eine S7-1200 CPU bis V3.0 (Artikelnummer)	Dann verwenden Sie diese Ersatzklemmenblöcke (4/Packung)	
	Artikelnummer Klemmenblöcke	Beschreibung der Klemmenblöcke
CPU 1211C DC/DC/DC (6ES7 211-1AE31-0XB0)	6ES7 292-1BC30-0XA0 6ES7 292-1AH30-0XA0 6ES7 292-1AP30-0XA0	3-polig, vergoldet 8-polig, vergoldet 14-polig, verzinnt, codiert
CPU 1211C DC/DC/Relais (6ES7 211-1HE31-0XB0)		
CPU 1211C AC/DC/Relais (6ES7 211-1BE31-0XB0)		
CPU 1212C DC/DC/DC (6ES7 212-1AE31-0XB0)		
CPU 1212C DC/DC/Relais (6ES7 212-1HE31-0XB0)		
CPU 1212C AC/DC/Relais (6ES7 212-1BE31-0XB0)		
CPU 1214C DC/DC/DC (6ES7 214-1AG31-0XB0)	6ES7 292-1BC3-0XA0 6ES7 292-1AM30-0XA0 6ES7 292-1AV30-0XA0	3-polig, vergoldet 12-polig, verzinnt 20-polig, verzinnt
CPU 1214C DC/DC/Relais (6ES7 214-1HG31-0XB0)		
CPU 1214C AC/DC/Relais (6ES7 214-1BG31-0XB0)		
CPU 1215C DC/DC/DC (6ES7 215-1AG31-0XB0)	6ES7 292-1BF30-0XB0 6ES7 292-1AM30-0XA0 6ES7 292-1AV30-0XA0	6-polig, vergoldet 12-polig, verzinnt 20-polig, verzinnt, codiert
CPU 1215C DC/DC/Relais (6ES7 215-1HG31-0XB0)		
CPU 1215C AC/DC/Relais (6ES7 215-1BG31-0XB0)		

Tabelle D- 2 S7-1200 SMs bis V3.0 - Ersatzklemmenblöcke

Sie haben ein S7-1200 SM bis V3.0 (Artikelnummer)	Dann verwenden Sie diese Ersatzklemmenblöcke (4/Packung)	
	Artikelnummer Klemmenblöcke	Beschreibung der Klemmenblöcke
SM1221 DI 8xDC (6ES7 221-1BF32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 8xDC (6ES7 222-1BF32-0XB0)		
SM1222 DO 8xRelais (6ES7 222-1HF32-0XB0)	6ES7 292-1AG40-0XA1	7-polig, verzinnt, links codiert
SM1231 AI 4x13 Bit (6ES7 231-4HD32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1232 AO 2x14 Bit (6ES7 232-4HB32-0XB0)		
SM1231 AI 4xTC (6ES7 231-5QD32-0XB0)		
SM1231 AI 4x16 Bit (6ES7 231-5ND32-0XB0)		
SM1221 DI 16xDC (6ES7 221-1BH32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 16xDC (6ES7 222-1BH32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 16xRelais (6ES7 222-1HH32-0XB0)	6ES7 292-1AG40-0XA0	7-polig, verzinnt, rechts codiert
SM1223 DI 8xDC/DO 8xDC (6ES7 223-1BH32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1223 8xDC/8xRelais (6ES7 223-1PH32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
	6ES7 292-1AG40-0XA0	7-polig, verzinnt, rechts codiert
SM1223 8xAC/8xRelais (6ES7 223-1QH32-0XB0)	6ES7 292-1AG40-0XA0	7-polig, verzinnt, rechts codiert

Sie haben ein S7-1200 SM bis V3.0 (Artikelnummer)	Dann verwenden Sie diese Ersatzklemmenblöcke (4/Packung)	
	Artikelnummer Klemmenblöcke	Beschreibung der Klemmenblöcke
SM1234 AI 4 / AO 2 (6ES7 234-4HE32-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1231 AI 8x13 BIT (6ES7 231-4HF32-0XB0)		
SM1232 AO 4x14 Bit (6ES7 232-4HD32-0XB0)		
SM1231 AI 4xRTD (6ES7 231-5PD32-0XB0)		
SM1231 AI 8xTC (6ES7 231-5QF32-0XB0)		
SM 1278 IO-Link (6ES7 278-4BD32-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt

Tabelle D- 3 S7-1200 SMs ab V3.2 - Ersatzklemmenblöcke

Sie haben ein S7-1200 SM ab V3.2 (Artikelnummer)	Dann verwenden Sie diese Ersatzklemmenblöcke (4/Packung)	
	Artikelnummer Klemmenblöcke	Beschreibung der Klemmenblöcke
SM1221 DI 8xDC (6ES7 221-1BF30-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 8xDC (6ES7 222-1BF30-0XB0)		
SM1222 DO 8xRelais (6ES7 222-1HF30-0XB0)		
SM1231 AI 4x13 Bit (6ES7 231-4HD30-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1232 AO 2x14 Bit (6ES7 232-4HB30-0XB0)		
SM1231 AI 4xTC (6ES7 231-5QD30-0XB0)		
SM1231 AI 4x16 Bit (6ES7 231-5ND30-0XB0)		
SM1221 DI 16xDC (6ES7 221-1BH30-0XB0)	6ES7 292-1AG30-0XA0	7-polig, verzinnt
SM1222 DO 16xDC (6ES7 222-1BH30-0XB0)		
SM1222 DO 16xRelais (6ES7 222-1HH30-0XB0)		
SM1223 DI 8xDC/DO 8xDC (6ES7 223-1BH30-0XB0)		
SM1223 8xDC/8xRelais (6ES7 223-1PH30-0XB0)		
SM1223 8xAC/8xRelais (6ES7 223-1QH30-0XB0)		
SM1234 AI 4 / AO 2 (6ES7 234-4HE30-0XB0)	6ES7 292-1BG30-0XA0	7-polig, vergoldet
SM1231 AI 8x13 BIT (6ES7 231-4HF30-0XB0)		
SM1232 AO 4x14 Bit (6ES7 232-4HD30-0XB0)		
SM1231 AI 4xRTD (6ES7 231-5PD30-0XB0)		
SM1231 AI 8xTC (6ES7 231-5QF30-0XB0)		
SM1222 DO 8xRelais (Umschaltung) (6ES7 222-1XF30-0XB0)	6ES7 292-1AL30-0XA0	11-polig, verzinnt
SM1223 DI 16xDC/DO 16xDC (6ES7 223-1BL30-0XB0)		
SM 1223 16xDC/16xRelais (6ES7 223-1PL30-1XB0)		
SM1231 AI 8xRTD (6ES7 231-5PF30-0XB0)	6ES7 292-1BL30-0XA0	11-polig, vergoldet

Index

&

&-Box (logische UND-Verknüpfung in FUP), 218

/

/=-Box (Zuweisung negieren in FUP), 219

=

=-Box (Zuweisung in FUP), 219

>

>=1-Box (logische ODER-Verknüpfung in FUP), 218

A

Abfragearchitektur, 941
Abfragearchitektur Master, 941
Abfragearchitektur Slave, 941
ABS (Absolutwert bilden), 251
Abstand, Luftströmung und Kühlung, 53
AC
  Erdung, 76
  Richtlinien für Potentialtrennung, 76
  Verdrahtungsrichtlinien, 75, 77
Achssteuerungsbefehle als Bewegungsfolge ausführen (MC_CommandTable), 609
ACOS (Arcuscosinuswert bilden), 254
ACT_TINT (Uhrzeitalarm aktivieren), 384
ADD (Addieren), 248
Ad-hoc-Modus, TCP und ISO-on-TCP, 659
Adressen im Speicher, 112, 114
Adressierung
  Boolesche Werte oder Bitwerte, 113
  Einzelne Eingänge (E) oder Ausgänge (A), 113
  Prozessabbild, 112
  Speicherbereiche, 112
Aktive/passive Kommunikation
  Parameter, 663
  Partner konfigurieren, 638, 808
  Verbindungs-IDs, 659
Aktive/passive Verbindung, 638
Aktualisierungs-OB, 99
Alarme
  Alarmlatenz, 100
  ATTACH (OB einem Alarmereignis zuweisen), 374
  CAN_DINT (Verzögerungsalarm abbrechen), 386
  DETACH (Zuweisung eines OB zu einem Alarmereignis aufheben), 374
  QRY_DINT (Status des Verzögerungsalarms abfragen), 386
  SRT_DINT (Verzögerungsalarm starten), 386
  Übersicht, 90
Aliasnamen in benutzerdefinierten Webseiten, 857
Alte Anweisung TSEND_C (Daten über Ethernet senden (TCP)), 680
Alte Anweisungen TCON, TDISCON, TSEND und TRCV, 698
Alte Anweisunng TRCV_C (Daten über Ethernet empfangen (TCP)), 680
An-/Abmelden, Standard-Webseiten, 830
Analoge E/A
  Darstellung Ausgang (Spannung), 1255, 1298
  Darstellung Ausgang (Strom), 1255, 1299
  Darstellung Eingang (Spannung), 1254, 1297
  Darstellung Eingang (Strom), 1254, 1298
  Konfiguration, 173
  Schrittantwortzeiten (CPU), 1169, 1181, 1193, 1205, 1221
  Schrittantwortzeiten (SB), 1297
  Schrittantwortzeiten (SM), 1253
  Statusanzeigen, 1118
  Umwandlung in physikalische Einheiten, 42, 118, 286
Analoge Signalboards
  SB 1231, 1293
  SB 1231 RTD, 1304
  SB 1231 Thermoelement, 1300
  SB 1232, 1295
Analoge Signalmodule
  SM 1231, 1244
  SM 1231 RTD, 1262
  SM 1231 Thermoelement, 1256
  SM 1232, 1248
  SM 1234, 1250
Analogwerte normalisieren, 286
Analogwerte skalieren, 42, 286
Ändern
  Beobachtungstabelle, 1131
  Variablen über Webserver, 841
  Zustand im Programmiereditor, 1130

Anfangswerte
Startwerte eines DBs erfassen und zurücksetzen, 1130
Anlauf nach NETZ-EIN, 85
Anlaufverarbeitung, 88
Anlauf-OB, 91
Anlaufparameter, 139
Anschlüsse
Ethernet-Protokolle, 807
Kommunikationsarten, 631
Konfiguration, 663
Partner, 638, 808
S7-Verbindung, 807
Typen, Verbindungen mit mehreren Teilnehmern, 807
Verbindungs-IDs, 659
Webserver, 892
Anschlusskontakte
Maximale Belastbarkeit, 1321
Antriebe, Einrichten des MM4-Antriebs, 971
Anweisung DeviceStates (Modulstatus eines E/A-Systems lesen), 393
Anweisung GEO2LOG (Hardwarekennung über Steckplatz ermitteln), 457
Anweisung Get_IM_Data (Identifikations- und Wartungsdaten lesen), 411
Anweisung Modbus_Comm_Load (SIPLUS I/O oder Port am PtP-Modul für Modbus RTU konfigurieren), 1001
Anweisungen
&-Box (logische UND-Verknüpfung in FUP), 218
-( )- (Schließerspule), 219
-(/)- (Öffnerspule), 219
-(N)- (Operand bei negativer Signalflanke setzen), 224
-(P)- (Operand bei positiver Signalflanke setzen), 223
-(RESET_BF) (Bitfeld rücksetzen), 221
-(SET_BF) (Bitfeld setzen), 221
/=-Box (Zuweisung negieren in FUP), 219
-|/|- (Öffnerkontakt), 217
-||- (Schließerkontakt), 217
-|N|- (Operand auf negative Signalflanke abfragen), 223
-|P|- (Operand auf positive Signalflanke abfragen), 223
=-Box (Zuweisung in FUP), 219
>=1-Box (logische ODER-Verknüpfung in FUP), 218
ABS (Absolutwert bilden), 251
ACOS (Arcuscosinuswert bilden), 254
ACT_TINT (Uhrzeitalarm aktivieren), 384
ADD (Addieren), 248
Alte Anweisung TRCV_C (Daten über Ethernet empfangen (TCP)), 680
Alte Anweisung TSEND_C (Daten über Ethernet senden (TCP)), 680
Alte Anweisung USS_DRV (Daten mit Antrieb tauschen), 1044
Alte Anweisung USS_PORT (Kommunikation über USS-Netzwerk bearbeiten), 1042
Alte Anweisung USS_RPM (Parameter aus dem Antrieb auslesen), 1047
Alte Anweisung USS_WPM (Parameter im Antrieb ändern), 1049
Alte Anweisungen TCON, TDISCON, TSEND und TRCV, 698
Alte USS-Statuscodes, 1050
Analogwerte skalieren, 42
Anweisung DeviceStates (Modulstatus eines E/A-Systems lesen), 393
Anweisung GEO2LOG (Hardwarekennung über Steckplatz ermitteln), 457
Anweisung Get_IM_Data (Identifikations- und Wartungsdaten lesen), 411
Anweisung ModuleStates (Modulstatusinformationen eines Moduls lesen), 399
ASIN (Arcussinuswert bilden), 254
ATAN (Arcustangenswert bilden), 254
ATH (ASCII-Zeichenkette in Hexadezimalzahl umwandeln), 345
ATTACH (OB einem Alarmereignis zuweisen), 374
Ausgang rücksetzen, 220
Ausgang setzen, 220
Beobachten, 1130
Bewegungssteuerung, 589
CALCULATE, 41
CALCULATE (Berechnen), 247
CAN_DINT (Verzögerungsalarm abbrechen), 386
CAN_TINT (Uhrzeitalarm löschen), 383
CASE (SCL), 307
CEIL (Aus Gleitpunktzahl nächsthöhere Ganzzahl erzeugen), 283
Chars_TO_Strg (Array aus CHAR in Zeichenkette umwandeln), 343
CONCAT (Zeichenketten verketten), 349
CONTINUE (SCL), 312
CONV (Wert umwandeln), 278
COS (Cosinuswert bilden), 254
CountOfElements (Anzahl ARRAY-Elemente abfragen), 274
CTD (Rückwärtszählen), 235
CTRL_HSC (Schnellen Zähler steuern), 465

CTRL_HSC_EXT (Schnellen Zähler steuern), 468
CTRL_PWM (Impulsdauermodulation), 413
CTU (Vorwärtszählen), 235
CTUD (Vorwärts- und Rückwärtszählen), 235
DataLogClose (Datenprotokoll schließen), 441
DataLogCreate (Datenprotokoll erstellen), 432
DataLogNewFile (Datenprotokoll in neuer Datei), 442
DataLogOpen (Datenprotokoll öffnen), 436
DataLogWrite (Datenprotokoll schreiben), 438
Datum, 323
DEC (Dekrementieren), 251
DECO (Decodieren), 316
DELETE (Zeichen in einer Zeichenkette löschen), 351
Deserialize, 260
DETACH (Zuweisung eines OB zu einem Alarmereignis aufheben), 374
Dezentrale Peripherie AS-i, 356
Dezentrale Peripherie PROFIBUS, 356
Dezentrale Peripherie PROFINET, 356
DIS_AIRT (Ausführung von Alarmen höherer Priorität und asynchronen Fehlerereignissen deaktivieren), 389
DIV (Dividieren), 248
DPNRM_DG (Diagnosedaten eines DP-Slaves lesen), 371
DPRD_DAT (Konsistente Daten eines DP-Normslaves lesen), 368
DPWR_DAT (Konsistente Daten eines DP-Normslaves schreiben), 368
Drag & Drop, 40
Drag & Drop zwischen Editoren, 45
Einfügen, 40
EN_AIRT (Ausführung von Alarmen höherer Priorität und asynchronen Fehlerereignissen aktivieren), 389
ENCO (Encodieren), 316
ENDIS_PW (Passwort-Legitimierung einschränken und freigeben), 295
EQ_ElemType (Vergleiche, ob Datentyp eines ARRAY-Elements GLEICH dem Datentyp einer Variablen ist), 245
EQ_Type (Vergleiche, ob Datentyp GLEICH dem Datentyp einer Variablen ist), 245
Ergänzen von Eingängen oder Ausgängen in KOP- und FUP-Anweisungen, 43
Erweiterbare Anweisungen, 43
EXIT (SCL), 313
EXP (Exponentialwert bilden), 254
EXPT (Potenzieren), 254
F_TRIG (Variable bei negativer Signalflanke setzen), 225
Favoriten, 40
FeldLesen (Feld lesen), 276
FeldSchreiben (Feld schreiben), 276
FILL_BLK (Speicher mit Bitmuster belegen), 266
FIND (Zeichen in einer Zeichenkette finden), 355
FLOOR (Aus Gleitpunktzahl nächstniedere Ganzzahl erzeugen), 283
FOR (SCL), 308
Forcefunktion, 1136
FRAC (Nachkommastellen ermitteln), 254
Gemeinsame Parameter, 739
GET (Daten aus entfernter CPU auslesen), 802
GET_DIAG (Diagnoseinformationen lesen), 405
GET_ERROR (Fehler lokal abrufen), 299
GET_ERROR_ID (Fehler-ID lokal abrufen), 300
Get_Features (Erweiterte Funktionen abrufen), 937
GOTO (SCL), 314
HSC (schneller Zähler), 471
HTA (Hexadezimalzahl in ASCII-Zeichenkette umwandeln), 345
IF-THEN (SCL), 306
IN_Range (Wert innerhalb Bereich), 243
INC (Inkrementieren), 251
INSERT (Zeichen in einer Zeichenkette einfügen), 352
INV (Einerkomplement erstellen), 316
IS_ARRAY (Auf ARRAY prüfen), 246
IS_NULL (Abfrage nach Pointer GLEICH Null), 246
JMP (Springen bei VKE = 1), 290
JMP_LIST (Sprungliste definieren), 291
JMPN (Springen bei VKE = 0), 290
Kalender, 323
Label (Sprungmarke), 290
LED (LED-Status lesen), 392
LEFT (linke Zeichen einer Zeichenkette lesen), 350
LEN (Länge einer Zeichenkette ermitteln), 348
LIMIT (Grenzwert setzen), 253
LN (Natürlichen Logarithmus bilden), 254
LOG2GEO (Steckplatz über Hardwarekennung ermitteln), 459
MAX (Maximum abrufen), 252
MAX_LEN (Maximale Länge einer Zeichenkette), 348
MB_CLIENT, 978
MB_CLIENT (Über PROFINET als Modbus TCP-Client kommunizieren), 1055
MB_COMM_LOAD (Port am PtP-Modul für Modbus RTU konfigurieren), 1074
MB_MASTER (Über PtP-Port als Modbus-Master kommunizieren), 1077

MB_SERVER (Über PROFINET als Modbus TCP-Server kommunizieren), 1062
MB_SLAVE (Über PtP-Port als Modbus-Slave kommunizieren), 1084
MC_ChangeDynamic (Dynamikeinstellungen der Achse ändern), 612
MC_CommandTable, 609
MC_Halt (Achse pausieren), 597
MC_Home (Referenzpunktfahrt der Achse durchführen), 594
MC_MoveAbsolute (Achse absolut positionieren), 599
MC_MoveJog (Achse im Tippbetrieb bewegen), 606
MC_MoveRelative (Achse relativ positionieren), 601
MC_MoveVelocity (Achse mit vordefinierter Geschwindigkeit bewegen), 603
MC_Power (Achse freigeben/sperren), 590
MC_ReadParam (Parameter eines Technologieobjekts lesen), 616
MC_Reset (Fehler bestätigen), 593
MC_WriteParam (Parameter des Technologieobjekts schreiben), 614
MID (mittlere Zeichen einer Zeichenkette lesen), 350
MIN (Minimum abrufen), 252
MOD (Divisionsrest einer Division), 249
Modbus_Comm_Load (SIPLUS I/O oder Port am PtP-Modul für Modbus RTU konfigurieren), 1001
Modbus_Master (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Master kommunizieren), 1004
Modbus_Slave (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Slave kommunizieren), 1011
MOVE (Wert kopieren), 257
MOVE_BLK (Bereich kopieren), 257
MUL (Multiplizieren), 248
MUX (Multiplexen), 318
N (Operand auf negative Signalflanke abfragen), 223
N_TRIG (VKE auf negative Signalflanke abfragen), 224
N=-Box und N-Spule (Operand bei negativer Signalflanke setzen), 224
NE_ElemType (Vergleiche, ob Datentyp eines ARRAY-Elements UNGLEICH dem Datentyp einer Variablen ist), 245
NE_Type (Vergleiche, ob Datentyp UNGLEICH dem Datentyp einer Variablen ist), 245
NEG (Zweierkomplement erstellen), 250
NORM_X (Normalisieren), 284
NOT (VKE invertieren), 219
NOT_NULL (Abfrage nach Pointer UNGLEICH Null), 246
NOT_OK (Ungültigkeit prüfen), 244
ODER (logische Verknüpfung), 315
OK (Gültigkeit prüfen), 244
OUT_Range (Wert außerhalb Bereich), 243
P (Operand auf positive Signalflanke abfragen), 223
P_TRIG (VKE auf positive Signalflanke abfragen), 224
P=-Box und P-Spule (Operand bei positiver Signalflanke setzen), 223
PEEK- und POKE-Varianten, 200, 269
PID_Compact (Universal-PID-Regler mit integrierter Einstellung), 485
PID_Temp (universeller PID-Regler für die Temperaturregelung), 502
PORT_CFG (Kommunikationsparameter dynamisch konfigurieren), 1022
Port_Config (Portkonfiguration), 918
Programmsteuerung (SCL), 305
PUT (Daten in entfernte CPU schreiben), 802
QRY_CINT (Weckalarmparameter abfragen), 380
QRY_DINT (Status des Verzögerungsalarms abfragen), 386
QRY_TINT (Status des Uhrzeitalarms abfragen), 385
R (Ausgang rücksetzen), 220
R_TRIG (Variable bei positiver Signalflanke setzen), 225
RALRM (Alarm empfangen), 360
RCV_CFG (Serielle Empfangsparameter dynamisch konfigurieren), 1025
RCV_PTP (Empfangsmeldungen aktivieren), 1033
RCV_RST (Empfangspuffer löschen), 1035
RD_ADDR (E/A-Adressen über Hardwarekennung ermitteln), 461
RD_LOC_T (Lokalzeit lesen), 326
RD_SYS_T (Uhrzeit lesen), 326
RDREC (Datensatz lesen), 357
RE_TRIGR, 104
RE_TRIGR (Zyklusüberwachungszeit neu starten), 297
READ_BIG (Daten in Big-Endian-Format lesen), 271
READ_DBL (Aus Datenbaustein im Ladespeicher lesen), 453
READ_LITTLE (Daten in Little-Endian-Format lesen), 271
Receive_Config (Empfangskonfiguration), 923
Receive_P2P (Punkt-zu-Punkt-Daten empfangen), 932
Receive_Reset (Empfangspuffer löschen), 934

RecipeExport (Rezeptexport), 424
RecipeImport (Rezeptimport), 426
REPEAT (SCL), 311
REPLACE (Zeichen in einer Zeichenkette ersetzen), 354
RESET_BF (Bitfeld rücksetzen), 221
RET (Rückgabewert), 294
RETURN (SCL), 314
RIGHT (rechte Zeichen einer Zeichenkette lesen), 350
ROL (Links rotieren) und ROR (Rechts rotieren), 322
ROUND (Zahl runden), 282
RS (Flipflop rücksetzen/setzen), 221
RT (Zeit zurücksetzen), 227
RTM (Betriebsstundenzähler), 331
RUNTIME (Programmlaufzeit messen), 303
S (Ausgang setzen), 220
S_CONV (Zeichenkette umwandeln), 334
S_MOV (Zeichenkette verschieben), 333
SCALE_X (Skalieren), 284
SEL (Selektieren), 318
SEND_CFG (Serielle Übertragungsparameter dynamisch konfigurieren), 1024
Send_Config (Sendekonfiguration), 920
Send_P2P (Punkt-zu-Punkt-Daten senden), 928
SEND_PTP (Sendepufferdaten übertragen), 1030
Serialize, 263
SET_BF (Bitfeld setzen), 221
SET_CINT (Weckalarm parametrieren), 378
Set_Features (Erweiterte Funktionen setzen), 938
SET_TIMEZONE (Zeitzone setzen), 330
SET_TINTL (Datum und Uhrzeit eines Uhrzeitalarms festlegen), 382
SGN_GET (RS232-Signale abfragen), 1036
SGN_GET (RS232-Signale abrufen), 935
SGN_SET (RS232-Signale festlegen), 1037
SHL (Links schieben) und SHR (Rechts schieben), 321
Signal_Set (RS232-Signale setzen), 936
SIN (Sinuswert bilden), 254
Spalten und Überschriften, 44, 667, 679, 687, 697, 955, 977, 1000, 1039, 1054, 1073
SQR (Quadrat bilden), 254
SQRT (Quadratwurzel bilden), 254
SR (Flipflop setzen/rücksetzen), 221
SRT_DINT (Verzögerungsalarm starten), 386
STP (Programm beenden), 298
Strg_TO_Chars (Zeichenkette in Array aus CHAR umwandeln), 343
STRG_VAL (Zeichenkette in Zahlenwert umwandeln), 334
SUB (Subtrahieren), 248
SWAP (Anordnung ändern), 268
SWITCH (Sprungverteilung), 292
T_ADD (Zeiten addieren), 324
T_COMBINE (Zeiten verknüpfen), 325
T_CONFIG (Schnittstelle konfigurieren), 731
T_CONV (Zeiten umwandeln und extrahieren), 323
T_DIAG, 709
T_DIFF (Zeitdifferenz), 325
T_RESET, 707
T_SUB (Zeiten subtrahieren), 324
TAN (Tangenswert bilden), 254
TCON, 688
TDISCON, 688
TM_MAIL (E-Mail senden), 1107
TOF (Ausschaltverzögerung), 227
TON (Einschaltverzögerung), 227
TONR (Speichernde Einschaltverzögerung), 227
TP (Impuls), 227
TRCV, 688
TRCV_C, 668, 748
TRUNC (Ganzzahl erzeugen), 282
TSEND, 688
TSEND_C, 668, 747
TURCV (Daten über Ethernet empfangen (UDP)), 725
TUSEND (Daten über Ethernet senden (UDP)), 725
Überwachungsstatus oder -wert, 1129
UFILL_BLK (Speicher ununterbrechbar mit Bitmuster belegen), 266
Uhr, 326
Uhrzeit, 323
UMOVE_BLK (Bereich ununterbrechbar kopieren), 257
Umwandlungsanweisungen in SCL, 279
UND (logische Verknüpfung), 315
USS_Drive_Control (Daten mit Antrieb tauschen), 960
USS_Port_Scan (Kommunikation über USS-Netzwerk bearbeiten), 959
USS_Read_Param (Parameter aus dem Antrieb auslesen), 963
USS_Write_Param (Parameter im Antrieb ändern), 964
USS-Statuscodes, 966
VAL_STRG (Zahlenwert in Zeichenkette umwandeln), 334
VariantGet (VARIANT Variablenwert lesen), 272
VariantPut (VARIANT Variablenwert schreiben), 273

Versionen von
Anweisungen, 44, 667, 679, 687, 697, 955, 977, 1000, 1039, 1054, 1073
Werte vergleichen, 242
WHILE (SCL), 310
WR_LOC_T (Lokalzeit einstellen), 326
WR_SYS_T (Uhrzeit einstellen), 326
WRIT_DBL (In Datenbaustein im Ladespeicher schreiben), 453
WRITE_BIG (Daten in Big-Endian-Format schreiben), 271
WRITE_LITTLE (Daten in Little-Endian-Format schreiben), 271
WRREC (Datensatz schreiben), 357
WWW (benutzerdefinierte Webseiten synchronisieren), 866
x-Box (logische Exklusiv-ODER-Verknüpfung in FUP), 218
XOR (logische Verknüpfung), 315
Zeit, 227
Zustand, 1130
Anwenderprogramm
Bausteine aus einer Online-CPU kopieren, 212
Codebausteine im Anwenderprogramm aufrufen, 181
Drag & Drop zwischen Editoren, 45
Einfügen von Anweisungen, 40
Ergänzen von Eingängen oder Ausgängen in KOP- und FUP-Anweisungen, 43
Erweiterbare Anweisungen, 43
Favoriten, 40
Laden in CPU, 211
Lineare und strukturierte Programme, 179
Memory Card, 136
Organisationsbausteine (OBs), 182
Passwortschutz, 208
Programmkarte, 136
Übertragungskarte, 136
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209
Anzeige erster Zyklus, 109
Arbeitsspeicher, 26
CPU 1211C, 1163
CPU 1212C, 1175
CPU 1214C, 1187
CPU 1215C, 1199
CPU 1217C, 1213
Arithmetik, 41, 247, 248
Arrays mit Variablen indexieren, 277
Arrays, Zugriff auf Elemente, 277
Artikelnummern
CPU 1214FC, CPU 1215FC, 1343
CPUs, 1339
CSM 1277 Ethernet Switch, 1344
Endhalter, 1345
Erweiterungskabel, 1345
FS Signalmodule, 1343
Grundlegende HMI-Panels, 1344
Klemmenblöcke, 1345
Kommunikationsschnittstellen (CM, CB und CP), 1342, 1342, 1342, 1342, 1343
Memory Cards, 1344
PM 1207 Stromversorgung, 1344
Programmiersoftware, 1348
Signalboards, Batterieboards, 1341
Signalmodule, 1340
Simulatoren, 1345
Steckverbinder und Klemmenanschlüsse, 1343
STEP 7, 1348
Visualisierungssoftware, 1348
WinCC, 1348
ASi
Adresse,
Analogwerte übertragen,
AS-i-Master CM 1243-2,
AS-i-Mastermodul CM1243-2 hinzufügen,
AS-i-Slave hinzufügen,
Digitalwerte übertragen,
Netzwerkverbindung,
RDREC (Datensatz lesen),
Slavekonfiguration mit STEP 7,
Slavekonfiguration ohne STEP 7,
Systemzuweisung,
Systemzuweisung von Slaveadressen,
WRREC (Datensatz schreiben),
AS-i
Anweisungen für die dezentrale Peripherie, 356
ASIN (Arcussinuswert bilden), 254
Assistent für den Zertifikatsimport, 894
ATEX-Zulassung, 1153
ATH (ASCII-Zeichenkette in Hexadezimalzahl umwandeln), 345
ATTACH (OB einem Alarmereignis zuweisen), 374
AT-Variablenüberlagerung, 134
Auf Werkseinstellungen zurücksetzen, 1123
Aufrufstruktur, 215
Ausgangsparameter, 184
Impulsausgänge, 414
Impulskanäle konfigurieren, 416
Auslösen
Trace, 1150
Werte in der Beobachtungstabelle, 1133
Ausschaltverzögerung (TOF), 227

Austauschen einer V3.0-CPU durch eine V4.1-CPU, 1349
AWP_Enum_Def, 858
AWP_Import_Fragment, 861
AWP_In_Variable, 852, 855
AWP_Out_Variable, 854
AWP_Start_Fragment, 860
AWP-Befehle, 849
Alias verwenden, 857
Definitionen verbinden, 862
Enum-Typ definieren, 858
Enum-Typ referenzieren, 858
Fragmente generieren, 860
Fragmente importieren, 861
Sondervariablen lesen, 854
Sondervariablen schreiben, 855
Variablen schreiben, 852

B

Batterieboard (BB)
Batterie einlegen, 1310
BB 1297, 1309
Baudrate, 903
Baugruppenträger- oder Stationsfehler-OB ("Rack or station failure OB"), 97
Bausteine
Alarme, 28, 100, 1164, 1177, 1188, 1201, 1215
Anfangswert eines FB, 184
Anlauf-OBs, 100
Anzahl der Codebausteine, 28, 81, 1164, 1177, 1188, 1201, 1215
Anzahl der OBs, 28, 100, 1164, 1177, 1188, 1201, 1215
Bausteinaufrufe, 81
Bausteine aus einer Online-CPU kopieren, 212
Beobachten, 28, 1164, 1177, 1188, 1201, 1215
Datenbaustein (DB), 81
Einzelinstanz- oder Multiinstanz-DB, 184
Ereignisse, 100
FB oder FC mit SCL aufrufen, 196
Funktion (FC), 81, 184
Funktionsbaustein (FB), 81, 184
Größe des Anwenderprogramms, 28, 81, 1164, 1177, 1188, 1201, 1215
Gültige FC-, FB- und DB-Nummern, 81
Instanz-Datenbaustein (DB), 184
Konsistenzprüfung, 215
Laden, 211
Lineare und strukturierte Programme, 179
Organisationsbausteine (OBs), 28, 81, 90, 100, 1164, 1177, 1188, 1201, 1215
Passwortschutz, 208
Schachtelungstiefe, 28, 81, 1164, 1177, 1188, 1201, 1215
Typen, 81
Zähler (Anzahl und Speicherbedarf), 28, 1165, 1177, 1189, 1201, 1215
Zeiten (Anzahl und Speicherbedarf), 28, 1165, 1177, 1189, 1201, 1215
Bausteine aus einer Online-CPU kopieren, 212
BB 1297, 1309
Bearbeiten im Betriebszustand RUN, (Laden im Betriebszustand RUN)
Bedienoberfläche
STEP 7 Projekt- und Portalansicht, 39
Bedienpanel, 31, 46
Betriebszustände der CPU, 85
Beispiele, Anweisungen
ATH (ASCII-Zeichenkette in Hexadezimalzahl wandeln), 346
CALCULATE, 41
CONTINUE, SCL, 312
CTRL_HSC_EXT, 470
DECO (Decodieren), 317
Deserialize, 262
DeviceStates, PROFIBUS und PROFINET, 395
EXIT, SCL, 313
GET_DIAG und Modi, 410
GOTO (SCL), 314
HTA (Hexadezimalzahl in ASCII-Zeichenkette wandeln), 346
LIMIT (Grenzwert setzen), 254
ModuleStates, PROFIBUS und PROFINET, 401
PEEK- und POKE-Varianten, 200, 269
RETURN, SCL, 314
ROR (Rechts rotieren), SCL, 322
RUNTIME (Programmlaufzeit messen), 304
S_CONV (Zeichenkette umwandeln), 341
Serialize, 265
SET_CINT Weckalarmausführung und Zeitparameter, 378
SHL (Links schieben), SCL, 321
STRG_VAL (Zeichenkette in Zahlenwert umwandeln), 342
SWAP (Anordnung ändern), 268
TM_MAIL, 1112
VAL_STRG (Zahlenwert in Zeichenkette umwandeln), 343
Zeitspulen, 229

Beispiele, Bewegungssteuerung
Achsverhalten, 618
Geschwindigkeit, 622
Geschwindigkeitskennlinie für die Referenzpunktfahrt bei der Bewegungssteuerung, 586
Konfiguration der Impulsausgangsgeschwindigkeit der CPU 1211C, CPU 1212C, CPU 1214C und CPU 1215C, 551
Konfiguration der Impulsausgangsgeschwindigkeit der CPU 1217C, 550
Konfigurieren einer MC_Tabelle mit einem Technologieobjekt, 573
Ruckbegrenzung, 588
Tippverhalten, 626
Beispiele, Kommunikation
ASi-Slaveadressierung,
CPU-Kommunikation mit einer gemeinsamen Verbindung zum Senden und Empfangen von Daten, 661
CPU-Kommunikation mit getrennten Verbindungen zum Senden und Empfangen von Daten, 660
CPU-Kommunikation über TSEND_C- oder TRCV_C-Verbindungen, 662
I-Device als IO-Gerät und IO-Controller, 758
PROFIBUS S7-Verbindung konfigurieren, 814
PROFINET S7-Verbindung konfigurieren, 812
PROFINET-Kommunikationsprotokolle, 656
Shared Device, 765
Shared I-Device, 771
T_CONFIG, IP-Parameter ändern, 737
T_CONFIG, IP-Parameter und PROFINET IO-Gerätenamen ändern, 738
Telecontrol, 1101
Beispiele, Modbus
Beispielprogramm für einen Modbus RTU-Master, 1018
Beispielprogramm für einen Modbus RTU-Slave, 1020
MB_CLIENT für mehrere Anforderungen mit gemeinsamer TCP-Verbindung, 995
MB_CLIENT für mehrere Anforderungen mit unterschiedlichen TCP-Verbindungen, 996
MB_SERVER für mehrere TCP-Verbindungen, 993
Modbus TCP MB_CLIENT für die Koordination mehrerer Anforderungen, 997
Modbus TCP MB_CLIENT Schreibanforderung für das Prozessabbild der Ausgänge, 997
Modbus TCP, Beispiele für Parameter von MB_HOLD_REG, 989
Modbus TCP, Halteregisteradresse, 991
Modbus TCP, MB_CLIENT-Verbindungsparameter, 982
Modbus TCP, MB_SERVER-Verbindungsparameter, 987
Beispiele, Modbus in älteren Systemen
Beispiele für Modbus RTU, Parameter von MB_HOLD_REG in älteren Systemen, 1085
Beispielprogramm für einen Modbus RTU-Master in älteren Systemen, 1091
Beispielprogramm für einen Modbus RTU-Slave in älteren Systemen, 1093
MB_CLIENT für die Koordination mehrerer Modbus TCP-Anforderungen in älteren Systemen, 1072
MB_CLIENT für mehrere Anforderungen mit unterschiedlichen Modbus TCP-Verbindungen in älteren Systemen, 1070
MB_CLIENT in älteren Systemen: mehrere Anforderungen mit gemeinsamer Modbus TCP-Verbindung,
MB_SERVER für mehrere Modbus TCP-Verbindungen in älteren Systemen, 1068
Modbus RTU, Halteregisteradressierung in älteren Systemen, 1088
Modbus TCP MB_CLIENT Schreibanforderung für das Prozessabbild der Ausgänge in älteren Systemen, 1071
Modbus TCP, Halteregisteradressierung in älteren Systemen, 1066
Modbus TCP, MB_HOLD_REG-Parameter in älteren Systemen, 1064
Beispiele, PID
PID_3Step, Konfigurationseinstellungen, 517
PID_Compact, Konfigurationseinstellungen, 516
PID_Temp, Konfigurationseinstellungen, 519
Beispiele, PtP-Kommunikation
Bedingung für das Meldungsende, 913
Bedingung für den Meldungsbeginn, 909
Beispiel für den Terminalemulator ausführen, 951
Konfiguration, 943
Meldungslänge innerhalb der Meldung, 913
PtP-Kommunikation in älteren Systemen, RCV_CFG, 1028
Receive_Config, 925
STEP 7-Programmierung, 949
Terminalemulator, 942, 951
Beispiele, USS-Kommunikation
Meldung von USS-Kommunikationsfehlern, 967
Meldung von USS-Kommunikationsfehlern in älteren Systemen, 1052
Beispiele, verschiedene
Analogwertverarbeitung, 118, 286
AT-Variablenüberlagerung, 134

Ausgewählte Bausteine im Betriebszustand RUN laden, 1141
Berechnung der Leistungsbilanz, 1337
CPU 1217C Differentialausgang und Anwendung, 1227
CPU 1217C Differentialeingang und Anwendung, 1226
Datenprotokollprogramm, 449
Drag & Drop zwischen Editoren, 45
ENO-Auswertung in SCL, 203
Konfigurationssteuerung (Optionsverwaltung), 164
Rezept, 420, 428
S7-1200 IO-Link-Master Anschluss, 1274
Scheiben eines Variablendatentyps, 133
Trace- und Logic-Analyzer-Funktion, 1150
verschachtelte CASE-Anweisungen, SCL, 308
Zugriff auf Array-Elemente, 277
Beispiele, Webserver
Aliasnamen, 851, 857
AWP-Deklarationen verbinden, 862
benutzerdefinierte Webseite, 870, 877
benutzerdefinierte Webseite für die Spracheinstellung, 884
DB-Fragmente, 862
Enum-Typen, 858, 859, 874
Sondervariablen lesen, 855
Sondervariablen schreiben, 856, 876
Sonderzeichen in AWP-Befehlen, 863
STEP 7-Programm für die Prüfung auf Fragmente, 891
Variablen lesen, 851, 873
Variablen schreiben, 853, 875
Zugriff über mobiles Gerät, 825
Bemessungsspannungen, 1159, 1159
Benutzerdefinierte Webseiten, 818, 847
Aktivieren mit WWW-Anweisung, 866
Aktualisieren, 849
Anlegen mit HTML-Editor, 848
AWP-Befehle für den Zugriff auf S7-1200 Daten, 849
Beispiel, 870
Einschränkungen Ladespeicher, 869
Entsprechende DBs in die CPU laden, 868
Fragmente erstellen, 860
Fragmente importieren, 861
Handhabung von Sonderzeichen, 862
HTML-Liste, 877
Konfiguration für mehrere Sprachen, 887
Konfigurieren, 865
Manuelle Steuerung von DB-Fragmenten, 887
mehrere Sprachen, 883
Programmbausteine generieren, 866
Programmbausteine löschen, 866
Programmieren in STEP 7, 866
Sondervariablen lesen, 854
Sondervariablen schreiben, 855
Über Steuer-DB aktivieren und deaktivieren, 887
Variablen lesen, 851
Variablen schreiben, 852
Zugriff über PC, 868
Benutzerdefinierte Webseiten aktualisieren, 849, 849
Benutzerdefinierte Webseiten anlegen, 848
Benutzerkonfiguration, Webserver, 821
Beobachten
Beobachtungstabelle, 1131
Forcefunktion, 1136
Forcetabelle, 1135
KOP-Status und Verwendung der Beobachtungstabelle, 1129
KOP-Zustand, 1130
Speicherauslastung, 1125
Startwerte eines DBs zurücksetzen, 1130
Werte eines DBs erfassen, 1130
Zykluszeit, 1125
Beobachtungstabelle
Ausgänge im Betriebszustand STOP freischalten, 1134
Beobachten, 1129
Forcen, 213
Funktionsweise, 1131
Memory Card, 136
Werte auslösen, 1133
Betriebsstundenzähler (RTM), 331
Betriebszustand, 46, 46
Betriebszustände der CPU, 85
STOP/RUN wechseln, 1125
Betriebszustand RUN, 85, 89, 1125
Bedienpanel, 46
Forcefunktion, 1136
Schaltflächen in der Funktionsleiste, 46
Betriebszustand STOP, 85, 1125
Ausgänge im Betriebszustand STOP freischalten, 1134
Bedienpanel, 46
Forcefunktion, 1136
Schaltflächen in der Funktionsleiste, 46
Bewegungsfolge (MC_CommandTable), 609
Bewegungssteuerung
Achse konfigurieren, 556, 566
Hardware- und Software-Endschalter, 578
Konfigurationsparameter für die Referenzpunktfahrt, 584
MC_ChangeDynamic (Dynamikeinstellungen der Achse ändern), 612

MC_CommandTable, 609
MC_Halt (Achse pausieren), 597
MC_Home (Referenzpunktfahrt der Achse durchführen), 594
MC_MoveAbsolute (Achse absolut positionieren), 599
MC_MoveJog (Achse im Tippbetrieb bewegen), 606
MC_MoveRelative (Achse relativ positionieren), 601
MC_MoveVelocity (Achse mit vordefinierter Geschwindigkeit bewegen), 603
MC_Power (Achse freigeben/sperren), 590
MC_ReadParam (Parameter eines Technologieobjekts lesen), 616
MC_Reset (Fehler bestätigen), 593
MC_WriteParam (Parameter des Technologieobjekts schreiben), 614
Phasenlage, 552
Referenzpunktfahrt (Abfolge bei der aktiven Referenzpunktfahrt), 586
Referenzpunktfahrt der Achse, 582
Übersicht, 547
Bewegungssteuerungsanweisungen, 589
Bibliothek für das alte USS-Protokoll
Statuscodes, 1050
Übersicht, 1038
USS_DRV (Daten mit Antrieb tauschen), 1044
USS_PORT (Kommunikation über USS-Netzwerk bearbeiten), 1042
USS_RPM (Parameter aus dem Antrieb auslesen), 1047
USS_WPM (Parameter im Antrieb ändern), 1049
Voraussetzungen für den Einsatz, 1040
Bibliothek für das USS-Protokoll
Statuscodes, 966
Übersicht, 952
USS_Drive_Control (Daten mit Antrieb tauschen), 960
USS_Port_Scan (Kommunikation über USS-Netzwerk bearbeiten), 959
USS_Read_Param (Parameter aus dem Antrieb auslesen), 963
USS_Write_Param (Parameter im Antrieb ändern), 964
Voraussetzungen für den Einsatz, 956
Bitverknüpfung
NOT-Operation (VKE invertieren), 219
Operationen AND (UND), OR (ODER) und XOR (EXKLUSIV ODER), 218
Operationen Steigende Flanke und Fallende Flanke, 223
Schließer- und Öffnerkontakte, 217
Schließer- und Öffnerspulen, 219
Setz- und Rücksetzoperationen, 220
Boolesche Werte oder Bitwerte, 113
Busstecker, 30

C

CALCULATE (Berechnen), 247
Analogwerte skalieren, 42
Für komplexe Gleichungen verwenden, 41
CAN_DINT (Verzögerungsalarm abbrechen), 386
CAN_TINT (Uhrzeitalarm löschen), 383
CANopen-Module
021620-B, 021630-B, 1332
CB 1241
Abschluss, 899
CB 1241 RS485, 1323
CEIL (Aus Gleitpunktzahl nächsthöhere Ganzzahl erzeugen), 283
CE-Zulassung, 1151
Char (Datentyp Character), 125
Chars_TO_Strg (Array aus CHAR in Zeichenkette umwandeln), 343
Codebaustein
Alarme, 28, 1164, 1177, 1188, 1201, 1215
Anfangswert eines FB, 184
Anzahl der Codebausteine, 28, 81, 1164, 1177, 1188, 1201, 1215
Anzahl der OBs, 28, 1164, 1177, 1188, 1201, 1215
Bausteinaufrufe, 81
Beobachten, 1164, 1177, 1188, 1201, 1215
Codebausteine im Anwenderprogramm aufrufen, 181
DB (Datenbaustein), 81, 186
FB (Funktionsbaustein), 81, 184
FC (Funktion), 81, 184
Größe des Anwenderprogramms, 28, 1164, 1177, 1188, 1201, 1215
Gültige FC-, FB- und DB-Nummern, 81
Instanz-Datenbaustein (DB), 184
Knowhow-Schutz, 208
Kopierschutz, 209
Lineare und strukturierte Programme, 179
Organisationsbausteine (OBs), 28, 182, 1164, 1177, 1188, 1201, 1215
Schachtelungstiefe, 28, 1164, 1177, 1188, 1201, 1215
Überwachung, 28
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209

Zähler (Anzahl und Speicherbedarf), 28, 1165, 1177, 1189, 1201, 1215
Zeiten (Anzahl und Speicherbedarf), 28, 1165, 1177, 1189, 1201, 1215
Codebausteine im Anwenderprogramm aufrufen, 181
Compact Switch Module, CSM 1277, 1332
CONCAT (Zeichenketten verketten), 349
CONTINUE, SCL, 312
CONV (Wert umwandeln), 278
Cookie siemens_automation_language, 883
Cookie, siemens_automation_language, 883
Cookie-Einschränkungen, Standard-Webseiten, 894
COS (Cosinuswert bilden), 254
CountOfElements (Anzahl ARRAY-Elemente abfragen), 274
CP-Modul
Webserver Startseite, 834
Zugriff auf Webserver, 826
CPU
Anlaufparameter, 139
Anlaufverarbeitung, 88
Anzahl der Kommunikationsverbindungen, 634
ASi,
ASi-Adresse,
AS-i-Port, 792
Auf Werkseinstellungen zurücksetzen, 1123
Ausgänge im Betriebszustand STOP freischalten, 1134
Bausteine aus einer Online-CPU kopieren, 212
Bausteine vergleichen und synchronisieren, 1127
Bedienpanel, 46
Bedienpanel (Online-CPU), 1125
Beobachtungstabelle, 1131
Betriebsarten RUN/STOP, 1125
Betriebszustände, 85
Eigenschaft Uhrzeitsynchronisation, 654
Einbau, 58, 60
Erdung, 76
Erweiterungskabel, 68
Ethernet-Port, 646
Forcen, 1135, 1136
Gerätekonfiguration, 151, 151
HSC-Konfiguration, 480
Impulsausgänge, 414
induktive Lasten, 79
IP-Adresse, 646
Klemmenblock, 67
Knowhow-Schutz, 208
Kommunikation, 636
Kommunikationsarten, 631
Kommunikationsboards (CB), 30
Kommunikationslast, 105
Laden in CPU, 211
Laden in Gerät, 651
Lampenlasten, 78
LED-Anzeigen, 1115
Leere Übertragungskarte, 149
Leistungsbedarf, 1335
Leistungsbilanz, 54
MAC- und IP-Adressen anzeigen, 651
MAC-Adresse, 646, 646, 651
Module hinzufügen, 156
Netzwerkverbindung, 637
Neues Gerät hinzufügen, 152, 152
Nicht spezifizierte CPU, 155
OBs verarbeiten, 182
Online, 1122
Online gehen, 1119
Passwortschutz, 205
PROFIBUS-Adresse, 787
PROFINET IO, 750
PROFINET-Port, 646
Programmausführung, 81
Richtlinien für Potentialtrennung, 76
RTM (Betriebsstundenzähler), 331
Schaltflächen RUN/STOP, 46
Schutzstufen, 205, 205
Signalboards (SB), 30
Startwerte eines DBs zurücksetzen, 1130
Thermischer Bereich, 53, 57
Übersicht, 25
Überwachung online, 1129
Verdrahtungsrichtlinien, 75, 77
Vergleichstabelle, 26
Verlorenes Passwort, 149
Vorgehensweise bei verlorenem Passwort, 149
Werte eines DBs erfassen, 1130
Zugriffsschutz, 205
Zuweisen einer IP-Adresse zu einer Online-CPU, 645
Zykluszeitkonfiguration, 105
CPU Memory Card
Einfügen, 137
Einsatzzwecke, 136
Programmkarte, 143
Übertragungskarte, 140
CPU-Eigenschaften, benutzerdefinierte Webseiten
Mehrere Sprachen einrichten, 887
STEP 7-Konfiguration, 865
CPU-Konfiguration
Betriebsparameter, 168
Impulskanäle, 416
Kommunikation mit HMI, 743
Mehrere CPUs, 745

Moduleigenschaften, 173
Zykluszeitüberwachung, 104
CPUs
CPU 1211C AC/DC/Relais, 1163
CPU 1211C DC/DC/DC, 1163
CPU 1211C DC/DC/Relais, 1163
CPU 1212C AC/DC/Relais, 1175
CPU 1212C DC/DC/DC, 1175
CPU 1212C DC/DC/Relais, 1175
CPU 1214C AC/DC/Relais, 1187
CPU 1214C DC/DC/DC, 1187
CPU 1214C DC/DC/Relais, 1187
CPU 1215C AC/DC/Relais, 1199
CPU 1215C DC/DC/DC, 1199
CPU 1215C DC/DC/Relais, 1199
CPU 1217C DC/DC/DC, 1213
Schrittantwortzeiten, 1169, 1181, 1193, 1205, 1221
CPU-Webseite Identifikation, 835
CSM 1277 Compact Switch Module, 1332
CTD (Rückwärtszählen), 235
C-Tick-Zulassung, 1153
CTRL_HSC (Schnellen Zähler steuern), 465
CTRL_HSC_EXT (Schnellen Zähler steuern), 468
CTS (Hardware-Flusskontrolle, PtP), 904
CTU (Vorwärtszählen), 235
CTUD (Vorwärts- und Rückwärtszählen), 235
cULus-Zulassung, 1152

D

Date
Datentyp Date, 123
DTL (Datentyp Date and Time long), 124
Datenaustausch zwischen E/A-Systemen, 758
Datenbaustein
CONF_DATA, 734
Einzelner FB mit Multiinstanz-DBs, 185
Fragmente in benutzerdefinierten Webseiten importieren, 861
Globaler Datenbaustein, 112, 186
Instanz-Datenbaustein, 112
Optimierter Zugriff, 186
Organisationsbausteine (OBs), 182
READ_DBL (Aus Datenbaustein im Ladespeicher lesen), 453
Standardzugriff, 186
Startwerte zurücksetzen, 1130, 1130
Struktur, 81
Übersicht, 81, 186
Werte erfassen, 1130, 1130
WRIT_DBL (In Datenbaustein im Ladespeicher schreiben), 453
Datenhantierungsbausteine (DHBs), 186
Datenprotokoll
Beispielprogramm, 449
DataLogClose (Datenprotokoll schließen), 441
DataLogCreate (Datenprotokoll erstellen), 432
DataLogNewFile (Datenprotokoll in neuer Datei), 442
DataLogOpen (Datenprotokoll öffnen), 436
DataLogWrite (Datenprotokoll schreiben), 438
Datenprotokolle anzeigen, 444
Datensatzstruktur, 431
Größenbegrenzung und Größenberechnung, 446
Überblick über Datenprotokolle, 431
Datentypen, 120
Any (Pointer), 130
Arrays, 127
Bool, Byte, Wort und Doppelwort, 121
PLC-Datentypeditor, 128
Pointer (Pointer), 129
Real, LReal (Gleitpunkt), 122
Struc, 128
Time, Date, TOD (Uhrzeit), DTL (Date and Time long), 123
Übersicht über Pointer-Datentypen, 128
USInt, SInt, UInt, Int, UDInt, Dint (Ganzzahl), 122
Variant (Pointer), 132
Zeichen und Zeichenfolgen, 125
Datenübertragung, anstoßen, 928, 1030
Datum
SET_TIMEZONE (Zeitzone setzen), 330
T_ADD (Zeiten addieren), 324
T_COMBINE (Zeiten verknüpfen), 325
T_CONV (Zeiten umwandeln und extrahieren), 323
T_DIFF (Zeitdifferenz), 325
T_SUB (Zeiten subtrahieren), 324
DB (Datenbaustein), (Datenbaustein)
DB-Fragmente (benutzerdefinierte Webseiten)
Generieren, 866
Mit AWP-Befehl importieren, 861
Mit AWP-Befehlen erstellen, 860
DBs für benutzerdefinierte Webseiten erstellen, 866
DBs für benutzerdefinierte Webseiten generieren, 866
DC
Ausgänge, 1160
Erdung, 76
induktive Lasten, 79
Richtlinien für Potentialtrennung, 76
Verdrahtungsrichtlinien, 75, 77
DEC (Dekrementieren), 251
DECO (Decodieren), 316
DELETE (Zeichen in einer Zeichenkette löschen), 351
Deserialize, 260

DETACH (Zuweisung eines OB zu einem Alarmereignis aufheben), 374
DeviceStates, Beispiel, 395
Diagnose
  Anweisung DeviceStates (Modulstatus eines E/A-Systems lesen), 393
  Anweisung Get_IM_Data (Identifikations- und Wartungsdaten lesen), 411
  Anweisung ModuleStates (Modulstatusinformationen eines Moduls lesen), 399
  Beobachtungstabelle, 1131
  Diagnosepuffer, 1126
  GET_DIAG (Diagnoseinformationen lesen), 405
  LED (LED-Status lesen), 392
  LED-Anzeigen, 1115
  Puffer, 110
  Speicherauslastung, 1125
  Statusanzeige, 109
  Zykluszeit, 1125
Diagnosefehler-OB, 95
Digitale E/A
  Impulsabgriff, 173
  Konfiguration, 173
  Statusanzeigen, 1116
Digitale Signalboards
  SB 1221, 1282
  SB 1222, 1284
  SB 1223, 1287, 1290
Digitale Signalmodule
  SM 1221, 1228
  SM 1222, 1230, 1232
  SM 1223, 1237, 1241
Digitaleingangsfilterzeit, 170
DIS_AIRT (Ausführung von Alarmen höherer Priorität und asynchronen Fehlerereignissen deaktivieren), 389
DIV (Dividieren), 248
Dokumentation, 4
DPNRM_DG, 371
DPRD_DAT (Konsistente Daten eines DP-Normslaves lesen), 368
DPWR_DAT (Konsistente Daten eines DP-Normslaves schreiben), 368
Drag & Drop zwischen Editoren, 45
Dynamische Verknüpfung, 209

E

E/A
  Adressierung, 117
  Analoge Statusanzeigen, 1118
  Darstellung Analogausgang (Spannung), 1255, 1298
  Darstellung Analogausgang (Strom), 1255, 1299
  Darstellung Analogeingang (Spannung), 1254, 1297
  Darstellung Analogeingang (Strom), 1254, 1298
  Digitale Statusanzeigen, 1116
  Forcefunktion, 1136
  In einer Beobachtungstabelle beobachten, 1131
  induktive Lasten, 79
  Schrittantwortzeiten (CPU), 1169, 1181, 1193, 1205, 1221
  Schrittantwortzeiten (SB), 1297
  Schrittantwortzeiten (SM), 1253
  Zustand in KOP beobachten, 1130
E/A-System, Datenaustausch, 758
Eigenschaft Uhrzeitsynchronisation, 654
Ein- und Ausgänge
  Beobachten, 1129
Einbau
  Abstand, 53
  CPU, 60
  Erdung, 76
  Erweiterungskabel, 68
  induktive Lasten, 79
  Klemmenblock, 67
  Kommunikationsboard (CB), 62
  Kommunikationsmodul (CM), 66
  Kühlung, 53
  Lampenlasten, 78
  Luftströmung, 53
  Montageabmessungen, 57
  Richtlinien, 51
  Richtlinien für Potentialtrennung, 76
  Signalboard (SB), 62
  Signalmodul (SM), 64
  Signalmodule (SM), 30
  Thermischer Bereich, 53, 57
  TS-Adapter auf DIN-Schiene, 73
  Überblick, 58
  Übersicht, 51
  Verdrahtungsrichtlinien, 75, 77
  Voraussetzungen, 38
  Wandmontage des TS-Adapters, 74
Einfügen eines Geräts
  Nicht spezifizierte CPU, 155
Einfügen von Anweisungen
  Drag & Drop, 40
  Drag & Drop zwischen Editoren, 45
  Favoriten, 40
Eingänge
  Impulsabgriffbits, 171
Eingangsfilterzeit, 170

Eingangssimulatoren, 1327
Eingeschränkte TSAPs und Portnummern, 740
Einschaltverzögerung (TON), 227
Einschränkungen
Benutzerdefinierte Webseiten, 869
Webserver, 892
Einstellungen, 44
Einstellungen von STEP 7 ändern, 44
Elektromagnetische Verträglichkeit (EMV), 1156, 1157
E-Mail, Senden mit TMAIL_C, 714
EN und ENO (Signalfluss), 202
EN_AIRT (Ausführung von Alarmen höherer Priorität und asynchronen Fehlerereignissen aktivieren), 389
ENCO (Encodieren), 316
Endebedingungen, 911
ENDIS_PW (Passwort-Legitimierung einschränken und freigeben), 295
Entwerfen einer Automatisierungslösung mit einem PLC-Gerät, 177, 179
Enum-Typen definieren, benutzerdefinierte Webseiten, 858, 858
Enum-Typen in benutzerdefinierten Webseiten, 858, 858
Enum-Typen zuweisen, benutzerdefinierte Webseiten, 858
EQ_ElemType (Vergleiche, ob Datentyp eines ARRAY-Elements GLEICH dem Datentyp einer Variablen ist), 245
EQ_Type (Vergleiche, ob Datentyp GLEICH dem Datentyp einer Variablen ist), 245
Ereignisausführung, 100
Ergänzen von Eingängen oder Ausgängen in KOP- und FUP-Anweisungen, 43
Erkennung zum Laden einer Online-CPU, 155
Ersetzen von Modulen, 49
Erweiterbare Anweisungen, 43
Erweiterte Bausteinschnittstelle
Laden im Betriebszustand RUN, 1144
Erweiterung der Leistungsmerkmale der S7-1200, 29
Erweiterungskabel, 1331
Ausbau, 68
Einbau, 68
Ethernet
Ad-hoc-Modus, 659
Alte Anweisung TRCV_C (Daten über Ethernet empfangen (TCP)), 680
Alte Anweisung TSEND_C (Daten über Ethernet senden (TCP)), 680
Alte Anweisungen TCON, TDISCON, TSEND und TRCV, 698
Anzahl der Kommunikationsverbindungen, 634
CSM 1277 Compact Switch Module, 1332
DPNRM_DG (Diagnosedaten eines DP-Slaves lesen), 371
DPRD_DAT (Konsistente Daten eines DP-Normslaves lesen), 368
DPWR_DAT (Konsistente Daten eines DP-Normslaves schreiben), 368
GET (Daten aus entfernter CPU auslesen), 802
IP-Adresse, 646
Kommunikationsarten, 631
MAC-Adresse, 646
Netzwerkverbindung, 637
PUT (Daten in entfernte CPU schreiben), 802
RALRM (Alarm empfangen), 360
RDREC (Datensatz lesen), 357
T_CONFIG (Schnittstelle konfigurieren), 731
TCON, 688
TDISCON, 688
TRCV, 688
TRCV_C, 668
TSEND, 688
TSEND_C, 668
TURCV (Daten über Ethernet empfangen (UDP)), 725
TUSEND (Daten über Ethernet senden (UDP)), 725
Übersicht, 656
Verbindungs-IDs, 659
WRREC (Datensatz schreiben), 357
Ethernet-Protokolle, 656
Verbindungen mit mehreren Teilnehmern, 807
EXIT, SCL, 313
EXP (Exponentialwert bilden), 254
EXPT (Potenzieren), 254

F

F_TRIG (Variable bei negativer Signalflanke setzen), 225
FAQs, 4
Favoriten, Funktionsleiste, 40
FB (Funktionsbaustein)
Übersicht, 81
FC (Funktion), 81, 184
Fehler
Diagnosefehler, 95
Häufige Fehler bei erweiterten Anweisungen, 464
Zeitfehler, 94
Fehler der Empfangskonfiguration, 927, 1030
Fehler der Sendekonfiguration, 922, 1025
Fehler durch inkompatible CPU-Version, 1116
Fehler durch unbekannte CPU-Version, 1116

Fehlerbehebung
Diagnosepuffer, 1126
LED-Anzeigen, 1115
FeldLesen (Feld lesen), 276
FeldSchreiben (Feld schreiben), 276
Feste Länge, 912
FILL_BLK (Speicher mit Bitmuster belegen), 266
Filterzeit, 170
FIND (Zeichen in einer Zeichenkette finden), 355
Firmware aktualisieren
aus STEP 7, 1124
mit einer Memory Card, 146
Firmware-Aktualisierung
über Webserver, 839, 839
Firmware-Update
aus STEP 7, 1124
mit einer Memory Card, 146
Flankenoperationen, steigend und fallend, 223
Flexible Maschinen, (Konfigurationssteuerung (Optionenhandling))
FLOOR (Aus Gleitpunktzahl nächstniedere Ganzzahl erzeugen), 283
Flusskontrolle, 903
Konfiguration, 903
Verwalten, 904
FM-Zertifizierung, 1152
FOR, SCL, 308
Forcen, 1135
Beobachtungstabelle, 1131
Eingänge der Peripherie, 1135, 1136
Eingänge und Ausgänge, 1136
Memory Card, 136
Speicherbereich E, 1135, 1136
Zyklus, 1136
Forcetabelle
Eingänge der Peripherie ansprechen, 1135
Forcefunktion, 1136
Forcen, 1135
FRAC (Nachkommastellen ermitteln), 254
Frequenz, Taktbits, 110
Funktion (FC)
Codebausteine im Anwenderprogramm aufrufen, 181
Gültige FC-Nummern, 81
Knowhow-Schutz, 208
Lineare und strukturierte Programme, 179
Übersicht, 81, 184
Funktion, I-Device, 754
Funktionen, neu, 33
Funktionsbaustein (FB)
Anfangswert, 184
Ausgangsparameter, 184
Codebausteine im Anwenderprogramm aufrufen, 181
Einzelner FB mit Multiinstanz-DBs, 185
Gültige FB-Nummern, 81
Instanz-Datenbaustein, 184
Knowhow-Schutz, 208
Lineare und strukturierte Programme, 179
Übersicht, 81, 184
Funkverbindung zum Webserver, 824
FUP (Funktionsplan), 194

G

Gerät
PROFINET IO, 750
PROFINET IO-Gerätenamen, 751
Shared, 765
Gerät ändern, 168
Geräteaustausch
V3.0-CPU durch eine V4.1-CPU, 1349
Vorgehensweise, 168
Gerätekonfiguration, 151, 742
Ändern eines Gerätetyps, 168
ASi,
AS-i-Port, 793
CPU konfigurieren, 168
Eigenschaft Uhrzeitsynchronisation (PROFINET), 654
Erkennen, 155
Ethernet-Port, 646
Laden in CPU, 211
Module hinzufügen, 156
Module konfigurieren, 173
Netzwerkverbindung, 637
Neues Gerät hinzufügen, 152
Nicht gesteckte Module, 49
PROFIBUS, 787
PROFINET-Port, 646
GET (Daten aus entfernter CPU auslesen), 802
Verbindung konfigurieren, 640
GET_DIAG (Diagnoseinformationen lesen), 405
GET_ERROR (Fehler lokal abrufen), 299
GET_ERROR_ID (Fehler-ID lokal abrufen), 300
Get_Features (Erweiterte Funktionen abrufen), 937
Globale Bibliothek
Überblick über das alte USS-Protokoll, 1038
Überblick über das USS-Protokoll, 952
Globaler Datenbaustein, 112, 186
GOTO, SCL, 314
Grundlegende Panels (HMI), 31
GSD-Datei, 761

H

Handbücher, 4
Hardware-Flusskontrolle, 904
Hardwarekonfiguration, 151
  ASi,
  AS-i-Port, 793
  CPU konfigurieren, 168
  Erkennen, 155
  Ethernet-Port, 646
  Laden in CPU, 211
  Module hinzufügen, 156
  Module konfigurieren, 173
  Netzwerkverbindung, 637
  Neues Gerät hinzufügen, 152
  PROFIBUS, 787
  PROFINET-Port, 646
Herunterladen
  Firmware-Update, 146
  Siemens-Sicherheitszertifikat auf PC, 833, 894
HMI-Geräte
  Netzwerkverbindung, 637
  PROFINET-Kommunikation konfigurieren, 743
  Übersicht, 31
Hochrüsten einer V3.0-CPU auf V4.1, 1349
HochspannungsIsolationsprüfung,
Hotline, 3
HSC (schneller Zähler)
  Funktionsweise, 471
  Konfiguration, 480, 480
HSC_Period, 469
HTA (Hexadezimalzahl in ASCII-Zeichenkette umwandeln), 345
HTML-Seiten
  Anwenderdefiniert, 847
  Liste, Beispiel für benutzerdefinierte Webseiten, 877
HTML-Seiten, benutzerdefiniert
  Aktualisieren, 849
  Entwickeln, 848
  Seitenanordnung, 865
  Speicherorte für Sprachen, 887
  Zugriff auf S7-1200 Daten, 849
HTTP-Variablen lesen, 854
HTTP-Verbindungen, Webserver, 892

I

I-Device (Intelligent IO Device)
  Funktionalität, 754
  Konfigurieren, 760
  Konfigurieren mit GSD-Datei, 761
  Shared, 771
I-Device (intelligentes IO-Gerät)
  Eigenschaften, 755
  untergeordnetes PN IO-System, 756
IF-THEN, SCL, 306
Impuls (TP), 227
Impulsabgriff, 171, 173
Impulsabgriffbits, Konfiguration der Digitaleingänge, 171
Impulsausgänge, 414
IN_Range (Wert innerhalb Bereich), 243
INC (Inkrementieren), 251
Induktive Lasten, 79
Industrieumgebungen
  Zulassungen, 1155
Informationsquellen, 4
INSERT (Zeichen in einer Zeichenkette einfügen), 352
Installation
  Leistungsbilanz, 54
  TS-Adapter SIM-Karte, 72
  TS-Adapter und TS-Modul, 70
Instanz-Datenbaustein, 112
INV (Einerkomplement erstellen), 316
IO-Link
  Adressbereich, 1276
  Anschlussbelegung, 1274
  Austausch, 1272
  Datensatz, 1277
  Diagnose, 1280
  Diagramm, 1275
  Fehlermeldungen, 1277, 1279, 1280
  Funktionen, 1272
  Gerätelagerung, 1273
  Geräteprofil, 1271
  Konfigurieren, 1276
  LED-Anzeige, 1279
  Parameter, 1276
  Parameter während der Laufzeit ändern, 1276
  Rücksetzen auf Werkseinstellungen, 1273
IO-Link-Master-Signalmodul, 1268
IP-Adresse, 647, 648
  CPU online konfigurieren, 1122
  Gerätekonfiguration, 168
  Konfigurieren, 646
  MAC-Adresse, 646
  Online zuweisen, 645
  Zuordnen, 642, 650
IP-Adresse des Routers, 648
IP-Router, 646
IS_ARRAY (Auf ARRAY prüfen), 246
IS_NULL (Abfrage nach Pointer GLEICH Null), 246

ISO on TCP
Ad-hoc-Modus, 659
ISO-on-TCP
Parameter, 663
Verbindungs-IDs, 659
Verbindungskonfiguration, 638
ISO-on-TCP-Protokoll, 656

J

JavaScript-Einschränkungen, Standard-Webseiten, 893
JMP (Springen bei VKE = 1), 290
JMP_LIST (Sprungliste definieren), 291
JMPN (Springen bei VKE = 0), 290

K

Kabel
Erweiterung, 1331
Netzwerkkommunikation, 898
Kalender, 323
Kein Neustart, 85
Klemmenblock, 67
Klemmenblock, Einbau und Ausbau, 67
Knowhow-Schutz
Passwortschutz, 208
Kommunikation
Abfragearchitektur, 941
Aktiv/passiv, 638, 663, 808
Anzahl der Verbindungen (PROFINET/PROFIBUS), 634
ASi-Adresse,
Eigenschaft Uhrzeitsynchronisation (PROFINET), 654
Flusskontrolle, 904
Hardwareanschluss, 741
IP-Adresse, 646
Kommunikationslast, 105
Konfiguration, 638, 663, 808
MAC-Adresse, 646
Netzwerk, 741
Netzwerkverbindung, 637
Parameter, 663
PROFIBUS-Adresse, 787
PROFINET und PROFIBUS, 631
Protokolle, 658
Sende- und Empfangsparameter, 906
TCON_Param, 663
Verbindungs-IDs, 659
Verlust, Ziehen oder Stecken von Modulen, 97
Zykluszeit, 105
Kommunikationsboard (CB)
Ausbau, 62
CB 1241 RS485, 1323
Einbau, 62
Gerätekonfiguration, 151
Konfiguration von Parametern, 173
LED-Anzeigen, 897, 1115
Module hinzufügen, 156
Programmieren, 940
RS485, 897
Übersicht, 30
Vergleichstabelle, 29
Kommunikationsmodul (CM)
AS-i-Mastermodul CM1243-2 hinzufügen, 791
Ausbau, 66
CM 1241 RS232, 1324
CM 1241 RS422/RS485, 1325
CM 1243-5 (DP-Master) hinzufügen, 785
Datenempfang, 932, 1033
Einbau, 66
Gerätekonfiguration, 151
Konfiguration des PtP-Beispielprogramms, 943
Konfiguration von Parametern, 173
LED-Anzeigen, 897, 1115
Leistungsbedarf, 1335
Module hinzufügen, 156
Programmieren, 940
RS232 und RS485, 897
Übersicht, 30
Vergleichstabelle, 29
Kommunikationsprozessor (CP)
Gerätekonfiguration, 151
Konfiguration von Parametern, 173
Module hinzufügen, 156
Übersicht, 30
Vergleichstabelle, 29
Kommunikationsschnittstellen
CB 1241 RS485, 1323
CM 1241 RS232, 1324
Gerätekonfiguration, 151
Konfiguration, 902
LED-Anzeigen, 1115
Module hinzufügen, 156
Programmieren, 940
RS232 und RS485, 897
Vergleichstabelle der Module, 29
Kompensation der kalten Verbindungsstelle, Thermoelement, 1259, 1302
Konfiguration
Anlaufparameter, 139

Anweisungen PID_Compact und PID_3Step, 516
Anweisungen PID_Temp, 519
ASi,
AS-i-Port, 792
CPU-Parameter, 168
Eigenschaft Uhrzeitsynchronisation (PROFINET), 654
Erkennen, 155
Ethernet-Port, 646
HSC (schneller Zähler), 480
IP-Adresse, 646
Kommunikation PLC/PLC, 745
Kommunikationslast, 105
Kommunikationsschnittstellen, 902
Laden in CPU, 211
MAC-Adresse, 646
Meldung empfangen, 907
Module, 173
Module hinzufügen, 156
Netzwerkverbindung, 637
PROFIBUS, 787
PROFIBUS-Adresse, 787
PROFINET-Port, 646
RS422, Betriebsarten, 946
RS485, Betriebsarten, 948
Schnittstellen, 902
Zykluszeit, 104
Konfiguration der gesendeten Meldung, 906
Konfiguration Empfangsparameter, 749
Konfiguration für Meldungssendung, 906
Konfiguration Sendeparameter, 638, 748, 808
Konfiguration, benutzerdefinierte Webseiten
Mehrere Sprachen einrichten, 887
STEP 7-Konfiguration, 865
Konfigurationssteuerung (Optionenhandling)
Steuerdatensatz, 161
Konfigurationssteuerung (Optionsverwaltung), 157
Beispiel, 164
Konsistenzprüfung, 215
Kontaktdaten, 3, 168
Kontakte, (Siehe Bitverknüpfung)
Konventionen bei Anführungszeichen, Webserver, 862
KOP (Kontaktplan)
Beobachten, 1130
Programmiereditor, 1130
Übersicht, 193
Überwachungsstatus oder -wert, 1129
Zustand, 1130, 1135
Kopierschutz
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209
Kühlen, 53
Kunden-Support, 3

L

Label (Sprungmarke), 290
Laden aus CPU
Anwenderprogramm, 212
Bausteine aus einer Online-CPU kopieren, 212
Laden im Betriebszustand RUN
aus STEP 7, 1140
Einschränkungen, 1146
Erweiterte Bausteinschnittstelle, 1144
Globale Einstellungen für die Speicherreserve, 1146
Laden ausgewählter Bausteine, 1141
Laden fehlgeschlagen, 1147
Laden ohne Reinitialisierung, 1144
Sicherheitsaspekte, 1148
Speicherreserve und remanente Speicherreserve, 1144
Übersetzungsfehler, 1143
Übersicht, 1138
Voraussetzungen, 1139
Laden in CPU
Anwenderprogramm, 211
DBs für benutzerdefinierte Webseiten, 868
MAC- und IP-Adressen anzeigen, 651
Projekt, 211
Ladespeicher, 26
Benutzerdefinierte Webseiten, 869
CPU 1211C, 1163
CPU 1212C, 1175
CPU 1214C, 1187
CPU 1215C, 1199
CPU 1217C, 1213
Memory Card, 136
Programmkarte, 136
Übertragungskarte, 136
Lampenlasten, 78
Länge PtP-Meldung, 913
Latenz, 100
Laufzeitfehler beim Senden, 930, 1032
Lebensdauer eines Relais, 1160
LED (LED-Status lesen), 392
LED-Anzeigen
CPU-Status, 1115
Kommunikationsschnittstelle, 897, 1115
Leerzeile, 908
LEFT (linke Zeichen einer Zeichenkette lesen), 350
Leistungsbilanz, 54
Beispiel, 1337

Formular für Berechnungen, 1338
Übersicht, 1335
Leitung im Leerlauf, 907
LEN (Länge einer Zeichenkette ermitteln), 348
Lesen aus DBs, E/A oder Speicher, 200, 269
LIMIT (Grenzwert setzen), 253
Lineare Programmierung, 179
LN (Natürlichen Logarithmus bilden), 254
LOG2GEO (Steckplatz über Hardwarekennung ermitteln), 459
Logic Analyzer, 1150
Luftströmung, 53

M

MAC- und IP-Adressen anzeigen, 651
MAC-Adresse, 646, 651
Manuelle Steuerung von DB-Fragmenten, 887
MAX (Maximum abrufen), 252
MAX_LEN (Maximale Länge einer Zeichenkette), 348
Maximale Anzahl Webserver-Verbindungen, 892
Maximale Meldungslänge, 912
MB_CLIENT, 978
MB_CLIENT (Über PROFINET als Modbus TCP-Client kommunizieren), 1055
MB_COMM_LOAD (Port am PtP-Modul für Modbus RTU konfigurieren), 1074
MB_MASTER (Über PtP-Port als Modbus-Master kommunizieren), 1077
MB_SERVER, 986
MB_SERVER (Über PROFINET als Modbus TCP-Server kommunizieren), 1062
MB_SLAVE (Über PtP-Port als Modbus-Slave kommunizieren), 1084
MC_ChangeDynamic (Dynamikeinstellungen der Achse ändern), 612
MC_CommandTable, 609
MC_Halt (Achse pausieren), 597
MC_Home (Referenzpunktfahrt der Achse durchführen), 594
MC_MoveAbsolute (Achse absolut positionieren), 599
MC_MoveJog (Achse im Tippbetrieb bewegen), 606
MC_MoveRelative (Achse relativ positionieren), 601
MC_MoveVelocity (Achse mit vordefinierter Geschwindigkeit bewegen), 603
MC_Power (Achse freigeben/sperren), 590
MC_ReadParam (Parameter eines Technologieobjekts lesen), 616
MC_Reset (Fehler bestätigen), 593
MC_WriteParam (Parameter des Technologieobjekts schreiben), 614
Mehrere AWP-Variablendefinitionen, 862
Meldung
Ende, 911
Länge, 912
Starten, 908
Meldungsempfang konfigurieren
PtP-Beispielprogramm, 944
PtP-Gerät, Konfiguration, 907
Meldungskonfiguration
Anweisungen, 940
Empfangen, 907
Senden, 906
Meldungssendung konfigurieren
PtP-Beispielprogramm, 944
PtP-Gerät, Konfiguration, 906
Memory Card, 1327
Firmware-Update, 146
Funktionsweise, 136
In CPU einfügen, 137
Inkompatibilitätsfehler, 1116
Leere Übertragungskarte für ein verlorenes Passwort, 149
Programmkarte, 143
Übersicht, 136
Übertragungskarte, 140
Memory Card in CPU stecken, 137
MicroMaster-Antrieb, verbinden, 969
MID (mittlere Zeichen einer Zeichenkette lesen), 350
MIN (Minimum abrufen), 252
Mobile Geräte
Webseitenaufbau, 828
Mobiles Gerät, Zugriff auf Webserver, 824
MOD (Divisionsrest einer Division), 249
Modbus
Adressen im Speicher, 974
Funktionscodes, 973
MB_CLIENT (Über PROFINET als Modbus TCP-Client kommunizieren), 1055
MB_COMM_LOAD (Port am PtP-Modul für Modbus RTU konfigurieren), 1074
MB_MASTER (Über PtP-Port als Modbus-Master kommunizieren), 1077
MB_SERVER (Über PROFINET als Modbus TCP-Server kommunizieren), 1062
MB_SLAVE (Über PtP-Port als Modbus-Slave kommunizieren), 1084
Modbus_Comm_Load (SIPLUS I/O oder Port am PtP-Modul für Modbus RTU konfigurieren), 1001
Modbus_Master (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Master kommunizieren), 1004
Modbus_Slave (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Slave kommunizieren), 1011

RTU-Kommunikation, 975
Stationsadressen im Netzwerk, 974
Versionen, 44, 955, 1000, 1039, 1073
MODBUS
MB_CLIENT, 978
MB_SERVER, 986
Modbus RTU
Beispiel Slave, 1020
Master-Programm, 1018
Modbus RTU in älteren Systemen
Beispiel Slave, 1093
Modbus TCP
Versionen, 977, 1054
Modbus_Master (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Master kommunizieren), 1004
Modbus_Slave (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Slave kommunizieren), 1011
Module
Kommunikationsboards (CB), 30
Kommunikationsmodul (CM), 30
Kommunikationsprozessor (CP), 30
Parameter konfigurieren, 173
Signalboard (SB), 30
Signalmodule (SM), 30
Thermischer Bereich, 53, 57
Vergleichstabelle, 29
ModuleStates, 399
ModuleStates Beispiel, 401
Montage
Abmessungen, 57
Abstand, 53
CPU, 60
Erdung, 76
Erweiterungskabel, 68
induktive Lasten, 79
Klemmenblock, 67
Kommunikationsboard (CB), 62
Kommunikationsmodul (CM), 66
Kühlung, 53
Lampenlasten, 78
Luftströmung, 53
Richtlinien, 51
Signalboard (SB), 62
Signalmodul (SM), 64
Thermischer Bereich, 53, 57
Trennung, 76
Übersicht, 58
Verdrahtungsrichtlinien, 75, 77
MOVE (Wert kopieren), 257
MOVE_BLK (Bereich kopieren), 257
MRES, Bedienpanel, 46
MUL (Multiplizieren), 248
MUX (Multiplexen), 318
My Documentation Manager, 4

N

N (Operand auf negative Signalflanke abfragen), 223
N_TRIG (VKE auf negative Signalflanke abfragen), 224
N=-Box und N-Spule (Operand bei negativer Signalflanke setzen), 224
NE_ElemType (Vergleiche, ob Datentyp eines ARRAY-Elements UNGLEICH dem Datentyp einer Variablen ist), 245
NE_Type (Vergleiche, ob Datentyp UNGLEICH dem Datentyp einer Variablen ist), 245
NEG (Zweierkomplement erstellen), 250
Network Time Protocol (NTP), 653
Netzwerkkommunikation, 741
Kabelabschluss, 898
Netzwerkverbindung
Geräte verbinden, 637
Mehrere CPUs, 744, 746, 751, 786, 792
Netzwerkverbindung erstellen
Zwischen PLC-Geräten, 637
Neue Funktionen, 33
Neues Gerät hinzufügen
CPU, 152
Nicht spezifizierte CPU, 155
Vorhandene Hardware erkennen, 155
Nicht gesteckte Module, 49
Nicht spezifizierte CPU, 155
NORM_X (Normalisieren), 284
NOT (VKE invertieren), 219
NOT_NULL (Abfrage nach Pointer UNGLEICH Null), 246
NOT_OK (Ungültigkeit prüfen), 244

O

OB, (Organisationsbaustein)
OB "Ziehen oder Stecken von Modulen", 97
ODER (logische Verknüpfung), 315
Offene Benutzerkommunikation
Herstellen einer Verbindung und Lesen von Daten mit der alten Anweisung TRCV_C, 680
Herstellen einer Verbindung und Lesen von Daten mit TRCV_C, 668
Herstellen einer Verbindung und Senden von Daten mit der alten Anweisung TSEND_C, 680
Herstellen einer Verbindung und Senden von Daten mit TSEND_C, 668
OK (Gültigkeit prüfen), 244

Online
Bedienpanel, 46, 1125
Beobachtungstabelle, 1129, 1130, 1131
Diagnosepuffer, 1126
Forcefunktion, 1136
Forcen, 1135
IP-Adresse, 1122
IP-Adresse zuordnen, 645
Online gehen, 1119
Schaltflächen RUN/STOP, 46
Speicherauslastung, 1125
Startwerte eines DBs zurücksetzen, 1130
Tageszeit, 1122
Tools, 1128
Überwachungsstatus oder -wert, 1129
Vergleichen und synchronisieren, 1127
Werte eines DBs erfassen, 1130
Zustand, 1130
Zykluszeit, 1125
Online- und Diagnose-Tools
Laden im Betriebszustand RUN, 1138
Online- und Offline-CPUs vergleichen und synchronisieren, 1127
Online-Gerätenamen
PROFINET IO, 1120
OPC, Projektierung, 1102
Optimierte Datenbausteine, 186
Optionsverwaltung (Konfigurationssteuerung), 157
Ordner, Sprachen für benutzerdefinierte Webseiten, 883
Organisationsbaustein
Anlaufverarbeitung, 88
Aufrufen, 90
Bearbeiten, 182, 182
Codebausteine im Anwenderprogramm aufrufen, 181
Erstellen, 183
Funktion, 90
Funktionsweise konfigurieren, 183
Knowhow-Schutz, 208
Lineare und strukturierte Programmierung, 179
Mehrere Zyklus-OBs, 183
Prioritätsklassen, 90
Übersicht, 81
Weckalarm, 92
Ortszeit
RD_LOC_T (Lokalzeit lesen), 326
WR_LOC_T (Lokalzeit einstellen), 326
OUT_Range (Wert außerhalb Bereich), 243

P

P (Operand auf positive Signalflanke abfragen), 223
P_TRIG (VKE auf positive Signalflanke abfragen), 224
P=-Box und P-Spule (Operand bei positiver Signalflanke setzen), 223
Panels (HMI), 31
Parameter BUFFER, SEND_P2P, 931
Parameter LENGTH, SEND_P2P, 931
Parameterkonfiguration
Empfangen, 749
LENGH und BUFFER für SEND_P2P, 931
Senden, 748
Parametrieren, 184
Parität, 903
Passive/aktive Kommunikation
Parameter, 663
Partner konfigurieren, 638, 808
Verbindungs-IDs, 659
Passwortschutz
Codebaustein, 208
CPU, 205
ENDIS_PW (Passwort-Legitimierung einschränken und freigeben), 295
Kopierschutz, 209
Leere Übertragungskarte, 149
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209
Verlorenes Passwort, 149
Zugriff auf die CPU, 205
Pause, 907, 908
PEEK, PEEK_WORD, PEEK_BOOL, PEEK_DWORD, PEEK_BLK, 200, 269
Phasenlage, 552
Phasenverschiebung, Weckalarm-OBs, 92
PID
Algorithmus PID_3Step, 482
Algorithmus PID_Compact, 482
Inbetriebnahme, 535
PID_3STEP (PID-Regler mit Einstellung für Ventile), 492
PID_Compact (Universal-PID-Regler mit integrierter Einstellung), 485
PID_Temp (universeller PID-Regler für die Temperaturregelung), 502
Übersicht, 482
PID-Ventileinstellung, 492
PLC
Anlaufverarbeitung, 88
Bausteine aus einer Online-CPU kopieren, 212
Beobachten, 1129
Beobachtungstabelle, 1131
Betriebszustände, 85

Eigenschaft Uhrzeitsynchronisation, 654
Einbau, 58, 60
Entwerfen einer Automatisierungslösung, 177
Erweiterungskabel, 68
Forcefunktion, 1136
Forcen, 1135
Gerätekonfiguration, 151
HSC-Konfiguration, 480
Klemmenblock, 67
Knowhow-Schutz, 208
Kommunikationslast, 105
Laden in CPU, 211
Leistungsbilanz, 54
Memory Card, 136
Module hinzufügen, 156
RTM (Betriebsstundenzähler), 331
Übersicht über die CPU, 25
Variablen, 112
Vergleichen und synchronisieren, 1127
Zuweisen einer IP-Adresse zu einer Online-CPU, 645
Zykluszeit, 104, 105
Zykluszeit, 104, 105
PLC-Gerät
Bausteine verwenden, 179
PM 1207 Stromversorgungsmodul, 1331
Podcasts, 4
Pointer
Datentyp Pointer, 129
Datentyp Variant, 132
Datenyp Any, 130
Übersicht über Pointer, 128
POKE, POKE_BOOL, POKE_BLK, 200, 269
PORT_CFG (Kommunikationsparameter dynamisch konfigurieren), 1022
Port_Config (Portkonfiguration), 918
Portalansicht, 39
Portkonfiguration, 902
Error, 920, 1024
PtP-Beispielprogramm, 943
Portnummern
eingeschränkt, 740
Zu Kommunikationspartnern zuweisen, 656
Potentiometermodul
Technische Daten, 1329
Priorität
Priorität bei der Verarbeitung, 100
Prioritätsklassen, 90
PROFIBUS
Adresse, 787
Adresse konfigurieren, 787
Anweisungen für die dezentrale Peripherie, 356
Anzahl der Kommunikationsverbindungen, 634
CM 1242-5 (DP-Slave), 782
CM 1243-5 (DP-Master), 782
CM 1243-5 (DP-Master) hinzufügen, 785
DPNRM_DG (Diagnosedaten eines DP-Slaves lesen), 371
DPRD_DAT (Konsistente Daten eines DP-Normslaves lesen), 368
DP-Slave hinzufügen, 785
DPWR_DAT (Konsistente Daten eines DP-Normslaves schreiben), 368
GET (Daten aus entfernter CPU auslesen), 802
Master, 782
Netzwerkverbindung, 637, 786
PUT (Daten in entfernte CPU schreiben), 802
RALRM (Alarm empfangen), 360
RDREC (Datensatz lesen), 357
S7-Verbindung, 807
Slave, 782
WRREC (Datensatz schreiben), 357
PROFIBUS und PROFINET
Beispiel DeviceStates, 395
ModuleStates Beispiel, 401
Profil-OB, 100
PROFINET
Ad-hoc-Modus, 659
Anweisungen für die dezentrale Peripherie, 356
Anzahl der Kommunikationsverbindungen, 634
CPU-zu-CPU-Kommunikation, 745
DPRD_DAT (Konsistente Daten eines DP-Normslaves lesen), 368
DPWR_DAT (Konsistente Daten eines DP-Normslaves schreiben), 368
Eigenschaft Uhrzeitsynchronisation, 654
Eigenschaften der Ethernet-Adresse, 648
Gerätenamen und Adressierung, 655
GET (Daten aus entfernter CPU auslesen), 802
IP-Adresse, 646
IP-Adresse konfigurieren, 168
Kommunikation PLC/PLC, 745
Kommunikation zwischen CPU und HMI-Gerät konfigurieren, 743
Kommunikationsarten, 631
MAC-Adresse, 646
Netzwerkverbindung, 637, 744, 746, 751
PUT (Daten in entfernte CPU schreiben), 802
RALRM (Alarm empfangen), 360
RDREC (Datensatz lesen), 357
S7-Verbindung, 807
Startzeit des Systems, 655
Testen eines Netzwerks, 650
Übersicht, 656

Uhrzeitsynchronisierung, 168
Verbindung beenden und neu aufbauen, 707
Verbindungs-IDs, 659
WRREC (Datensatz schreiben), 357
Zuweisung der IP-Adresse, 655
PROFINET IO
CPU zuweisen, 751
Geräte, 750
Gerätenamen, 751
Gerätenamen online zuweisen, 1120
Gerätenamen zuweisen, 751
Hinzufügen eines Geräts, 750
Online-Gerätenamen, 1120
PROFINET RT, 656
PROFINET-Anweisungen
Alte Anweisung TRCV_C (Daten über Ethernet empfangen (TCP)), 680
Alte Anweisung TSEND_C (Daten über Ethernet senden (TCP)), 680
Alte Anweisungen TCON, TDISCON, TSEND und TRCV, 698
T_CONFIG (Schnittstelle konfigurieren), 731
T_DIAG, 709
T_RESET, 707
TCON, 688
TDISCON, 688
TRCV, 688
TRCV_C, 668, 748
TSEND, 688
TSEND_C, 668
TURCV (Daten über Ethernet empfangen (UDP)), 725
TUSEND (Daten über Ethernet senden (UDP)), 725
Programm
Bausteine aus einer Online-CPU kopieren, 212
Codebausteine im Anwenderprogramm aufrufen, 181
Laden in CPU, 211
Lineare und strukturierte Programme, 179
Memory Card, 136
Organisationsbausteine (OBs), 182
Passwortschutz, 208
Prioritätsklassen, 90
Startwerte eines DBs zurücksetzen, 1130
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209
Werte eines DBs erfassen, 1130
Programm beobachten, 213
Programm testen, 213
Programmausführung, 81
Programmiereditor
Beobachten, 1130
Startwerte eines DBs zurücksetzen, 1130
Werte eines DBs erfassen, 1130
Zustand, 1130
Programmierung
Algorithmus PID_3Step, 482
Algorithmus PID_Compact, 482
Anfangswert eines FB, 184
Anzahl der Codebausteine, 81
Bausteinaufrufe, 81
Betriebszustände der CPU, 85
Codebausteine im Anwenderprogramm aufrufen, 181
Codebausteine vergleichen und synchronisieren, 1127
Datenbaustein (DB), 81
Drag & Drop zwischen Editoren, 45
Einfügen von Anweisungen, 40
Ergänzen von Eingängen oder Ausgängen in KOP- und FUP-Anweisungen, 43
Erweiterbare Anweisungen, 43
Favoriten, 40
Funktion (FC), 184
Funktionsbaustein (FB), 81, 184
FUP (Funktionsplan), 194
Gültige FC-, FB- und DB-Nummern, 81
Instanz-Datenbaustein (DB), 184
KOP (Kontaktplan), 193
Lineares Programm, 179
Nicht gesteckte Module, 49
Nicht spezifizierte CPU, 155
PID_3STEP (PID-Regler mit Einstellung für Ventile), 492
PID_Compact (Universal-PID-Regler mit integrierter Einstellung), 485
PID_Temp (universeller PID-Regler für die Temperaturregelung), 502
Prioritätsklassen, 90
PtP-Anweisungen, 940
RTM (Betriebsstundenzähler), 331
SCL (Structured Control Language), 195, 195, 196
Signalfluss (EN und ENO), 202
Strukturiertes Programm, 179
Systemzeit, 326
Übersicht über den PID-Regler, 482
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209
Programminformationen
In der Aufrufstruktur, 215
Programmkarte
Anlaufparameter konfigurieren, 139
Anlegen, 143
Funktionsweise, 136

In CPU einfügen, 137
Übersicht, 136
Programmsteuerung (SCL), 305
CASE, 307
CONTINUE, 312
EXIT, 313
FOR, 308
GO TO, 314
IF-THEN, 306
REPEAT, 311
RETURN, 314
WHILE, 310
Programmstruktur, 181
Programmzyklus-OB, 90
Projekt
Codebaustein schützen, 208
Laden in CPU, 211
Leere Übertragungskarte, 149
Programmkarte, 143
Übertragungskarte, 140
Vergleichen und synchronisieren, 1127
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209
Verlorenes Passwort, 149
Zugang zur CPU einschränken, 205
Zugriffsschutz, 205
Projektansicht, 39, 39
Protokoll
Frei programmierbare Kommunikation, 900
ISO on TCP, 656
Kommunikation, 900
Modbus, 900
PROFINET RT, 656
TCP, 656
UDP, 656
USS, 900
Protokoll der frei programmierbaren Kommunikation, 900
Protokolle, Kommunikation, 658
Prozessabbild
Beobachten, 1130
Forcefunktion, 1136
Forcen, 1135
Überwachungsstatus oder -wert, 1129
Zustand, 1130, 1135
Prozessalarm-OB, 93
Prüfen der Verbindung, 709
PTO (Impulsfolge)
CTRL_PWM (Impulsdauermodulation), 413
Funktionsweise, 414
Impulskanäle konfigurieren, 416
Kann nicht geforct werden, 1136
PtP-Anweisungen Rückgabewerte, 915
PtP-Fehlerklassen, 917, 1022
PtP-Kommunikation, 900
Anschlüsse konfigurieren, 902
Beispielprogramm, 942
Beispielprogramm, Ausführen, 951
Beispielprogramm, STEP 7-Programmierung, 949
Konfiguration des Beispielprogramms, 943
Parameter konfigurieren, 906
Programmieren, 940
Terminalemulator für Beispielprogramm, 951
Punkt-zu-Punkt-Kommunikation, 900
Punkt-zu-Punkt-Programmierung, 940
PUT (Daten in entfernte CPU schreiben), 802
Verbindung konfigurieren, 640
PWM (Impulsdauermodulation)
CTRL_PWM (Impulsdauermodulation), 413
Funktionsweise, 414
Impulskanäle konfigurieren, 416
Kann nicht geforct werden, 1136

Q

QRY_CINT (Weckalarmparameter abfragen), 380
QRY_DINT (Status des Verzögerungsalarms abfragen), 386
QRY_TINT (Status des Uhrzeitalarms abfragen), 385
Querverweis zum Anzeigen der Verwendung, 214

R

R (Ausgang rücksetzen), 220
R_TRIG (Variable bei positiver Signalflanke setzen), 225
RALRM (Alarm empfangen), 360, 364
RCV_CFG (Serielle Empfangsparameter dynamisch konfigurieren), 1025
RCV_PTP (Empfangsmeldungen aktivieren), 1033
RCV_RST (Empfangspuffer löschen), 1035
RD_ADDR (E/A-Adressen über Hardwarekennung ermitteln), 461
RD_LOC_T (Lokalzeit lesen), 326
RD_SYS_T (Uhrzeit lesen), 326
RDREC (Datensatz lesen), 357, 364
RE_TRIGR (Zyklusüberwachungszeit neu starten), 297
READ_BIG (Daten in Big-Endian-Format lesen), 271
READ_DBL (Aus Datenbaustein im Ladespeicher lesen), 453
READ_LITTLE (Daten in Little-Endian-Format lesen), 271
Receive_Config (Empfangskonfiguration), 923

Receive_P2P (Punkt-zu-Punkt-Daten empfangen), 932
Receive_Reset (Empfangspuffer löschen), 934
Remanente Bausteinvariablen
  Laden im Betriebszustand RUN, 1145
Remanenter Speicher, 26, 106
  CPU 1211C, 1163
  CPU 1212C, 1175
  CPU 1214C, 1187
  CPU 1215C, 1199
  CPU 1217C, 1213
REPEAT, SCL, 311
REPLACE (Zeichen in einer Zeichenkette ersetzen), 354
RESET_BF (Bitfeld rücksetzen), 221
RET (Rückgabewert), 294
RETURN, SCL, 314
Rezept
  Beispielprogramm, 428
  DB-Struktur, 420
  RecipeExport (Rezeptexport), 424
  RecipeImport (Rezeptimport), 426
  Übersicht, 419
Rezeptprogramm Beispiel, 428
Richtlinien
  Einbau der CPU, 60
  Erdung, 76
  induktive Lasten, 79
  Installation, 51
  Lampenlasten, 78
  Trennung, 76
  Verdrahtungsrichtlinien, 75, 77
  Vorgehensweise für den Einbau, 58
Richtlinien für Potentialtrennung, 76
RIGHT (rechte Zeichen einer Zeichenkette lesen), 350
ROL (Links rotieren) und ROR (Rechts rotieren), 322
ROUND (Zahl runden), 282
RS (Flipflop rücksetzen/setzen), 221
RS232- und RS485-Kommunikationsmodule, 897
RS485-Stecker
  Abschluss, 898
RT (Zeit zurücksetzen), 227
RTS (Hardware-Flusskontrolle, PtP), 904
RTS-Einschaltverzögerung, - Ausschaltverzögerung, 906
Ruckbegrenzung, 588
Rückgabewerte
  Anweisungen für die offene Benutzerkommunikation, 739
  PtP-Anweisungen, 915
Rückgabewerte der Anweisungen für die offene Benutzerkommunikation, 739
Rückgabewerte Empfangs-Laufzeit, 932, 1033
RUNTIME (Programmlaufzeit messen), 303

S

S (Ausgang setzen), 220
S_CONV (Zeichenkette umwandeln), 334
S_MOV (Zeichenkette verschieben), 333
S7-Kommunikation
  Verbindung konfigurieren, 640
SCALE_X (Skalieren), 284
Schachtelungstiefe, 81
Schaltflächen RUN/STOP, 46
Schaltpläne
  CB 1241 RS 485, 1323
  CPU 1211C, 1171
  CPU 1212C, 1183
  CPU 1214C, 1195
  CPU 1215C, 1208
  CPU 1217C, 1224
  SB 1221, 1283
  SB 1222, 1286
  SB 1223, 1289, 1292
  SB 1231, 1294
  SB 1231 RTD, 1306
  SB 1231 Thermoelement, 1303
  SB 1232, 1296
  SM 1221, 1229
  SM 1222, 1233
  SM 1223, 1239, 1243
  SM 1231, 1245
  SM 1231 RTD, 1263
  SM 1231 Thermoelement, 1257
  SM 1232, 1249
  SM 1234, 1252
  SM 1278 IO-Link-Master, 1270
Scheibe (eines Variablendatentyps), 132
Schließer-/Öffnerkontakt, 217
Schließer-/Öffnerspule, 219
Schneller Zähler, 465
  Funktionsweise, 471
  Kann nicht geforct werden, 1136
  Konfiguration, 480
Schnittstellenkonfiguration
  Anweisungen, 940
Schreiben in DBs, E/A oder Speicher, 200, 269
Schutz
  CPU, 205
  Knowhow-Schutz für einen Codebaustein, 208
Schutzart, 1159
Schutzbeschaltungen für induktive Lasten, 79
Schutzgrad, 1159

Schutzstufe
Codebaustein, 208
CPU, 205
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209
Verlorenes Passwort, 149
SCL (Structured Control Language)
ABS (Absolutwert bilden), 251
ACOS (Arcuscosinuswert bilden), 254
ACT_TINT (Uhrzeitalarm aktivieren), 384
ADD (Addieren), 248
Adressierung, 196
Algorithmus PID_3Step, 482
Algorithmus PID_Compact, 482
Alte Anweisung USS_DRV (Daten mit Antrieb tauschen), 1044
Alte Anweisung USS_PORT (Kommunikation über USS-Netzwerk bearbeiten), 1042
Alte Anweisung USS_RPM (Parameter aus dem Antrieb auslesen), 1047
Alte Anweisung USS_WPM (Parameter im Antrieb ändern), 1049
Anweisung DeviceStates (Modulstatus eines E/A-Systems lesen), 393
Anweisung GEO2LOG (Hardwarekennung über Steckplatz ermitteln), 457
Anweisung Get_IM_Data (Identifikations- und Wartungsdaten lesen), 411
Anweisung ModuleStates (Modulstatusinformationen eines Moduls lesen), 399
ASIN (Arcussinuswert bilden), 254
ATAN (Arcustangenswert bilden), 254
ATH (ASCII-Zeichenkette in Hexadezimalzahl umwandeln), 345
ATTACH (OB einem Alarmereignis zuweisen), 374
Ausdrücke, 196
Bausteine aufrufen, 181
Bedingungen, 196
Bitverknüpfung, 217
CAN_DINT (Verzögerungsalarm abbrechen), 386
CAN_TINT (Uhrzeitalarm löschen), 383
CASE, 307
CEIL (Aus Gleitpunktzahl nächsthöhere Ganzzahl erzeugen), 283
Chars_TO_Strg (Array aus CHAR in Zeichenkette umwandeln), 343
CONCAT (Zeichenketten verketten), 349
CONTINUE, 312
CONV (Wert umwandeln), 278
COS (Cosinuswert bilden), 254
CountOfElements (Anzahl ARRAY-Elemente abfragen), 274
CTD (Rückwärtszählen), 235
CTRL_HSC (Schnellen Zähler steuern), 465
CTRL_PWM (Impulsdauermodulation), 413
CTU (Vorwärtszählen), 235
CTUD (Vorwärts- und Rückwärtszählen), 235
DataLogClose (Datenprotokoll schließen), 441
DataLogCreate (Datenprotokoll erstellen), 432
DataLogNewFile (Datenprotokoll in neuer Datei), 442
DataLogOpen (Datenprotokoll öffnen), 436
DataLogWrite (Datenprotokoll schreiben), 438
DEC (Dekrementieren), 251
DECO (Decodieren), 316
DELETE (Zeichen in einer Zeichenkette löschen), 351
Deserialize, 260
DETACH (Zuweisung eines OB zu einem Alarmereignis aufheben), 374
DIS_AIRT (Ausführung von Alarmen höherer Priorität und asynchronen Fehlerereignissen deaktivieren), 389
DIV (Dividieren), 248
EN und ENO (Signalfluss), 202
EN_AIRT (Ausführung von Alarmen höherer Priorität und asynchronen Fehlerereignissen aktivieren), 389
ENCO (Encodieren), 316
EXIT, 313
EXP (Exponentialwert bilden), 254
EXPT (Potenzieren), 254
F_TRIG (Variable bei negativer Signalflanke setzen), 225
FB oder FC aufrufen, 196
FILL_BLK (Speicher mit Bitmuster belegen), 266
FIND (Zeichen in einer Zeichenkette finden), 355
FLOOR (Aus Gleitpunktzahl nächstniedere Ganzzahl erzeugen), 283
FOR, 308
FRAC (Nachkommastellen ermitteln), 254
GET (Daten aus entfernter CPU auslesen), 802
GET_DIAG (Diagnoseinformationen lesen), 405
GOTO, 314
HTA (Hexadezimalzahl in ASCII-Zeichenkette umwandeln), 345
IF-THEN, 306
IN_Range (Wert innerhalb Bereich), 243
INC (Inkrementieren), 251
INSERT (Zeichen in einer Zeichenkette einfügen), 352
INV (Einerkomplement erstellen), 316

JMP_LIST (Sprungliste definieren), 291
LED (LED-Status lesen), 392
LEFT (linke Zeichen einer Zeichenkette lesen), 350
LEN (Länge einer Zeichenkette ermitteln), 348
LIMIT (Grenzwert setzen), 253
LN (Natürlichen Logarithmus bilden), 254
LOG2GEO (Steckplatz über Hardwarekennung ermitteln), 459
MAX (Maximum abrufen), 252
MAX_LEN (Maximale Länge einer Zeichenkette), 348
MB_CLIENT (Über PROFINET als Modbus TCP-Client kommunizieren), 1055
MB_COMM_LOAD (Port am PtP-Modul für Modbus RTU konfigurieren), 1074
MB_MASTER (Über PtP-Port als Modbus-Master kommunizieren), 1077
MB_SERVER (Über PROFINET als Modbus TCP-Server kommunizieren), 1062
MB_SLAVE (Über PtP-Port als Modbus-Slave kommunizieren), 1084
MC_ChangeDynamic (Dynamikeinstellungen der Achse ändern), 612
MC_CommandTable, 609
MC_Halt (Achse pausieren), 597
MC_Home (Referenzpunktfahrt der Achse durchführen), 594
MC_MoveAbsolute (Achse absolut positionieren), 599
MC_MoveJog (Achse im Tippbetrieb bewegen), 606
MC_MoveRelative (Achse relativ positionieren), 601
MC_MoveVelocity (Achse mit vordefinierter Geschwindigkeit bewegen), 603
MC_Power (Achse freigeben/sperren), 590
MC_ReadParam (Parameter eines Technologieobjekts lesen), 616
MC_Reset (Fehler bestätigen), 593
MC_WriteParam (Parameter des Technologieobjekts schreiben), 614
MID (mittlere Zeichen einer Zeichenkette lesen), 350
MIN (Minimum abrufen), 252
MOD (Divisionsrest einer Division), 249
Modbus_Comm_Load (SIPLUS I/O-Port am PtP-Modul für Modbus RTU konfigurieren), 1001
Modbus_Master (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Master kommunizieren), 1004
Modbus_Slave (Über SIPLUS I/O oder den PtP-Port als Modbus RTU-Slave kommunizieren), 1011
MOVE (Wert kopieren), 257
MOVE_BLK (Bereich kopieren), 257
MUL (Multiplizieren), 248
MUX (Multiplexen), 318
NEG (Zweierkomplement erstellen), 250
NORM_X (Normalisieren), 284
NOT_OK (Ungültigkeit prüfen), 244
ODER (logische Verknüpfung), 315
OK (Gültigkeit prüfen), 244
Operatoren, 196
OUT_Range (Wert außerhalb Bereich), 243
PEEK- und POKE-Varianten, 200, 269
PID_3STEP (PID-Regler mit Einstellung für Ventile), 492
PID_Compact (Universal-PID-Regler mit integrierter Einstellung), 485
PID_Temp (universeller PID-Regler für die Temperaturregelung), 502
PORT_CFG (Kommunikationsparameter dynamisch konfigurieren), 1022
Priorität von Operatoren, 196
Programmiereditor, 195
Programmsteuerung, 305
PUT (Daten in entfernte CPU schreiben), 802
QRY_CINT (Weckalarmparameter abfragen), 380
QRY_DINT (Status des Verzögerungsalarms abfragen), 386
QRY_TINT (Status des Uhrzeitalarms abfragen), 385
R_TRIG (Variable bei positiver Signalflanke setzen), 225
RCV_CFG (Serielle Empfangsparameter dynamisch konfigurieren), 1025
RCV_PTP (Empfangsmeldungen aktivieren), 1033
RCV_RST (Empfangspuffer löschen), 1035
RD_ADDR (E/A-Adressen über Hardwarekennung ermitteln), 461
RD_LOC_T (Lokalzeit lesen), 326
RD_SYS_T (Uhrzeit lesen), 326
READ_BIG (Daten in Big-Endian-Format lesen), 271
READ_DBL (Aus Datenbaustein im Ladespeicher lesen), 453
READ_LITTLE (Daten in Little-Endian-Format lesen), 271
RecipeExport (Rezeptexport), 424
RecipeImport (Rezeptimport), 426
REPEAT, 311
REPLACE (Zeichen in einer Zeichenkette ersetzen), 354
RETURN, 314
RIGHT (rechte Zeichen einer Zeichenkette lesen), 350

ROL (Links rotieren) und ROR (Rechts rotieren), 322
ROUND (Zahl runden), 282
RTM (Betriebsstundenzähler), 331
RUNTIME (Programmlaufzeit messen), 303
S_CONV (Zeichenkette umwandeln), 334
S_MOV (Zeichenkette verschieben), 333
SCALE_X (Skalieren), 284
SEL (Selektieren), 318
SEND_CFG (Serielle Übertragungsparameter dynamisch konfigurieren), 1024
SEND_PTP (Sendepufferdaten übertragen), 1030
Serialize, 263
SET_CINT (Weckalarm parametrieren), 378
SET_TIMEZONE (Zeitzone setzen), 330
SET_TINTL (Datum und Uhrzeit eines Uhrzeitalarms festlegen), 382
SGN_GET (RS232-Signale abfragen), 1036
SGN_SET (RS232-Signale festlegen), 1037
SHL (Links schieben) und SHR (Rechts schieben), 321
SIN (Sinuswert bilden), 254
SQR (Quadrat bilden), 254
SQRT (Quadratwurzel bilden), 254
SRT_DINT (Verzögerungsalarm starten), 386
Steuerungsanweisungen, 196, 305, 306, 307, 308, 310, 311, 312, 313, 314, 314
Strg_TO_Chars (Zeichenkette in Array aus CHAR umwandeln), 343
STRG_VAL (Zeichenkette in Zahlenwert umwandeln), 334
SUB (Subtrahieren), 248
SWAP (Anordnung ändern), 268
SWITCH (Sprungverteilung), 292
T_ADD (Zeiten addieren), 324
T_COMBINE (Zeiten verknüpfen), 325
T_CONV (Zeiten umwandeln und extrahieren), 323
T_DIFF (Zeitdifferenz), 325
T_SUB (Zeiten subtrahieren), 324
TAN (Tangenswert bilden), 254
TM_MAIL (E-Mail senden), 1107
TRUNC (Ganzzahl erzeugen), 282
Übersicht, 195
Übersicht über den PID-Regler, 482
UFILL_BLK (Speicher ununterbrechbar mit Bitmuster belegen), 266
UMOVE_BLK (Bereich ununterbrechbar kopieren), 257
Umwandlungsanweisungen, 279
UND (logische Verknüpfung), 315
USS_Drive_Control (Daten mit Antrieb tauschen), 960
USS_Port_Scan (Kommunikation über USS-Netzwerk bearbeiten), 959
USS_Read_Param (Parameter aus dem Antrieb auslesen), 963
USS_Write_Param (Parameter im Antrieb ändern), 964
VAL_STRG (Zahlenwert in Zeichenkette umwandeln), 334
Var-Abschnitt, 195
VariantGet (VARIANT Variablenwert lesen), 272
VariantPut (VARIANT Variablenwert schreiben), 273
Werte vergleichen, 242
WHILE, 310
WR_LOC_T (Lokalzeit einstellen), 326
WR_SYS_T (Uhrzeit einstellen), 326
WRIT_DBL (In Datenbaustein im Ladespeicher schreiben), 453
WRITE_BIG (Daten in Big-Endian-Format schreiben), 271
WRITE_LITTLE (Daten in Little-Endian-Format schreiben), 271
WWW (benutzerdefinierte Webseiten synchronisieren), 866
XOR (logische Verknüpfung), 315
Zeiten, 227
SEL (Selektieren), 318
SEND_CFG (Serielle Übertragungsparameter dynamisch konfigurieren), 1024
Send_Config (Sendekonfiguration), 920
Send_P2P (Punkt-zu-Punkt-Daten senden), 928
SEND_P2P (Punkt-zu-Punkt-Daten senden)
Parameter LENGH und BUFFER, 931
SEND_PTP (Sendepufferdaten übertragen), 1030
Serialize, 263
Serielle Kommunikation, 900
Service und Support, 3
SET_BF (Bitfeld setzen), 221
SET_CINT (Weckalarm parametrieren), 378
Set_Features (Erweiterte Funktionen setzen), 938
SET_TIMEZONE (Zeitzone setzen), 330
SET_TINTL (Datum und Uhrzeit eines Uhrzeitalarms festlegen), 382
SGN_GET (RS232-Signale abfragen), 1036
SGN_GET (RS232-Signale abrufen), 935
SGN_SET (RS232-Signale festlegen), 1037
Shared Device
Konfiguration, 765
Konzept, 762
Shared I-Device, Konfiguration, 771
SHL (Links schieben) und SHR (Rechts schieben), 321
SHL, SCL, 321

SHR, SCL, 321
Sicherheit
Kopierschutz, 209
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209
Verlorenes Passwort, 149
Zugriffsschutz, 205
Siemens-Sicherheitszertifikat importieren, 894
Siemens-Sicherheitszertifikat, Webseiten, 833, 894
Signal_Set (RS232-Signale setzen), 936
Signalboards (SB)
Ausbau, 62
Darstellung Analogausgang (Spannung), 1255, 1298
Darstellung Analogausgang (Strom), 1255, 1299
Darstellung Eingang (Spannung), 1254, 1297
Darstellung Eingang (Strom), 1254, 1298
Einbau, 62
Konfiguration von Parametern, 173
Leistungsbedarf, 1335
Module hinzufügen, 156
SB 1221 DI 4 x 24 V DC, 200 kHz, 1282
SB 1221 DI 4 x 5 V DC, 200 kHz, 1282
SB 1222 DO 4 x 24 V DC, 200 kHz, 1284
SB 1222 DO 4 x 5 V DC, 200 kHz, 1284
SB 1223 DI 2 x 24 VDC / DO 2 x 24 VDC, 200 kHz, 1287
SB 1223 DI 2 x 24 VDC, DO 2 x 24 V DC, 1290
SB 1223 DI 2 x 5 VDC / DO 2 x 5 VDC, 200 kHz, 1287
SB 1231 AI 1 x 12 Bit, 1293
SB 1231 AI 1 x 16 Bit RTD, 1304
SB 1231 AI 1 x 16 Bit Thermoelement, 1300
SB 1232 AO 1 x 12 Bit, 1295
Übersicht, 30
Signalmodule (SM)
Ausbau, 65
Darstellung Analogausgang (Spannung), 1255, 1298
Darstellung Analogausgang (Strom), 1255, 1299
Darstellung Analogeingang (Spannung), 1254, 1297
Darstellung Analogeingang (Strom), 1254, 1298
Einbau, 64
Erweiterungskabel, 68
Konfiguration von Parametern, 173
Leistungsbedarf, 1335
Module hinzufügen, 156
Schrittantwortzeiten, 1253
SM 1221 DI 16 x 24 V DC, 1228
SM 1221 DI 8 x 24 V DC, 1228
SM 1222 DO 16 x 24 V DC, 1232
SM 1222 DO 16 x Relais, 1232
SM 1222 DO 8 x 24 V DC, 1230
SM 1222 DO 8 x Relais, 1230
SM 1222 DO 8 x Relais (Umschaltung), 1230
SM 1223 DI 16 x 24 VDC, DO 16 x 24 V DC, 1237
SM 1223 DI 16 x 24 VDC, DO 16 x Relais, 1237
SM 1223 DI 8 x 120/230 V AC / DO 8 x Relais, 1241
SM 1223 DI 8 x 24 VDC, DO 8 x 24 V DC, 1237
SM 1223 DI 8 x 24 VDC, DO 8 x Relais, 1237
SM 1231 AI 4 x 13 Bit, 1244
SM 1231 AI 4 x 16 Bit, 1244
SM 1231 AI 4 x 16 Bit TC, 1256
SM 1231 AI 4 x RTD x 16 Bit, 1262
SM 1231 AI 8 x 13 Bit, 1244
SM 1231 AI 8 x 16 Bit TC, 1256
SM 1231 AI 8 x RTD x 16 Bit, 1262
SM 1232 AO 2 x 14 Bit, 1248
SM 1232 AO 4 x 14 Bit, 1248
SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit, 1250
SM 1278 4xIO-Link-Master, 1268
Übersicht, 30
Signalverarbeitungsfehler, 936, 937, 1036, 1038
Simulatoren, 1327
SIN (Sinuswert bilden), 254
SM 1231 RTD
Auswahltabellen, 1265, 1307
SM und SB
Gerätekonfiguration, 151, 151
Vergleichstabelle, 29
Smartphone, Zugriff auf Webserver, 824
SMS, 1101
Snubber-Schutzbeschaltungen für induktive Lasten, 79
Software-Flusskontrolle, 905
Sommerzeit TimeTransformationRule, 329
Sonderzeichen
Benutzerdefinierte Webseiten, 862
Spalten und Überschriften in Taskcards, 44
Spannungsversorgungsmodul
PM1207, 1331
Speicher
A (Prozessabbild der Ausgänge), 115
Adressen der Peripherieeingänge (Forcetabelle), 1135
Arbeitsspeicher, 106
Beobachten der Speicherauslastung, 1125
E (Prozessabbild der Eingänge), 114
L (Lokaldatenspeicher), 112
Ladespeicher, 106
M (Bitmerker), 116
Remanenter Speicher, 106
Systemspeicher, 108

Taktmerker, 108
Temporärer Speicher, 116
Speicherbereich A
Impulsausgänge, 414
Impulskanäle konfigurieren, 416
Speicherbereich E
Adressen der Peripherieeingänge (Forcetabelle), 1135
Beobachten, 1129
Beobachtungstabelle, 1129
Forcefunktion, 1136, 1136
Forcen, 1135
Forcetabelle, 1135
KOP beobachten, 1130
Speicherbereiche
Adressierung von Booleschen Werten oder Bitwerten, 113
Direkter Zugriff, 112
Prozessabbild, 112
Speicherkarte
Anlaufparameter konfigurieren, 139
Verlorenes Passwort, 149
Sprachen wechseln, benutzerdefinierte Webseiten, 883
Sprachen, benutzerdefinierte Webseiten, 883
Sprachumschaltung für benutzerdefinierte Webseiten programmieren, 883
Spulen, (Siehe Bitverknüpfung)
SQR (Quadrat bilden), 254
SQRT (Quadratwurzel bilden), 254
SR (Flipflop setzen/rücksetzen), 221
SRT_DINT (Verzögerungsalarm starten), 386
Standard-Datenbausteine, 186
Standard-Hutschiene, 58
Standard-Maschinenprojekte, (Konfigurationssteuerung (Optionenhandling))
Standard-Webseite "Datenprotokolle", 845
Standard-Webseite "Diagnose", 836
Standard-Webseite "Einführung", 833
Standard-Webseite "Identifikation", 835
Standard-Webseite "Kommunikation", 840
Standard-Webseite "Modulinformationen", 837
Standard-Webseite "Start", 834
Standard-Webseite "Variablenzustand", 841
Standard-Webseiten, 817
An- und Abmelden, 830
Aufbau, 827
Betriebszustand wechseln, 834
Cookie-Einschränkungen, 894
Datenprotokolle, 845
Diagnose, 836
Einführung, 833
Identifikation, 835
JavaScript-Einschränkungen, 893
Kommunikation, 840
Modulinformationen, 837
Sicherer Zugriff, 824
Start, 834
Variablenzustand, 841
Zugriff über PC, 823
Startbedingungen, 908
STARTUP, Betriebszustand
Forcefunktion, 1136
Startwerte eines DBs zurücksetzen, 1130
STEP 7
Ändern von Einstellungen, 44
Anfangswert eines FB, 184
Anlaufverarbeitung, 88
Anzahl der Codebausteine, 81
ASi,
AS-i-Port, 792
Bausteinaufrufe, 81
Bausteine aus einer Online-CPU kopieren, 212
Bedienpanel, 46
Beobachten, 1129, 1130
Betriebszustände, 85
Codebausteine im Anwenderprogramm aufrufen, 181
CPU konfigurieren, 168
Datenbaustein (DB), 81
Drag & Drop zwischen Editoren, 45
Eigenschaft Uhrzeitsynchronisation (PROFINET), 654
Einfügen von Anweisungen, 40
Ergänzen von Eingängen oder Ausgängen in einer KOP- oder FUP-Anweisung, 43
Erweiterbare Eingänge und Ausgänge, 43
Ethernet-Port, 646
Favoriten, 40
Forcefunktion, 1136
Forcen, 1135
Funktion (FC), 184
Funktionsbaustein (FB), 81, 184
Funktionsweise, 1131
Gerätekonfiguration, 151
Gültige FC-, FB- und DB-Nummern, 81
HSC-Konfiguration, 480
Instanz-Datenbaustein (DB), 184
Kommunikationslast, 105
Laden in CPU, 211
Lineare und strukturierte Programme, 179
Memory Card, 136
Module hinzufügen, 156
Module konfigurieren, 173
Netzwerkverbindung, 637

Neues Gerät hinzufügen, 152
Nicht gesteckte Module, 49
Passwortschutz, 208
Portalansicht und Projektansicht, 39
Prioritätsklasse (OB), 90
PROFIBUS, 787
PROFINET IO-Gerät hinzufügen, 750
PROFINET-Port, 646
Programmkarte, 136
RTM (Betriebsstundenzähler), 331
Schaltflächen RUN/STOP, 46
Startwerte eines DBs zurücksetzen, 1130
Vergleichen und synchronisieren, 1127
Werte eines DBs erfassen, 1130
Zuweisen einer IP-Adresse zu einer Online-CPU, 645
Zykluszeit, 104, 105
Zykluszeit, 104, 105
STEP 7 Webseiten, 4
STEP 7-Programmierung
Benutzerdefinierte Webseiten, 866
PtP-Beispielprogramm, 949
Steuer-DB für benutzerdefinierte Webseiten
Anforderungsbefehle und -zustände, 887
Globale Befehle, 887
Parameter WWW-Anweisung, 866
Stoppbits, 903
Stoßwellenfestigkeit, 1156
STP (Programm beenden), 298
Strg_TO_Chars (Zeichenkette in Array aus CHAR umwandeln), 343
STRG_VAL (Zeichenkette in Zahlenwert umwandeln), 334
String
Datentyp String, 333
Datentyp STRING, 125
S_MOV (Zeichenkette verschieben), 333
Übersicht über Zeichenkettenoperationen, 347
Strukturierte Programmierung, Bausteinstruktur, 179
SUB (Subtrahieren), 248
Subnetzmaske, 647
Support, 3
SWAP (Anordnung ändern), 268
SWITCH (Sprungverteilung), 292
Synchronisation
Eigenschaft Uhrzeitsynchronisation (PROFINET), 654
Systemspeicherbyte, 109
Systemuhr
RD_SYS_T (Uhrzeit lesen), 326
WR_LOC_T (Lokalzeit einstellen), 326
WR_SYS_T (Uhrzeit einstellen), 326
Systemvoraussetzungen, 38

T

T_ADD (Zeiten addieren), 324
T_COMBINE (Zeiten verknüpfen), 325
T_CONFIG (Schnittstelle konfigurieren), 731
T_CONV (Zeiten umwandeln und extrahieren), 323
T_DIAG, 709
T_DIFF (Zeitdifferenz), 325
T_RESET, 707
T_SUB (Zeiten subtrahieren), 324
Tablet, Zugriff auf Webserver, 824
Tageszeit
CPU online konfigurieren, 1122
Taktmerkerbyte, 110
TAN (Tangenswert bilden), 254
Taskcards
Spalten und Überschriften, 44, 667, 679, 687, 697, 955, 977, 1000, 1039, 1054, 1073
TCON, 688
Konfiguration, 638
Verbindungs-IDs, 659
Verbindungsparameter, 663
TCON, TDISCON, TSEND und TRCV
Versionen, 687, 697
TCON_Param, 663
TCP
Ad-hoc-Modus, 659
Parameter, 663
Protokolle, 656
Verbindungs-IDs, 659
Verbindungskonfiguration, 638, 638
TCP/IP-Kommunikation, 656
TDISCON, 688
Technische Daten, 1151
Allgemeine technische Daten, 1151
BB 1297, 1309
Bemessungsspannungen, 1159
CB 1241 RS485, 1323
CM 1241 RS232, 1324
CM 1241 RS422/485, 1325
CPU 1211C AC/DC/Relais, 1163
CPU 1211C DC/DC/DC, 1163
CPU 1211C DC/DC/Relais, 1163
CPU 1212C AC/DC/Relais, 1175
CPU 1212C DC/DC/DC, 1175
CPU 1212C DC/DC/Relais, 1175
CPU 1214C AC/DC/Relais, 1187
CPU 1214C DC/DC/DC, 1187
CPU 1214C DC/DC/Relais, 1187

CPU 1215C AC/DC/Relais, 1199
CPU 1215C DC/DC/DC, 1199
CPU 1215C DC/DC/Relais, 1199
CPU 1217C DC/DC/DC, 1213
Darstellung Analogausgang (Spannung), 1255, 1298
Darstellung Analogausgang (Strom), 1255, 1299
Darstellung Analogeingang (Spannung), 1254, 1297
Darstellung Analogeingang (Strom), 1254, 1298
Eingangssimulatoren, 1327
Elektromagnetische Verträglichkeit (EMV), 1156
Industrieumgebungen, 1155
Memory Cards, 1327
Potentiometermodul, 1329
SB 1221 DI 4 x 24 V DC, 200 kHz, 1282
SB 1221 DI 4 x 5 V DC, 200 kHz, 1282
SB 1222 DO 4 x 24 V DC, 200 kHz, 1284
SB 1222 DO 4 x 5 V DC, 200 kHz, 1284
SB 1223 DI 2 x 24 VDC / DO 2 x 24 VDC, 200 kHz, 1287
SB 1223 DI 2 x 24 VDC, DO 2 x 24 V DC, 1290
SB 1223 DI 2 x 5 VDC / DO 2 x 5 VDC, 200 kHz, 1287
SB 1231 AI 1 x 12 Bit, 1293
SB 1231 AI 1 x 16 Bit RTD, 1304
SB 1231 AI 1 x 16 Bit Thermoelement, 1300
SB 1232 AO 1 x 12 Bit, 1295
Schrittantwortzeiten (CPU), 1169, 1181, 1193, 1205, 1221
Schrittantwortzeiten (SB), 1297
Schrittantwortzeiten (SM), 1253
Signalmodul SM 1231 AI 4 x RTD x 16 Bit, 1262
Signalmodul SM 1231 AI 8 x RTD x 16 Bit, 1262
SM 1221 DI 16 x 24 V DC, 1228
SM 1221 DI 8 x 24 V DC, 1228
SM 1222 DO 16 x 24 V DC, 1232
SM 1222 DO 16 x Relais, 1232
SM 1222 DO 8 x 24 V DC, 1230
SM 1222 DO 8 x Relais, 1230
SM 1222 DO 8 x Relais (Umschaltung), 1230
SM 1223 DI 16 x 24 VDC, DO 16 x 24 V DC, 1237
SM 1223 DI 16 x 24 VDC, DO 16 x Relais, 1237
SM 1223 DI 8 x 120/230 V AC / DO 8 x Relais, 1241
SM 1223 DI 8 x 24 VDC, DO 8 x 24 V DC, 1237
SM 1223 DI 8 x 24 VDC, DO 8 x Relais, 1237
SM 1231 AI 4 x 13 Bit, 1244
SM 1231 AI 4 x 16 Bit, 1244
SM 1231 AI 4 x 16 Bit TC, 1256
SM 1231 AI 8 x 13 Bit, 1244
SM 1231 AI 8 x 16 Bit TC, 1256
SM 1232 AO 2 x 14 Bit, 1248
SM 1232 AO 4 x 14 Bit, 1248
SM 1234 AI 4 x 13 Bit / AO 2 x 14 Bit, 1250
SM 1278 4xIO-Link-Master, 1268
Umgebungsbedingungen, 1158
Zulassungen, 1151
Technischer Support, 3
Technischer Support von Siemens, 3
Technologieanweisungen, 465
Technologiemodul, SM 1278 4xIO-Link-Master, 1268
Technologieobjekte
Bewegungssteuerung, 555
PID, 483
Telecontrol, 1097
TeleService über GPRS, 1097
TeleService-Adapter und -Modul, 70
Teleservice-Kommunikation
TM_MAIL (E-Mail senden), 1107
Terminalemulator für PtP-Beispielprogramm, 951
Testen im Betriebszustand RUN, 1138, 1148
Thermischer Bereich, 53, 57
Thermoelement
Filterauswahltabelle SB 1231 Thermoelement, 1302
Filterauswahltabelle SM 1231 Thermoelement, 1260
Grundlegende Funktionsweise, 1259, 1302
Kompensation der kalten Verbindungsstelle, 1259, 1302
SB 1231 AI 1 x 16 Bit, 1300
SB 1231 Filterauswahltabelle, 1303
SM 1231 Thermoelement-Auswahltabelle, 1260
TIA Portal, Portalansicht und Projektansicht, 39
Time
Datentyp Time, 123
DTL (Datentyp Date and Time long), 124
RD_LOC_T (Lokalzeit lesen), 326
SET_TIMEZONE (Zeitzone setzen), 330
T_DIFF (Zeitdifferenz), 325
TOD (Datentyp Time of Day), 124
TimeTransformationRule für Sommerzeit, 329
TM_MAIL (E-Mail senden), 1107
TMAIL_C, 714
Trace-Funktion, 1150
TRCV, 688
Verbindungs-IDs, 659
TRCV (Daten über Ethernet empfangen (TCP))
Ad-hoc-Modus, 659
Parameterkonfiguration, 749
TRCV_C
Ad-hoc-Modus, 659
TRCV_C (Daten über Ethernet empfangen (TCP)), 668
Verbindungs-IDs, 659
Verbindungsparameter, 663

TRCV_C (Daten über Ethernet empfangen (TCP))
Konfiguration, 638
TRUNC (Ganzzahl erzeugen), 282
TS-Adapter, 29
Einbau auf DIN-Schiene, 73
Einbau eines TS-Moduls, 70
SIM-Karte, 72
Wandmontage, 74
TSAP (Transport Service Access Point), 641
Allgemeine Parameter konfigurieren, 746, 807
Anweisungen zum Zuweisen zu Geräten, 656
Definition, 657
Eingeschränkte TSAPs und Portnummern, 740
TSEND, 688
Verbindungs-IDs, 659
TSEND_C (Daten über Ethernet senden (TCP)), 668
Anweisungskonfiguration, 748
Konfiguration, 638
Verbindungs-IDs, 659
Verbindungsparameter, 663
TSEND_C und TRCV_C
alte Versionen, 679
Versionen, 667
TURCV (Daten über Ethernet empfangen (UDP)), 725
Konfiguration, 638
Verbindungsparameter, 663
TUSEND (Daten über Ethernet senden (UDP)), 725
Konfiguration, 638
Parameter, 663

U

Übertragungsbaustein (T-Baustein), 746
Übertragungskarte, 140
Anlaufparameter konfigurieren, 139
Funktionsweise, 136
In CPU einfügen, 137
Leere Übertragungskarte für ein verlorenes Passwort, 149
Übersicht, 136
Verlorenes Passwort, 149
Übertragungskarte (Programmkarte), 1327
Überwachen von Variablen über Webserver, 841
UDP
Parameter, 663
Verbindungskonfiguration, 638
UDP-Protokoll, 656
UFILL_BLK (Speicher ununterbrechbar mit Bitmuster belegen), 266
Uhr
Echtzeituhr, 110
RD_LOC_T (Lokalzeit lesen), 326
RD_SYS_T (Uhrzeit lesen), 326
WR_LOC_T (Lokalzeit einstellen), 326
WR_SYS_T (Uhrzeit einstellen), 326
Uhrzeit
T_COMBINE (Zeiten verknüpfen), 325
Uhrzeit-OB, 98
Umgebungsbedingungen
Betriebsbedingungen, 1158
Transport und Lagerung, 1158
UMOVE_BLK (Bereich ununterbrechbar kopieren), 257
Umwandlung (SCL-Anweisungen), 279
UND (logische Verknüpfung), 315
Unterschiede
in Anweisungen TSEND_C und TRCV_C, 667
in den Anweisungen TCON, TDISCON, TSEND und TRCV, 687
in Modbus RTU-Anweisungen, 998
in Modbus TCP-Anweisungen, 976
in PtP-Anweisungen, 901
in USS-Anweisungen, 953

V

VAL_STRG (Zahlenwert in Zeichenkette umwandeln), 334
Variable
Forcefunktion, 1136
Scheibe, 132
Überlagerung, 134
Überwachungsstatus oder -wert, 1129
Variablen überwachen und ändern über Webserver, 841
Variablenindex für ein Array, 277
VariantGet (VARIANT Variablenwert lesen), 272
VariantPut (VARIANT Variablenwert schreiben), 273
Verbindung zwischen lokaler und Partner-CPU, 638
Verbindungen
Anzahl der Verbindungen (PROFINET/PROFIBUS), 634
Verbindungen mit mehreren Teilnehmern
Ethernet-Protokolle, 807
Verbindungsarten, 807
Verdrahtungsrichtlinien, 77
Abstand für Luftströmung und Kühlung, 53
Erdung, 76
Voraussetzungen, 75
Vergleichstabelle
CPU-Varianten, 26
HMI-Geräte, 31
Module, 29
Verknüpfung mit einer CPU, einer Memory Card oder einem Passwort, 209

Verlorenes Passwort, 149
Verlust der CPU-Kommunikation an Modulen, 97
Verpolschutz, 1160
Verschiedene PtP-Parameterfehler, 916
Versionen von
Anweisungen, 44, 667, 679, 687, 697, 955, 977, 1000, 1039, 1054, 1073
Verzeichnisse, Sprachen für benutzerdefinierte Webseiten, 883
Verzögerungsalarme, 386
Verzögerungsalarm-OB, 91
Visualisierung, HMI-Geräte, 31
Voraussetzungen PC, 38
Voraussetzungen, Installation, 38

W

Warmstart, 85
Warteschlangen, 100
Wartezeit, 903
WChar (Datentyp Word Character), 125
Webseiten
  STEP 7-Service, -Support und -Dokumentation, 4
Webserver, 817
  Aktivieren, 819
  Aktualisierungsrate, 819
  Benutzerdefinierte Webseiten, 847
  Benutzerkonfiguration, 821
  Darstellung auf einem mobilen Gerät, 828
  Einschränkungen, 892
  Konventionen bei Anführungszeichen, 862
  maximale Anzahl HTTP-Verbindungen, 892
  mobiles Gerät Zugriff, 824
  Standard-Webseiten, 823
  Zugriff über CP-Modul, 826
Wechsel von RUN nach STOP, 111
Weckalarm-OB, 92
Werte eines Online-DBs erfassen, 1130
Werte vergleichen, 242
WHILE, SCL, 310
WR_LOC_T (Lokalzeit einstellen), 326
WR_SYS_T (Uhrzeit einstellen), 326
WRIT_DBL (In Datenbaustein im Ladespeicher schreiben), 453
WRITE_BIG (Daten in Big-Endian-Format schreiben), 271
WRITE_LITTLE (Daten in Little-Endian-Format schreiben), 271
WRREC (Datensatz schreiben), 357, 364
WString (Datentyp Word String), 125
WWW (benutzerdefinierte Webseiten synchronisieren), 866

X

x-Box (logische EXKLUSIV-ODER-Verknüpfung in FUP), 218
XON / XOFF, 905
XOR (logische Verknüpfung), 315

Z

Zahlen
  Binär, 121
  Ganzzahl, 122
  Real, 122
Zähler
  Anzahl, 28, 1165, 1177, 1189, 1201, 1215
  CTD (Rückwärtszählen), 235
  CTRL_HSC (Schnellen Zähler steuern), 465
  CTU (Vorwärtszählen), 235
  CTUD (Vorwärts- und Rückwärtszählen), 235
  Funktionsweise des HSC, 471
  Größe, 28, 1165, 1177, 1189, 1201, 1215
  HSC-Konfiguration, 480
  Operation (Standardzähler), 237
Zeichen für das Ende einer Meldung, 912
Zeichen für den Beginn einer Meldung, 908
Zeichenabstand, 912
Zeichenfolge
  Meldungsbeginn, 908
  Meldungsende, 912
Zeichenposition, Meldungslänge, 913
Zeit
  RD_SYS_T (Uhrzeit lesen), 326
  T_ADD (Zeiten addieren), 324
  T_CONV (Zeiten umwandeln und extrahieren), 323
  T_SUB (Zeiten subtrahieren), 324
  WR_LOC_T (Lokalzeit einstellen), 326
  WR_SYS_T (Uhrzeit einstellen), 326
Zeit akkumulieren (TONR), 227
Zeit zurücksetzen (RT), 227
Zeiten
  Anzahl, 28, 1165, 1177, 1189, 1201, 1215
  Funktionsweise, 230
  Größe, 28, 1165, 1177, 1189, 1201, 1215
  RT (Zeit zurücksetzen), 227
  TOF (Ausschaltverzögerung), 227
  TON (Einschaltverzögerung), 227
  TONR (Zeit akkumulieren), 227
  TP (Impuls), 227
Zeitfehler-OB, 93
Zugreifen
  Benutzerdefinierte Webseiten, 868
  Datenprotokolle auf PC, 845

Zugriffsschutz, CPU, 205
Zulassung für das Seewesen, 1154
Zulassung koreanische Zertifizierung, 1153
Zulassungen
    ATEX, 1153
    CE, 1151
    C-Tick, 1153
    cULus, 1152
    FM, 1152
    Koreanische Zertifizierung, 1153
    Seewesen, 1154
Zurücksetzen auf Werkseinstellungen, 1123
Zustand
    LED-Anzeigen, 1115
    LED-Anzeigen (Kommunikationsschnittstelle), 897
Zustands-OB, 99
Zyklus
    Forcefunktion, 1136, 1136
    Übersicht, 104
Zykluszeit
    Beobachten, 1125
    Konfiguration, 105
    Übersicht, 104
Zykluszeitüberwachung (Anweisung RE_TRIGR), 297