rexroth
A Bosch Company

IndraWorks 15VRS

SafeLogic - Projektkonfiguration, Stand 02V08

Anwendungsbeschreibung
R911398634

Ausgabe 02



Titel IndraWorks
15VRS
SafeLogic - Projektkonfiguration, Stand 02V08

Art der Dokumentation Anwendungsbeschreibung

Dokumentations-Type DOK-IWORKS-SL**PRJ*V15-AP02-DE-P

Interner Ablagevermerk RS-98882f737fb5ce130a347e8601e418dc-2-de-DE-17

Änderungsverlauf Ausgabe 02, 2021-04
Siehe Tab. 1-1 "Änderungsverlauf" auf Seite 1

Schutzvermerk © Bosch Rexroth AG 2021
Alle Rechte vorbehalten, auch bzgl. jeder Verfügung, Verwertung, Reproduktion, Bearbeitung, Weitergabe sowie für den Fall von Schutzrechtsanmeldungen.

Verbindlichkeit Die angegebenen Daten dienen allein der Produktbeschreibung und sind nicht als zugesicherte Eigenschaften im Rechtssinne zu verstehen. Änderungen im Inhalt der Dokumentation und Liefermöglichkeiten der Produkte sind vorbehalten.

Redaktion Entwicklung Automationssysteme Motion Logic Control AnSt (MiNi)

Inhaltsverzeichnis

Seite

1 Zu dieser Dokumentation ... 1
1.1 Gültigkeit der Dokumentation ... 1
1.2 Gliederung der Dokumentation ... 2
1.3 Gebrauch der Sicherheitshinweise ... 3
1.3.1 Aufbau der Sicherheitshinweise ... 3
1.3.2 Erläuterung der Signalwörter und der Signalgrafik ... 3
1.3.3 Verwendete Symbole ... 4
1.3.4 Erläuterung der Signalgrafik auf dem Gerät ... 4
1.4 Erforderliche und ergänzende Dokumentationen IndraWorks ... 4
1.4.1 Engineering ... 4
1.4.2 Engineering & Visualisierung ... 6
1.4.3 Visualisierung ... 6
1.5 Bezeichnungen und Abkürzungen ... 6
1.6 Kundenfeedback ... 6

2 Einleitung ... 9

3 Normen und Standards ... 11

4 Planung des Gesamtsystems ... 19
4.1 Planung der Geräte und Allokation ... 20
4.2 Planung der Reaktionszeiten ... 21
4.2.1 Reaktionszeiten bei Feldbussteuerungen ... 21
4.2.2 Reaktionszeit bei Querkommunikation (T3) ... 23
Reaktionszeitgarantie bei Querkommunikation ... 24
Gefahr des Undersamplings ... 25
Überwachungszeit bei Querkommunikation ... 25
4.3 Planung der Adressen ... 27

5 Voraussetzungen und Verwendung ... 29
5.1 Systemvoraussetzungen ... 29
5.2 Qualifiziertes Personal ... 29
5.3 IndraWorks SafetyManager-Lizenz ... 29
5.4 Hinweise ... 31

6 Projektentwicklung in SafeLogic ... 33
6.1 Umsetzung des PLCopen Architekturmodells ... 33
6.2 Vorgehensweise bei Erstellung einer Sicherheitsapplikation ... 34
6.3 Projektstruktur ... 36
6.4 Sicherheitssteuerung ... 38
6.5 Safety Logik ... 41
6.6 Sicherheitsapplikationsobjekt SafetyApp ... 41

Seite

6.7 Logische E/As ... 47
6.7.1 Logische E/As - Überblick ... 47
6.7.2 Verwendungstypen der "Logischen E/As" ... 51
Logische E/As von "Sicheren Feldgeräten" ... 51
Logische E/As von "Nicht-sicheren Feldgeräten" ... 51
Logische E/As für den Datenaustausch Standard-/Sicherheitssteuerung ... 52
6.7.3 Editor der "Logischen E/As" ... 54
Information und E/A-Abbild ... 54
Sichere Konfiguration und Sichere Parametrierung ... 57
6.7.4 Verwendung von "Logischen E/As" im Projekt ... 59
6.8 Programmorganisationseinheiten (POUs) ... 60
6.9 Safety Task ... 63
6.10 Globale Variablenliste (GVL) ... 65
6.11 Safety Netzwerkvariablenliste (NVL) ... 67
6.12 Bibliotheksverwalter ... 71
6.13 Variablendeklaration ... 73
6.14 Bibliotheken ... 75
6.15 Geräteverwaltung ... 76
6.16 Zugriffschutz ... 77
6.16.1 Überblick Zugriffsschutz ... 77
6.16.2 Benutzerverwaltung im Projekt ... 78
6.16.3 Admin-Passwort ... 80
6.16.4 Zugriffschutz bei Anbindung an Quellcodeverwaltung ... 80

7 Programmierung ... 81
7.1 Überblick Programmierung ... 81
7.2 Programmierrichtlinien ... 82
7.2.1 Übersicht ... 82
7.2.2 Automatisch geprüfte Programmierrichtlinien ... 82
7.2.3 Manuell zu prüfende Programmierrichtlinien ... 84
7.2.4 Allgemeine Regeln für sicherheitsgerichtete Funktionsbausteine ... 84
7.2.5 Defensives Programmieren ... 84
7.2.6 Empfehlungen zur Dokumentierung des Codes ... 85
7.3 Datenfluss sicherheitsgerichteter Signale ... 86
7.4 IEC 61131-3-Konformität ... 87
7.5 Sprachelemente ... 88
7.5.1 Überblick über die Sprachelemente ... 88
7.5.2 Regeln für Bezeichner von Safety-Objekten und -Variablen ... 90
7.5.3 Globale Variablenliste (GVL) ... 91
7.5.4 Safety Netzwerkvariablenliste (NVL) ... 91
Austausch von Informationen innerhalb eines Projektes und über Projektgrenzen ... 94
7.5.5 POUs ... 96
7.5.6 Netzwerke ... 97
Überblick Netzwerke ... 97
Grundlegendes zu Variablen ... 100
Variablen für Basic-POUs ... 103

Seite

Variablen für Extended-POUs ... 104
Operatoren ... 106
FUP-Zuweisungen ... 110
Sprung/Return und Sprungmarke ... 111
FB-Aufrufe ... 112
7.6 E/A-Zugriff ... 113
7.6.1 Anbindung der Sicherheitsapplikation an 1oo1- und 1oo2-Eingabegeräte ... 114
7.7 Querkommunikation mit Safety Netzwerkvariablen ... 116
7.8 Taskkonfiguration ... 120
7.9 Programmierbeispiel ... 120

8 Online-Betrieb der Sicherheitssteuerung ... 125
8.1 Eingeloggter Zustand, Einführung ... 125
8.2 Verbindung zur Sicherheitssteuerung ... 126
8.3 Einloggen auf der Steuerung und Wechsel in den Debug-Betrieb ... 129
8.4 Mehrfacher Download ... 133
8.5 Erzeugen und Neustart der Bootapplikation ... 136
8.6 Zustände einer Sicherheitssteuerung ... 138
8.6.1 Betriebszustände und Laden einer Sicherheitsapplikation ... 138
8.6.2 Debug-Modus und organisatorische Sicherheit ... 143
8.6.3 Beenden der Applikation ... 146
8.7 Debuggen und Diagnose ... 146
8.7.1 Monitoring ... 146
8.7.2 Ablaufkontrolle ... 147
8.7.3 Debug-Betrieb der Sicherheitssteuerung ... 147
8.7.4 Debuggen-Befehle: Schreiben/Forcen ... 148
8.7.5 Debuggen-Befehle: Start, Stop, Reset kalt ... 150
8.8 Online-Informationen aus der Sicherheitssteuerung ... 151
8.9 Abstimmung mit der Standardsteuerung ... 151
8.10 Deaktivieren von sicheren Busteilnehmern ... 153
8.10.1 Anwendung ... 155
8.10.2 Einlernvorgang ... 158
8.10.3 Anforderungen ... 159
8.10.4 Abnahmeprotokolle ... 160

9 Pinnen ... 161

10 Software-Verifikation ... 167
10.1 Software-Verifikation, Allgemeines ... 167
10.2 Anforderungen an Verifikation/Validierung ... 168
10.2.1 PL-e Sicherheitsapplikationen ... 168
10.2.2 SIL3 Sicherheitsapplikationen ... 168
10.3 Statische Verifikation und Tests ... 169
10.3.1 Statische Verifikation ... 169
10.3.2 Überprüfung der Programmierrichtlinien ... 169

Seite

Automatische Prüfung der Programmierrichtlinien ... 169
Manuelle Prüfung der Programmierrichtlinien ... 170
10.3.3 Verwendung von Bausteinen ... 171
10.3.4 Testen gegen die Spezifikation ... 172
Prüfung gegen die Spezifikation ... 172
Schnittstellen ... 172
Globale Kontrollflussanalyse ... 172
Lokale Kontrollflussanalyse im Extended Level ... 174
Datenflussanalyse ... 174
Datenflussanalyse im Extended Level ... 176
10.3.5 Verifikation gegen den Systemplan ... 177
10.4 Dynamische Verifikation und Tests ... 177
10.4.1 Dynamische Verifikation und Validierung ... 177
10.4.2 Online-Tests ... 179
10.4.3 Online-Test im Extended Level ... 180
10.5 Vollständiger Funktionstest der Applikation ... 180
10.6 Verifikation in der fertigen Maschine ... 181

11 Software-Abnahme und Dokumentation ... 183
11.1 Voraussetzungen und Nachweise für die Abnahme ... 183
11.2 Funktionen für die Abnahme ... 185
11.2.1 Archivierung ... 185
11.2.2 Ausdruck Abnahmedokumentation ... 186

12 Software-Aktualisierung ... 187
12.1 Überblick Versionierung ... 187
12.2 Aktualisieren der Geräteversion ... 187
12.3 Aktualisieren der Firmware, die Ausführungsversion ... 188
12.4 Aktualisieren der IndraWorks-Version ... 189

13 Betrieb und Administration der Steuerung ... 191
13.1 IT-Sicherheit im Betrieb ... 191
13.1.1 Schutzmaßnahmen für die Sicherheitssteuerung ... 191
13.1.2 Schutz der Sicherheitssteuerung vor Schreibzugriff ... 195
13.1.3 Schutz der Sicherheitssteuerung vor Fernzugriff ... 196
13.1.4 Beobachtung Security-relevanter Ereignisse ... 196
13.2 Beobachtung von Fehlern im Betrieb ... 197
13.2.1 Beobachtung der Kommunikationsfehler-Häufigkeit ... 197
13.2.2 Anwenderverhalten bei undefiniertem oder sicherheitsrelevantem Fehler ... 197
13.3 Diagnose von Fehlern im Betrieb ... 197
13.3.1 Verbindung zur Sicherheitssteuerung für Fernzugang ... 198
13.3.2 Logbuch ... 198
13.3.3 Informationen zu Firmware und Bootapplikation ... 200
13.3.4 Diagnose der Kommunikation ... 202
13.3.5 Informationen zur Hardware ... 204

Seite

13.3.6 Diagnose auf der Standard-SPS-Steuerung ... 205
13.4 Administration mit IndraWorks ... 209
13.4.1 Bootapplikation starten ... 210
13.4.2 Bootapplikation löschen ... 210
13.5 Verfahren für die Wartung ... 211
13.5.1 Temporärer Wechsel in den unsicheren Betrieb ... 211
13.5.2 Verlassen des Betriebsmodus - Debuggen der Maschine ... 212
13.6 Wartung und Service ... 212
13.6.1 Neue Bootapplikation aufspielen ... 212
13.6.2 Firmware-Update ... 212
13.7 Hardware-Tausch ... 213
13.8 Änderungen in Netzwerken und Feldbussen ... 215

14 Verfahren bei Änderungen und Wiederverwendung der abgenommenen Software ... 217
14.1 Verfahren bei Änderungen und Wiederverwendung der Software ... 217
14.2 Wiederverwendung eines abgenommenen Safety-Projekts ... 218
14.3 Wiederverwendung von Funktionsbausteinen ... 219
14.4 Änderungen im Projekt ... 222
14.4.1 Änderungen in Projekten mit Querkommunikation ... 224

15 Außerbetriebnahme der Steuerung - SPS-Ausbau ... 227

16 Sicherheitsbibliotheken ... 229
16.1 Genauigkeit von Zeitmessungen ... 229
16.2 SafetyStandard.library ... 229
16.2.1 SF_RS ... 230
16.2.2 SF_SR ... 231
16.2.3 SF_CTD ... 231
16.2.4 SF_CTUD ... 232
16.2.5 SF_CTU ... 234
16.2.6 SF_TOF ... 235
16.2.7 SF_TON ... 236
16.2.8 SF_TP ... 237
16.2.9 SF_F_TRIG ... 238
16.2.10 SF_R_TRIG ... 239
16.3 SafetyUtilities.library ... 240
16.3.1 BoolToByte ... 241
16.3.2 BoolToWord ... 242
16.3.3 BoolToDWord ... 244
16.3.4 SF_BoolToByte ... 247
16.3.5 SF_BoolToWord ... 249
16.3.6 SF_BoolToDWord ... 250
16.3.7 ByteToBool ... 253
16.3.8 WordToBool ... 254
16.3.9 DWordToBool ... 256

Seite

16.3.10 SF_ByteToBool ... 258
16.3.11 SF_WordToBool ... 259
16.3.12 SF_DWordToBool ... 260
16.3.13 SF_IOConfig ... 262
16.3.14 SF_IOConfigCntl ... 266
16.3.15 SF_StartupWarning ... 268
16.3.16 SF_ClockGenerator ... 280
16.3.17 BinaryDecoder_4_16 ... 282
16.3.18 BinaryEncoder_16_4 ... 285
16.3.19 SF_BinaryDecoder_4_16 ... 287
16.3.20 SF_BinaryEncoder_16_4 ... 290
16.4 SafetyPLCopen.library ... 292
16.4.1 Allgemeine Regeln für sicherheitsgerichtete Funktionsbausteine ... 293
16.4.2 SF_Antivalent ... 300
16.4.3 SF_EDM ... 306
16.4.4 SF_EmergencyStop ... 314
16.4.5 SF_EnableSwitch ... 320
16.4.6 SF_Equivalent ... 327
16.4.7 SF_ESPE ... 333
16.4.8 SF_GuardLocking ... 335
16.4.9 SF_GuardMonitoring ... 341
16.4.10 SF_ModeSelector ... 347
16.4.11 SF_MutingPar ... 356
16.4.12 SF_MutingPar_2Sensor ... 370
16.4.13 SF_MutingSeq ... 379
16.4.14 SF_OutControl ... 389
16.4.15 SF_SafetyRequest ... 395
16.4.16 SF_TestableSafetySensor ... 401
16.4.17 SF_TwoHandControlTypeII ... 412
16.4.18 SF_TwoHandControlTypeIII ... 417
16.5 SafetyPLCopenPress.library ... 422
16.5.1 Standard Ein- und Ausgänge ... 423
16.5.2 SF_SingleValveMonitoring ... 425
16.5.3 SF_SingleValveCycleMonitoring ... 437
16.5.4 SF_DoubleValveMonitoring ... 453
16.5.5 SF_ValveGroupControl ... 470
16.5.6 SF_CamMonitoring ... 487
16.5.7 SF_CamshaftMonitor ... 500
16.5.8 SF_CycleControl ... 508
16.5.9 SF_PressControl ... 522
16.6 SafetyDriveMotion.library ... 536
16.6.1 SF_MotionEmergencyStop ... 539
16.6.2 SF_MotionSafeStop ... 545
16.6.3 SF_SMMSelector ... 552
16.6.4 SF_AxisGroup ... 567

Seite

17 Feldbusse und Netzwerkvariablen ... 573
17.1 Feldbusse und Netzwerkvariablen, Allgemeines ... 573
17.2 SafetyProfisafeHost.library ... 577
17.2.1 ProfisafeHost ... 579
17.2.2 Profisafe-F-Parameter ... 586
17.2.3 Profisafe-I-Parameter ... 589
17.2.4 PROFIsafe-spezifische Nachweise für die Abnahme ... 589
17.3 SafetyCSos.library ... 589
17.3.1 CSosOriginator ... 591
17.3.2 CSos-Parameter ... 598
17.4 SafetyNetVar.library ... 599
17.4.1 NetVarSender ... 603
17.4.2 NetVarReceiver ... 605
17.4.3 Safety NetVar-spezifische Nachweise für die Abnahme ... 611

18 Multi-Device Funktionalität mit SafeLogic ... 613
18.1 Inbetriebnahme der Multi-Device Funktionalität mit Safety-Steuerung ... 613
18.2 Individuelle Steuerungseinstellungen ... 615
18.2.1 Gerätenamen der Standard-Steuerung ändern ... 615
18.2.2 IP-Adresse der Standard-Steuerung ändern ... 616
18.2.3 Gerätenamen der Safety-Steuerung ändern ... 616
18.2.4 Administrationspasswort für die Safety-Steuerung ... 616
18.2.5 Fernwartung für die Safety-Steuerung ... 617
18.3 Gemeinsame (einheitliche) Steuerungseinstellungen ... 618
18.3.1 Projekt pinnen ... 618
18.3.2 Benutzerverwaltung ... 618
18.3.3 SafetyApp umbenennen ... 618
18.4 Einschränkungen durch die Verwendung der Multi-Device Funktionalität ... 618
18.4.1 Sichere Feldbusse ... 618
18.4.2 Safety Netzwerkvariablen (NVL) ... 619
18.5 Maximalprojekt erstellen ... 619
18.6 Ergänzende Hinweise ... 619
18.7 FAQ – Multi-Device ... 621

19 Versionsliste der Safety-Bausteine ... 623

20 Spezifische Sicherheitshinweise für applikative Bibliotheksbausteine ... 625

21 Liste zulässiger oder geänderter Funktionen ... 627
21.1 Zulässige Befehle ... 627
21.2 Zulässige Ansichten ... 629
21.3 Geänderte Standardfunktionen ... 632

Seite

22 Fehler ... 635
22.1 Safety-Fehlermeldungen ... 635
22.2 Safety-Diagnosen, CSS ... 654
22.2.1 SF_2730 Time Coordination Message Check ... 654
22.2.2 SF_2832 Safety Message Check ... 654
22.2.3 SF_283A Safety Message Check ... 654
22.2.4 SF_3118 Unsuccessful Forward_Open response ... 654
22.2.5 Weitere CSS-Fehlercodes ... 655

23 Referenz Safety-Benutzeroberfläche ... 663
23.1 Objekte ... 663
23.1.1 Objekt 'Safety Netzwerkvariablenliste (Sender)' ... 663
23.1.2 Objekt 'Safety Netzwerkvariablenliste (Empfänger)' ... 665
23.2 Befehle ... 668
23.2.1 Befehl 'Senderinformationen Aktualisieren' ... 668
23.2.2 Befehl 'Alle zugehörigen Empfänger anzeigen' ... 669
23.2.3 Befehl 'Gehe zu Sender' ... 669
23.2.4 Befehl 'Querverweise ausgeben - Safety' ... 670

24 Referenz Safety-Programmierung ... 675
24.1 Programmierlevel Basic - Extended ... 675
24.2 Safety-FUP-Sprachelemente ... 675
24.2.1 Netzwerke ... 676
24.2.2 Variablen ... 676
24.2.3 Operatoren ... 678

25 Glossar, SafeLogic ... 681

26 IEC 61131-3 Compliance ... 691
26.1 Zusammenfassung der Erweiterungen zur IEC 61131-3 ... 691
26.2 Compliance Liste: Unterstützte Features ... 692
26.2.1 Compliance Liste: Unterstützte Features, Allgemeines ... 692
26.2.2 Compliance Statement: Common elements ... 692
26.2.3 Compliance Statement: Common graphical elements ... 696
26.2.4 Liste zusätzlich reservierter Schlüsselwörter ... 696
26.3 Compliance Liste: Implementationsabhängige Parameter ... 699
26.4 Compliance Liste: Fehlerbedingungen ... 701
26.4.1 Compliance Liste: Fehlerbedingungen, Allgemeines ... 701
26.4.2 Compliance Liste: Nicht erkannte Fehler ... 703
26.4.3 Zusammenfassung Laufzeitfehler ... 703
26.5 Literaturangabe ... 703

27 Service und Support ... 705

Seite

28 Abnahmeprotokolle ... 707
28.1 Deaktivieren sicherer Teilnehmer im SafeLogic-Programm ... 707
28.2 Maschinenabnahme "Deaktivieren sicherer Teilnehmer" ... 708
28.3 Abnahme des Safety-Programms ... 709
28.4 Abnahme einer POU ... 710

Index ... 711

1 Zu dieser Dokumentation

Ausgaben dieser Dokumentation

Ausgabe	Stand	Bemerkung
01	2020-11	**Erstausgabe 15VRS** **Überarbeitung**: • Kap. 16.3.14 "SF_IOConfigCntl" auf Seite 266
02	2021-04	**Überarbeitung zur Version 15V12** *Neu:* • Kap. 4.2.1 "Reaktionszeiten bei Feldbussteuerungen" auf Seite 21 • Kap. 20 "Spezifische Sicherheitshinweise für applikative Bibliotheksbausteine" auf Seite 625 *Überarbeitung:* • Kap. 4.2 "Planung der Reaktionszeiten" auf Seite 21 • Kap. 5.4 "Hinweise" auf Seite 31 • Kap. 5.1 "Systemvoraussetzungen" auf Seite 29 • Kap. 5.2 "Qualifiziertes Personal" auf Seite 29 • "Erkennung von Undersampling (Sender und Empfänger)" auf Seite 119 • Kap. 8.3 "Einloggen auf der Steuerung und Wechsel in den Debug-Betrieb" auf Seite 129 • Kap. 14.1 "Verfahren bei Änderungen und Wiederverwendung der Software" auf Seite 217 • Kap. 14.2 "Wiederverwendung eines abgenommenen Safety-Projekts" auf Seite 218 • Kap. 14.4 "Änderungen im Projekt" auf Seite 222 • Kap. 8.2 "Verbindung zur Sicherheitssteuerung" auf Seite 126 • Kap. 8.6.2 "Debug-Modus und organisatorische Sicherheit" auf Seite 143

Tab. 1-1: Änderungsverlauf

1.1 Gültigkeit der Dokumentation

Übersicht über Zielgruppen und Produktphasen

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

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

Die Dokumentation unterstützt den Anwender in den Phasen:

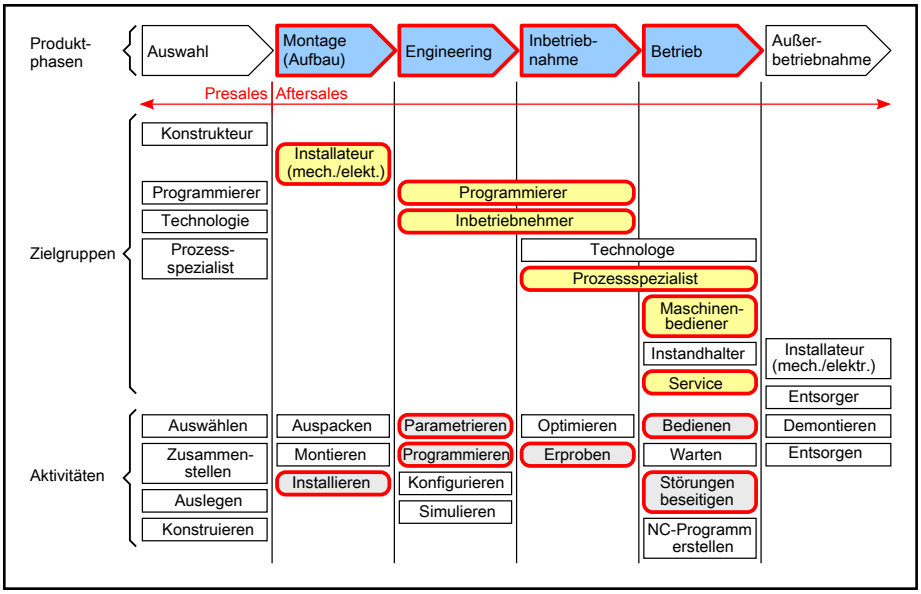


Abb. 1-1: Anwendungsphasen

1.2 Gliederung der Dokumentation

Die Dokumentation dient der Beschreibung folgender Punkte:

Kap. 3 "Normen und Standards" auf Seite 11
Kap. 4 "Planung des Gesamtsystems" auf Seite 19
Kap. 5 "Voraussetzungen und Verwendung" auf Seite 29
Kap. 6 "Projektentwicklung in SafeLogic" auf Seite 33
Kap. 7 "Programmierung" auf Seite 81
Kap. 8 "Online-Betrieb der Sicherheitssteuerung" auf Seite 125
Kap. 9 "Pinnen" auf Seite 161
Kap. 10 "Software-Verifikation" auf Seite 167
Kap. 11 "Software-Abnahme und Dokumentation" auf Seite 183
Kap. 12 "Software-Aktualisierung" auf Seite 187
Kap. 13 "Betrieb und Administration der Steuerung" auf Seite 191
Kap. 14 "Verfahren bei Änderungen und Wiederverwendung der abgenommenen Software" auf Seite 217
Kap. 15 "Außerbetriebnahme der Steuerung - SPS-Ausbau" auf Seite 227
Kap. 16 "Sicherheitsbibliotheken" auf Seite 229
Kap. 17 "Feldbusse und Netzwerkvariablen" auf Seite 573
Kap. 19 "Versionsliste der Safety-Bausteine" auf Seite 623
Kap. 21 "Liste zulässiger oder geänderter Funktionen" auf Seite 627
Kap. 22 "Fehler" auf Seite 635
Kap. 23 "Referenz Safety-Benutzeroberfläche" auf Seite 663
Kap. 24 "Referenz Safety-Programmierung" auf Seite 675
Kap. 25 "Glossar, SafeLogic" auf Seite 681
Kap. 26 "IEC 61131-3 Compliance" auf Seite 691
Kap. 27 "Service und Support" auf Seite 705.

Zusätzlich werden zur Verfügung gestellt:

Kap. 28 "Abnahmeprotokolle" auf Seite 707

1.3 Gebrauch der Sicherheitshinweise

1.3.1 Aufbau der Sicherheitshinweise

Die Sicherheitshinweise sind wie folgt aufgebaut:

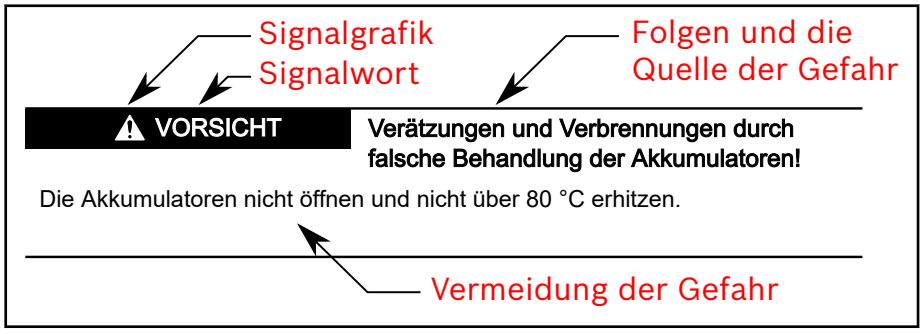


Abb. 1-2: Aufbau der Sicherheitshinweise

1.3.2 Erläuterung der Signalwörter und der Signalgrafik

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

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

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

⚠ GEFAHR

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

⚠ WARNUNG

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

⚠ VORSICHT

Bei Nichtbeachtung dieses Sicherheitshinweises können mittelschwere oder leichte Körperverletzung eintreten.

HINWEIS

Bei Nichtbeachtung dieses Sicherheitshinweises können Sachschäden eintreten.

1.3.3 Verwendete Symbole

Fingerzeige werden wie folgt dargestellt:

Dies ist ein Hinweis.

Tipps werden wie folgt dargestellt:

Dies ist ein Tipp.

1.3.4 Erläuterung der Signalgrafik auf dem Gerät



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

1.4 Erforderliche und ergänzende Dokumentationen IndraWorks

1.4.1 Engineering

Dokumentationstitel mit Typenschlüssel und Materialnummer

IndraWorks 15VRS Software-Installation DOK-IWORKS-SOFTINS*V15-CORS-DE-P, R911393449 Die Dokumentation beschreibt den Installationsprozess von IndraWorks.
IndraWorks 15VRS Engineering DOK-IWORKS-ENGINEE*V15-APRS-DE-P, R911393302 Diese Dokumentation beschreibt die Verwendung von IndraWorks, in das sich die Rexroth Engineering-Tools integrieren. Beschrieben werden das Arbeiten mit IndraWorks und die Bedienung der Oszilloskop-Funktion.
IndraWorks 15VRS SPS-Programmiersystem IndraLogic 2G DOK-IWORKS-IL2GPRO*V15-APRS-DE-P, R911396134 Diese Dokumentation beschreibt das SPS-Programmierwerkzeug IndraLogic 2G und seine Verwendung. Beschrieben werden die Basisverwendung, Erste Schritte, die Visualisierung, Menüpunkte und die Editoren.
IndraWorks 15VRS Basis-Bibliotheken IndraLogic 2G DOK-IL*2G*-BASLIB**V15-LIRS-DE-P, R911398632 Das vorliegende Handbuch beschreibt die systemübergreifenden SPS-Bibliotheken.
IndraWorks 14VRS WinStudio 7.4 DOK-IWORKS-WINSTUD*V14-APRS-DE-P, R911341584 Dieses "Benutzerhandbuch und Technische Referenz" unterstützt Sie dabei, mit Ihrer Software "Rexroth WinStudio"™ optimale Ergebnisse zu erzielen. Das Dokument bietet technische Informationen und Schritt-für-Schritt-Anleitungen zum Anlegen von webfähigen HMI/SCADA-Programmen.
IndraWorks CamBuilder 14VRS DOK-IWORKS-CAMBUIL*V14-APRS-DE-P, R911387567 Diese Dokumentation dient zur Beschreibung der Grundlagen und der Bedienung des CamBuilders, des Werkzeugs zur Kurvenscheibenbearbeitung.

IndraWorks 15VRS Feldbusse DOK-IWORKS-FB******V15-APRS-DE-P, R911393282 Die Dokumentation dient der Beschreibung der von den Systemen MLC und MTX unterstützten Feldbus- und lokalen Peripherieanschaltungen. Der Schwerpunkt dieser Dokumentation liegt in der Konfiguration, Parametrierung, Inbetriebnahme und Diagnose der verschiedenen Peripherieanschaltungen. Sie ist die Grundlage für die Online-Hilfe.
WebAssistant DOK-GENERL-WEB*ASSIST*-APRS-DE-P, R911381468 Diese Dokumentation beschreibt den WebAssistant. Der WebAssistant ist ein webbasiertes Diagnosewerkzeug, das zum Zugriff auf ein Steuerungssystem über eine Ethernet-Hochgeschwindigkeitsverbindung dient. Der WebAssistant ermöglicht OEMs, Endanwendern und Kundenbetreuern von überall auf ein System zuzugreifen und Ferndiagnosen zu stellen. Voraussetzung ist ein PC mit Internet Explorer Version 11 oder neuer, Firefox Version 45 oder neuer oder Google Chrome Version 54 oder neuer. Die folgenden Steuerungsvarianten werden unterstützt: • XM2 • XM4 • VPx
IndraWorks SafeLogic 15VRS Die ersten Schritte DOK-IWORKS-SL*STEP*V15-CORS-DE-P, R911400169 Diese Dokumentation ermöglicht einen leichten Einstieg in die Sicherheitssteuerungstechnik SafeLogic. Mittels Projektbeispielen werden dem Anwender Kenntnisse in den Bereichen Installation, Konfiguration, Inbetriebnahme, Fehlersuche und Diagnose vermittelt.
IndraWorks 15VRS Feldbus-Bibliotheken DOK-IWORKS-FB*LIB**V15-LIRS-DE-P, R911393274 Das vorliegende Handbuch beschreibt die Bibliotheken der Feldbusse: RIL_ProfibusDP_02, RIL_ProfibusDPSlave, RIL_ProfinetIO, RIL_ProfinetIODevice, RIL_EtherNetIPAdapter, RIL_MappingList, RIL_SERCOSIII, RIL_Inline, einschließlich deren Diagnosen, sowie die Fehlerreaktion der Bausteine.
IndraWorks 14VRS Fernwartung DOK-IWORKS-REMOTE**V14-APRS-DE-E, R911382627 Diese Dokumentation unterstützt den Anwender bei der Installation und Benutzung einer Fernwartungssoftware, die zu den aktuellen Betriebssystemen Win7 - Win10 kompatibel ist.
IndraWorks Remote Condition Monitoring RMB_TechRCM DOK-IWORKS-TEC*RCM*V14-LIRS-DE-E, R911343670 Diese Dokumentation beschreibt die einzelnen Funktionen des "Remote Condition Monitoring"-Systems sowie die wichtigsten Anwendungsfälle zur schnellen Zustandsbeurteilung der angeschlossenen Maschinen/Anlagen.
IndraWorks 15VRS SafeLogic - Projektkonfiguration DOK-IWORKS-SL**PRJ*V15-APRS-DE-P, R911398634 Diese Dokumentation beschreibt die Erstellung und Programmierung von SafeLogic Projekten in IndraWorks Engineering.

Tab. 1-2: IndraWorks-Dokumentationsübersicht - Engineering

1.4.2 Engineering & Visualisierung

Dokumentationstitel mit Typenschlüssel und Materialnummer
IndraWorks 14VRS Energy Efficiency Management DOK-IWORKS-4EE*****V14-APRS-DE-P, R911339228 Dieses Dokument beschreibt die Anwendung der Energy Efficiency Management Funktion innerhalb der Systeme MLC und MTX. Anhand von Beispiel-Applikationen wird dem Anwender die Projektierung, Konfiguration und Visualisierung des Systems erläutert.

Tab. 1-3: IndraWorks-Dokumentationsübersicht - Engineering & Visualisierung

1.4.3 Visualisierung

Dokumentationstitel mit Typenschlüssel und Materialnummer
IndraWorks 14VRS HMI DOK-IWORKS-HMI*****V14-APRS-DE-P, R911343567 Diese Dokumentation beschreibt die HMI-Bedienoberfläche IndraWorks Operation, deren Bedienung und Funktionen und die Konfiguration in IndraWorks Engineering.

Tab. 1-4: IndraWorks-Dokumentationsübersicht - Visualisierung

1.5 Bezeichnungen und Abkürzungen

Begriff	Erklärung
MLC	Kompaktes Motion-Logic-System mit Motion-, Robot- und Logic-Control-Funktionalität
MTX	Universelle CNC-Steuerung
IndraWorks Engineering	Projektierungs- und Inbetriebnahmetool von Bosch Rexroth
IndraDrive	Antriebsregelgerät
Wizard	Assistent, der den Anwender durch mehrere Dialoge für eine ergonomische Dateneingabe führt
Funktionsbaustein	stellt applikationsspezifische Funktionalitäten zur Verfügung
IEC	International Electrotechnical Commission (Normungsgremium für Elektrotechnik)
Sercos	sercos (serial realtime communication system) interface ist eine weltweit genormte digitale Schnittstelle zur Kommunikation zwischen Steuerungen und Antrieben
XML	Die Extensible Markup Language (engl. für "erweiterbare Auszeichnungssprache"), abgekürzt XML, ist eine Auszeichnungssprache zur Darstellung hierarchisch strukturierter Daten in Form von Textdaten
SDDML	sercos Device Description Markup Language. Gerätebeschreibungssprache für sercos III Geräte. Ist eine XML-Beschreibung für sercos III Geräte und orientiert sich am sercos III Gerätemodell

Tab. 1-5: Verwendete Bezeichnungen und Abkürzungen

Siehe auch

- Kap. 25 "Glossar, SafeLogic" auf Seite 681

1.6 Kundenfeedback

Anregungen, Wünsche oder Verbesserungen von unseren Kunden haben bei uns einen hohen Stellenwert. Senden Sie uns Ihre Anmerkungen zu den Do-

kumentationen per E-Mail an Feedback.Documentation@boschrexroth.de. Sie können direkt im elektronischen PDF-Dokument Kommentare einfügen und uns die PDF-Datei zusenden.

2 Einleitung

IndraWorks Safety Manager ist eine Software zum Programmieren von Sicherheitssteuerungen.

Dieses Handbuch ist Teil des IndraWorks Safety Manager-Produktpakets und vor Erstellen einer Sicherheitsapplikation mit IndraWorks Safety Manager unbedingt zu lesen.

Das Handbuch enthält neben der Beschreibung der Funktionen von IndraWorks Safety Manager auch Hinweise, welche bei der Erstellung Ihrer Sicherheitsapplikation unbedingt zu beachten sind.

IndraWorks Safety Manager ist geeignet zur Entwicklung von Sicherheitsapplikationen im Sinne von Anwendungssoftware bis zu SIL3 nach IEC 61508 / 62061 und Kat 4 PL-e nach ISO 13849.

Diese Dokumentation beschreibt:

- wie eine Sicherheitsapplikation verwaltet wird
- wie die Bausteine programmiert werden
- wie sichere E/A-Module zu konfigurieren und zu verwenden sind
- wie man eine Sicherheitsapplikation vor unerlaubtem Zugriff schützt
- wie man Sicherheitsapplikationen versioniert
- wie der Datenaustausch zwischen Sicherheitsapplikation und Standardapplikation konfiguriert wird
- wie eine Sicherheitsapplikation auf eine Sicherheitssteuerung geladen und getestet wird
- wie eine Sicherheitsapplikation verifiziert und zur Abnahme gebracht werden kann

Teil des Handbuchs ist die genaue Beschreibung des Sprachumfangs, der zur Erstellung einer Sicherheitsapplikation zur Verfügung steht. Dies beinhaltet alle Datentypen, alle Operatoren und alle zur Verfügung stehenden Funktionsbausteine. Der Sprachumfang richtet sich nach dem von der PLCopen für Sicherheitssoftware festgelegten Sprachumfang (Basic Level und Extended Level).

Die zur Verfügung stehenden Datentypen und Operatoren sind in Überblick über die Sprachelemente, Seite 88, beschrieben.

Die vom System angebotenen Funktionsbausteine befinden sich in den Kapiteln Sicherheitsbibliotheken, Seite 229, und Feldbusse und Netzwerkvariablen, Seite 573.

Voraussetzung für das Verstehen des Handbuchs sind gute IndraWorks-Kenntnisse und Kenntnisse der IEC 61131-3.

3 Normen und Standards

Es wird der Stand der Normen zum 01.09.2017 berücksichtigt.

Die gültigen Sicherheitsnormen

Maschinenhersteller und Betreiber technischer Anlagen sind dafür verantwortlich, dass Maschinen bzw. Anlagen grundlegenden Sicherheits- und Gesundheitsschutzanforderungen genügen. Werden die jeweils maßgeblichen Normen eingehalten, so wird gewährleistet, dass der Maschinen- bzw. Gerätehersteller oder der Errichter einer Anlage seine Sorgfaltspflicht erfüllt hat und damit der Stand der Technik erreicht ist.

Die Maschinenrichtlinie regelt ein einheitliches Schutzniveau zur Unfallverhütung für Maschinen beim Inverkehrbringen innerhalb des europäischen Wirtschaftsraums (EWR) sowie der Schweiz und der Türkei.

Die Nachweispflicht für die Einhaltung der zu Grunde liegenden Anforderungen liegt beim Gerätehersteller. Der Nachweis kann durch Einhalten der entsprechenden Europäischen Normen (EN) erbracht werden.

Diese Normen ("harmonisierte Normen") sind im Amtsblatt der EU aufgelistet.

- EN IEC 62061 und EN ISO 13849: Für den funktionalen Teil der Sicherheit von Steuerungssystemen
- Spezifische harmonisierte Normen ("Produktnormen"): Für den funktionalen Teil der Sicherheit einzelner Maschinentypen (Werkzeugmaschinen, Holzbearbeitungsmaschinen, Druckmaschinen, Verpackungsmaschinen etc.)

Für den Maschinenbauer haben diese Normen die absolut höhere Priorität. Mit Einhaltung dieser Normen kann davon ausgegangen werden, dass die grundlegenden Sicherheitsanforderungen der Maschinenrichtlinie erfüllt werden.

IndraWorks Safety Manager ist dafür geeignet, die Sicherheitsfunktionen der Maschine konform zu diesen Normen zu programmieren:

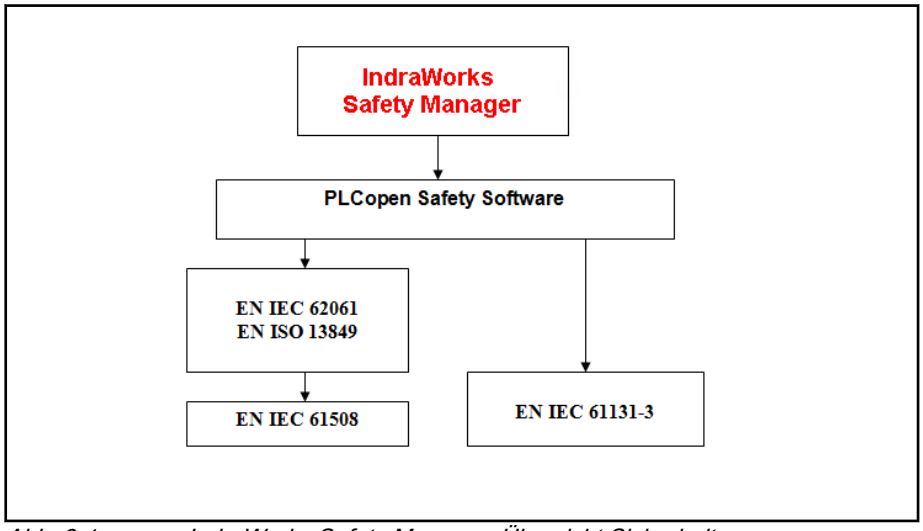


Abb. 3-1: IndraWorks Safety Manager, Übersicht Sicherheitsnormen

Norm / Standard	Beschreibung
EN IEC 61508	Sicherheits-Grundnorm, die das Gesamtsystem betrachtet Definiert SIL, LVL, organisatorische Anforderungen
EN IEC 62061 EN ISO 13849	Bereichs-Sicherheitsnorm für funktionale Sicherheit von Steuerungssystemen Harmonisierte EU Normen EN IEC 62061: Abstufung nach SIL EN ISO 13849: Abstufung nach PL
EN IEC 61131-3	Speicherprogrammierbare Steuerungen, Teil 3: Programmiersprachen, 2. Ausgabe Definiert fünf Programmiersprachen, unter anderem auch FUP
PLCopen TC5 "Safety Software"	Programmierrichtlinien Trennung Signale (sicher - unsicher) Zertifizierte Funktionsbausteine

Sicherheitsstufen

EN IEC 62061 und EN ISO 13849 stellen je nach Höhe des Schadenrisikos unterschiedliche Anforderungen an die Zuverlässigkeit der programmierten Sicherheitsfunktionen der Maschinen.

EN ISO 13849 definiert dazu verschiedene Zuverlässigkeitsstufen genannt PL-a bis PL-e.

EN IEC 62061 verweist dazu auf die Sicherheitsgrundnorm EN IEC 61508, wo die Zuverlässigkeitsstufen SIL1 bis SIL4 definiert sind.

Die Norm EN IEC 61508 beschreibt die allgemeine Zertifizierung elektrischer, elektronischer und programmierbar elektronischer Systeme (kurz: E/E/PES). Nach dieser Norm können auch integrierbare Softwareteile zertifiziert werden. Sie ist die Grundnorm, auf welche EN IEC 62061 und EN ISO 13849 verweisen.

Die beiden Stufensysteme entsprechen sich etwa wie folgt:

PL	SIL
a	keine Entsprechung
b	1
c	1
d	2
e	3
keine Entsprechung	4

Tab. 3-1: Entsprechung von PL zu SIL

IndraWorks Safety Manager ist geeignet für Maschinen der Stufen SIL1 bis SIL3 und PL-a bis PL-e.

Anforderungen an die Software-Entwicklung

Damit die programmierte Sicherheitsfunktion die notwendige Zuverlässigkeitsstufe erreicht, muss bereits die Entwicklung der Steuerungssoftware bestimmten Ansprüchen genügen.

*Insgesamt definiert EN IEC 61508 den **Zyklus des Gesamtsystems** mit 16 Phasen, beginnend mit*

- **Phase1:** "Konzept" und endet mit
- **Phase16:** "Außerbetriebnahme".

IndraWorks Safety Manager wird als Werkzeug in EN IEC 61508

- **Phase10:** "Realisierung, Sicherheits-Software Lebenszyklus" eingesetzt.

Die Norm fordert unter anderem die Definition eines Sicherheitsplans, das Erstellen von Software-Spezifikationen, die Festlegung von Programmierrichtlinien und die Planung der Testaktivitäten.

Durch die Verwendung einer Programmiersprache mit eingeschränktem Sprachumfang (Limited Variability Language, kurz: LVL) reduzieren sich die Anforderungen der Norm. Denn mit diesen Sprachen lässt sich die Programmlogik klarer ausdrücken, leichter verstehen und verifizieren. Diese leichteren Anforderungen sind in den Normen EN IEC 62061 und EN ISO 13849 definiert. (Bei Verwendung von uneingeschränkten Programmiersprachen verweisen beide Normen auf die Grundnorm 61508 (mit allgemeinen, komplexen Anforderungen).)

Zu den LVLs zählen insbesondere die grafischen Sprachen der EN IEC 61131-3, wie z. B. FUP oder KOP. Nicht dazu gehören die textuellen Sprachen, wie C, ST, Assembler oder AWL.

Das PLCopen-Konzept für Sicherheitssoftware

Für Maschinenbauer ist es oft kostenintensiv und kompliziert alle Normen einzuhalten, da sie mit mehreren sicherheitsbezogenen Normen konfrontiert werden. Bei der Konstruktion von Maschinen und Anlagen wird der sicherheitsgerichtete und der funktionale Teil der Applikation häufig voneinander getrennt entwickelt und erst am Schluss zusammengeführt. Auf diese Weise werden die Sicherheitsaspekte zwar berücksichtigt, sie werden jedoch nicht von Anfang an in die gesamte Philosophie des Systems integriert. Moderne, softwarebasierte Steuerungssysteme und Sicherheitsprotokollschichten für industrielle Feldbusse erlauben inzwischen die verzahnte Entwicklung von sicherheitsgerichtetem und funktionalem Teil der Applikation von Anfang an.

Gemäß aller maßgeblichen Sicherheitsnormen sind die allgemeinen Grundanforderungen an eine Sicherheitsapplikation:

- Unterscheidung zwischen Sicherheits- und Nicht-Sicherheits- Funktionalitäten
- Verwendung von geeigneten Programmiersprachen bzw. Teilmengen von Programmiersprachen
- Verwendung validierter Softwarebausteine
- Verwendung von geeigneten Programmierrichtlinien
- Verwendung von anerkannten Fehler-reduzierenden Maßnahmen für den Lebenszyklus der sicherheitsgerichteten Software

Um den Aufwand für das Erfüllen dieser hohen Anforderungen zu reduzieren, hat das PLCopen Komitee TC eine Lösung für alle Punkte erarbeitet. Diese standardisierte Lösung wird von IndraLogic-Safety in allen Aspekten unterstützt.

PLCopen-Konzept 1: Integration der Entwicklungsumgebungen

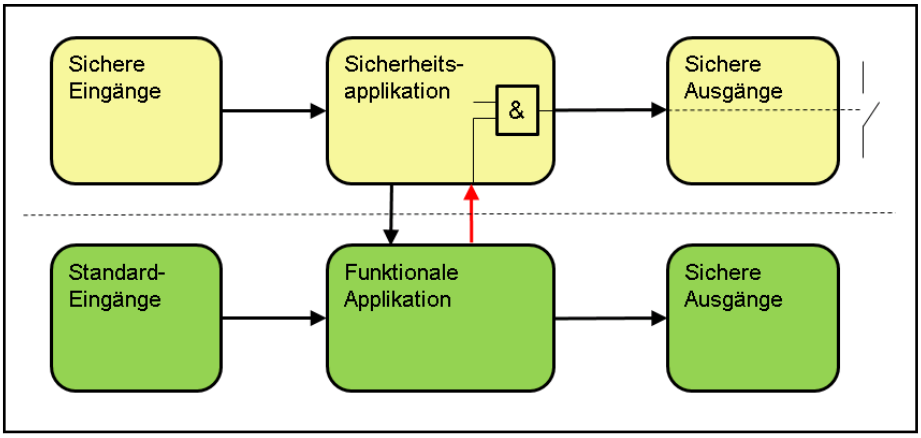


Abb. 3-2: PLCopen Software-Architekturmodell

Der sicherheitsgerichtete und der funktionale Teil der Applikation wird im Software-Architekturmodell als funktionale Applikation bzw. Sicherheitsapplikation dargestellt. Bei SafeLogic läuft die funktionale Applikation auf der Hauptsteuerung und die Sicherheitsapplikation auf der Sicherheitssteuerung. Ziel der PLCopen ist, statt verschiedener Entwicklungsumgebungen für die jeweiligen Teilapplikationen, die Entwicklungsumgebung der funktionalen Applikation um einen integralen Teil für die Sicherheitsapplikation zu erweitern.

- Die Sicherheitsapplikation bearbeitet die sicheren Eingänge und steuert die sicheren Ausgänge, während die funktionale Applikation die nicht-sicheren Eingänge verarbeitet und die nicht-sicheren Ausgänge steuert. Die funktionale Applikation besitzt Lesezugriff auf die sicheren Eingänge und auf die globalen Variablen (angezeigt durch den linken Pfeil).
- Die nicht-sicheren Signale können in der Sicherheitsapplikation nur zur Kontrolle des Programmflusses verwendet werden und können nicht direkt mit sicheren Ausgängen verbunden werden (angezeigt durch den rechten Pfeil und den UND-Operator).

SafeLogic ist so eine Erweiterung der Standard-Entwicklungsumgebung IndraWorks (für funktionale Applikationen) um einen Teil für die Entwicklung von Sicherheitsapplikationen. Die gesamte Applikation der Maschine bestehend aus funktionaler Applikation und Sicherheitsapplikation mit entsprechenden Ein- und Ausgängen und Datenaustausch zwischen ihnen wird in einer gemeinsamen Entwicklungsumgebung in einem gemeinsamen Projekt entwickelt.

Erweiterung des PLCopen-Konzepts mit Querkommunikation

Die Querkommunikation zwischen Steuerungen stellt eine Erweiterung der klassischen 3-stufigen Architektur Eingänge--Steuerung-- Ausgänge wie im PLCopen Software Architekturmodell dar. Durch Verwendung von Safety-Netzwerkvariablen in der Sicherheitsapplikation ist auch eine sichere Querkommunikation zwischen den Sicherheitssteuerungen möglich.

Im folgenden Architekturmodell wird die Verwendung von Safety-Netzwerkvariablen berücksichtigt. Es gibt eine zweite Sicherheitsapplikation und somit auch eine weitere funktionale Applikation. Die Signale der sicheren Eingänge werden von den PLCopen-FBs ausgewertet, in einer der beiden Sicherheitsapplikationen verarbeitet und anschließend in den PLCopen-FBs für die Ansteuerung der sicheren Ausgänge ausgewertet.

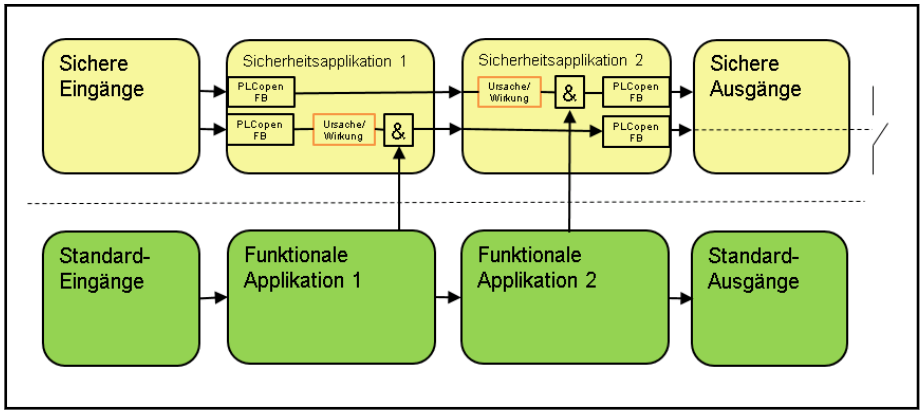


Abb. 3-3: Software-Architekturmodell mit Querkommunikation

Bei Verwendung von Netzwerkvariablen können die Daten auf zwei verschiedene Arten zwischen Sicherheitsapplikationen kommuniziert werden:

- Ausgänge der Sensorüberwachung, die durch PLCopen-Bausteine realisiert werden
- Anforderung einer Sicherheitsfunktion, die im Modell als Ursache/ Wirkung darstellt ist.

PLCopen-Konzept 2: Trennung der sicheren Aspekte von nicht-sicheren Aspekten

Um deutlich zwischen sicherheitsrelevanten Signalen und Standardsignalen zu unterscheiden, wurden neue Datentypen mit der Kennzeichnung "SAFE" definiert. Dadurch erkennt der Entwickler, dass die Signale sicherheitsrelevant sind und mit besonderer Sorgfalt behandelt werden müssen. Des Weiteren kann durch diese Kennzeichnung die Verbindung von Daten automatisch geprüft werden, um alle unerlaubten Verbindungen zwischen Standardsignalen und sicherheitsrelevanten Signalen zu entdecken. Obwohl die SAFE-Datentypen nicht garantieren können, dass der Signal- Status sicher ist (zum Beispiel bei nicht korrekt verdrahteter Peripherie), ist es dennoch ein organisatorisches Tool zur Minimierung von Fehlern im Anwendungsprogramm. Dies vereinfacht und verkürzt die Verifikation des Signalflusses. SAFE-Datentypen können innerhalb einer sicherheitsrelevanten Umgebung verwendet werden. Sie sollten zur Unterscheidung von sicheren und nicht-sicheren Signalen verwendet werden mit dem Ziel, die Validierung und Zertifizierung zu vereinfachen.

SafeLogic behandelt nur Signale von und zu sicheren Feldgeräten als Werte eines SAFE-Datentyps und überprüft automatisch die korrekte Verknüpfung von SAFE-Datentypen in der Applikationslogik.

PLCopen-Konzept 3: geeignete Sprache mit Programmierrichtlinien

Die PLCopen hat einige Regeln zur einheitlichen Definition sicherheitsbezogener Funktionsbausteine aufgestellt und sie in ihren Bausteinen angewendet. Diese "General Rufes for safety-related function blocks" gelten für PLCopen-konforme Funktionsbausteine.

Als Sprachen werden von der PLCopen der Funktionsplan (FUP) und die Anweisungsliste (AWL) bevorzugt, da mit diesen Sprachen Programmiereinheiten relativ leicht erstellt, gut gelesen und geprüft werden können. Die Ansicht des Programmcodes ist klar strukturiert und ähnelt der traditionellen diskreten Verdrahtung von Sicherheitsbaugruppen. In SafeLogic steht zur Implementierung des Programmcodes der Sicherheitsapplikation die Programmiersprache FUP zur Verfügung.

Aufgrund der Anforderungen nach Programmierrichtlinien hat die PLCopen Arbeitsgruppe "Safety Software" Programmierregeln in zwei Abstufungen definiert: Basic Level und Extended Level.

Basic Level: Der grundlegende Ansatz ist, dass das Sicherheitsprogramm nur aus zertifizierten Funktionsbausteinen besteht, die in grafischer Form

leicht miteinander verbunden werden können. Wenn zusätzlich die Struktur der Verbindung zwischen den FBs eingeschränkt wird, kann eine Ansicht im Stil moderner IDEs (d. h. Integrierte Entwicklungsumgebung) entwickelt werden, die ähnlich der traditionellen diskreten Verdrahtung von Sicherheitsbaugruppen funktioniert. Die Programme haben eine klare Struktur und können leicht gelesen werden. Darüber hinaus wird die Zeit für die Freigabe des Programms signifikant verkürzt, da es aus bereits zertifizierten Bausteinen besteht.

Extended Level: Bei Projekten, für welche der gegenwärtige Stand der zertifizierten Funktonsbausteine nicht ausreichend ist, kann der Anwender die erforderlichen Bausteine (oder sogar das Programm) im Extended Level erstellen. Hierfür wird ein erweiterter Sprachumfang zur Verfügung gestellt. Allerdings kann die Validierung der Funktionalitäten dieser Bausteine und Programme wesentlich aufwendiger, und demzufolge zeitaufwendiger sein, da die Programme dem vollständigen Verifikationsprozess unterliegen. Wenn die Bausteine verifiziert/validiert sind, können sie, zusammen mit den oben beschriebenen Vorteilen im Basic Level verwendet werden.

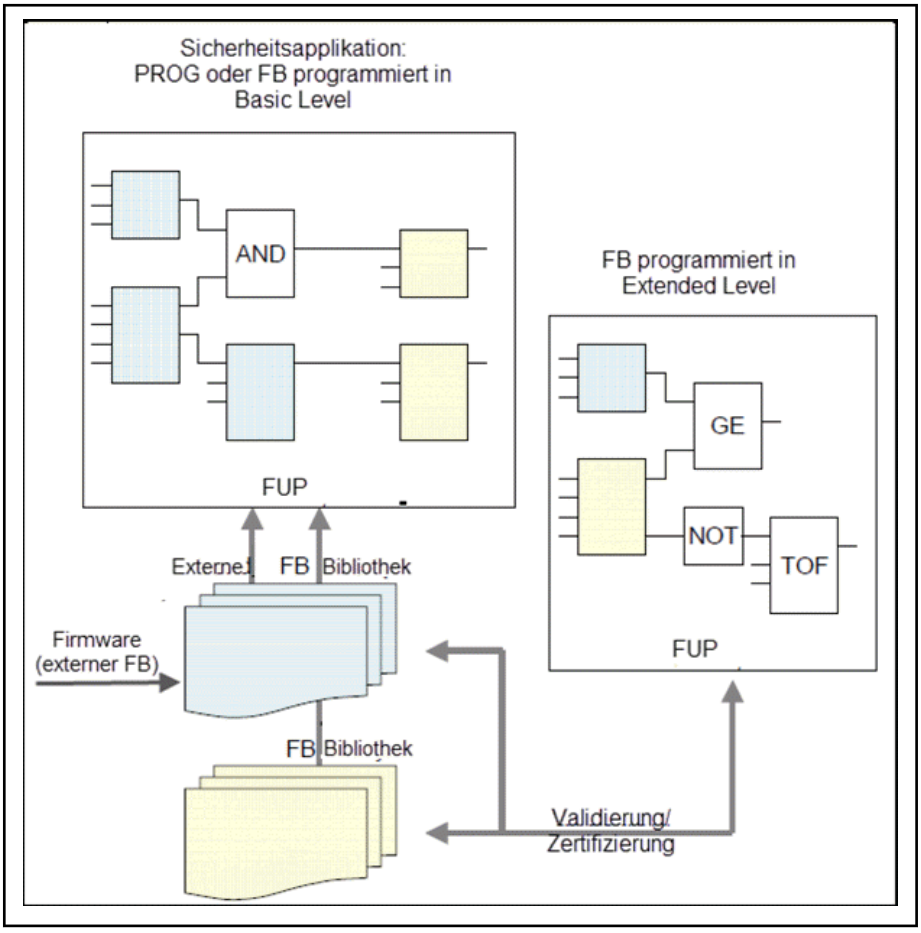


Abb. 3-4: Empfohlener Anwendungsbereich der beiden Programmierlevel

SafeLogic unterstützt diese PLCopen-Regeln durch automatische Prüfung dieser Programmierrichtlinien und der Verknüpfungsregeln. Es prüft Bausteine, die als Basic markiert sind, gegen die Basic-Regeln und Bausteine, die als Extended markiert sind, gegen die Extended-Regeln. Bei Verstoß lässt sich eine Applikation nicht auf die Steuerung laden. So ist das Einhalten der Regeln automatisch nachgewiesen.

Nur noch einige Regeln müssen manuell geprüft werden (siehe Manuelle Prüfung der Programmierrichtlinien, Seite 170).

PLCopen-Konzept 4: standardisierte Funktionsbausteine

Die PLCopen hat einen Satz von Funktionsbausteinen definiert, welche immer wiederkehrende Funktionalitäten von Sicherheitsapplikationen definieren. Dieser Satz von Funktionsbausteinen unterstützt die typische Safety Ausstattung in der Fertigungsautomation. Die unten aufgeführte Tabelle (Tab. "Zuordnung der Sicherheitsvorrichtungen zu den PLCopen FBs" auf Seite 17) zeigt wie die Funktionalitäten der Safety Ausstattung den einzelnen PLCopen Bausteinen zugeordnet werden.

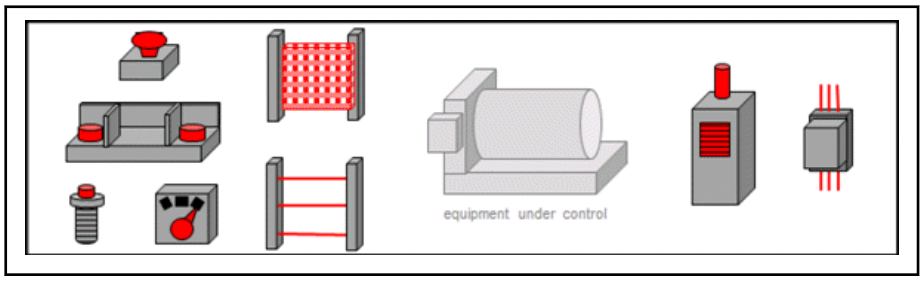


Abb. 3-5: Gerätetypen der Safety Ausstattung in der Fertigungsautomation

Zuordnung der Sicherheitsvorrichtungen zu den PLCopen FBs

Bedienelemente	Notaus-Schalter (EN ISO 13850)	SF_EmergencyStop
	Zweihandbedienung (EN 574, ISO 13851)	SF_TwoHandControlTypeII / III
	Zustimmtaster, Betriebsartenwahlschalter (IEC 60204)	SF_EnableSwitch, SF_ModeSelector
Sensoren	Schutztür (EN ISO 14120)	SF_SafetyGuardMonitoring, SF_SafetyGuardLocking (Safety Guard Interlocking with Locking)
	Lichtschranke (IEC 61496 / 62046)	SF_ESPE, SF_Testable Safety-Sensor
	Muting-Sensoren (IEC 61496 / 62046)	SF_MutingSeq, SF_MutingPar, SF_MutingPar_2Sensor
Ausgänge	Antrieb/Ventil mit sicherem Zustand (IEC 60204)	SF_SafetyRequest
	Relais mit Rückführkontakt (IEC 60204)	SF_EDM
	unsicher geschalteter Ausgang mit Überwachung (IEC 60204)	SF_OutControl

IEC 61131-3

Die IEC 61131-3 beschreibt die Programmiersprachen, die zur Programmierung von Steuerungen eingesetzt werden. Sie definiert die Sprache FUP, von der die PLCopen Teilmengen definiert hat und für die die PLCopen Programmierregeln definiert hat.

4 Planung des Gesamtsystems

Für den sicherheitsbezogenen Teil des Steuerungssystems einer Maschine können, abhängig von den Kommunikationsmöglichkeiten der eingesetzten Sicherheitssteuerungen und ihrer Hauptsteuerungen, verschiedene Topologien gewählt werden:

- **T1**: Die Sicherheitsfunktionen der Maschine werden innerhalb eines oder mehrerer Feldbusse jeweils lokal mit einer einzigen Sicherheitssteuerung am Feldbus-Master realisiert
- **T2**: Die Sicherheitsfunktionen der Maschine werden innerhalb eines oder mehrere Feldbusse jeweils lokal mit mehreren dezentralen, nicht untereinander kommunizierenden Sicherheitssteuerungen realisiert.
- **T3**: Die Sicherheitsfunktionen der Maschine werden in mehreren Feldbussen mit mehreren per Safety Netzwerkvariablen untereinander kommunizierenden Sicherheitssteuerungen realisiert.

Nachdem die Sicherheitsrisiken in einer Maschine identifiziert (Risikoanalyse) und Sicherheitsfunktionen zur Risikominderung definiert wurden, kann der sicherheitsgerichtete Teil des Steuerungssystems der Maschine geplant werden. Dabei sind mehrere Punkte gleichzeitig zu berücksichtigen:

1. **Allokation:** Für die Sicherheitsfunktionen der Maschine müssen entsprechende Feldgeräte, Feldbusse und Sicherheitssteuerungen eingeplant werden (Planung der Geräte und Allokation, Seite 20).

   Daraus ergeben sich die Ein-/Ausgangssignale und funktionalen Anforderungen für die zu entwickelnden Sicherheitsapplikationen der einzelnen Sicherheitssteuerungen.

2. **Reaktionszeiten:** Damit die für eine Sicherheitsfunktion nötigen Reaktionszeiten in der Maschine erreicht werden, müssen entsprechend performante Geräte und entsprechende Zeiteinstellungen in den Geräten eingeplant werden (Kap. 4.2 "Planung der Reaktionszeiten" auf Seite 21).

   Daraus ergeben sich Anforderungen an Konfigurationseinstellungen in den Sicherheitsapplikationen.

3. **Eindeutigkeit:** Für die Kommunikation zwischen Sicherheitssteuerungen und sicheren Feldgeräten und zwischen Sicherheitssteuerungen untereinander müssen je nach Feldbus verschiedene eindeutige Adressen und IDs vergeben werden. Bei Topologien T2 und T3 muss dieses über die einzelne Sicherheitssteuerungen hinweg geplant werden (Planung der Adressen, Seite 27).

   Daraus ergeben sich Anforderungen an Konfigurationseinstellungen in den Sicherheitsapplikationen.

4. **Systemanforderungen:** Für Ethernet-basierte Feldbusse (zum Beispiel Sercos, PROFINET) und für Netzwerkvariablen, können zusätzliche Systemanforderungen an Abgrenzung des Netzwerks, Teilnehmer im Netzwerk und die Kommunikationsfehlerrate entstehen.

   Für die spezifischen Anforderungen siehe die Abschnitte "Systemanforderungen" in Feldbusse und Netzwerkvariablen, Allgemeines, Seite 573

5. **Zugriffschutz:** Der Schutz der Sicherheitsfunktionen erfordert Schutzmaßnahmen auf verschiedenen Ebenen. Insbesondere muss bei Querkommunikation (T3) das dazu verwendete Netzwerk auf Maschinenebene geschützt werden (Kap. 13.1 "IT-Sicherheit im Betrieb" auf Seite 191).

4.1 Planung der Geräte und Allokation

Für die Realisierung der Sicherheitsfunktionen der Maschine müssen entsprechende Sensoren und Aktoren, entsprechende sichere Feldgeräte in entsprechenden Feldbussen und entsprechende Sicherheitssteuerungen eingeplant werden. Die Sicherheitsfunktionen müssen diesen zugeordnet werden (Allokation).

☞ Die maximale Anzahl sicherer Feldgeräte für eine Sicherheitsfunktion kann durch den Feldbus begrenzt sein, z. B. auf 100 bei PROFIsafe.

(Siehe SafetyProfisafeHost.library, Seite 577).

Typischerweise wird eine Sicherheitsfunktion durch eine Sicherheitssteuerung und die ihr exklusiv zugeordneten sicheren Feldgeräte im gleichen Feldbus realisiert (immer so bei T1, T2). Die für die Sicherheitsfunktion einer Sicherheitssteuerung benötigten Sensoren und Aktoren sollten der Einfachheit halber genau an dieser Sicherheitssteuerung (und damit der Hauptsteuerung) angeschlossen werden.

☞ Die maximale Anzahl sicherer Feldgeräte im gleichen Feldbus oder von Sicherheitssteuerungen im gleichen Feldbus (nur T2) kann feldbusspezifisch begrenzt sein (Feldbusse und Netzwerkvariablen, Allgemeines, Seite 573).

Jede geplante Sicherheitssteuerung benötigt eine Sicherheitsapplikation. Für diese muss ein Software-Entwicklungsprozess gestartet werden. Für die Sicherheitsapplikationen muss eine Spezifikation erstellt werden (nicht notwendigerweise als separates Dokument). Zur Verifikation muss für jede Sicherheitsapplikation ein funktionaler bzw. Blackbox-Test gegen die Spezifikation eingeplant und in der Verifikationsphase durchgeführt werden.

Siehe Vollständiger Funktionstest der Applikation, Seite 180.

Aufteilung mit Querkommunikation

Eine Sicherheitsfunktion kann auf Feldgeräte und auf Sicherheitssteuerungen in mehreren Feldbussen (auch Feldbusse verschiedenen Typs) aufgeteilt werden, wenn man eine Kommunikation zwischen den Sicherheitssteuerungen dieser Feldbusse herstellt (T3). Es muss dann auch das entsprechende Netzwerk zwischen den Steuerungen in die Planung einbezogen werden.

Bei so einer Aufteilung erfolgt die Überwachung der Sensoren, Aktoren und Feldgeräte immer in der Sicherheitssteuerung im gleichen Feldbus wie die Feldgeräte. Auf welcher Sicherheitssteuerung die Entscheidung zum Auslösen der Sicherheitsfunktion (Cause/Effect) zu realisieren ist, muss festgelegt werden (PLCopen-Konzept 1: Integration der Entwicklungsumgebungen, Seite 14)

auf der Sicherheitssteuerung mit den Sensoren oder auf der Sicherheitssteuerung mit den Aktoren; oder sind beide Sicherheitssteuerungen an der Entscheidung beteiligt?

Safety-Netzwerkvariablen erfüllen die Funktion der sicheren Querkommunikation in SafeLogic. Sie erlauben die Konfiguration und den Betrieb eines sicheren Datenaustauschs zwischen SafeLogic-Sicherheitssteuerungen in einem Projekt.

Sie sind gemacht für den Fall, dass mehrere Sicherheitssteuerungen für ihre Sicherheitsfunktionen den gleichen Sensor (z.B. einen Notaus-Schalter) benötigen: Sie wählen eine Sicherheitssteuerung aus, an die Sie den Sensor anschließen. Diese Sicherheitssteuerung muss den Sensor überwachen und

das überwachte Sensorsignal über den NetVar-Mechanismus an die anderen Sicherheitssteuerungen weiterverteilen.

Das Prinzip: Ein IndraLogic-Projekt enthält beide Sicherheitssteuerungen mit ihren Hauptsteuerungen, und ggf. noch weitere. Die Sicherheitssteuerung mit den Sensoren veröffentlicht die Daten und die anderen Sicherheitssteuerungen des Projekts können die Daten lesen. Der Kommunikationsweg erfolgt dabei über die Hauptsteuerungen der entsprechenden Sicherheitssteuerungen.

4.2 Planung der Reaktionszeiten

Die Reaktionszeit einer Sicherheitsfunktion (Gesamtreaktionszeit) ist die Zeit von dem Gefährdungsereignis, das ein Sensor dieser Sicherheitsfunktion detektiert, bis zur Reaktion eines Aktors dieser Sicherheitsfunktion, die die Sicherheit herstellt. Allgemein gilt, eine Sicherheitsfunktion kann mehrere Aktoren umfassen. Für jede Verknüpfung Sensor → Aktor muss die Reaktionszeit einzeln berechnet werden.

Bei der Reaktionszeit ist zu unterscheiden:

- Die tatsächliche Reaktionszeit R_t in der Maschine für ein bestimmtes Ereignis zu einem Zeitpunkt t
- Die im störungsfreien Betrieb technisch bedingte Obergrenze (Worst-Case) R_{wc} tatsächlicher Reaktionszeiten. D.h. $R_{wc} \geq R_t$.
- Die durch das Sicherheitssystem auch bei einem Fehler garantierte Obergrenze der Reaktionszeit R_g, zu der spätestens eine Reaktion erfolgt, notfalls auch indem die sicheren Ausgänge den sicheren Zustand herstellen. Für den reibungslosen Betrieb (ohne notfallmäßigen sicheren Zustand durch die sicheren Ausgänge) ist $R_g \geq R_{wc}$ erforderlich.
- Die durch Sicherheitsanforderungen geforderte, maximal zulässige Reaktionszeit R_{req} für die funktionale Sicherheit der Maschine. Für die Sicherheit ist $R_{req} \geq R_g$ erforderlich.

Die tatsächlichen Reaktionszeiten, ihr Worst-Case R_{wc}, und damit die kürzesten sinnvollen Reaktionszeitgarantien Rg hängen von der Topologie und von verschiedenen geräte-, feldbus- und applikationsabhängigen Faktoren ab. Einen Teil dieser Faktoren kann der Anwender über IndraWorks durch Konfiguration beeinflussen (Zykluszeiten, Überwachungszeiten).

Damit die für eine Sicherheitsfunktion geforderten Reaktionszeiten in der Maschine erreicht werden können und sicher erreicht werden, müssen entsprechende koordinierte Zeiteinstellungen in den verschiedenen Sicherheitsapplikationen inklusive ihrer sicheren Feldgeräte und in den Feldbus-Mastern eingeplant werden.

4.2.1 Reaktionszeiten bei Feldbussteuerungen

Reaktionszeit bei Topologie T1/T2

Für jede Sicherheitsfunktion, die innerhalb eines Feldbusses lokal mit einer Sicherheitssteuerung realisiert ist (T1/T2), lässt sich die tatsächliche Reaktionszeit durch folgende Einstellungen beeinflussen: Zykluszeit der Sicherheitssteuerung und Zykluszeit der Mapping-Applikation der Standardsteuerung.

Jeder Sensor (zum Beispiel Lichtschranke oder Not-Halt-Schalter) wandelt das physikalische Signal in ein elektrisches Signal um. Das Signal kann mit einem Eingangsmodul (zum Beispiel einem sicheren Eingang) verbunden werden, der das Signal in logische Information umwandelt.

Jedes dieser Elemente hat eine minimale (= Bearbeitung) und eine maximale Verzögerungszeit (= Bearbeitung und Warten). Die tatsächliche Verzögerungszeit bzw. das Verzögerungszeitintervall liegt zwischen der minimalen und maximalen Verzögerungszeit.

Aufgrund der Annahme, dass alle Komponenten asynchron arbeiten, ist der schlimmste Fall für jede Komponente die doppelte Verzögerung der Komponente.

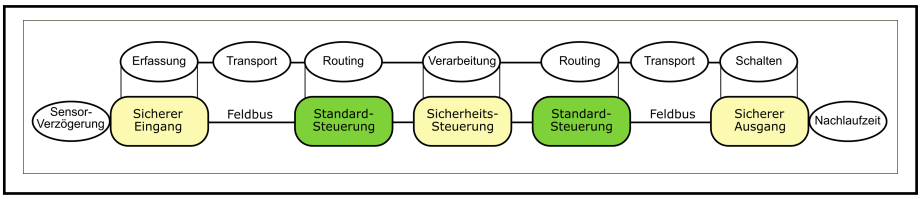
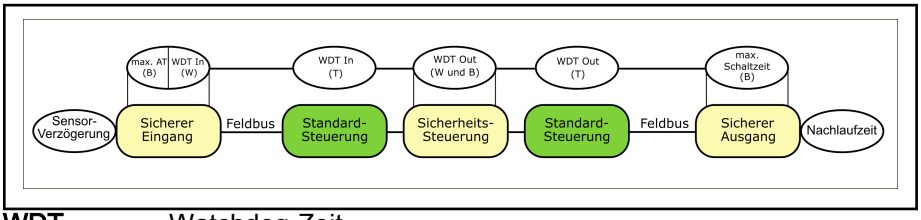


Abb. 4-1: Zeitfaktoren für Berechnung der tatsächlichen Reaktionszeit Rt

Informationen zu den einzelnen Zeitfaktoren:

- **Sensorverzögerung**: Der Sensor wandelt einen physikalischen Zustand in ein elektrisches Signal um (Dauer siehe Datenblatt des Sensors)
- **Erfassung**: Das Eingangsmodul wandelt das Sensorsignal in Feldbus-PDOs (logische Information) um. Dauer siehe Datenblatt des Eingangsmoduls
- **Feldbustransport**: Abhängig von der eingestellten Feldbusgeschwindigkeit
- **Routing** in der Standardsteuerung: Abhängig von der Zykluszeit der Mapping-Applikation
- **Transport** in die Sicherheitssteuerung (systemabhängig)
- **Verarbeitung**: Die Sicherheitssteuerung verarbeitet die logische Information des Eingangsmoduls in logische Ausgangsinformation (Feldbus-PDOs). Dauer abhängig von der Zykluszeit der Sicherheitsapplikation
- **Feldbustransport**: Dauer abhängig von der eingestellten Feldbusgeschwindigkeit
- Schalten durch den sichereren Ausgang: Umwandlung des Feldbus-PDOs in elektrisches Signal für den Aktor, siehe Datenblatt des Ausgangsmoduls
- **Nachlaufzeit**: Umsetzen des elektrischen Signals in eine physikalische Aktion durch den Aktor, siehe Datenblatt des Aktors



WDT	Watchdog-Zeit
AT	Erfassungszeit
W	Wartzeit
B	Bearbeitungszeit
T	Transportzeit

Abb. 4-2: Zeitfaktoren für die Berechnung der garantierten Obergrenze der Reaktionszeit Rg

Die Faktoren innerhalb jeder Sicherheitskomponente und die Kombination der Faktoren zwischen den Sicherheitskomponenten haben aus Sicherheitsgründen ihre überlagerte Zeitüberwachung gegen ein Zeitlimit, genannt

Watchdog-Zeit. Die Zeitüberwachung ergreift die notwendigen Maßnahmen, um den sicheren Zustand zu aktivieren, immer wenn ein Fehler oder eine Störung innerhalb dieser Komponente oder der davorliegenden Komponente auftritt. Diese Watchdog-Zeiten werden in der Sicherheitsapplikation konfiguriert und mit ihnen lässt sich die garantierte Obergrenze Rg der Gesamtreaktionszeit berechnen.

- **WDT In**: Konfigurierte Watchdog-Zeit der Feldbuskommunikation zwischen sicherem Eingangsmodul und Sicherheitssteuerung
  - Erstmalig als maximale Wartezeit, die durch das sichere Feldbusprotokoll im Eingangsmodul anfallen kann, bevor es ein geändertes Signal vom Sensor an die Sicherheitssteuerung abschicken kann. So lange muss das Signal des Sensors mindestens anstehen, um zu garantieren, dass es von der Sicherheitssteuerung gesampelt werden kann.
  - Zum zweiten Mal als maximale Transportzeit, in der das abgeschickte Signal in der Sicherheitssteuerung als Eingangssignal ankommt. Diese Zeit überwacht die Performance des Feldbusses und der Standardsteuerung als Vermittler.

    Dauer Feldbustransport + Dauer Routing ≤ WDT In
- **WDT Out**: Konfigurierte Watchdog-Zeit der Feldbuskommunikation zwischen Sicherheitssteuerung und sicherem Ausgangsmodul
  - Erstmalig als maximale Wartezeit, inklusive eines Applikationszyklus zur Abbildung der neuen Variablenwerte auf Ausgangssignale, bevor die Sicherheitssteuerung ein geändertes Ausgangssignal an das Ausgangsgerät abschicken kann. So lange muss die Applikation der Sicherheitssteuerung das Signal mindestens anstehen lassen, um zu garantieren, dass es vom Ausgangsmodul gesampelt werden kann.
  - Zum zweiten Mal als maximale Transportzeit, in der das abgeschickte Signal im Ausgangsmodul ankommt. Diese Zeit überwacht die Performance des Feldbusses und der Standardsteuerung als Vermittler.

    Dauer Routing + Dauer Feldbustransport ≤ WDT Out



Die zugesicherte Gesamtreaktionszeit-Obergrenze R_g beträgt:

Sensorverzögerung

+ max. Erfassungszeit

+ 2 x WDT In

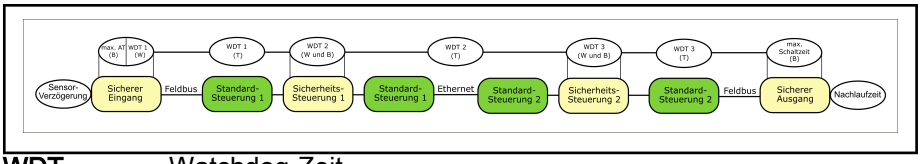
+ 2 x WDT Out

+ max. Schaltzeit

+ Nachlaufzeit

4.2.2 Reaktionszeit bei Querkommunikation (T3)

Bei einer Sicherheitsfunktion, an der zwei per Safety Netzwerkvariablen kommunizierende Sicherheitssteuerungen beteiligt sind, lässt sich die Gesamtreaktionszeit durch folgende Einstellungen beeinflussen: Zykluszeit der Sender-Sicherheitsapplikation, Zykluszeit der Empfänger-Sicherheitsapplikation, Zykluszeit der Mapping- Applikation der Standardsteuerung.



WDT Watchdog-Zeit
AT Erfassungszeit
W Wartzeit
B Bearbeitungszeit
T Transportzeit

Abb. 4-3: Reaktionszeit bei Querkommunikation

Reaktionszeitgarantie bei Querkommunikation

Die zugesicherte Gesamtreaktionszeit-Obergrenze R_g beträgt:

Sensorverzögerung
+ max. Erfassungszeit
+ 2 x WDT 1
+ 2 x WDT 2
+ 2 x WDT 3
+ max. Schaltzeit
+ Nachlaufzeit

In einem Projekt mit Safety-Netzwerkvariablen gibt es mehrere Watchdog-Zeiten WDT (siehe Abb. 4-3), die in Summe die Gesamtreaktionszeit ergeben. Dabei enthält jede WDT immer mindestens eine Zykluszeit der Applikation, bedingt durch das Zeitmessverfahren in der Sicherheitssteuerung von Zyklusanfang bis Zyklusanfang (Reaktionszeit bei Querkommunikation (T3), Seite 23).

- **WDT 1**: Konfigurierte Überwachungszeit der Feldbuskommunikation zwischen sicherem Eingangsgerät und Sicherheitssteuerung 1
  - Erstmalig als maximale Wartezeit, die durch das sichere Feldbusprotokoll im Eingangsgerät anfallen kann, bevor es ein geändertes Signal vom Sensor an Sicherheitssteuerung 1 abschicken kann. So lange muss das Signal des Sensors mindestens anstehen, um zu garantieren, dass es von Sicherheitssteuerung 1 gesampelt werden kann.
  - Zum zweiten Mal als maximale Transportzeit, in der das abgeschickte Signal in Sicherheitssteuerung 1 als Eingangssignal ankommt. Diese Zeit überwacht die Performance des Feldbusses und der Standardsteuerung als Vermittler.
  - WDT 1 entspricht "WDT In", siehe Kap. 4.2.1 "Reaktionszeiten bei Feldbussteuerungen" auf Seite 21
- **WDT 2**: Konfigurierte Überwachungszeit der Querkommunikation zwischen Sicherheitssteuerung 1 und Sicherheitssteuerung 2
  - Erstmalig als maximale Wartezeit, inklusive eines Applikationszyklus zur Abbildung der neuen Eingangssignale auf Netzwerkvariablen, bevor Sicherheitssteuerung 1 neue Variablenwerte an Sicherheitssteuerung 2 abschicken kann. So lange müssen die Werte in Sicherheitssteuerung 1 mindestens anstehen, um zu garantieren, dass sie von Sicherheitssteuerung 2 gesampelt werden können (Querkommunikation mit Safety Netzwerkvariablen, Seite 116).

  - Zum zweiten Mal als maximale Transportzeit, in der die abgeschickten Variablenwerte in Sicherheitssteuerung 2 ankommen. Diese Zeit überwacht die Performance der Ethernet-Verbindung und der beiden Standardsteuerungen als Vermittler.

- **WDT 3**: Konfigurierte Überwachungszeit der Feldbuskommunikation zwischen Sicherheitssteuerung 2 und sicherem Ausgangsgerät
  - Erstmalig als maximale Wartezeit, inklusive eines Applikationszyklus zur Abbildung der neuen Variablenwerte auf Ausgangssignale, bevor Sicherheitssteuerung 2 ein geändertes Ausgangssignal an das Ausgangsgerät abschicken kann. So lange muss das Signal in Sicherheitssteuerung 2 mindestens anstehen, um zu garantieren, dass es vom Ausgangsgerät gesampelt werden kann.
  - Zum zweiten Mal als maximale Transportzeit, in der das abgeschickte Signal im Ausgangsgerät ankommt. Diese Zeit überwacht die Performance des Feldbusses und der Standardsteuerung als Vermittler.
  - WDT 3 entspricht "WDT Out", siehe Kap. 4.2.1 "Reaktionszeiten bei Feldbussteuerungen" auf Seite 21

Gefahr des Undersamplings

In jedem Schritt der Sicherheitskette muss ein Signal "lang genug" in der Quelle (sicheres Eingangsgerät, Sicherheitsteuerung 1, Sicherheitsteuerung 2) anliegen (nämlich so lange wie die WDT-Zeit der Verbindung), damit es auch im Worst-Case zeitlichen Verhaltens eine Chance hat, weitergeschickt zu werden. Das heißt, wenn das Sensorsignal im Eingangsgerät oder eine Netzwerkvariable in Sicherheitsteuerung 1 oder das Ausgangssignal in Sicherheitsteuerung 2 nur kurzzeitig von Wert A auf Wert D wechselt und wieder zurück auf A, kann es sein, dass der Wert D nie übertragen wird (Undersampling). Wenn mit dem verschluckten Wert D eine Sicherheitsfunktion angefordert werden sollte, versagt die Sicherheitsfunktion und die funktionale Sicherheit geht verloren. Wenn ein Glied in der Sicherheitskette das Signal nicht automatisch auf das Mindestmaß für den nächsten Schritt verlängert (WDT-Zeit k), dann muss das Signal schon im vorherigen Glied entsprechend länger anliegen als bloß WDT-Zeit k-1.

Zum Beispiel, wenn Sicherheitssteuerung 1 das Signal vom Eingangsgerät (oder einer anderen Sicherheitssteuerung 0) empfängt und unverarbeitet weiter veröffentlicht, ist folgendes zu berücksichtigen:



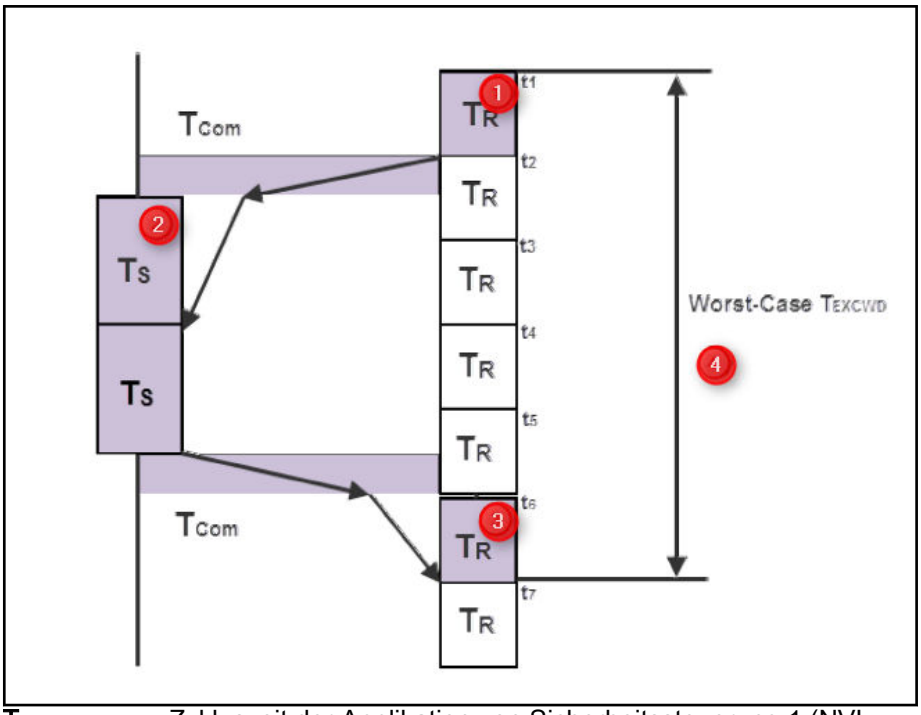
Eine Standzeit T_Q des Signals im Eingangsgerät, die gerade sicherstellt, dass das Signal in Sicherheitssteuerung 1 ankommt ($T_Q \geq$ WDT 1) reicht noch nicht, um in Sicherheitssteuerung 1 eine danach anschließende Standzeit T_S zu garantieren, die die NVL-Verbindung zu Sicherheitssteuerung 2 braucht ($T_S \geq$ WDT 2). Die Standzeit muss dazu ($T_Q \geq$ WDT 1 + Zykluszeit Sicherheitssteuerung 1 + WDT 2) sein. Bei einer Sicherheitskette mit mehreren NVL-Verbindungen hintereinander sind die Zeiten aller involvierten Verbindungen und Steuerungen zu kombinieren.

Überwachungszeit bei Querkommunikation

Bei Querkommunikation mit Netzwerkvariablen hängt die Überwachungszeit der Verbindung (WDT 2) vom Empfänger (Sicherheitssteuerung 2) ab.

Eine Überwachungszeit kleiner als der Worst-Case des Kommunikationszyklus der Netzwerkvariablen führt zu keinem stabil laufenden System. Im Kommunikationszyklus kommt es zu Zeitverlusten, da die Applikationszyklen von

Sender und Empfänger nicht synchron laufen und auch die systemabhängige Transportzeit berücksichtigt werden muss.



T_S	Zykluszeit der Applikation von Sicherheitssteuerung 1 (NVL-Sender)
T_R	Zykluszeit der Applikation von Sicherheitssteuerung 2 (NVL-Empfänger)
T_{Com}	Transportzeit von einer Sicherheitssteuerung über deren Standardsteuerung zur anderen Standardsteuerung und deren Sicherheitssteuerung

Abb. 4-4: Sequenzdiagramm eines Netzwerkvariablen-Kommunikationszyklus (links Sender, rechts Empfänger)

- (1): Der Empfänger schickt das Protokoll zum Ende des Zyklus ab. Da seine Zeitbasis aber für die Dauer des Zyklus eingefroren ist, beginnt die überwachte Zeit schon mit dem Startzeitpunkt t_1 des Zyklus.
- (2): Das vom Empfänger gesendete Protokoll kommt zu spät (nach Beginn des Zyklus des Senders). Dies bedeutet, dass sich das Lesen des Protokolls bis zum Start des nächsten Zyklus verzögert.
- (3): Die von Empfänger gesendeten Daten kommen verspätet (nach Beginn des Zyklus des Empfängers). Dies bedeutet, dass sich der Prüfung der Daten und der Überwachungszeit bis zum nächsten Zyklusbeginn (im Beispiel: t_7) verzögert.
- (4): Worst-Case T_{excWD} des Kommunikationszyklus

Aus diesen Verzögerungsfaktoren ergibt sich die kleinste sinnvolle Überwachungszeit (WDT 2).

☞ Die Überwachungszeit (**WDT 2**) für eine Safety Netzwerkvariablen-Verbindung muss größer sein als:

2 x Transportzeit T_{Com} + 2 x Zykluszeit Sender T_s + 2 x Zykluszeit Empfänger T_R.

4.3 Planung der Adressen

Adressen für Feldgeräte

Sicherheitsprotokolle wie Profisafe oder CSos sichern die Verbindung zwischen Sicherheitssteuerung und sicheren Feldgeräten über den Standardfeldbus mittels Safety-Adressen ab (F_Dest_Add und F_Source_Add bei Profisafe und SDID und NetworkID bei CSos). Deren eindeutige Vergabe innerhalb eines Feldbusses muss geplant werden, insbesondere bei mehreren Sicherheitssteuerungen im gleichen Feldbus (Topologie T2).

Für genaue Anweisungen zur Eindeutigkeit der entsprechenden feldbus-spezifischen Parameter siehe Feldbusse und Netzwerkvariablen, Allgemeines, Seite 573

Manche Feldgeräte haben mehrere parallele Safety-Verbindungen gleichzeitig. In dem Fall muss für jede Safety-Verbindung eine eigene Adresse und, je nach Feldbus eine eigene Verbindungs-ID eingeplant werden.

Bedeutung der Adresseindeutigkeit:

- bei Topologie T1: eindeutige Slave-Adressen in der Sicherheitssteuerung
- bei Topologie T2: gemeinsamer Slave-Adressraum für alle Sicherheitssteuerungen unter gleicher Standardsteuerung
- bei Topologie T3: gemeinsamer Slave-Adressraum für alle querkommunizierenden Sicherheitssteuerungen

Adressen bei Querkommunikation

Für Netzwerkvariablen werden Safety-Adressen, Connection IDs und Listenidentifikatoren innerhalb eines Projekts automatisch eindeutig vergeben. Listenidentifikatoren dienen der IndraLogic-internen Kennung. Für jede Verbindung Sender-Empfänger braucht IndraLogic zwei Identifikatoren (einen in jede Richtung). (Beim Empfänger ist ihre Anzahl also 2, beim Sender 2 x Max. Anzahl Empfänger).

☞ Netzwerkvariablen verwenden standardmäßig UDP Broadcasts im Maschinennetzwerk.

Um die Netzlast zu reduzieren, können die IP-Adressen der gegenüberliegenden Standardsteuerung fix eingestellt werden.

Für diesen Fall ist es sinnvoll, die IP-Adressen aller Steuerungen im Maschinennetzwerk bei der Planung des Gesamtsystems zu vergeben.

Mehrere IndraLogic-Projekte

Wenn die Software für die Maschine in mehreren getrennten IndraLogic-Projekten entwickelt wird, müssen für Netzwerkvariablen maschinenweit eindeutige Safety-Adressen, Connection IDs und Listenidentifikatoren bei der Planung des Gesamtsystems vergeben werden und manuell eingestellt werden.

5 Voraussetzungen und Verwendung

5.1 Systemvoraussetzungen

Als Mindestausstattung für kleinere IndraWorks Safety Manager-Projekte mit maximal 100 Bausteinen, maximal 10 Visualisierungen und maximal 8 Feldbusteilnehmern gelten folgende Systemvoraussetzungen:

- 2 GByte RAM
- 1 GHz Pentium
- 5 GByte freier Festplattenspeicher

Das Programmiersystem IndraWorks Safety Manager ist nur freigegeben für:

- Betriebssystem: Windows 10, 64 bit
- Bildschirmauflösung: 96 dpi

5.2 Qualifiziertes Personal

Voraussetzung für die Anwendung sicherheitstechnischer Produkte ist eine ausreichende Qualifikation.

So muss das Personal, welches Sicherheitsapplikationen mit dem vorliegenden Produkt plant, erstellt und in Betrieb nimmt

- mit den geltenden Normen und Vorschriften vertraut sein
- mit den einschlägigen Sicherheitskonzepten in der Automatisierungstechnik vertraut sein
- mit grundlegenden Unfallverhütungs- und Arbeitssicherheitsvorschriften und eventuell besonderen Betriebsvorschriften vertraut sein
- die herstellerspezifischen Handbücher der Sicherheitssteuerung und der verbundenen sicheren Feldgeräte und Schutzeinrichtungen kennen
- die Sicherheitshinweise in diesem Handbuch gelesen und verstanden haben
- gute Kenntnisse der IndraLogic Programmierung vorweisen

5.3 IndraWorks SafetyManager-Lizenz

Es gibt zwei Arten von SafetyManager-Lizenzen:

- IndraWorks SafetyManager (R911322463)

  Diese Lizenz wird zum Entwickeln von Safety-Programmen benötigt.

  Sie ermöglicht:

  – das Einloggen in die Safety-Steuerung inklusive aller Debug-Handlungen
  – das Setzen von Passwörtern
  – das Herunterladen von Safety-Programmen
  – das Erzeugen von Bootprojekten
  – das Abrufen des Steuerungslogbuchs
  – den Firmwareupdate

- IndraWorks SafetyManager Lite (R911383338)

  Diese Lizenz wird für die Ferndiagnose bzw. den Service von Safety-Applikationen benötigt.

  Sie ermöglicht:

- das Einloggen in die Steuerung über den Fernwartungszugang
- das Debuggen ohne die Möglichkeit, Werte zu ändern
- das Abrufen des Steuerungslogbuchs

Die Lizenzen werden wie folgt installiert:

1. Lizenzübersicht öffnen

   Die Lizenzübersicht ist über **Extras** ▸ **Optionen...** und dann im Baum unter **Allgemein** ▸ **Softwarelizenzen** erreichbar.

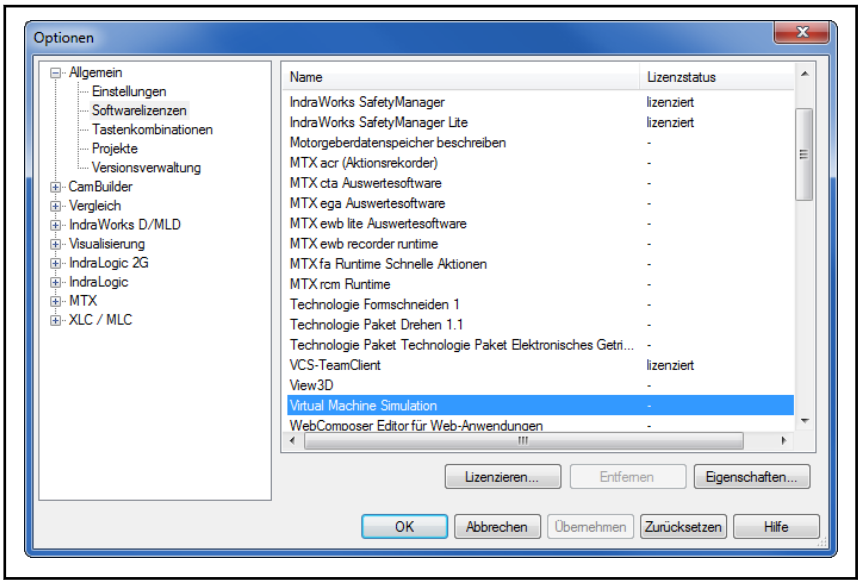


Abb. 5-1: Lizenzübersicht

2. Lizenzieren

   Über die Schaltfläche **Lizenzieren...** gelangt man in den Lizenzdialog. Hier kann der Aktivierungsschlüssel eingetragen werden.

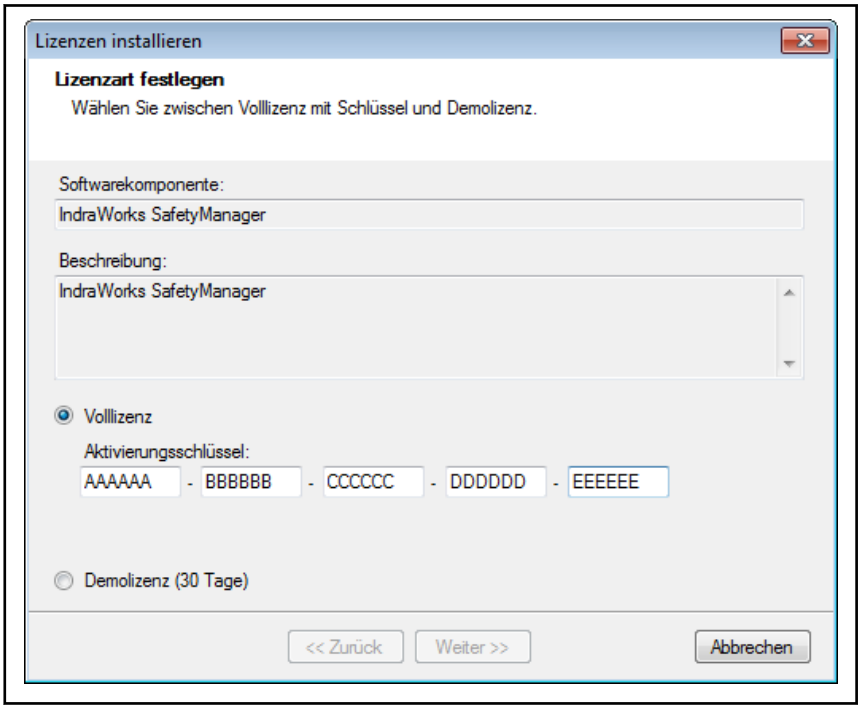


Abb. 5-2: Lizenzdialog

Nach der Bestätigung mit **Weiter >>** wird der Aktivierungsschlüssel geprüft und die Lizenz übernommen.

Die IndraWorks SafetyManager-Lizenz (Lite) ist jetzt installiert und aktiviert.

5.4 Hinweise

Ein Projekt, das mit SafeLogic erstellt wurde, garantiert nicht, dass die gesamte Anwendung sicher ist.

⚠ VORSICHT

Es müssen vom Anwender geeignete Sicherheitskonzepte für ihre Anlagen, Maschinen und Programme erstellt werden.

⚠ VORSICHT

Hersteller und Betreiber von Maschinen mit SafeLogic Steuerungen müssen bei Entwicklung, Einbau, Inbetriebnahme, Betrieb und technischen Überprüfungen alle in ihrem Land geltenden nationalen und internationalen Rechtsvorschriften, Sicherheitsregeln und Normen in eigener Verantwortung einhalten.

⚠ VORSICHT

Die Hinweise der Anwendungsbeschreibung von SafeLogic und der herstellerspezifischen Handbücher der eingesetzten sicherheitstechnischen Geräte müssen beachtet werden.

⚠ VORSICHT

Zum Arbeiten an Sicherheitsapplikationen dürfen nur die Befehle, Werkzeuge, Benutzeroberflächen- Steuerelemente, Editor-Registerkarten und Ansichten verwendet werden, die in folgendem Kapitel aufgelistet sind:

Kap. 21 "Liste zulässiger oder geänderter Funktionen" auf Seite 627

6 Projektentwicklung in SafeLogic

Im folgenden werden die Struktur des Projektbaums und die Objekte des Projektbaums beschrieben, welche für die Erstellung einer Sicherheitsapplikation mit SafeLogic gemäß der Richtlinien IEC 62061 und ISO 13849 zur Verfügung stehen.

Das Anlegen eines Projekts und die Bedienung der Oberfläche erfolgt nach dem Bedienkonzept von IndraWorks.

☞ Das Erstellen von Sicherheitsapplikationen setzt eine IndraWorks Engineering-Installation voraus, welche die SafeLogic Projektierung mit dem System MLC oder MTX unterstützt.

Befehle von IndraWorks, die in SafeLogic nicht zur Verfügung stehen, sind in der Regel sichtbar, aber nicht aktivierbar.

Kompatibilitätsmodus

Der Kompatibilitätsmodus legt für ein IndraWorks-Projekt fest, welche Version von "IndraWorks Engineering" mindestens erforderlich ist, um das Projekt bearbeiten zu können. Wird ein Projekt geöffnet, das eine ältere Versionskennung besitzt, ist zum Bearbeiten des Projekts als Kompatibilitätsmodus immer die höchste, zertifizierte IndraWorks-Version auszuwählen. Die Kompatibilität des Projekts zur ursprünglichen Version geht hierbei verloren!

⚠ VORSICHT

Das Bearbeiten des Projekts ist nur in der höchsten, zertifizierten IndraWorks-Version zulässig!

Das Bearbeiten im Kompatibilitätsmodus ist nicht zulässig!

⚠ VORSICHT

Der Download der Safety-Applikation bzw. das Erzeugen der Bootapplikation muss innerhalb von 8 Stunden nach dem Einschalten der Safety-Steuerung erfolgen!

Andernfalls ist die Safety-Steuerung aus- und wieder einzuschalten.

☞ Um den korrekten Hochlauf der Safety-Steuerung über die Bootapplikation vor der Abnahme zu überprüfen, ist die Safety-Steuerung aus- und wieder einzuschalten.

Gelbe Sicherheitsstrukturen

Durch gelbe Strukturen werden die Safety-Objekte in der Projektstruktur und die Safety-Editoren deutlich hervorgehoben und unterscheiden sich dadurch optisch klar von den entsprechenden Standard-Objekten und Standard-Editoren.

Zusätzlich sind die sicheren Signalflüsse ebenfalls gelb markiert. Dies unterstützt und erleichtert die Tätigkeiten bei der Entwicklung, Verifikation und Abnahme der Sicherheitsapplikation.

6.1 Umsetzung des PLCopen Architekturmodells

Das PLCopen Architekturmodell wird in IndraWorks auf folgende Art und Weise umgesetzt:

- Die Sicherheitssteuerung wird unter der Hauptsteuerung eingefügt (siehe Sicherheitssteuerung auf Seite 38)

- Alle physikalischen Geräte werden im Gerätebaum unter der Hauptsteuerung eingefügt. Sie werden automatisch mit der richtigen Steuerung verknüpft. Dabei werden die E/As der Sicherheitssteuerung zusätzlich als logische E/As unter der Sicherheitsapplikation eingefügt (siehe Logische E/As - Überblick auf Seite 47)
- Der Datenaustausch zwischen der Sicherheitssteuerung und der Hauptsteuerung erfolgt über die Objekte "GVL für logischen Austausch" der Standardapplikation und "Logisches Austauschgerät" der Sicherheitsapplikation (siehe Logische E/As für den Datenaustausch Standard-/ Sicherheitssteuerung auf Seite 52).

Zum PLCopen-Architekturmodell siehe Abb. 3-4 "Empfohlener Anwendungsbereich der beiden Programmierlevel" auf Seite 16

6.2 Vorgehensweise bei Erstellung einer Sicherheitsapplikation

Eine Sicherheitsapplikation kann unter einer Standardapplikation durch Hinzufügen des Safety-Funktionsmoduls (kurz Safety-FM) bzw. Safety-Erweiterungsmodul angelegt werden.

Möglichkeiten, ein Safety-FM im Projekt anzulegen:

- beim Anlegen der Steuerung
- nachträgliches Zuordnen durch Drag and Drop oder
- nachträgliches Zuordnen durch Copy and Paste

Die Möglichkeiten werden am Beispiel einer XM Steuerung mit Funktionsmodul **Safety_CPU_XFE01_1_SY_01** gezeigt.

Das Beispiel gilt analog für VPx Steuerungen mit dem Funktionsmodul **Safety_CPU_PFC01_1_SY_01**.



Pro Steuerung kann maximal ein Safety-Funktionsmodul angeschlossen werden!

FM zusammen mit Standardsteuerung anlegen

Wird in einem Projekt eine Standardsteuerung angelegt, dann besteht die Möglichkeit gleich ein Safety-FM mit anzulegen.

Alternativ können Sie bei bereits angelegten Steuerungen nachträglich ein Safety-Funktionsmodul hinzufügen. Öffnen Sie hierzu das Kontextmenü des Steuerungsknotens im Projektexplorer und rufen Sie den "Einfügen-Wizard" auf. Wählen Sie im Dialog "Schnittstellen" unter der Rubrik "Extensionmodule" ein Safety-FM aus.

Mittels Tooltip wird auf dieser Dialogseite auf eventuelle Fehlkonfigurationen hingewiesen.



Abb. 6-1: Safety-FM im Wizard hinzufügen

FM durch Drag and Drop anlegen

Ist bereits eine Standardsteuerung im Projekt enthalten, so kann jederzeit nachträglich ein Safety-FM zur Standardsteuerung hinzugefügt werden.

Dies kann mittels Drag and Drop erfolgen, indem das Safety-FM mit der Maus vom Bibliotheksexplorer auf den Knoten der Standardsteuerung gezogen und dort fallen gelassen wird.

Das Einfügen eines Safety-FM ist nur an einer geeigneten Standardsteuerung möglich, eine falsche Konfiguration kann somit nicht entstehen.

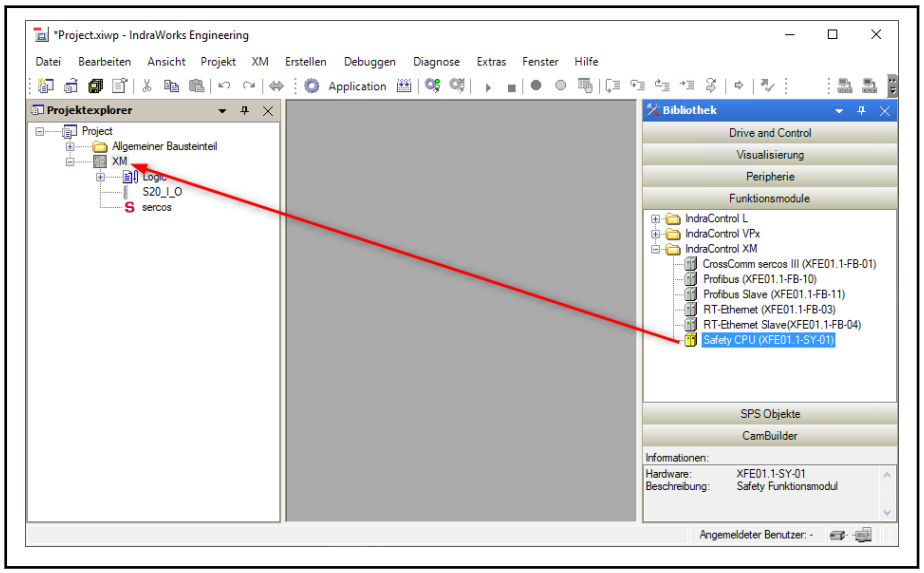


Abb. 6-2: Safety-FM per Drag and Drop einfügen

FM durch Copy and Paste anlegen

Alternativ kann das Safety-FM auch per Copy and Paste an die Standardsteuerung angefügt werden.

Hierzu ist das Safety-FM im Bibliotheksexplorer auszuwählen, mittels <Strg>+<C> oder über den Kontextmenü-Befehl **Kopieren** zu kopieren.

Danach ist die Standardsteuerung per Mausklick im Projektexplorer auszuwählen.

Nun kann das Safety-FM entweder mittels <Strg>+<V> oder über den Kontextmenü-Befehl **Einfügen** eingefügt werden.

Auch hierbei wird überwacht, dass das Safety-FM nur den dafür zulässigen Objekten im Projektexplorer zugeordnet werden kann.

Nach dem Anlegen des Safety-Funktionsmoduls muss die Benutzerverwaltung aufgebaut werden.

Beschreibung der Benutzerverwaltung und der Safety-Benutzerkonfiguration siehe Benutzerverwaltung im Projekt auf Seite 78.

6.3 Projektstruktur

Nach dem Einfügen des Safety-Funktionsmoduls ins Projekt, werden unterhalb des Safety-FM-Knotens die erforderlichen Objekte für die Applikationserstellung automatisch erzeugt.

Gelbe Safety-Strukturen

Durch gelbe Strukturen werden die Safety-Objekte in der Projektstruktur und die Safety-Editoren deutlich hervorgehoben und unterscheiden sich dadurch optisch klar von den entsprechenden Standard- Objekten und Standard-Editoren.

Zusätzlich sind die sicheren Signalflüsse ebenfalls gelb markiert. Dies unterstützt und erleichtert die Tätigkeiten bei der Entwicklung, Verifikation und Abnahme der Sicherheitsapplikation.

Safety-Objekte

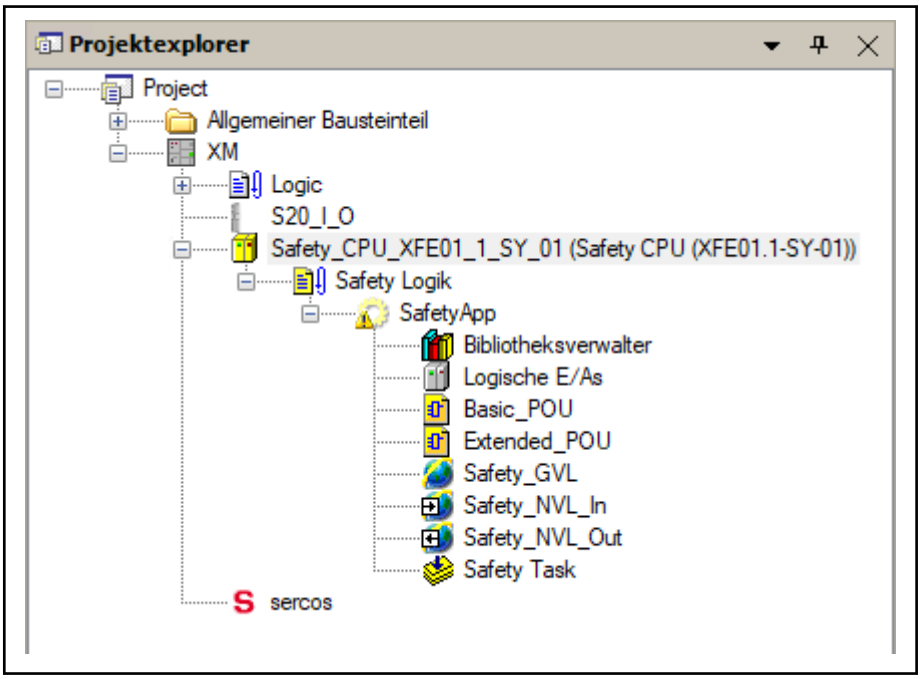


Abb. 6-3: Sicherheitsapplikation im Projektexplorer

Die Objekte unterhalb des Safety-Funktionsmoduls unterliegen den folgenden Einschränkungen:

- Es ist nur eine Applikation erlaubt. Diese wird automatisch angelegt.
- Es gibt nur eine Task. Diese wird automatisch angelegt.
- Der Bibliotheksverwalter wird automatisch eingefügt.

Die folgenden Objekte bzw. Knotenpunkte sind unterhalb des Safety-FM genau einmal vorhanden:

- **Safety Logik:** Logischer Knotenpunkt der Sicherheitssteuerung, unter dem genau eine Sicherheitsapplikation existieren kann.
- **SafetyApp:** Knotenpunkt, unter dem die zur Sicherheitsapplikation gehörenden Objekte liegen.

Objekt, das die Ausführungsversion des Codes und den aktuell festgehaltene Stand (Pin) der Applikation festlegt (siehe Pinnen auf Seite 161).

Der Editor des Objekts verwaltet die Liste der aktuell zur Sicherheitsapplikation gehörenden Objekte.

- **Bibliotheksverwalter:**

  Enthält die auf der eingefügten Sicherheitssteuerung zur Verfügung stehenden Bibliotheken:

  - SafetyStandard.library, Seite 229
  - SafetyUtilities.library, Seite 240
  - SafetyPLCopen.library, Seite 292
  - SafetyPLCopenPress.library, Seite 422
  - SafetyDriveMotion.library, Seite 536
  - SafetyProfisafeHost.library, Seite 577
  - SafetyCSos.library, Seite 589
  - SafetyNetVar.library, Seite 599, enthält die Treiberbausteine zum Austausch von Netzwerk-Daten zwischen einem Sender und einem oder mehreren Empfängern.

- **Logische E/As:** Knotenpunkt, dem logische E/A-Objekte hinzugefügt werden können. Diese hinzugefügten logischen E/As dienen dem Austausch von Daten und E/As mit der Standardsteuerung (siehe Logische E/As - Überblick auf Seite 47)
- **Safety Task:** Dieses Objekt listet alle Programme auf, die auf die Steuerung geladen und ausgeführt werden können (siehe Safety Task auf Seite 63).

Die folgenden Objekte können unterhalb der Applikation manuell (auch mehrfach) hinzugefügt werden:

- **Safety-BasicPOU:** POU (Programm oder Funktionsbaustein) auf Programmierlevel Basic auf Seite 81
- **Safety-ExtendedPOU:** POU (Programm oder Funktionsbaustein) auf Programmierlevel Extended auf Seite 81
- **Globale Variablenliste (Safety):** Deklaration der nur innerhalb der Sicherheitsapplikation gültigen globalen Variablen (siehe Globale Variablenliste (GVL) auf Seite 91)
- **Safety_NVL_OUT:** NVL-Sender, Liste von Netzwerkvariablen zur Realisierung einer Querkommunikation zwischen den Sicherheitsapplikationen mehrerer Steuerungen, bei bedarf mehrfach (Editor einer Sender NVL auf Seite 68).
- **Safety_NVL_IN:** NVL-Empfänger, Liste von Netzwerkvariablen zur Realisierung einer Querkommunikation zwischen den Sicherheitsapplikationen mehrerer Steuerungen, bei bedarf mehrfach (Editor einer Empfänger NVL auf Seite 71).

Alle Objekte können über die Benutzerverwaltung im Projekt auf Seite 78 vor unberechtigtem Zugriff geschützt werden.

⚠ VORSICHT

Damit nur berechtigte Personen das neue Geräteobjekt bearbeiten können, muss der Entwickler gleich nach dem Einfügen der Sicherheitssteuerung die Änderungsrechte explizit in dem Dialog "Eigenschaften" unter dem Tab "Zugriffskontrolle" wie folgt konfigurieren:

- + für Safety
- + für Safety.ExtendedLevel
- - für Everyone

Kopieren und Verschieben von Safety-Objekten

Es gibt hinsichtlich der Kopieren- und Verschieben-Funktionalität keine Unterschiede zwischen Standard- und Safety-Objekten.

Objekte aus dem Bereich der Standard-Steuerung können nicht unterhalb einer Safety-Steuerung eingefügt werden und umgekehrt.

Weiterhin ist es nicht möglich, Safety-Objekte unterhalb des "Allgemeinen Bausteinteils" abzulegen.

Weiteres Vorgehen bei Safety Projekt mit Benutzerverwaltungs-Template

1. Eigenschaften-Dialog der Sicherheitssteuerung öffnen: Sicherheitssteuerung im Projektbaum selektieren und Kontextmenü-Befehl **Eigenschaften** aktivieren.
2. Registerkarte **Zugriffskontrolle** öffnen.
3. Die Zellen in den Spalten **Bearbeiten** und **Entfernen** der Benutzergruppe **Everyone** durch Doppelklick öffnen, jeweils **Verbieten** auswählen und mit **OK** bestätigen.
4. Die Zellen in den Spalten **Bearbeiten** und **Entfernen** der Benutzergruppen **Safety** und **Safety.ExtendedLevel** durch Doppelklick öffnen, jeweils **Erlauben** auswählen und mit **OK** bestätigen.

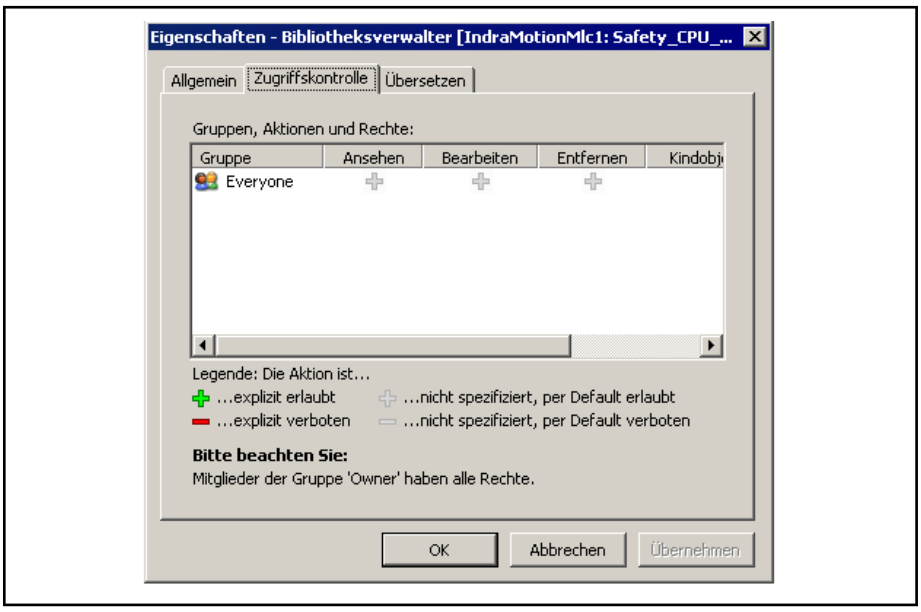


Abb. 6-4: Eigenschaften der Sicherheits-SPS, Registerkarte 'Zugriffskontrolle'

6.4 Sicherheitssteuerung

Hinzufügen der Sicherheitssteuerung

Bei der Topologie "Vater-Kind"-Steuerung wird die Sicherheitssteuerung unter der Standardsteuerung angeordnet. Dies erfolgt durch Selektion der Stan-

dardsteuerung und Aktivierung des Kontextmenübefehls **Hinzufügen...** mit Auswahl des Funktionsmoduls Safety-FM.

Beim Einfügen der Sicherheitssteuerung werden immer der logische Knotenpunkt **Safety Logik**, das Sicherheitsapplikationsobjekt **SafetyApp**, das Task-Objekt **Safety Task**, der **Bibliotheksverwalter** und der Knotenpunkt **Logische E/As** automatisch mit eingefügt.

Die Sicherheitssteuerung wird automatisch mit der Standardsteuerung auf eine neuere Version aktualisiert. Dabei werden eventuell Bibliotheken durch neuere Versionen ersetzt.

Informationen am Ikon der Sicherheitssteuerung

Tritt an der Sicherheitssteuerung ein Problem auf, so wird dies mit einem kleinen Warn-Ikon signalisiert. Genauere Informationen erhält man über den Tooltip.

Ein typisches Beispiel für eine solche Warnung ist das Fehlen der SafetyManager-Lizenz.

Objekteigenschaften der Sicherheitssteuerung

Eine im Projektbaum existierende Sicherheitssteuerung besitzt einen Eigenschaftendialog mit den Registerkarten **Allgemein** und **Zugriffskontrolle**. Der Eigenschaftendialog wird durch Selektion der Sicherheitssteuerung im Projektbaum und Aktivierung des Kontextmenü-Befehls **Eigenschaften...** geöffnet.

Editor der Sicherheitssteuerung

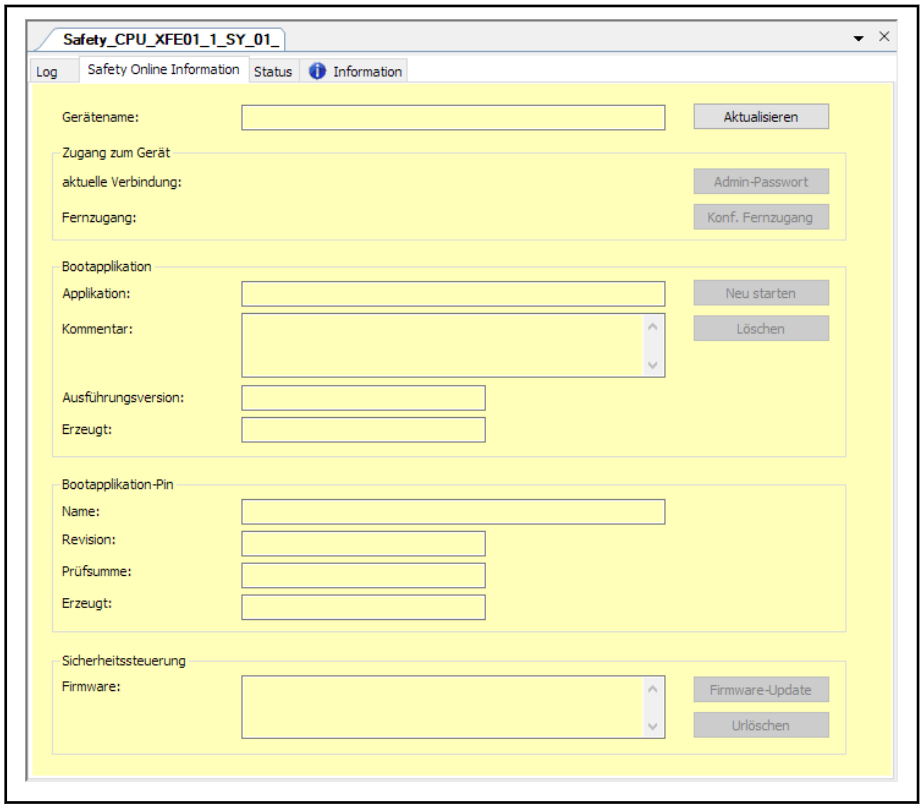


Abb. 6-5: Editor der Sicherheitssteuerung, Registerkarte 'Safety Online Information'

Der Dialog enthält folgende Registerkarten:

- **Log:** Beschreibung siehe Logbuch auf Seite 198

- **Safety Online Information:** Diese Registerkarte enthält sicherheitsspezifische Informationen und Befehle der Sicherheitssteuerung. (Siehe Safety Online Information auf Seite 40)
- **Status:** siehe Diagnose der Kommunikation auf Seite 202
- **Information:** Anzeige von allgemeinen Informationen. (Siehe Informationen zur Hardware auf Seite 204)

Safety Online Information

Diese Registerkarte (siehe Abb. 6-5 "Editor der Sicherheitssteuerung, Registerkarte 'Safety Online Information'" auf Seite 39) bietet Informationen zur Sicherheitssteuerung. Der Steuerungsinhalt kann über Befehle manipuliert werden.

Ist kein Safety-Funktionsmodul erreichbar, bleiben die Textausgabefelder leer.

Befehle und Textausgaben:

- Befehl **Aktualisieren**: Damit die aktuellen Informationen zur Bootapplikation in der Steuerung ausgegeben werden, muss zunächst dieser Befehl ausgeführt werden.
- **Knotenname**: Der Name der Sicherheitssteuerung, auf der eingeloggt ist.

Informationen und Befehle zur Bootapplikation: (Diese Information kann nur ausgegeben werden, wenn auf der Steuerung ein Bootprojekt hinterlegt ist.)

- **Name:** Name des aktuell geladenen Safety-Applikationsobjekts
- **Kommentar:** Kommentar, wie er im Eigenschaften-Dialog des Safety-Applikationsobjekts (Registerkarte **Safety**, Abschnitt **Kommentar**) hinterlegt wurde.
- **Ausführungsversion:** Die Ausführungsversion, wie sie im Eigenschaften-Dialog der Safety-Applikationsobjekts eingestellt wurde.
- **Erzeugt:** Datum und Uhrzeit, zu der das Bootprojekt erzeugt wurde.
- Befehl **Löschen**: Löscht das Bootprojekt in der Sicherheitssteuerung und entlädt die laufende Applikation. (Siehe Bootapplikation löschen auf Seite 210)
- Befehl **Neu starten**: Entlädt die aktuelle Applikation, lädt und startet das Bootprojekt. (Siehe Neustart der Bootapplikation auf Seite 137)
- Befehl **Firmware-Update**: Lädt eine (andere) Firmware-Release aus einer Datei auf die Steuerung, um eine dort bereits vorhandene Firmware zu ersetzen. (Siehe Firmware-Update auf Seite 212)
- Befehl **Urlöschen**: Erlaubt das Entladen der Applikation, Löschen der Bootapplikation und Rücksetzen des Passworts. Dieser Befehl kann ohne Passwort ausgeführt werden und erlaubt das Rücksetzen der Steuerung, auch wenn ein Passwort vergessen wurde. (Siehe Urlöschen auf Seite 227)

Unter **Bootapplikation Pin** wird die Pin-Kennung zum in der Sicherheitssteuerung abgelegten Bootprojekt ausgegeben (siehe Pinnen auf Seite 161). Sie entspricht den Informationen, wie sie im Editorfenster der Safety-Applikationsobjekts angezeigt werden.



Wurde das Bootprojekt zu einer nicht gepinnten oder nach dem Pinnen veränderten Bootapplikation erzeugt, sind die Ausgabefelder leer.

- **Name:** Die Bezeichnung des gepinnten Standes

- **Revision:** Die Revisionsnummer
- **Prüfsumme:** Die Prüfsumme über die gepinnte Applikation, zu der das Bootprojekt erzeugt wurde. (Pin-Prüfsumme)
- **Erzeugt:** Datum und Uhrzeit der letzten Änderung

Informationen und Kommandos zur Sicherheitssteuerung

- **Firmware:** Aus der Sicherheitssteuerung wir die genaue Bezeichnung der Firmware sowie der Hardware-Index ausgelesen.
- Befehl **Passwort setzen:** Setzt ein Passwort für alle schreibenden Zugriffe auf die Steuerung.

  Dieses Passwort ist initial leer.

  Schreibende Zugriffe auf die Sicherheitssteuerung (z. B. Projekt laden) müssen mit diesem Passwort bestätigt werden.

  (Siehe Schutz der Sicherheitssteuerung vor Schreibzugriff auf Seite 195)

6.5 Safety Logik

Den logischen Knotenpunkt **Safety Logik** gibt es aus Gründen der Symmetrie zur Standardsteuerung. Er hat in der Sicherheitssteuerung keine Bedeutung.

Der logische Knotenpunkt **Safety Logik** wird automatisch mit der Sicherheitssteuerung in den Projektbaum eingefügt. Unter einer Sicherheitssteuerung kann es nur genau einen Knoten **Safety Logik** geben. Ihm kann als Objekt nur eine **Sicherheitsapplikation** hinzugefügt werden.

Safety Logik besitzt einen Eigenschaftendialog mit den Registerkarten **Allgemein** und **Zugriffskontrolle**. Der Eigenschaftendialog wird durch Selektion von **Safety Logik** im Projektbaum und Aktivierung des Kontextmenü-Befehls **Eigenschaften...** geöffnet.

Safety Logik kann mit keinem Editor geöffnet werden.

6.6 Sicherheitsapplikationsobjekt SafetyApp

Aufbau des Projektbaums der Sicherheitsapplikation

Der Projektbaum wird im Gerätefenster (**Geräte** des Menüs **Ansicht**) dargestellt.

Die für die Sicherheitssteuerung und ihre Programmierung relevanten Objekte befinden sich unterhalb der Sicherheitssteuerung. Direkt darunter liegt immer der symbolische Knotenpunkt **Safety Logik** .

Unter jedem **Safety Logik**-Objekt kann sich immer nur **ein** Sicherheitsapplikationsobjekt **SafetyApp** (Standard-Name) befinden, welches folgende Safety-Objekte enthalten kann:



Abb. 6-6: Bsp. Safety-Applikationsobjekt mit Objekten

*Folgende Objekte müssen in einem Sicherheitsapplikationsobjekt genau **einmal** vorhanden sein:*

- Bibliotheksverwalter
- Logische E/As
- Safety Task

⚠ VORSICHT

Werden unterhalb des "SafetyApp" Ordnerstrukturen verwendet (siehe Online-Hilfe von Standard-IndraWorks), so dürfen diese Ordner nicht kopiert werden.

Hinzufügen des Safety-Applikationsobjekts

Das Hinzufügen des Sicherheitsapplikationsobjekt im Projektbaum erfolgt automatisch beim Hinzufügen der Sicherheitssteuerung

Im Eigenschaften-Dialog (Selektion des Objekts **SafetyApp** und Aktivierung des Befehls **Eigenschaften...**) kann in der Registerkarte **Allgemein** ein Name und in der Registerkarte **Safety** ein Kommentar für das Safety-Applikationsobjekt editiert werden (siehe Abb. 6-7 "Eigenschaften des Sicherheitsapplikationsobjekts, Registerkarte 'Safety'" auf Seite 43).

Objekteigenschaften des Safety-Applikationsobjekts



Der Eigenschaften-Dialog des Safety-Applikationsobjekts und aller Objekte des Safety-Applikationsobjekts besitzt die Registerkarten **Allgemein** (mit Namen, Objekttyp und Editor, mit welchem das Objekt geöffnet wird) und **Zugriffskontrolle** (Zugriffsrechte der Benutzergruppen für das Objekt).

Zusätzliche, objektspezifische Registerkarten werden bei den einzelnen Objekten beschrieben.

Der Eigenschaftendialog wird durch Selektion des Objekts im Projektbaum und Aktivierung des Kontextmenübefehls **Eigenschaften** geöffnet.

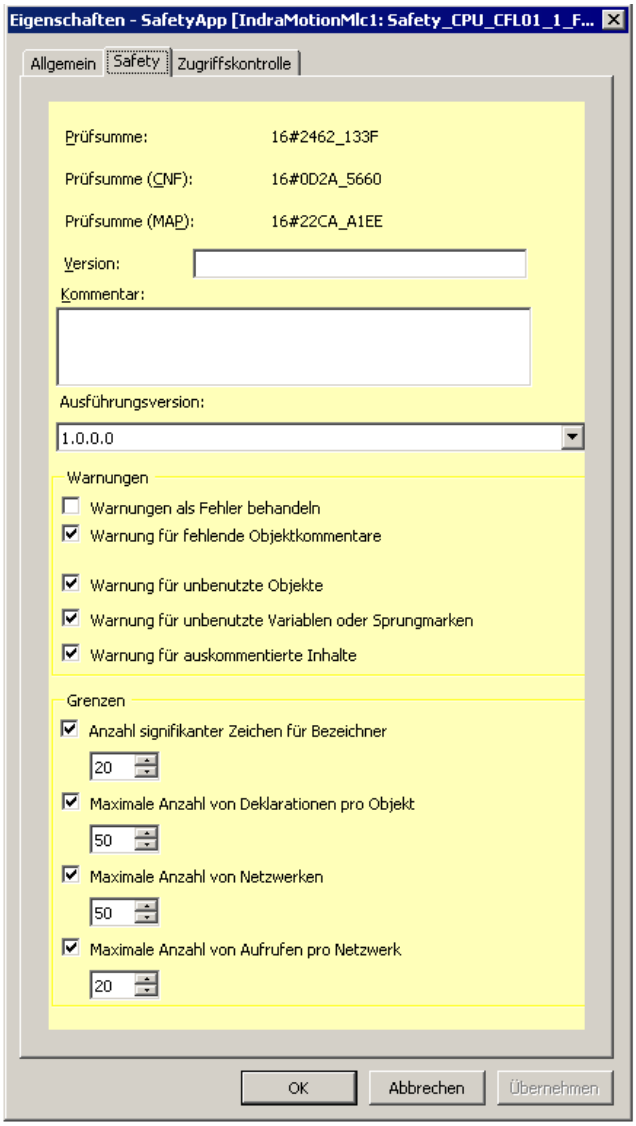


Abb. 6-7: Eigenschaften des Sicherheitsapplikationsobjekts, Registerkarte 'Safety'

Pin-Prüfsumme:

Die Pin-Prüfsumme ist eine Prüfsumme über alle Objekte der Sicherheitsapplikation einschließlich der verwendeten Bibliotheksbausteine.

Prüfsumme eines Objekts:

Im Gegensatz zur Pin-Prüfsumme identifiziert die Prüfsumme der einzelnen Objekte der Sicherheitsapplikation zusammen mit der Objektversion den Objekt-Inhalt. Die Prüfsumme ist von Bedeutung, wenn einzelne Objekte einer Sicherheitsapplikation in einer anderen Applikation wiederverwendet werden.

Die Registerkarte Safety enthält:

- Informationen

  - **Prüfsumme:** Information über die Prüfsumme des Sicherheitsapplikationsobjekts
  - **Prüfsumme (CNF):** (falls Aspekt **Sichere Applikationsparameter** vorhanden, siehe Editor des Safety-Applikationsobjekts mit Vergleichsansicht auf Seite 45) Information über die Prüfsumme der Konfiguration
  - **Prüfsumme (MAP):** (falls Aspekt **Sichere Applikationsparameter** vorhanden, siehe Editor des Safety-Applikationsobjekts mit Vergleichsansicht auf Seite 45) Information über die Prüfsumme des Mappings
  - **Version** (editierbar)

    Die Version kann vom Entwickler frei vergeben werden und dient dazu, dass die Version des Objekts in der Vergleichsansicht leicht erkannt werden kann.
  - **Kommentar** (editierbar)
  - **Ausführungsversion** (ggf. auswählbar)

    Mit Hilfe der Ausführungsversion wird die Kompatibilität zwischen (abgenommener) Bootapplikation und LZS überwacht.
- Einstellbare Warnungen und Begrenzungen, die durch den Safety-Checker geprüft werden (automatisch geprüfte Programmierrichtlinien siehe Automatische Prüfung der Programmierrichtlinien auf Seite 169)



Die Ausführungsversion ist Teil der Abnahme der Sicherheitsapplikation

---

Ausführungsversion: In der Regel sollte der Entwickler immer die neuste Ausführungsversion auswählen.

Weitere Informationen zur Ausführungsversion siehe Aktualisieren der Firmware, die Ausführungsversion auf Seite 188



Beim Übersetzen des Sicherheitsapplikation durch den Safety-Checker erzeugte Warnungen und Fehler werden im Meldungsfenster ausgegeben.

---

*Warnungen und Begrenzungen der Registerkarte **Safety** mit ihren Standard-Werten:*

- **Warnungen als Fehler behandeln**: Nicht aktiv

  Werden Warnungen als Fehler behandelt, kann die Applikation nicht geladen werden.
- **Warnung für fehlende Objektkommentare für Applikation und POUs**: Aktiv

  Berücksichtigt Kommentar und Version
- **Warnung für unbenutzte Objekte**: Aktiv
- **Warnung für unbenutzte Variablen oder Sprungmarken**: Aktiv
- **Warnung für auskommentierte Inhalte**: Aktiv
- **Anzahl signifikanter Zeichen für Bezeichner**: 20

  Es wird eine Warnung ausgegeben, wenn die ersten 20 Zeichen zweier Bezeichner gleich sind.
- **Maximale Anzahl von Deklarationen pro Objekt**: 50
- **Maximale Anzahl von Netzwerken**: 50

- **Maximale Anzahl von Aufrufen pro Netzwerk**: 20

Genauere Informationen zu den Warnungen und Grenzen siehe Optional automatisch geprüfte Programmierrichtlinien auf Seite 82

Editor des Safety-Applikationsobjekts mit Vergleichsansicht

Der Editor wird durch Selektion des Objekts **SafetyApp** und Aktivierung des Kontextmenübefehls **Öffnen** geöffnet.

Im Projektvergleich kann durch Doppelklick auf das Sicherheitsapplikationsobjekt der Vergleichseditor geöffnet werden, der die Unterschiede in den Pin-Kennungen und den Objektlisten der Pins beider Applikationen anzeigt.

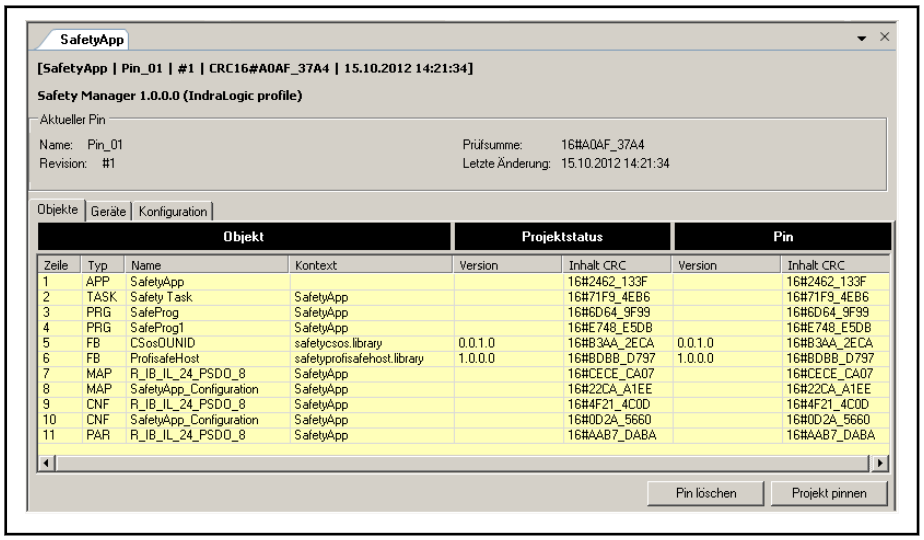


Abb. 6-8: Beispiel, Editor des Safety-Applikationsobjekts mit Vergleichsansicht (Objektliste)

Registerkarten des Editors des Sicherheitsapplikationsobjekt

- Objekte
- Geräte
- Konfiguration
- optional: Sichere Applikationsparameter

  Die optionale Registerkarte **Applikationsparameter** ist geräteabhängig.

*Der Applikationseditor mit den Registerkarten **Objekte** und **Geräte** enthält folgende Informationen:*

- Pin-Informationen oder falls Applikation verändert wurde: **In Work**
- Aktuelle IndraWorks Safety Manager-Version
- **Aktueller Pin** der Sicherheitsapplikation bestehend aus:
  - **Name**
  - **Prüfsumme** (Pin Prüfsumme), die Prüfsumme wird über die gesamte gepinnte Applikation erstellt.
  - **Revision**
  - **Letzte Änderung**

Genauere Informationen zu den Pin-Informationen siehe Pinnen auf Seite 161

Objekte

☞ Die Registerkarte **Objekte** des Editors des Safety-Applikationsobjekts ist die **Vergleichsansicht**!

Sie zeigt an, welche Objekte seit dem letzten Pinnen verändert wurden.

Vergleichsansicht (Registerkarte 'Objekte') zeigt folgende Informationen an:

- Auflistung der zur Sicherheitsapplikation gehörenden Objekte mit
  - **Zeile:** Laufende Nummer der Einträge
  - **Typ:** Objekttyp (siehe Tab. 6-1 "Angezeigte Werte für Typ und Name in der Objektliste (abhängig vom Objekttyp)" auf Seite 46)
  - **Name:** Name des Objekts
  - **Version** und **Prüfsumme**: Version und Prüfsumme des Objekts im aktuellen Projekt (Projektstatus)
  - **Version** und **Prüfsumme**: Version und Prüfsumme des Objekts des zuletzt gepinnten Projekts

☞ Die Objektliste enthält auch Einträge für alle Safety-Objekte aus den Bibliotheken, welche die Applikation einbindet.

Dabei werden nur jene aufgenommen, welche aus den Objekten der Applikation referenziert werden. Dies schließt jedoch auch Objekte aus den Bibliotheken mit ein, welche nicht aus Anwender-Code sondern aus System-Code aufgerufen werden, insbesondere aus den E/A-Treibern.

Objekttyp	Typ	Name
Applikation	APP	Name des Applikationsobjekts
Task	TASK	Name des Taskobjekts
Programm POU	PRG	Name des Programmbausteins
Funktionsbaustein POU	FB	Name des POU-Objekts des Funktionsbausteins
Globale Variablenliste	GVL	Name des GVL-Objekts
E/A-Mapping	MAP	Name des logischen Geräteobjekts/Applikationsparameter
Sichere Konfiguration	CNF	Name des logischen Geräteobjekts/Applikationsparameter
Sichere Parametrierung	PAR	Name des logischen Geräteobjekts

Tab. 6-1: Angezeigte Werte für Typ und Name in der Objektliste (abhängig vom Objekttyp)

Zusätzlich enthält die Vergleichsansicht die Schaltflächen **Pin löschen** und **Projekt pinnen** (siehe Automatische Prüfung der Programmierrichtlinien auf Seite 169).

Geräte

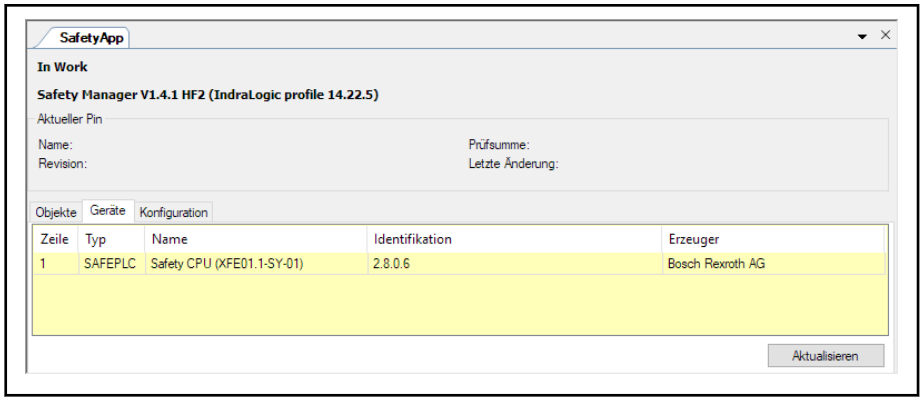


Abb. 6-9: Editor des Safety-Applikationsobjekts mit Geräteliste

Die Registerkarte 'Geräte' zeigt folgende Informationen an:

- Auflistung der zur Sicherheitsapplikation gehörenden Feldgeräte und der Sicherheitssteuerung, zu der die Sicherheitsapplikation gehört, jeweils mit den dazugehörigen Informationen der Beschreibungsdateien.
  - **Zeile:** Laufende Nummer der Einträge
  - **TYP:** Typ des Gerätes (siehe Tab. 6-2 "Angezeigte Werte für Typ in der Geräteliste" auf Seite 47)
  - **Name:** Name des Gerätes im Projektbaum
  - **Identifikation:** Gerätespezifische Informationen zur Identifikation
  - **Erzeuger:** Informationen über den Erzeuger der gerätespezifischen Informationen

Typ	Beschreibung
SAFEPLC	Sicherheitssteuerung, der die Applikation zugeordnet ist
SAFEDEV	Sicheres logisches Gerät der Applikation
STDDEV	Standard logisches Gerät der Applikation
XVARDEV	Standard logisches Gerät zum Variablenaustausch

Tab. 6-2: Angezeigte Werte für Typ in der Geräteliste

Die Schaltfläche **Liste aktualisieren** dient dem Aktualisieren der Geräte-Liste, wenn sich bei geöffnetem Editor die Geräte (Logische E/As) oder die Sicherheitssteuerung im Projektbaum geändert haben.

Konfiguration

Das Registerblatt Konfiguration zeigt an:

- **Safety Device ID (SDID):** Geräte-ID des Zielgeräts. Die SDID muss für jedes Gerät eindeutig vergeben werden. Sie darf keine Beziehung zur sercos Adresse des Gerätes haben.
- **Safety Network Number (SNN):** Eindeutige ID des Safety-Netzwerks. Die SNN hat eine Länge von 6 Bytes. Alle CSos-Geräte müssen dieselbe SNN besitzen.

6.7 Logische E/As

6.7.1 Logische E/As - Überblick

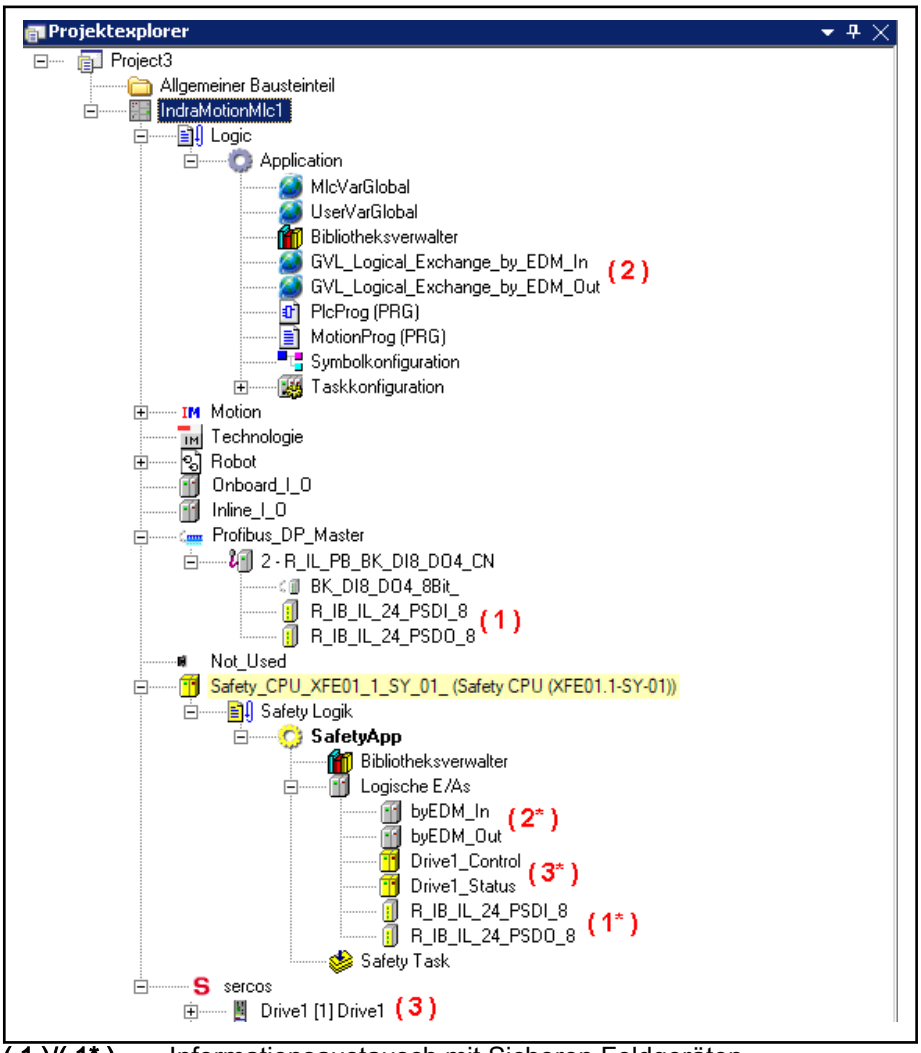
Die logischen E/As dienen dem Austausch von Daten zwischen der Standard- und der Sicherheitssteuerung. Zwei Arten von Daten können ausgetauscht werden: Globale Variablen und E/A-Daten. Die Sicherheitssteuerung selbst verfügt über keine E/A-Daten. Diese müssen in der Standardsteuerung konfiguriert und dann als logische E/As mit der Sicherheitssteuerung ausge-

tauscht werden. Wird in IndraWorks in der Steuerungskonfiguration ein E/A Modul mit sicheren E/As eingefügt, so wird automatisch ein passendes logisches E/A in der Sicherheitssteuerung eingefügt (gleicher Name wie das E/A Modul auf Standardseite). Dieses logische E/A enthält alle sicheren E/A-Kanäle sowie die Sicherheitsparameter des Moduls, so dass sämtliche sicherheitsrelevanten Informationen unter der Sicherheitsapplikation zu finden sind.

Prinzip der "Logischen E/As"

Arten der "Logischen E/As"

- Logische E/As von sicheren Feldgeräten (siehe Logische E/As von Sicheren Feldgeräten auf Seite 51)
- Logische E/As von Standard-Feldgeräten (siehe Logische E/As von Nicht-sicheren Feldgeräten auf Seite 51)
- Logische E/As von globalen Variablen zum Austausch mit der Standardsteuerung des Projekts (**logische Austauschgeräte**) (siehe Logische E/As für den Datenaustausch Standard-/Sicherheitssteuerung auf Seite 52)



(1)/(1*) Informationsaustausch mit Sicheren Feldgeräten
(2)/(2*) Informationsaustausch über Globale Variablenlisten
(3)/(3*) Informationsaustausch mit Sicheren IndraDrive-Antrieben

Abb. 6-10: Datenaustausch zwischen Standard- und Sicherheitssteuerung

Die logischen E/As der Sicherheitsapplikation sind mit physikalischen Geräten bzw. **GVLs für logischen Austausch** (spezielles Objekt auf Standardseite)

der Standardapplikation verlinkt. Dies bedeutet, dass es zu jedem physikalischen Gerät, dessen Ein-/Ausgangssignale in der Sicherheitsapplikation verarbeitet werden, genau ein "Logisches E/A" in der Sicherheitsapplikation gibt.

Ebenso existiert für jede **GVL für logischen Austausch** der Standardsteuerung genau ein "Logisches E/A" unter der Sicherheitsapplikation. Die Zuordnung ist variabel. Die maximale Größe des logischen E/A-Bereichs ist auf 4 KB festegelegt. Wird sie überschritten, so ist die Applikation nicht lauffähig. Es erfolgt ein Logbucheintrag.



Die sicherheitsgerichteten Parameter eines Gerätes, dessen Ein-/Ausgänge in der Sicherheitsapplikation verarbeitet werden, können nur in dem entsprechenden "Logischen E/A" in der Sicherheitsapplikation editiert werden.



Verwaltet werden die Beschreibungen der logischen E/As in der Kategorie **Logische Geräte** des Geräte-Repositorys.



Zur besseren Übersicht können im Projektbaum unter **Logische E/As** Ordner hinzugefügt werden, um die "Logischen E/As" zu gruppieren.

Vorteile der "Logischen E/As"

Durch das Konzept der logischen E/As ergeben sich folgende Vorteile für die Entwicklung und die Verifikation einer IndraWorks Safety Manager-Sicherheitsapplikation:

- Die Parametrierung der sicheren Parameter (z. B. F-Parameter bei Profisafe) von Feldgeräten erfolgt nur in den logischen E/As der Sicherheitsapplikation. Bei eingerichteter Safety-Benutzerverwaltung kann diese Parametrierung nur von Mitgliedern der Benutzergruppe Safety vorgenommen werden.
- Geänderte Zuweisungen von physikalischen Feldgeräten und GVLs für logischen Austausch verändern die Sicherheitsapplikation nicht, da die Änderungen von Zuweisungen unter der Standardapplikation erfolgen und durch einen Download auf die Standardsteuerung wirksam werden.
- Eine bereits verifizierte und abgenommene IndraWorks Safety Manager-Sicherheitsapplikation kann losgelöst vom ursprünglichen Projekt komplett in eine anderes Projekt integriert werden, ohne dass diese neue Sicherheitsapplikation erneut verifiziert werden muss. Den logischen E/As der Sicherheitsapplikation müssen dabei passende Feldgeräte und GVLs für logischen Austausch neu zugewiesen werden.

Hinweise zu den "Logischen E/As"

Die Zuweisung auf die logischen-E/As erfolgt abhängig vom Namen der Applikation und des logischen E/A-Objekts.

Daraus ergeben sich folgende Hinweise:

- Wird der Name eines logischen E/As geändert, so wird der Name des physikalischen Geräts, bzw. der GVL für logischen Austausch automatisch nachgeführt und muss nicht neu auf das "Logische E/A" zugeordnet werden. (siehe Verbindung eines Feldgeräts mit dem Logischen E/A auf Seite 52 und Verbindung der GVL für Logischen Austausch auf das Logische Austauschgerät auf Seite 53)
- Das Umbenennen (außer bei "Default-Zuweisung") oder Verschieben von Feldgeräten im Gerätebaum, bzw. von GVLs für Logischen Austausch in der Hauptapplikation ändert nichts an der Zuweisung auf das logische E/A und erfordert auch keine Änderung der Sicherheitsapplika-

tion. Eine Änderung der Sicherheitsapplikation erfolgt beim Umbenennen automatisch: das zugeordnete logische E/A wird analog zum Feldgerät umbenannt.

Wird das Feldgerät in eine parallele andere Steuerung verschoben, so ist die Verknüpfung gelöst und das Feldgerät muss neu auf das logische E/A der Sicherheitsapplikation zugeordnet werden.

- Das Löschen des Feldgeräts im Gerätebaum bzw. der GVL für logischen Austausch in der Standardapplikation bedeutet, dass das zugeordnete logische E/A auf nichts mehr gemappt wird (nicht mehr versorgt wird).
- Das Löschen des logischen E/A-Objekts bedeutet, dass keine sicherheitsgerichteten Einstellungen mehr auf das Feldgerät gemappt werden bzw., dass die von "Safety"-Variablen in der GVL für Logischen Austausch nicht mehr versorgt werden.
- Wenn ein anderes logisches E/A-Objekt den alten Namen eines umbenannten oder gelöschten logischen E/A-Objekts erhält, dann werden ab jetzt dessen Einstellungen auf das zugeordneten Feldgerät gemappt bzw. dessen Variablenwerte mit der Austausch-GVL ausgetauscht. Die Zuweisung wird somit indirekt geändert.

  Wurde ein logisches E/A gelöscht und es wird ein neues logisches E/A mit dem Namen des gelöschten logischen E/As eingefügt, dann ist die Zuweisung wieder aktiv.
- Wird ein physikalisches E/A-Objekt kopiert, das mit einem logischen E/A-Objekt verlinkt ist, so wird das logische E/A-Objekt mit seinen Daten ebenfalls kopiert.

Ersatzwerte

Wenn nicht anders festgelegt, wird für Werte, die nicht aktualisiert werden können eine "0" geliefert. Weitere Informationen siehe Abstimmung mit der Standardsteuerung auf Seite 151.

Objekteigenschaften

Der Eigenschaften-Dialog von allen logischen E/As enthält zusätzlich zu den Registerkarten **Allgemein** und **Zugriffskontrolle** die Registerkarte **Safety** (siehe Abb. 6-11 "Eigenschaften von logischen E/As, Registerkarte 'Safety'" auf Seite 50). Informationen der Registerkarte **Safety** siehe Sicherheitsapplikationsobjekt SafetyApp auf Seite 41.

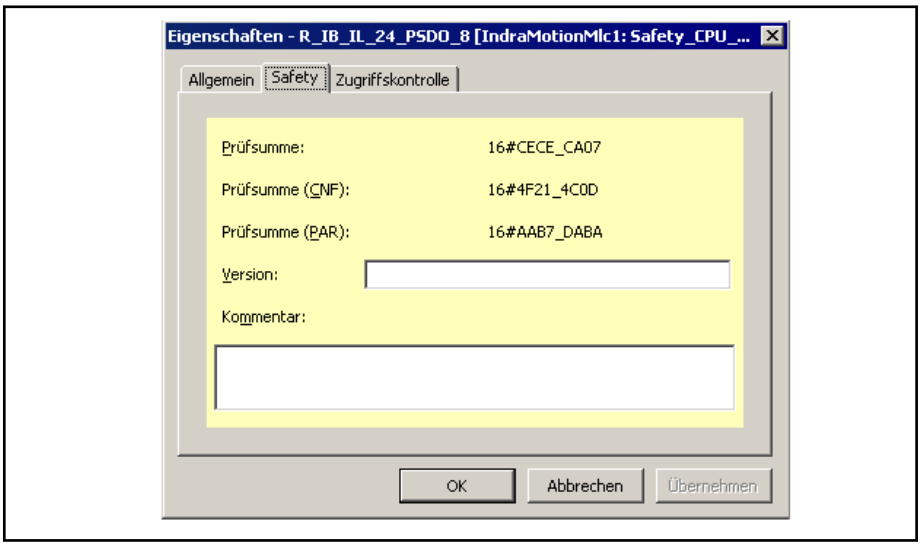


Abb. 6-11: Eigenschaften von logischen E/As, Registerkarte 'Safety'

6.7.2 Verwendungstypen der "Logischen E/As"

Logische E/As von "Sicheren Feldgeräten"

Diese "Logischen E/As" dienen dem Austausch von sicherheitsgerichteten E/As zwischen der Sicherheits- und der Standardsteuerung.

☞ Informationen, wie Sie sichere 1oo1-, bzw. 1oo2- Eingabegeräte mit der Sicherheitsapplikation verbinden, und worauf Sie dabei achten müssen, finden Sie in Kap. 7.6 "E/A-Zugriff" auf Seite 113

Für diesen Austausch muss zuerst das sichere Feldgerät unterhalb der Standardsteuerung eingefügt werden.

Hinzufügen eines sicheren Feldgeräts unter der Standardsteuerung

1. Im Projektbaum den entsprechenden Feldbus-Slave unter der Standardsteuerung selektieren.
2. Kontextmenübefehl **Hinzufügen ▸ Hinzufügen...** aktivieren.
3. Im sich öffnenden Dialog das gewünschte sichere Feldgerät auswählen und durch Doppelklick bzw. die Schaltfläche **Hinzufügen** einfügen. Sicherheitsmodule sind mit einem gelben Ikon gekennzeichnet.
4. Den Dialog über die Schaltfläche **Schließen** beenden.

☞ Beim Einfügen eines sicheren Feldgeräts unterhalb der Standardsteuerung wird das entsprechende "Logische E/As" automatisch unter den Knotenpunkt **Logische E/As** der Sicherheitsapplikation eingefügt.

Typkonsistenz der E/A-Kanäle

HINWEIS **Die Typkonsistenz der EA-Kanäle ist nur gewährleistet,**

- wenn der Applikationsstand auf der Sicherheitssteuerung und auf der Standardsteuerung dem Stand des gleichen übersetzbaren Projekts entsprechen

  und

- wenn die Feldgeräte im Projekt den Feldgeräten in der Maschine entsprechen.

  Je nach Bussystem wird ein Mismatch hier automatisch erkannt (z.B. bei Profibus).

Logische E/As von "Nicht-sicheren Feldgeräten"

Diese "Logischen E/As" dienen dem Informationsaustausch von Standard-E/As zwischen der Standard- und der Sicherheitssteuerung.

Werden Standard-E/As in der Sicherheitsapplikation verwendet, so sind diese Daten **nicht** sicher!

Zuerst wird das gewünschte Standard-Feldbusgerät geräteabhängig unter der Standardsteuerung eingefügt. Die Vorgehensweise entspricht IndraWorks-Standard.

☞ In der aktuellen Version von IndraWorks können nur PROFINET- und Profibus-Geräte als Standard-Geräte in die Sicherheitsapplikation eingebunden werden.

Anschließend muss das entsprechende "Logische E/A" (Generiertes logisches Gerät) unterhalb des Sicherheitsapplikationsobjekts eingefügt werden:

Hinzufügen des "Generierten logischen Gerätes"

1. Im Projektbaum das Standard-Feldbusgerät selektieren
2. Kontextmenübefehl **Mit Safety CPU verknüpfen** aktivieren

   Das Logische Austauschgerät wird in der Sicherheitssteuerung unterhalb des Knotens **Logische E/As** hinzugefügt.
   Die Zuordnung zwischen Standard-Feldbusgerät und "Logischem E/A" wird automatisch eingetragen.

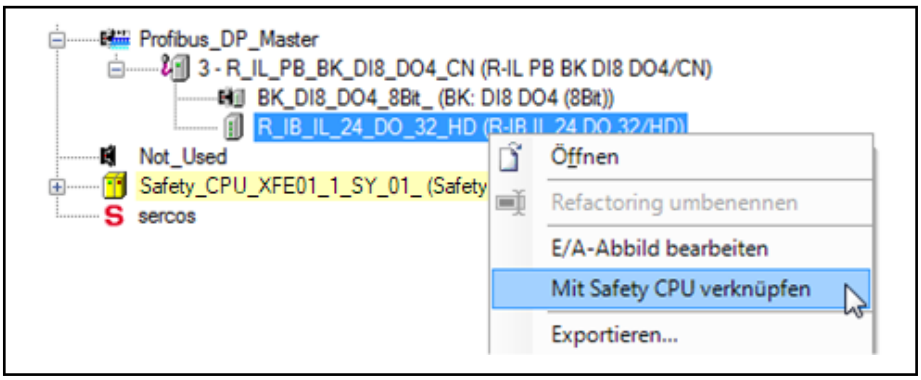


Abb. 6-12: Dialog 'Logisches Gerät hinzufügen'

Editieren der Variablen siehe E/A-Abbild auf Seite 55

Verbindung eines Feldgeräts mit dem "Logischen E/A"

Die Verbindung zwischen dem Standard-Feldbusgerät unter der Standardsteuerung und dem entsprechenden "Logischen E/A" unterhalb der Sicherheitssteuerung erfolgt automatisch beim Anlegen des "Logischen E/As".

HINWEIS **Die Typkonsistenz der EA-Kanäle ist nur gewährleistet,**

- wenn der Applikationsstand auf der Sicherheitssteuerung und auf der Standardsteuerung dem Stand des gleichen übersetzbaren Projekts entsprechen

  und

- wenn die Feldgeräte im Projekt den Feldgeräten in der Maschine entsprechen.

  Je nach Bussystem wird ein Mismatch hier automatisch erkannt (z.B. bei Profibus).

Logische E/As für den Datenaustausch Standard-/Sicherheitssteuerung

Der Datenaustausch zwischen Sicherheitssteuerung und Standardsteuerung erfolgt über Variablen, die in den "Logischen E/As" (Logischen Austauschgeräten) definiert werden.

Auf der Standardseite wird dazu eine **GVL_für_logischen_Austausch** angelegt, die mit dem entsprechenden **Logischen Austauschgerät** verbunden wird.

Diese Daten werden in den POUs der Sicherheitssteuerung wie Ein-/Ausgänge (VAR_EXTERNAL) verwendet, in der Standardapplikation sind sie als globale Variablen verfügbar.

☞ Der Datenfluss zwischen zwei Variablen ist eindeutig. Dies bedeutet, dass die gleiche Variable der einen Applikation nicht mit der Variablen der anderen Applikation ausgetauscht werden kann.

Hinzufügen der "GVL für logischen Austausch"

1. Im Projektbaum das Standardapplikationsobjekt selektieren
2. Kontextmenübefehl **Hinzufügen** mit Auswahl **GVL für logischen Austausch...** aktivieren
3. Im Dialog **GVL für Logischen Austausch** hinzufügen kann ein Name für die GVL editiert werden. Der Default-Name lautet **GVL_für_Logischen Austausch**.
4. Schaltfläche **Fertigstellen** aktivieren

Hinzufügen des "Logischen Austauschgeräts"

1. Im Projektbaum den Knotenpunkt **Logische E/As** des Safety-Applikationsobjekts selektieren
2. Im Kontextmenü **Hinzufügen** ▸ **Logische Austauschgeräte** das gewünschte logische E/A auswählen

Editieren der Austauschvariablen siehe E/A-Abbild auf Seite 55.

*Bei der Auswahl des **Logischen Austauschgeräts** werden für die einzelnen Variablen festgelegt:*

- **die Austauschrichtung:** IN oder OUT

  IN: von der Standardapplikation in die Sicherheitsapplikation

  OUT: von der Sicherheitsapplikation zur Standardapplikation
- **der Datentyp:** BYTE, DINT, INT oder WORD

☞ Die Austauschvariablen können nur im logischen E/A editiert werden.

In der **GVL für Logischen Austausch** können Variablen nicht eingegeben oder verändert werden.

☞ Es können nur Daten ausgetauscht werden, deren Variablen einen Standard-Datentyp besitzen.

Variablen des Datentyps mit Präfix SAFE können nicht zwischen einer Sicherheitssteuerung und einer Standardsteuerung ausgetauscht werden.

Verbindung der "GVL für Logischen Austausch" auf das "Logische Austauschgerät"

1. GVL für Logischen Austausch der Standardapplikation im Projektbaum selektieren
2. Kontextmenübefehl **Öffnen** aktivieren
3. Combobox **Abbild für Logischen Austausch** anklicken
4. Aus der sich öffnenden Liste das entsprechende logische Austauschobjekt auswählen

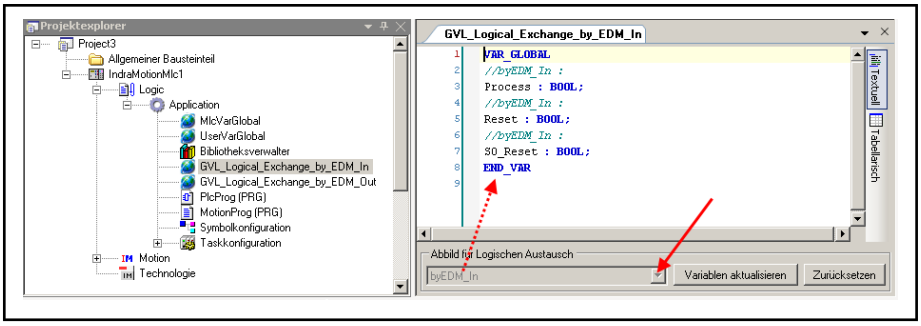


Abb. 6-13: Editor der GVL für Logischen Austausch mit geöffneter Combobox

In der Combobox stehen alle noch nicht gemappten logischen Austauschobjekte zur Verfügung.

Wird ein logisches Austauschgerät ausgewählt, so werden die Variablen implizit aktualisiert.

Werden im verbundenen logischen Austauschgerät Änderungen vorgenommen, so muss in der GVL für Logischen Austausch die Schaltfläche **Aktualisieren** aktiviert werden, um die Variablenliste zu aktualisieren.

Durch Aktivierung der Schaltfläche **Zurücksetzen** wird eine bestehende Verbindung aufgelöst.

Automatische Konvertierung zwischen BOOL und Binärdatentypen

Logische Austauschgeräte, die Binärdatentypen wie z.B. BYTE und WORD verwenden, können in der Safety-Steuerung als BOOL aufgelegte Bits automatisch in ein BYTE- oder WORD-Datentyp konvertieren. Hierbei werden alle BOOL-Variablen entsprechend ihrer Bitposition abgebildet. Diese Funktion lässt sich über die Schaltfläche "Kombinierten Typ verwenden" in der GVL für logische Austauschobjekte unterhalb der Standardsteuerung aktivieren. Statt der einzelnen BOOL-Variablen wird dann der kombinierte Datentyp angelegt.

Hinweise zum Datenaustausch zwischen Standard- und Sicherheitssteuerung

Eine Änderung des Variablenaustauschs kann nur durch einen erneuten Download der Sicherheitsapplikation und der Standardapplikation aktiv werden.

Ersatzwerte

Solange die Applikation nicht beendet ist, aber gerade keine aktuellen Werte ausgetauscht werden können, werden für den Variablenaustausch Ersatzwerte verwendet, siehe Ersatzwerte auf Seite 152.

HINWEIS **Die Typkonsistenz der EA-Kanäle ist nur gewährleistet,**

- wenn der Applikationsstand auf der Sicherheitssteuerung und auf der Standardsteuerung dem Stand des gleichen übersetzbaren Projekts entsprechen

  und

- wenn die Feldgeräte im Projekt den Feldgeräten in der Maschine entsprechen.

  Je nach Bussystem wird ein Mismatch hier automatisch erkannt (z.B. bei Profibus).

6.7.3 Editor der "Logischen E/As"

Information und E/A-Abbild

Information

Der Editor aller "Logischen E/As" besitzt die Registerkarte **Information** (siehe Abb. 6-14 "Registerkarte 'Informationeines logischen E/A" auf Seite 55),

welche genaue Informationen und gegebenenfalls ein Bild des jeweiligen "Logischen E/As" enthält.

Als Informationen werden aufgelistet: Name, Hersteller, Kategorien, Typ, ID, Version, Bestellnummer und Beschreibung.

Mögliche Kategorien

- Sichere Module
- Generierte logische Geräte
- Logische Austauschgeräte

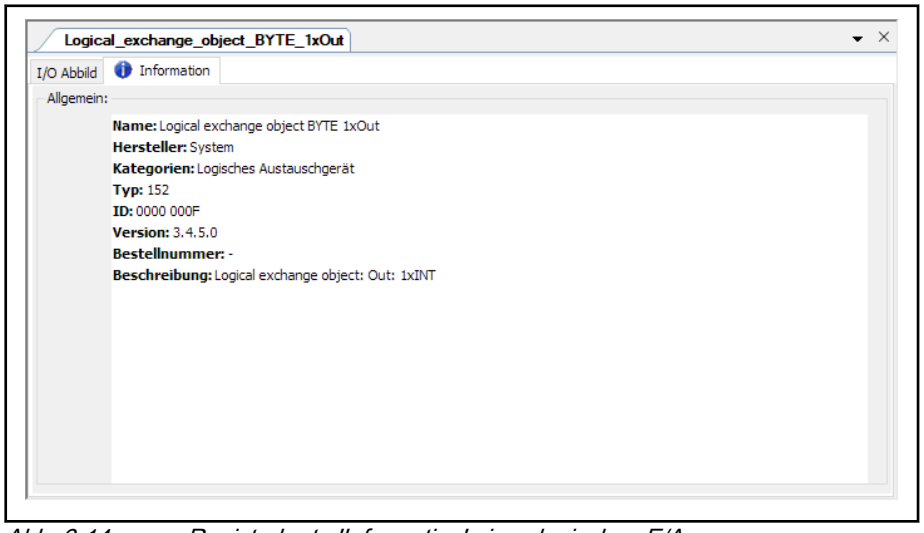


Abb. 6-14: Registerkarte 'Information' eines logischen E/A

E/A-Abbild

Im **E/A-Abbild** werden die Variablen, die dem Zugriff der Sicherheitsapplikation auf die E/As dienen, definiert.

Im IndraWorks Safety Manager können E/A-Kanäle nur auf neue Variablen, d. h. nicht auf bereits existierende Variablen, gemappt werden.

Auf Eingangskanäle gemappte Variablen enthalten die entsprechenden Eingangssignale von Feldgeräten und können somit gelesen werden. Variablen, die auf Ausgangskanäle gemappt sind, können geschrieben werden und setzen Ausgangssignale in Feldgeräten.

Für jeden Eingangs- bzw. Ausgangskanal eines E/A-Moduls, welchem eine Variable zugeordnet wurde, wird in der Sicherheitsapplikation eine implizite globale Variable mit dem entsprechenden Namen und dem in der Spalte **Typ** angegebenen Datentyp angelegt.

Bereiche der Registerkarte E/A Abbild

- **Liste der Variablen des E/A Abbilds mit:** Variable (Name), Kanal (Input bzw. Output), Typ, Einheit und Beschreibung

  Die Informationen der Spalten **Kanal**, **Typ** (IEC Datentyp) , **Einheit** und **Beschreibung** sind in der Gerätebeschreibungsdatei festgelegt und können nicht verändert werden.
- **Physikalisches E/A:** Anzeige des mit diesem "Logischen E/A" verbundenen Objekts der Standardapplikation
- **Instanzen:** Liste der impliziten Instanzen. Diese stehen der Sicherheitsapplikation als globale Variablen zur Verfügung, siehe Verwendung von Logischen E/As im Projekt auf Seite 59.

Editieren der Variablen

Die Mapping-Variablen werden in der Registerkarte **E/A Abbild** des "Logischen E/As" editiert und angezeigt. Um eine Variable zu editieren, muss die entsprechende Zelle mit einem Doppelklick geöffnet werden.

Weiterhin ist es möglich eine oder mehrere Zeilen zu kopieren und wieder einzufügen. Dies funktioniert auch mit externen Anwendungen.

Durch Aktivierung der Schaltfläche **Abbild zurücksetzen** werden alle in der Tabelle eingetragenen Mapping-Variablen gelöscht, d. h. die Abbildung vom physikalischen Gerät auf das "Logische E/A" wird zurückgesetzt.

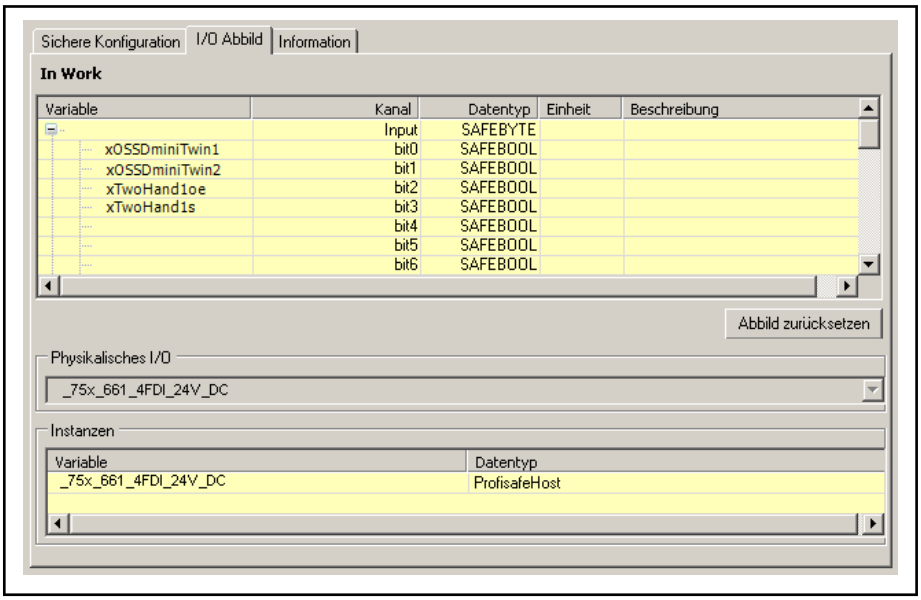


Abb. 6-15: Registerkarte 'E/A Abbild' des "Logischen E/As" eines sicheren Feldgeräts

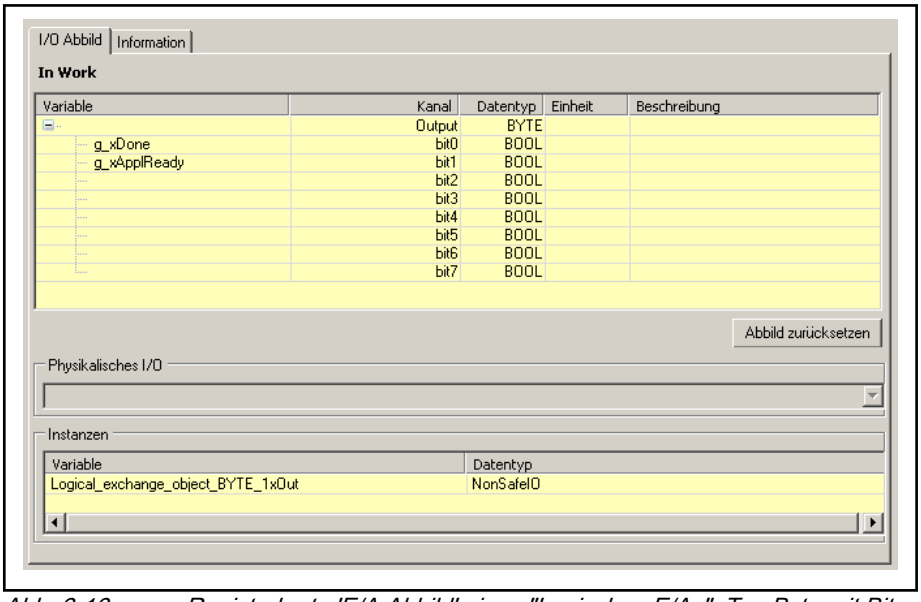


Abb. 6-16: Registerkarte 'E/A Abbild' eines "Logischen E/As", Typ Byte mit Bit-Zugriff

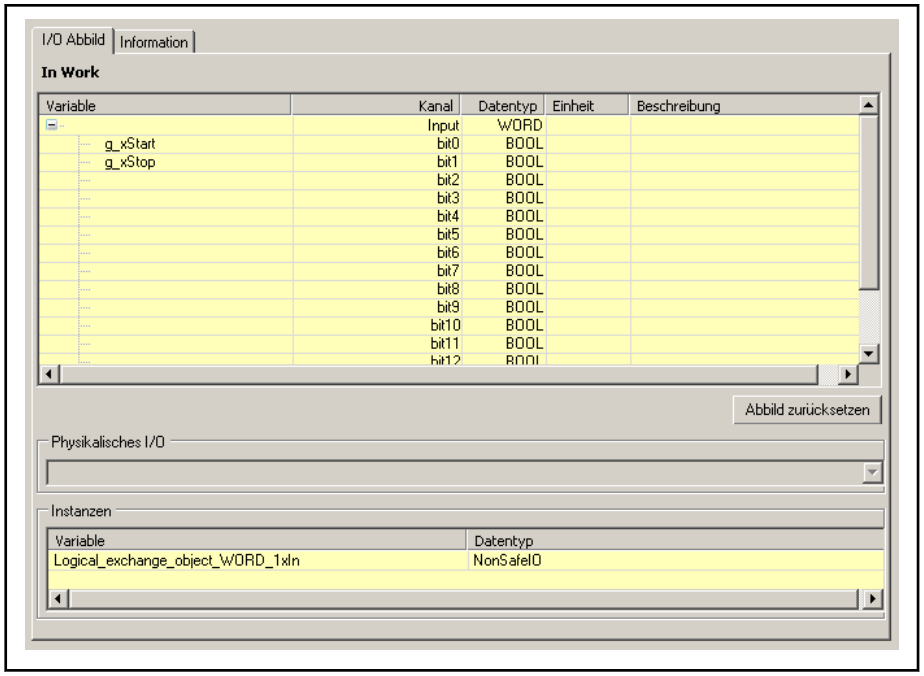


Abb. 6-17: Registerkarte 'E/A Abbild' eines "Logischen E/As", Typ WORD

Durchgeführte Änderungen in der Registerkarte **E/A Abbild** werden rot gekennzeichnet.

Es wird dabei immer nur die zuletzt durchgeführte Änderung markiert.

Mit Schließen des Editors werden sämtliche Änderungsmarkierungen entfernt.

Verwendung der im **E/A Abbild** definierten Variablen im Projekt siehe Verwendung von Logischen E/As im Projekt auf Seite 59.

Der Editor der "Logischen E/As" von Standardgeräten besitzt keine Registerkarte **Sichere Konfiguration** und **Sichere Geräteparametrierung**. Die Konfiguration und die Geräteparametrierung erfolgt unter der Standardapplikation wie in IndraWorks-Standard.

Wenn ein Gerät der Sicherheitssteuerung zugewiesen ist, so zeigt die Mapping-Registerkarte des Geräteeditors nur diese Zuweisung und nicht mehr das Kanalmapping auf die Variablen der Standardsteuerung.

Sichere Konfiguration und Sichere Parametrierung

Die Registerkarte **Sichere Parametrierung** und **Sichere Konfiguration** des Editors der "Logischen E/As" gibt es nur bei den sicherheitsgerichteten E/As.

Der Geräteeditor ist für die Darstellung und Bearbeitung der Geräteparameter bestimmter Geräte geeignet. Genaue Informationen finden Sie in der Dokumentation des jeweiligen Geräts, bzw. beim jeweiligen Gerätehersteller.

⚠ VORSICHT **Die Absicherung von sicheren Feldbusprotokollen ist von den eingestellten Parametern abhängig.**

Feldbusspezifische Anforderungen an die Einstellung bestimmter Parameter sind zu berücksichtigen.

☞ Der Anwender ist dafür verantwortlich, dass die Geräte entsprechend der Gerätedokumentation des jeweiligen Geräts, bzw. des jeweiligen Geräteherstellers korrekt parametriert werden.

☞ Der Gerätehersteller muss den Anwender über die Bedingungen für die Berechnung der System- Kennwerte informieren.

Im oberen Bereich steht die Pin-Information, bzw. **In Work**. Im anschließenden Bereich sind jeweils die Parameter in tabellarischer Form aufgelistet.

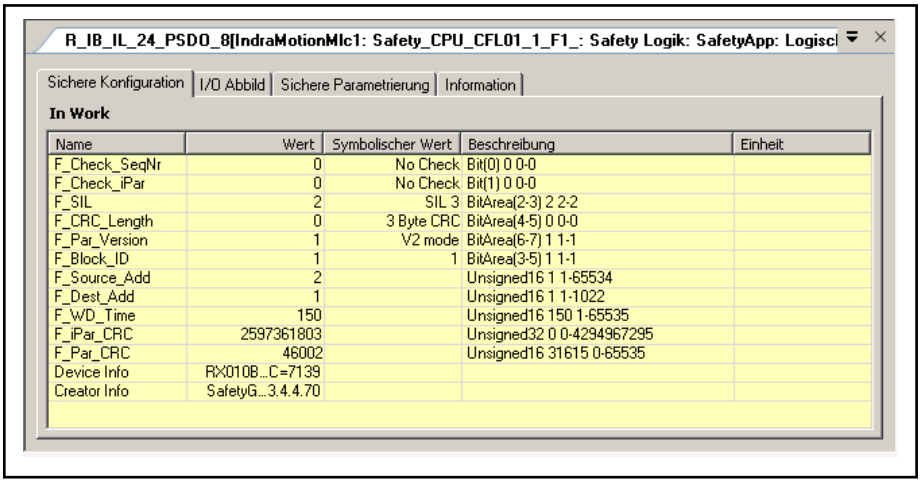
Sichere Konfiguration, enthält die Konfigurationsparameter für die sichere Kommunikation (siehe Abb. 6-18 "Beispiel: 'Sichere Konfiguration'" auf Seite 58).

Sichere Parametrierung, listet die Parameter des jeweiligen sicheren Gerätes auf (siehe Abb. 6-19 "Registerkarte 'Sichere Parametrierung" auf Seite 59).

Die Standard-Werte für die Parameter stammen aus der jeweiligen Gerätebeschreibungsdatei (SDD).

HINWEIS **Feldbusspezifische Anforderungen an die Einstellung bestimmter Parameter sind zu berücksichtigen.**

☞ Integer-Parameter können je nach Gerätebeschreibung auch hexadezimal angezeigt werden. Diese optionale Einstellung muss in der Gerätebeschreibungsdatei erfolgen.



Name	Wert	Symbolischer Wert	Beschreibung	Einheit
F_Check_SeqNr	0	No Check	Bit(0) 0 0-0	
F_Check_iPar	0	No Check	Bit(1) 0 0-0	
F_SIL	2	SIL 3	BitArea(2-3) 2 2-2	
F_CRC_Length	0	3 Byte CRC	BitArea(4-5) 0 0-0	
F_Par_Version	1	V2 mode	BitArea(6-7) 1 1-1	
F_Block_ID	1	1	BitArea(3-5) 1 1-1	
F_Source_Add	2		Unsigned16 1 1-65534	
F_Dest_Add	1		Unsigned16 1 1-1022	
F_WD_Time	150		Unsigned16 150 1-65535	
F_iPar_CRC	2597361803		Unsigned32 0 0-4294967295	
F_Par_CRC	46002		Unsigned16 31615 0-65535	
Device Info	RX010B...C=7139			
Creator Info	SafetyG...3.4.4.70			

Abb. 6-18: Beispiel: 'Sichere Konfiguration'

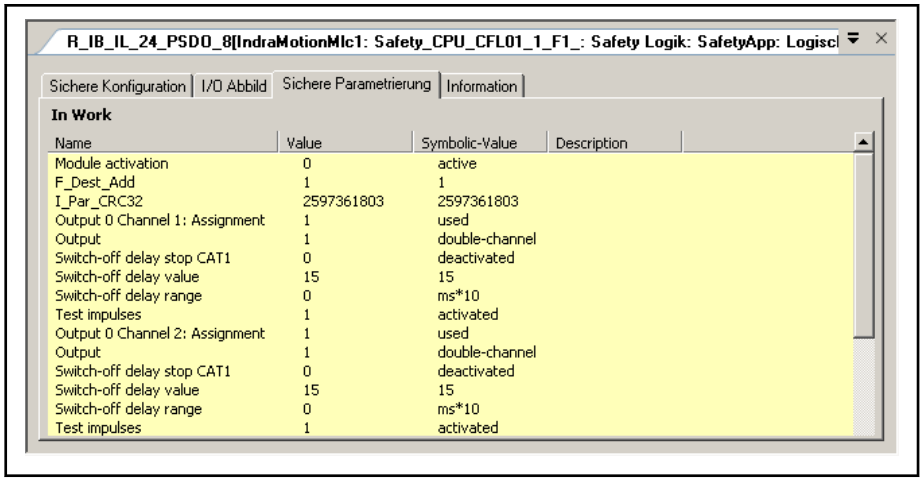


Abb. 6-19: Registerkarte 'Sichere Parametrierung'

Durchgeführte Änderungen in den Registerkarten **Sichere Konfiguration** und **Sichere Parametrierung** werden rot gekennzeichnet.

Es wird dabei immer nur die zuletzt durchgeführte Änderung markiert.

Mit Schließen des Editors werden sämtliche Änderungsmarkierungen entfernt.

Detaillierte Beschreibung der busspezifischen sicheren Parameter

- Profisafe-Geräte siehe Profisafe-F-Parameter auf Seite 586.
- CSos-Geräte siehe CSos-Parameter auf Seite 598.

6.7.4 Verwendung von "Logischen E/As" im Projekt

Jede im **E/A Abbild** eines "Logischen E/As" deklarierte Mapping-Variable (Kanalvariablen) und alle erzeugten Instanzen der "Logischen E/As" von sicheren Geräten und von Standardgeräten stehen dem Entwickler bei der Programmierung der Sicherheitsapplikation als globale Variablen zur Verfügung.

Um diese Variablen zur Code-Implementierung in eines Programms verwenden zu können, müssen sie im Deklarationsteil der POUs als VAR_EXTERNAL deklariert werden (Variablendeklaration siehe Grundlegendes zu Variablen auf Seite 100).

Alternativ zur expliziten Deklaration können diese Variablen und Instanzen im Implementierungsteil von Programmen entweder in der Eingabehilfe (siehe Abb. 6-20 "Eingabehilfe der Variablendeklaration mit Kategorie Globale Variablen und GVL IoConfigRemote_Globals" auf Seite 60) oder in der automatisch angezeigten Auswahlliste "Intellisense" (siehe Abb. 6-21 "Bespiel: Intellisense mit globalen Variablen von IoConfigRemote_Globals" auf Seite 60) ausgewählt werden.

In der Eingabehilfe der Variablendeklaration befinden sie sich in der Kategorie **Globale Variablen** in der GVL **IoConfigRemote_Globals**.

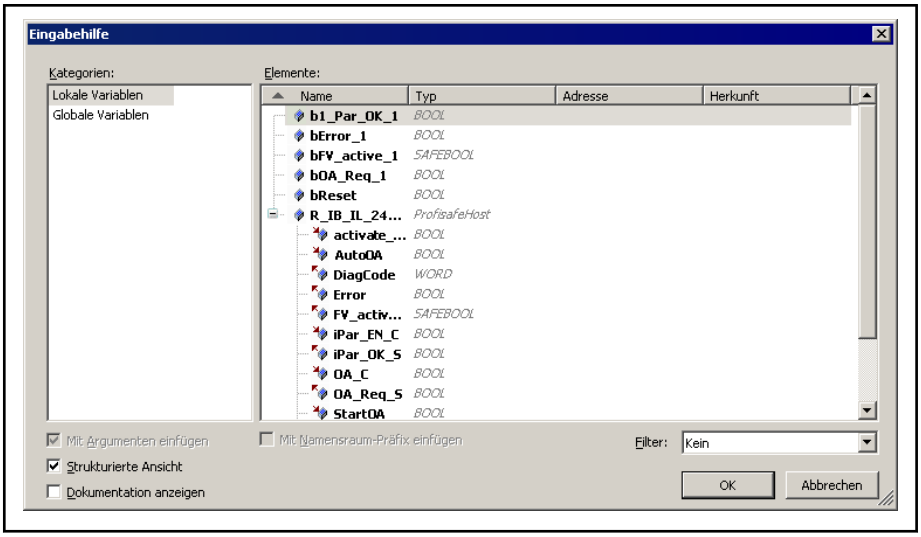


Abb. 6-20: Eingabehilfe der Variablendeklaration mit Kategorie "Globale Variablen" und GVL "IoConfigRemote_Globals"

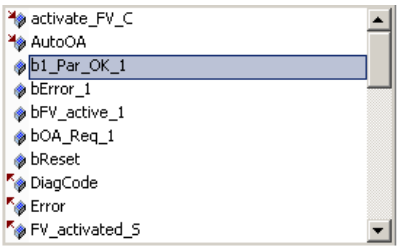


Abb. 6-21: Bespiel: Intellisense mit globalen Variablen von "IoConfigRemote_Globals"

☞ Variablen, die auf einen E/A-Kanal gemappt werden, können beim Debuggen des Programms geschrieben und geforct werden!

6.8 Programmorganisationseinheiten (POUs)

POUs (Programm Organisation Units) sind die Programmierobjekte des IndraWorks Safety Managers, die entweder als Programm (**PROGRAM**) oder als Funktionsbaustein (**FUNCTION_BLOCK**) deklariert sind.

Im Projektbaum der Sicherheitsapplikation können beliebig viele POUs hinzugefügt werden.

Merkmale von Programm und Funktionsbaustein:

- **Programm:**

  Ein Programm kann nicht von anderen Programmen aufgerufen werden, kann jedoch Instanzen von Funktionsbausteinen aufrufen.

  Programme werden direkt von der Safety Task aufgerufen. Welche Programme aufgerufen werden, wird im Objekt **Safety Task** festgelegt. Nur die aufgerufenen Programme werden auf der Steuerung ausgeführt.

- **Funktionsbaustein:**

  Funktionsbausteine werden immer über eine Instanz aufgerufen, die eine Kopie des Funktionsbausteins ist, welche die Daten enthält. Jede Instanz hat einen Bezeichner (Instanzname) und eine Datenstruktur, die ihre Eingabe-, Ausgabe- und internen Variablen beinhaltet.

In Funktionsbausteinen können Instanzen von anderen Funktionsbausteinen aufgerufen werden.

Die Deklaration und Verwendung von Funktionsbausteinen erfolgt wie in Standard-IndraWorks und wird hier nicht weiter erläutert.

Hinzufügen einer POU

1. Im Projektbaum **SafetyApp** selektieren
2. Kontextmenü-Befehl **Objekt hinzufügen** mit Auswahl **Safety Basic-POU** oder **Safety Extended-POU** aktivieren
3. Im Dialog **Basic-** bzw. **Extended-POU hinzufügen** den Namen und Kommentar der POU eingeben und Baustein-Typ **PROGRAM** oder **FUNCTION_BLOCK** auswählen (siehe Abb. 6-22 "Erstellen einer Extended-POU, Bausteintyp: FUNCTION_BLOCK" auf Seite 62).
4. Im Fall, dass der Bausteintyp **FUNCTION_BLOCK** ist, kann die Checkbox **Verwendung Einmaliger Aufruf** gesetzt werden. Wird diese Checkbox nicht gesetzt, kann die POU mehrfach aufgerufen werden.

   Bei dem Bausteintyp **PROGRAM** ist die Checkbox für den einmaligen Aufruf automatisch gesetzt und kann nicht geändert werden.
5. Schaltfläche **Öffnen** aktivieren

☞ **Kommentierung von POUs**

Zu jeder POU sollten im Feld "Kommentar" gemäß PLCopen folgende Informationen enthalten sein:

- Autor
- Erstelldatum der POU
- Freigabedatum
- Version
- Versionsgeschichte
- funktionale Beschreibung (einschließlich E/AParameter)

HINWEIS **Erzeugen, Bearbeiten, Entfernen von Extended-POUs**

Bei einem Safety-Projekt mit dem Safety-Benutzerkonfiguration, bei dem Extended-POUs nur von Mitgliedern der Benutzergruppe **Safety.ExtendedLevel** erzeugt werden können, muss der Entwickler im Eigenschaftendialog unter dem Tab **Zugriffskontrolle** der Benutzergruppe **Safety** das Recht zum **Bearbeiten** und **Entfernen** der Extended-POU verbieten, falls die Extended-POU auch nur von Mitgliedern der Benutzergruppe **Safety.ExtendedLevel** bearbeitet und entfernt werden soll.

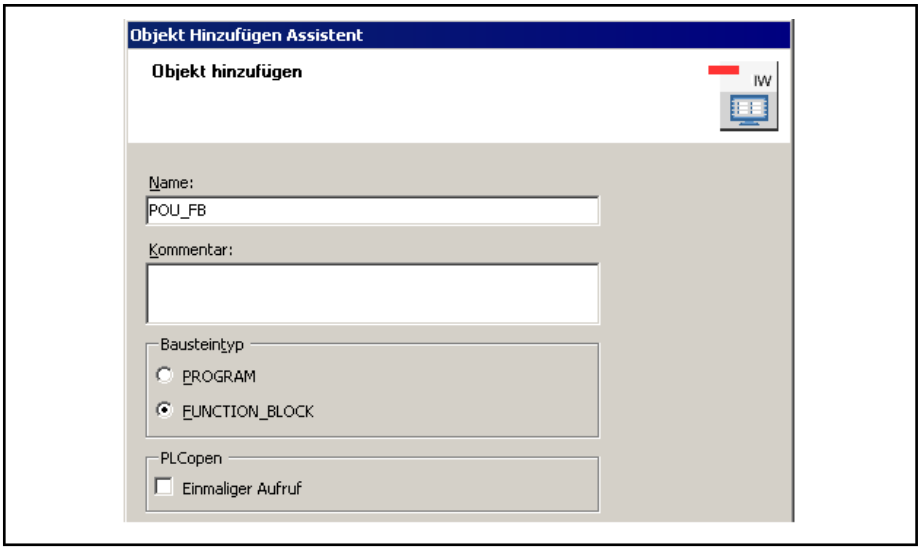


Abb. 6-22: Erstellen einer Extended-POU, Bausteintyp: "FUNCTION_BLOCK"

Editor einer POU und Erstellung des Programmcodes in POUs siehe POUs auf Seite 96.

Objekteigenschaften einer POU

Jede POU der Sicherheitsapplikation besitzt einen Eigenschaftendialog mit den Registerkarten **Allgemein**, **Safety** und **Zugriffskontrolle**.

Der Eigenschaftendialog wird durch Selektion der entsprechenden POU im Projektbaum und Aktivierung des Kontextmenü-Befehls **Eigenschaften...** geöffnet.

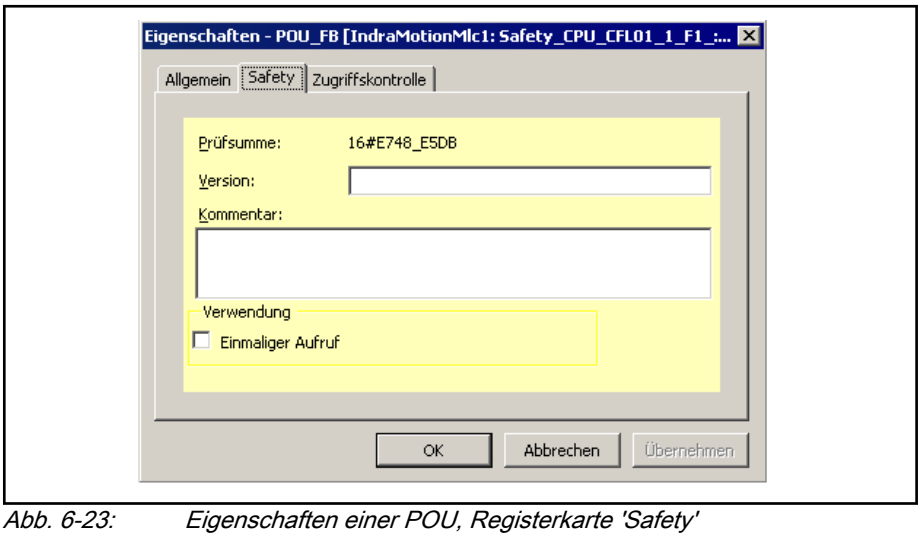


Abb. 6-23: Eigenschaften einer POU, Registerkarte 'Safety'

- **Prüfsumme:** Prüfsumme zu dieser POU
- **Version** (editierbar): Die Version kann der Entwickler frei vergeben. Mit Hilfe der Version kann in der Vergleichsansicht des Safety-Applikationsobjekts schnell erkannt werden, um welche Version des Objekts es sich handelt.
- **Kommentar** (editierbar)
- **Verwendung**

  *Die Checkbox **Einmaliger Aufruf** ist*

  – beim Bausteintyp **PROGRAM** automatisch gesetzt. Die Einstellung kann nicht verändert werden

– beim Bausteintyp **FUNCTION_BLOCK** aktivierbar und kann gesetzt oder zurückgesetzt werden.

Details zum POU-Editor siehe POUs auf Seite 96.

6.9 Safety Task

Im Projektnavigator muss genau eine  **Safety Task** unter dem Sicherheitsapplikationsobjekt vorhanden sein. Diese Safety Task ruft gemäß einer editierbaren Reihenfolge die Programme (POUs vom Typ PROGRAM) des Safety-Applikationobjekts auf. Sie definiert also diejenigen Programme, die auf der Steuerung ausgeführt werden. Die Programmauswahl und ihre Aufrufreihenfolge werden durch den Entwickler festgelegt. Die im Editor aufgelisteten Programme entsprechen den Programm-POUs des Projektbaums.

Hinzufügen der Safety Task

Das Objekt **Safety Task** wird beim Hinzufügen der Sicherheitssteuerung automatisch unter dem Sicherheitsapplikationsobjekt hinzugefügt.

Erfolgt das Hinzufügen der Safety Task nicht automatisch oder wurde sie gelöscht, so kann das Objekt über das Kontextmenü-Befehl **Hinzufügen** mit der Auswahl **Safety Task** dem Sicherheitsapplikationsobjekt hinzugefügt werden.

Objekteigenschaften der Safety Task

Die Registerkarte **Safety** des Eigenschaften-Dialogs enthält die Prüfsumme des Objekts "Safety Task" und editierbare Felder für die Version und den Kommentar der Safety Task.

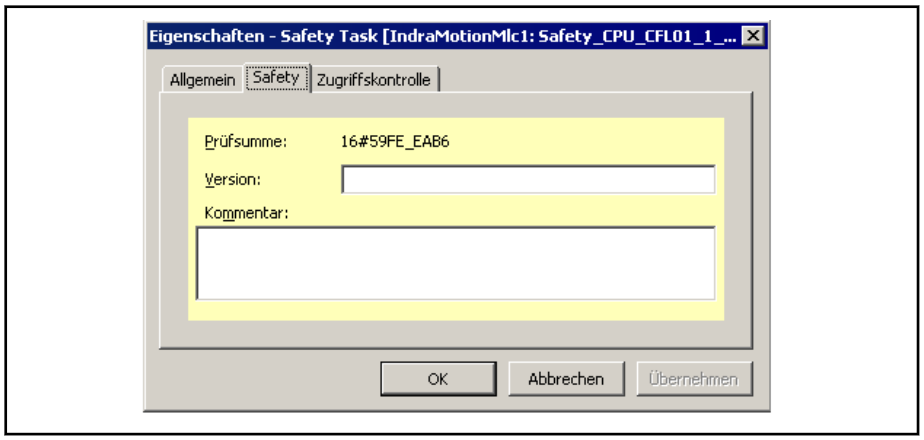


Abb. 6-24: Eigenschaften der Safety Task, Registerkarte 'Safety'

Editor der Safety Task

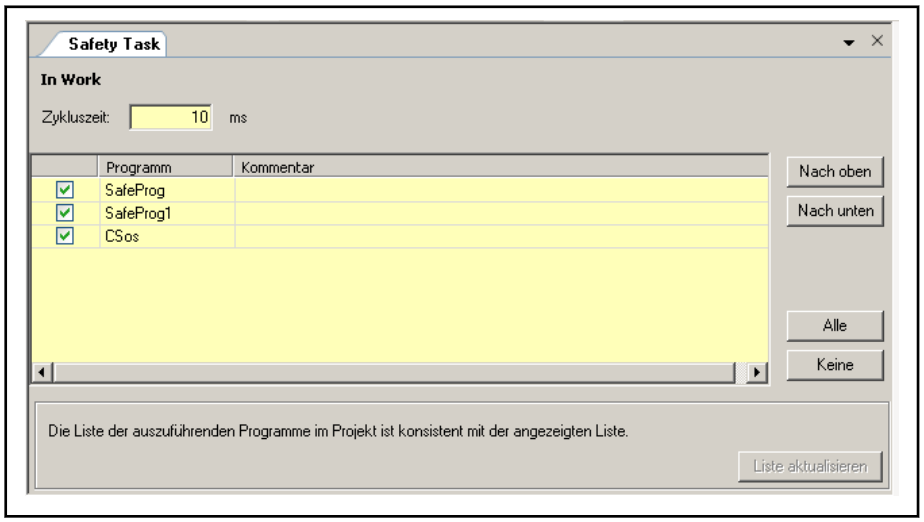


Abb. 6-25: Editor Safety Task

Der Editor des Task-Objekts besteht aus 3 Bereichen:

- Pin-Information
- Editierbare Anzeige der Zykluszeit
- Liste der Programme mit Schaltflächen

Zykluszeit

Der Entwickler kann den Wert der Zykluszeit, verändern. Sie wird in ganzen ms (Millisekunden) eingegeben. Als Default wird eine für das angewählte Gerät sinnvolle Zykluszeit voreingestellt (im Beispiel 10 ms). Die Zykluszeit kann zwischen 3 und 150 ms in 1 ms Schritten eingestellt werden.

- Wird ein kleinerer Wert als der für diese Sicherheitssteuerung zulässige eingegeben, so wird der Minimalwert verwendet.
- Wird ein größerer Wert als der für diese Sicherheitssteuerung zulässige eingegeben, so wird der Maximalwert verwendet.

☞ Durch eine Veränderung der Zykluszeit ändern sich auch die Pin-Prüfsumme und die Prüfsumme des Objekts Safety Task!

Programmliste

Die Programmliste enthält alle Programme (POUs vom Typ PROGRAM) der Sicherheitsapplikation.

Die Markierung in der ersten Spalte der Programmliste zeigt an, welche Programme ausgeführt werden. Durch Deaktivierung der Checkbox in der ersten Spalte können die markierten Einträge der Liste auskommentiert werden. Programme von auskommentierten Einträgen werden auf der Steuerung nicht ausgeführt. Auskommentierungen werden durch Aktivieren der Schaltfläche rückgängig gemacht, die entsprechende Programme werden von der Task aufgerufen.

*Die Schaltflächen **Nach oben** und **Nach unten** dienen dazu, die Aufrufreihenfolge der Programme zu ändern:*

- **Nach oben**: der aktuell selektierte Programmeintrag wird in der Programmliste um eine Position nach oben verschoben
- **Nach unten**: der aktuell selektierte Programmeintrag wird in der Programmliste um eine Position nach unten verschoben

Durch Aktivieren der Schaltfläche **Alle** werden alle Programme der Liste markiert. Aktivierung der Schaltfläche **Keine** werden alle Programme abgewählt.

Die zuletzt durchgeführte Änderung in Feldern der Programmliste wird rot markiert. Diese farbliche Markierung wird beim Schließen des Editors entfernt.

Aktualisierung der Programmliste:

Die Programmliste des Safety Task-Objekts wird bei Änderungen der Projektstruktur im Projektbaum automatisch aktualisiert. **Ausnahme:**

☞ Besitzt der Entwickler in der Benutzerverwaltung nicht das Recht zum Bearbeiten des Objekts Safety Task des Projekts, so wird die Programmliste des Task-Objekts nicht automatisch aktualisiert. In diesem Fall ist die Schaltfläche **Aktualisieren** aktivierbar, kann aber nur von einem Entwickler aktiviert werden, der das Recht zum Bearbeiten des Objekts Safety Task besitzt.

Zur Dokumentierung kann zu jedem Programmeintrag in der Liste ein Kommentar eingegeben werden.

6.10 Globale Variablenliste (GVL)

Die Safety Globale Variablenliste (GVL) dient der Darstellung, Deklaration und Bearbeitung global deklarierter Variablen, die innerhalb der ganzen Sicherheitsapplikation gültig sind.

Einer Sicherheitsapplikation können mehrere GVLs hinzugefügt werden. Über die in den GVL enthaltenen Variablen können also die Programme der SafetyApp per VAR_EXTERNAL Daten austauschen.

Im Projektbaum wird eine Safety GVL mit dem Symbol  dargestellt.

Die in der GVL der Sicherheitsapplikation deklarierten Variablen sind nicht projektweit, sondern nur innerhalb der Sicherheitsapplikation global gültig.

Hinzufügen einer GVL

1. Im Projektbaum das Sicherheitsapplikationsobjekt **SafetyApp** selektieren
2. Kontextmenü-Befehl **Hinzufügen** mit Auswahl **Safety-GVL...** selektieren
3. Im Dialogfenster **Objekt hinzufügen Assistent** den Namen und optional Kommentar für die GVL eingeben.
4. Schaltfläche **Fertigstellen** aktivieren

Es öffnet sich der Editor zur Deklaration der globalen Variablen.

Objekteigenschaften GVL

Der Eigenschaftendialog einer GVL enthält die Registerkarten **Allgemein**, **Safety** und **Zugriffskontrolle**

Die Registerkarte **Safety** des Eigenschaftendialogs enthält die **Prüfsumme** (Prüfsumme der GVL) und editierbare Felder für die Version und den Kommentar der GVL.

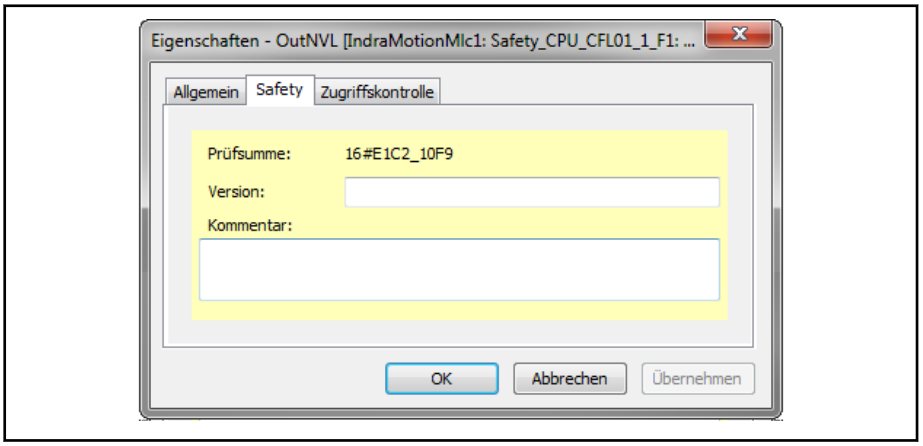


Abb. 6-26: Eigenschaften der GVL, Registerkarte 'Safety'

Die Version kann der Entwickler frei vergeben. Mit Hilfe der Version kann in der Vergleichsansicht des Safety-Applikationsobjekts schnell erkannt werden, um welche Version des Objekts es sich handelt.

Editor einer GVL

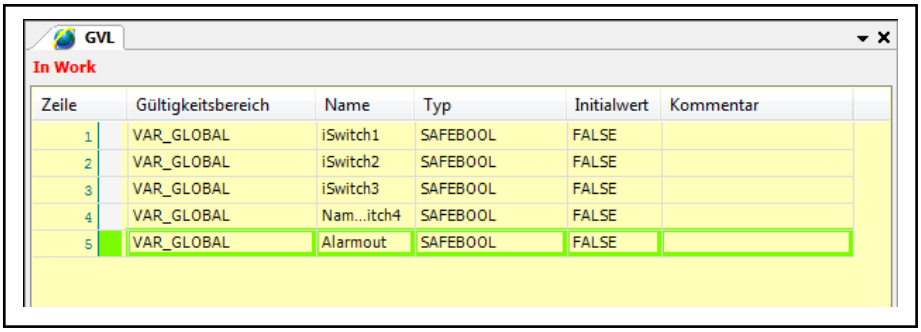


Abb. 6-27: GVL-Editor

Der Editor besteht aus zwei Bereichen:

- Anzeige der Pin-Information (falls nicht gepinnt: In Work)
- Tabelle zur Variablendeklaration mit den Spalten: Zeile, Bereich, Name, Typ, Initialwert, Kommentar

  Grüne Markierung: in der aktuellen Sitzung editiert.

Deklaration einer globalen Variablen siehe Variablendeklaration auf Seite 73.

Gültigkeitsbereiche für globale Variablen

- VAR_GLOBAL
- VAR_GLOBAL CONSTANT

Datentypen für globale Variablen

- INT
- DINT
- TIME
- BOOL
- BYTE
- WORD
- DWORD
- SAFEINT
- SAFEDINT
- SAFETIME
- SAFEBOOL
- SAFEBYTE
- SAFEWORD
- SAFEDWORD

☞ In einer GVL deklarierte Variablen stehen in der Eingabehilfe der Sicherheitsapplikation unter der Kategorie "Globale Variablen" zur Verfügung.

Globale Variable ändern

1. GVL des Safety-Applikationsobjekts im Projektbaum selektieren
2. Kontextmenübefehl **Öffnen** aktivieren
3. Im Editor (siehe Abb. 6-27 "GVL-Editor" auf Seite 66) die zu ändernde Zelle mit Doppelklick öffnen
4. Inhalt der Zelle ändern

6.11 Safety Netzwerkvariablenliste (NVL)

Die NVL (Safety **N**etzwerk**V**ariablen**L**iste) enthält Variablendeklarationen, die zu oder von einer anderen Sicherheitssteuerung übertragen werden. Die Variablen der NVL können in sämtlichen Programmen der Sicherheitsapplikation verwendet werden.

Einer Sicherheitsapplikation können mehrere NVLs hinzugefügt werden. Über die in den NVL enthaltenen Variablen können also die Programme der SafetyApp per VAR_EXTERNAL Daten austauschen.

Im Projektexplorer werden eine Safety-Sende-NVL mit dem Symbol und eine Safety-Empfangs-NVL mit dem Symbol dargestellt.

Siehe auch

- Objekt 'Safety Netzwerkvariablenliste (Sender)', Seite 663
- Objekt 'Safety Netzwerkvariablenliste (Empfänger)', Seite 665

Die in der NVL der Sicherheitsapplikation deklarierten Variablen sind nicht projektweit, sondern nur innerhalb der Sicherheitsapplikation global gültig.

Wenn die Querkommunikation mittels der Objekte "Safety-Netzwerkvariablenliste (Sender)" und "Safety-Netzwerkvariablenliste (Empfänger)" konfiguriert wurde und ein Download auf Sicherheits- und Hauptsteuerungen durchgeführt wurde, wird die Kommunikationsverbindung über die Hauptsteuerungen der Sicherheitssteuerung automatisch hergestellt.

Die Sicherheitssteuerung tauscht immer den Variablenwert aus, den die Variable am Ende des Applikationszyklus hat. Alle ausgetauschten Telegramme (Variablenwerte und Empfangsbestätigungen) werden synchron zum Applikationszyklus in der Ausgangsphase abgeschickt und in der Eingangsphase empfangen werden.

Ein Sender kann die gleiche Variable an mehrere Empfänger senden und ist dabei programmatisch und funktional unabhängig von seinen Empfängern. Die Empfänger müssen sich zum Aufbau der abgesicherten Kommunikation beim Sender anmelden.

Ein konfigurierter Variablenaustausch läuft automatisch an, automatisch weiter und nach Wegfall eines Unterbrechungsgrunds automatisch wieder an wenn folgende Bedingungen erfüllt sind:

- Das Routing der Hauptsteuerungen von Sender und Receiver(n) läuft.
- Die Sicherheitsapplikationen von Sender und Empfänger laufen.
- Die Kommunikationsstrecke ist schnell genug, die Zykluszeiten sind klein genug und die Watchdog-Zeit ist groß genug.

Für die Querkommunikation werden die Bausteine NetVarReceiver und NetVarSender der Bibliothek SafetyNetVar.library verwendet, zusätzlich wird für jede Sender- Empfänger-Beziehung eine Instanz des Bausteins "NetVarSenderStack" erzeugt.

Siehe auch:

- NetVarReceiver auf Seite 605
- NetVarSender auf Seite 603
- SafetyNetVar.library auf Seite 599

Hinzufügen einer NVL-Verbindung

1. Im Projektbaum das Sicherheitsapplikationsobjekt **SafetyApp** der Sender-Sicherheitssteuerung selektieren
2. Kontextmenü-Befehl **Hinzufügen** mit Auswahl **Safety-NVL-Sender...** selektieren

3. Im Editor des Objekts "Safety-NVL (Sender)" in das Eingabefeld "Safety-Adresse dieser Variablenliste" die Safety-Adresse eingeben.
4. Im Projektbaum das Sicherheitsapplikationsobjekt **SafetyApp** der Empfänger-Sicherheitssteuerung selektieren
5. Kontextmenü-Befehl **Hinzufügen** mit Auswahl **Safety-NVL-Receiver...** selektieren
6. In die Eingabefelder "Connection ID" und "Watchdog-Zeit" die entsprechenden Werte eingeben.
7. Die Sender-NVL in der Auswahlbox "Zugeordneter Sender:" auswählen

   Die Variablenliste der Empfänger-NVL wird mit den Variablen der Sender-NVL gefüllt.

Es öffnet sich der Editor zur Deklaration der globalen Variablen.

Objekteigenschaften NVL

Der Eigenschaftendialog einer NVL enthält die Registerkarten **Allgemein**, **Safety** und **Zugriffskontrolle**

Die Registerkarte **Safety** des Eigenschaftendialogs enthält die **Prüfsumme** (Prüfsumme der NVL) und editierbare Felder für die Version und den Kommentar der NVL.

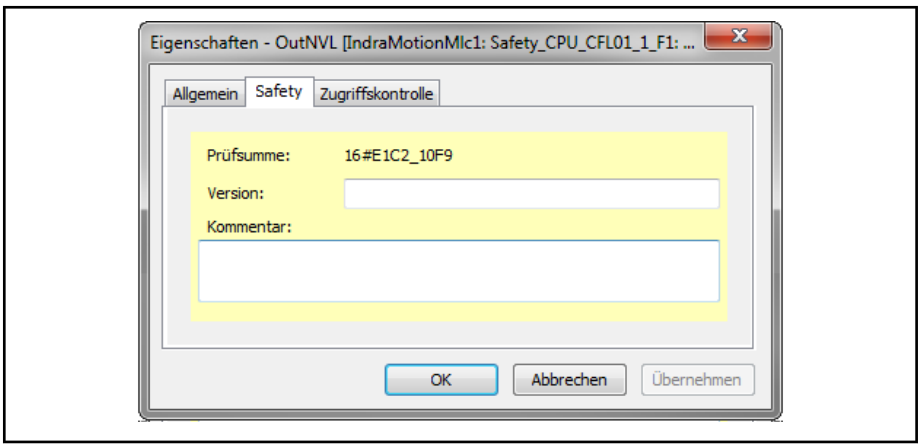


Abb. 6-28: Eigenschaften der NVL, Registerkarte 'Safety'

Die Version kann der Entwickler frei vergeben. Mit Hilfe der Version kann in der Vergleichsansicht des Safety-Applikationsobjekts schnell erkannt werden, um welche Version des Objekts es sich handelt.

Editor einer Sender NVL

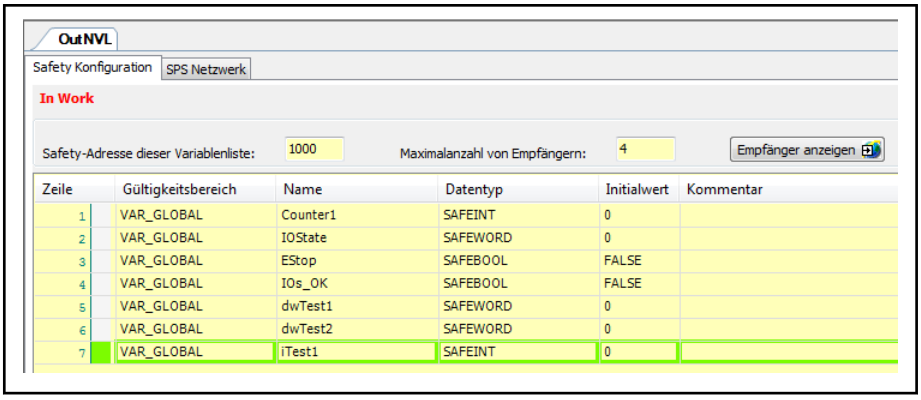


Abb. 6-29: Sender NVL-Editor

Der Editor besteht aus zwei Bereichen:

- Anzeige der Pin-Information (falls nicht gepinnt: **In Work**)

- Tabelle zur Variablendeklaration mit den Spalten: Zeile, Bereich, Name, Typ, Initialwert, Kommentar

  Grün markiert: letzte Eingabe

Deklaration einer globalen Variablen siehe Variablendeklaration auf Seite 73.

Gültigkeitsbereiche für globale Variablen

- VAR_GLOBAL

Datentypen für Safety Netzwerkvariablen

- SAFEBOOL
- SAFEWORD
- SAFEINT

☞ In einer NVL deklarierte Variablen stehen in der Eingabehilfe der Sicherheitsapplikation unter der Kategorie "Globale Variablen" zur Verfügung.

Globale Netzwerk-Variable ändern

1. Sender-NVL des Safety-Applikationsobjekts im Projektbaum selektieren
2. Kontextmenübefehl **Öffnen** aktivieren
3. Im Editor die zu ändernde Zelle mit Doppelklick öffnen (Sender NVL-Editor auf Seite 68)
4. Inhalt der Zelle ändern
5. Empfänger-NVL des Safety-Applikationsobjekts im Projektbaum selektieren
6. Kontextmenübefehl **Öffnen** aktivieren
7. Im Editor die Schaltfläche **Aktualisieren** drücken (siehe Konfiguration einer Empfänger-NVL auf Seite 71)

   Die Variablenliste der Empfänger-NVL wird mit den Variablen der Sender-NVL aktualisiert.

Netzwerk-Editor einer Sender-NVL

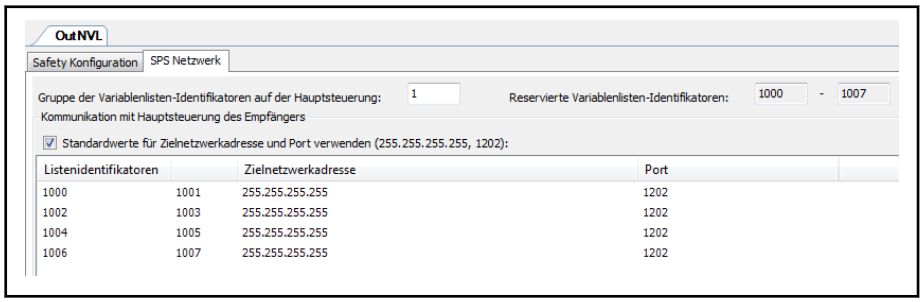


Abb. 6-30: Netzwerkkonfiguration einer Sender-NVL

Gruppe der Variablenlisten-Identifikatoren auf der Hauptsteuerung	Erlaubte Werte: 1 bis 255 Der angezeigte Wert wird beim Erstellen des Safety NVL(Sender) automatisch errechnet. Wenn Sie den Wert verändern, wird die Liste mit den Variablenlisten-Identifikatoren neu erstellt und die Anzeige Reservierte Variablenlisten-Identifikatoren aktualisiert. In diesem Zuge wird die Option zur Verwendung von Standardwerten aktiviert, wodurch manuelle Änderungen in der Liste zurückgesetzt werden.
Reservierte Variablenlisten-Identifikatoren	Angabe des Bereich: von "Kleinster der Gruppe zugeordneter Wert" bis "Größter der Gruppe zugeordneter Wert" der Listenidentifikatoren, die für diese Safety Netzwerkvariablenliste (Sender) reserviert sind. Werte aus diesem Bereich dürfen nicht für die Variablenlistenkennung von Standard Netzwerkvariablen oder andere Safety-Netzwerkvariablen im gleichen Netzwerk verwendet werden!
Standardwerte für Zielnetzwerkadresse und Port verwenden (255.255.255.255, 1209):	Rücksetzen der Zielnetzwerkadressen und des Ports der Liste auf den Standardwert
Liste mit Variablenlisten-Identifikatoren Spalten:	Anzahl der Einträge (Zeilen) entspricht dem Wert von Maximalanzahl von Empfängern der Registerkarte Safety Konfiguration
Listenidentifikatoren	Erster Listenidentifikator für ein Sender-Empfänger Verbindungspaar.
	Nicht editierbar. Wird abhängig vom ersten Listenidentifikator gesetzt: Listenidentifikator + 1

Zielnetzwerkadresse	Editierbar, wenn Standardwerte für Zielnetzwerkadresse und Port verwenden (255.255.255.255, 1209): nicht aktiviert ist. Kann verwendet werden, um eine Verbindung zu einer Safety Netzwerkvariablenliste (Empfänger) an einer Steuerung mit einer bestimmten IP-Netzwerkadresse aufzubauen, anstatt die Broadcast Adresse zu verwenden. In diesem Fall kann es erforderlich sein, dass auch die Gegenstelle (auf Empfängerseite) entsprechend konfiguriert wird. Für den Fall dass die IP Adressen der Steuerungen derart eingetragen werden, müssen ebenfalls die Listenidentifikatoren auf Empfängerseite manuell passend gesetzt werden.
Port	Kann verwendet werden um einen benutzerdefinierten Port für die Kommunikation mit dem Empfänger vorzunehmen, dessen Eintrag angepasst wird. Die Einstellung für den Port muss für die jeweilige Sender-Empfänger Verbindung übereinstimmen, damit eine Kommunikation zustandekommen kann. Es wird empfohlen den Port 1209 zu verwenden. Port 1202 wird bereits durch die Standard-Steuerung belegt.

Tab. 6-3:

Editor einer Empfänger NVL

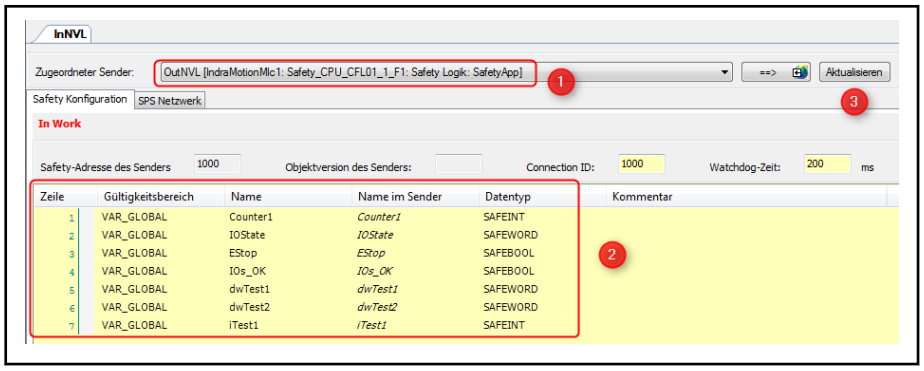


Abb. 6-31: Konfiguration einer Empfänger-NVL

Mit der Übernahme des Namens des "Zugeordneten Senders" (1), werden alle Variablen (2) aus der Sender-NVL übernommen.

Wird die Sender-NVL geändert, bitte die Empfängerseite aktualisieren (3).

Die Namen können in der Spalte "Name" geändert werden.

6.12 Bibliotheksverwalter

Im Bibliotheksverwalter werden die von der Sicherheitsapplikation verwendbaren Bibliotheken verwaltet. Die Liste der Bibliotheken stammt aus der Gerätebeschreibung der Sicherheitssteuerung, welche die Sicherheitsapplikation mit dem Bibliotheksverwalter enthält. Alle Bibliotheken, die für diese Sicherheitssteuerung zur Verfügung stehen, werden automatisch eingefügt.

Der Bibliotheksverwalter des IndraWorks Safety Manager entspricht dem Bibliotheksverwalter von IndraWorks. Details siehe Online-Hilfe von IndraWorks.

Genauere Beschreibungen der Sicherheitsbibliotheken siehe:

- SafetyStandard.library, Seite 229
- SafetyUtilities.library, Seite 240
- SafetyPLCopen.library, Seite 292
- SafetyPLCopenPress.library, Seite 422
- SafetyDriveMotion.library, Seite 536
- SafetyProfisafeHost.library, Seite 577
- SafetyCSos.library, Seite 589
- SafetyNetVar.library, Seite 599

Das Objekt **Bibliotheksverwalter** wird automatisch mit der Sicherheitssteuerung in den Projektbaum eingefügt. Dieses Objekt kann unter einem Safety-Applikationsobjekt nur einmal vorhanden sein.

Objekteigenschaften

Der Eigenschaftendialog enthält, wie in IndraWorks-Standard, die Registerkarten **Allgemein**, **Zugriffskontrolle** und **Übersetzen**.

Editor des Bibliotheksverwalters

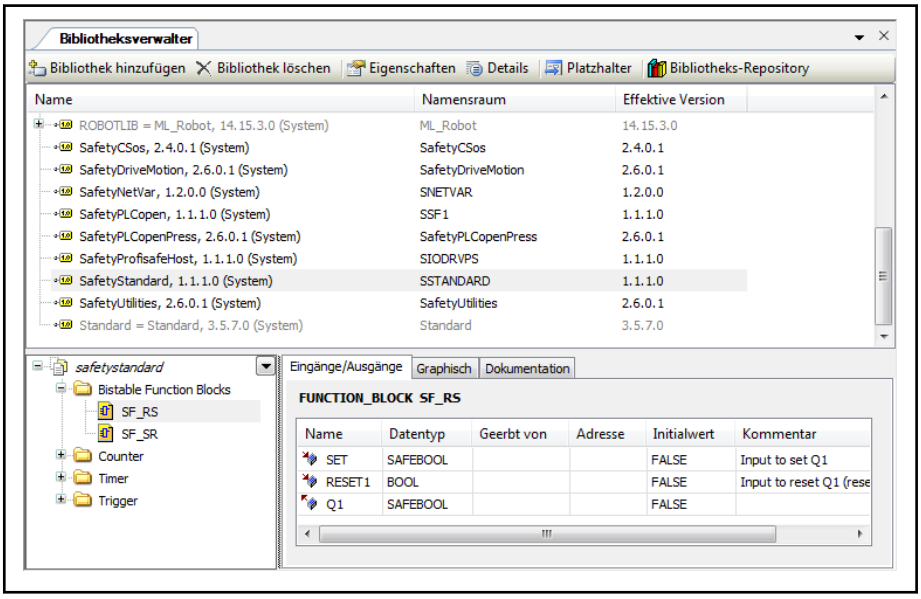


Abb. 6-32: Beispiel: Editor des Bibliotheksverwalters

Der Editor besteht aus folgenden drei Bereichen:

- Bibliotheksliste
- Bausteinliste (einer selektierten Bibliothek, am Beispiel **SafetyStandard**)
- Bausteinbeschreibung (eines in der Bausteinliste selektierten Bausteins, am Beispiel **SF_RS**) mit den Registerkarten: **Eingänge/Ausgänge**, **Graphisch** und **Dokumentation**

Weitere Details zu diesem Editor siehe Online-Hilfe von IndraWorks-Standard.

Hinweise zum Bibliotheksverwalter

Der Bibliotheksverwalter ist für den Nachweis der in der Sicherheitsapplikation verwendeten Bibliotheksbausteine (IEC-Bausteine und externe Bausteine) bei der Verifikation und bei der Abnahme nicht geeignet.

Der Nachweis der ausführungsrelevanten Stände muss über die Vergleichsansicht erfolgen (Pinnen auf Seite 161).

Abnahmedokumentation siehe Software-Abnahme und Dokumentation auf Seite 183.

Die Liste der Bibliotheken kann sich durch ein Aktualisieren der Sicherheitsteuerung ändern.

6.13 Variablendeklaration

Variable deklarieren

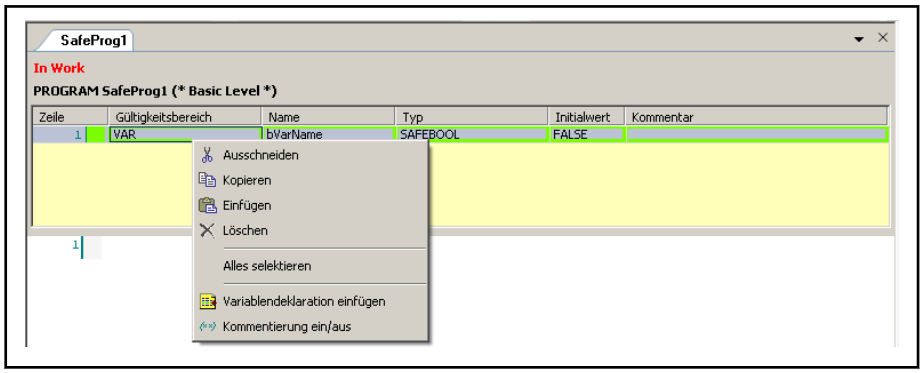


Abb. 6-33: Editor zur Variablendeklaration mit geöffnetem Kontextmenü

1. Im Variablendeklarationseditor das Kontextmenü aktivieren
2. Im Menüfenster den Befehl **Variablendeklaration einfügen** aktivieren
3. Im Dialog **Safety Variable deklarieren** die Felder Bereich, Name, Typ, Initialwert und optional Kommentar editieren bzw. auswählen
4. Schaltfläche **OK** aktivieren

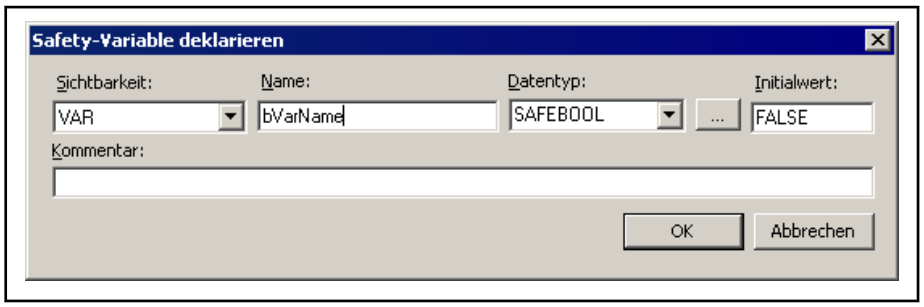


Abb. 6-34: Dialog 'Safety-Variable deklarieren'

Eingabefelder des Dialog "Safety-Variable deklarieren"

- Sichtbarkeit
- Name
- Datentyp
- Initialwert
- Kommentar

Bestehende Deklarationen können durch einen Doppelklick auf das entsprechende Feld der Deklarationstabelle geändert werden.

Datentypen

In der Sicherheits-Programmierung mit IndraWorks Safety Manager werden zwischen sicherheitsbezogenen und nicht-sicherheitsbezogenen Daten unterschieden.

Die nicht-sicherheitsbezogenen Daten sind die IEC-Standard-Datentypen.

Bei den sicherheitsbezogenen Daten sind die IEC Standard-Datentypen mit dem Präfix SAFE erweitert.

Die Mapping-Variablen der Eingangs- und Ausgangskanäle von sicheren Feldgeräten haben immer einen Datentyp SAFExxx, während die Mapping-Variablen von nicht-sicheren Feldgeräten immer einen nicht-SAFE-Datentyp haben.

Datentyp	Bitlänge	Wertebereich	Beschreibung
BOOL	1	0,1	0 entspricht FALSE 1 entspricht TRUE
DINT	32	- 2.147.483.648 ... 2.147.483.647	
INT	16	-32.768 ... 32.767	
TIME	32	0 ... 2.147.483,647 s	Zeitdauer
WORD	16	0 ... 65.535 (16#00 ... 16#FFFF)	

Tab. 6-4: IEC-Standard-Datentypen

Die Datentypen BYTE, DWORD, SAFEBYTE und SAFEDWORD können nur in den logischen E/As auftreten und in Extended-Level Programmen als Kanalvariablen (Kategorie: globale Variablen, Deklaration als VAR_EXTERNAL) verwendet werden (siehe Verwendung von Logischen E/As im Projekt auf Seite 59).

Datentyp	Bitlänge	Wertebereich	Beschreibung
SAFEBOOL	1	0,1	0 entspricht FALSE 1 entspricht TRUE
SAFEDINT	32	- 2.147.483.648 ... 2.147.483.647	
SAFEINT	16	-32.768 ... 32.767	
SAFETIME	32	0 ... 2.147.483,647 s	
SAFEWORD	16	0 ... 65.535 (16#00 ... 16#FFFF)	

Tab. 6-5: SAFE-Datentypen

Die Datentypen REAL/LREAL sind in der Safety-Programmierung nicht zulässig. Werden sie dennoch verwendet, führt dies zu einem Übersetzungsfehler.

Benutzerdefinierte Datentypen

Die Datentyp-Kategorie **Benutzerdefinierte Typen** enthält Funktionsbausteine der Sicherheitsapplikation und die Bibliotheken. Diese Kategorie kann in der Eingabehilfe, die im Dialog Abb. 6-34 "Dialog 'Safety-Variable deklarieren'" auf Seite 73 durch Aktivierung der Schaltfläche  geöffnet wird, ausge-

wählt werden. Bei bereits deklarierten Variablen wird die Eingabehilfe des Datentyps wie folgt geöffnet:

1. Im Deklarationsfenster die Typ-Zelle der entsprechenden Variable auswählen
2. In der Auswahlliste des Typs das Symbol **...** anklicken

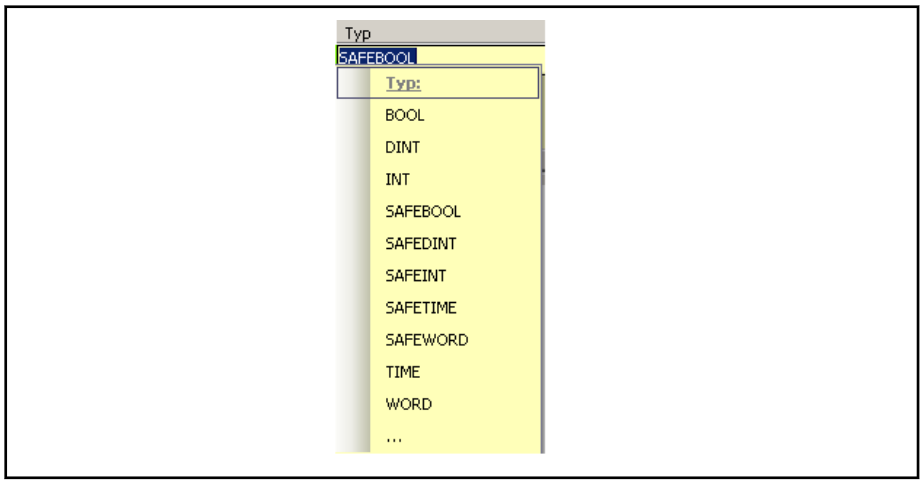


Abb. 6-35: Deklarationsfenster: Auswahlliste des Typs

6.14 Bibliotheken

Für die Programmierung mit SafeLogic stehen dem Entwickler Sicherheitsbibliotheken mit bereits zertifizierten Funktionsbausteinen zur Verfügung. Folgende Bibliotheken sind Teil der Installation des IndraWorks Safety Managers:

Genauere Beschreibungen der Sicherheitsbibliotheken siehe:

- SafetyStandard.library, Seite 229
- SafetyUtilities.library, Seite 240
- SafetyPLCopen.library, Seite 292
- SafetyPLCopenPress.library, Seite 422
- SafetyDriveMotion.library, Seite 536
- SafetyProfisafeHost.library, Seite 577
- SafetyCSos.library, Seite 589
- SafetyNetVar.library, Seite 599

Diese Bibliotheken sind automatisch Teil der Sicherheitsapplikation. Eine Installation ist nicht notwendig.

Je nach Safety-Feldbus können weitere feldbusspezifische Bibliotheken zur Verfügung stehen.

Die Verwaltung dieser Bibliotheken erfolgt in IndraWorks-Standard.



Das Bibliotheks-Repository und der Bibliotheksverwalter sind für den Nachweis der in der Sicherheitsapplikation verwendeten Bibliotheksbausteine bei der Verifikation und der Abnahme nicht geeignet.

Der Nachweis der ausführungsrelevanten Stände muss über die Vergleichsansicht erfolgen, siehe Pinnen, Seite 161.

Abnahmedokumentation siehe Software-Abnahme und Dokumentation auf Seite 183.

6.15 Geräteverwaltung

☞ Wurden Gerätebeschreibungen mit Standard-IndraWorks ohne Safety-Erweiterung installiert, so können sie in SafeLogic nicht verwendet werden. Die entsprechenden Gerätebeschreibungen müssen nach Installation von IndraWorks Safety Manager neu installiert werden, um sie in SafeLogic verwendbar zu machen.

Sicherheitsgeräte werden analog zu Standardgeräten in der „Gerätedatenbank“ (Menü **Extras ▸ Gerätedatenbank...**) verwaltet, installiert und deinstalliert.

Für Detailinformationen wird auf die Dokumentation **Rexroth IndraWorks 15VRS Feldbusse**, DOK-IWORKS-FB******V15-APRS-DE-P, R911393282, verwiesen.

In der Gerätedatenbank sind die sicherheitsrelevanten Geräte wie folgt eingruppiert:

- Das Safety-Funktionsmodul befindet sich unterhalb der Steuerungen in der Unterkategorie "Sicherheitssteuerungen".
- logische Geräte der sicheren Feldgeräte befinden sich in der Unterkategorie "sichere Module" des jeweiligen Feldbusses.
- physikalische Geräte für sichere Feldgeräte werden nur explizit aufgeführt, wenn sie nicht als Module oder Submodule als Teil des Feldgeräts installiert werden. Sie werden nicht in eigenen Unterkategorien aufgeführt.
- logische Geräte der Standard-Feldgeräte werden in der Unterkategorie "Generierte Logische Geräte" der Kategorie "Logische Geräte" verwaltet.
- In der Kategorie "Logische Geräte" befindet sich die Unterkategorie "Logische Austauschgeräte" zum Datenaustausch mit der Standardsteuerung.

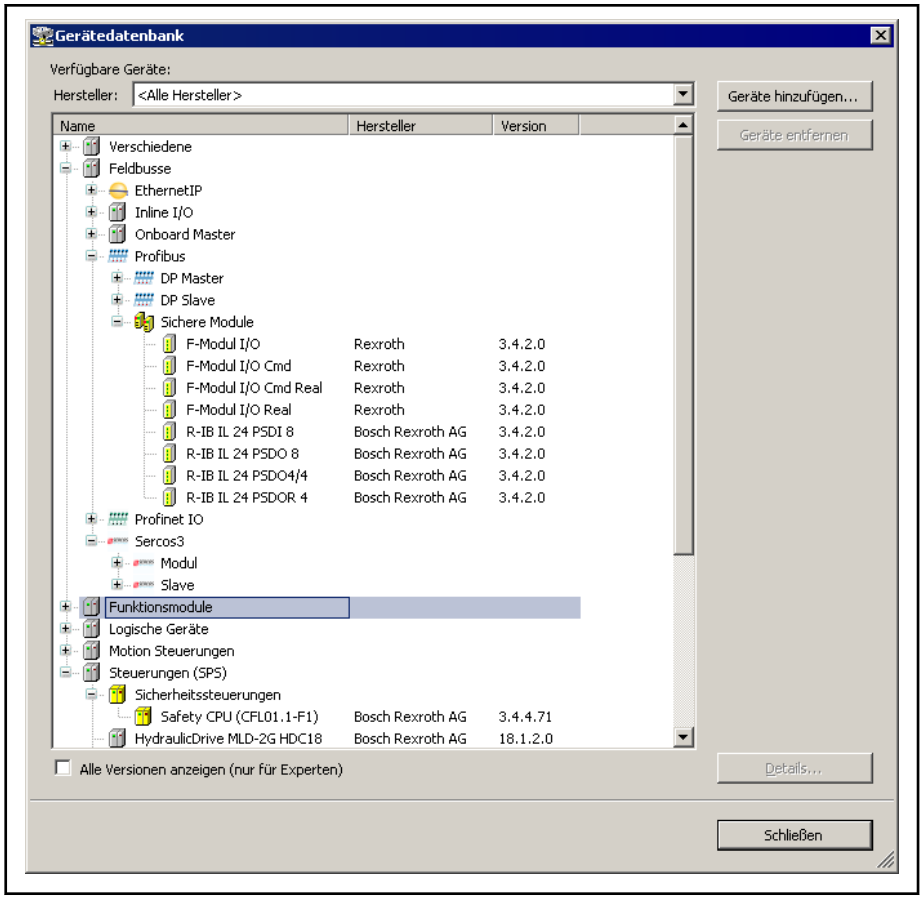


Abb. 6-36: Safety–Komponenten in der Gerätedatenbank

Genaue Informationen zu den logischen E/As siehe Logische E/As - Überblick auf Seite 47.

Sichere Feldgeräte, die im Geräte-Repository installiert wurden, können im Projektbaum auf Standardseite wie jedes installierte Gerät eingefügt werden.

Die sicherheitsgerichteten Parameter der sicherheitsgerichteten Feldgeräte werden in den jeweiligen logischen E/As (siehe Sichere Konfiguration und Sichere Parametrierung auf Seite 57) editiert. Die Parametrierung der Standard-Feldgeräte und der Standard-Parameter der sicherheitsgerichteten Geräte erfolgt wie in IndraWorks-Standard.

6.16 Zugriffschutz

6.16.1 Überblick Zugriffsschutz

Übersicht über Safety-Zugriffsschutz

In der Projekt-Benutzerverwaltung können Rechte so vergeben werden, dass Safety-Objekte eines Projekts nur von bestimmten Benutzergruppen (z. B. "Safety-Entwickler") erstellt bzw. modifiziert werden können.

Der gesamte Zugriffschutz von IndraWorks Safety Manager-Sicherheitsapplikationen wird durch folgende Mechanismen realisiert:

- Benutzerverwaltung im Projekt
- Admin-Passwort



Ohne Benutzerverwaltung im Projekt besitzt jede Person, die an dem Projekt mitarbeitet, alle Rechte an den Standard- und Sicherheitsapplikationen im Projekt!

6.16.2 Benutzerverwaltung im Projekt

Im IndraWorks Safety Manager ist eine Safety-Benutzerkonfiguration für Projekte mit Sicherheitsapplikation integriert.

Der Projektleiter kann diese Safety-Benutzerkonfiguration für sein Projekt verwenden oder aber auch eine eigene Benutzerverwaltung erstellen.

☞ Der Projektleiter muss sich bereits zu Beginn der Erstellung des neuen Projekts im Dialog **Neues Projekt...** entscheiden, ob im Projekt eine Benutzerverwaltung angelegt werden soll. (siehe Vorgehensweise bei Erstellung einer Sicherheitsapplikation auf Seite 34).

Wird ein **Leeres Projekt**, d. h. ein Projekt ohne Safety-Benutzerkonfiguration erzeugt, so muss der Projektleiter selbst eine Benutzerverwaltung für das Safety Projekt erstellen (siehe Online-Hilfe von IndraWorks).

☞ Die Rechtevergabe der Benutzerverwaltung befindet sich im Menü **Projekt** ▸ **Benutzerverwaltung** ▸ **Rechte...**.

☞ Jeder Benutzer gehört automatisch zu der Gruppe **Everyone**.

Einstellungen der Safety-Benutzerkonfiguration

Diese Benutzerverwaltung enthält bereits folgende Einstellungen:

Benutzergruppen

- **Owner**
- **Safety**
- **Safety.ExtendedLevel**
- **Everyone**

Benutzer

- **Owner**, der der Gruppe **Owner** angehört
- **saf**, der der Gruppe **Safety**
- **ext**, der den Gruppen **Safety.ExtendedLevel** und **Safety** angehört.

Die voreingestellten Benutzer haben noch kein Passwort.

Die Gruppe **Owner** enthält bereits den Benutzer **Owner**, noch ohne Passwort.

Der Benutzer **Owner** kann den Benutzergruppen Safety, Safety.ExtendedLevel und Owner entsprechende Benutzer über das Menü **Projekt** ▸ **Benutzerverwaltung** ▸ **Benutzer und Gruppen** zuordnen und an die Benutzer Passwörter vergeben.

Der Benutzer **Owner** kann neue Benutzergruppen definieren. Eine neue Benutzergruppe hat anfangs die Rechte der Benutzergruppe **Everyone** des Safety-Benutzerkonfiguration. Durch Zuordnung einer neuen Gruppe zu anderen bereits existierendem Benutzergruppen (z. B. **Safety**, **Safety.ExtendedLevel**), erhält die neue Gruppe die Zugriffsrechte der zugeordneten Gruppe.

☞ Der **Projektleiter** muss für den Benutzer "Owner" ein geeignetes, sicheres Passwort vergeben, um den Zugriffschutz des Projekts zu gewährleisten.

Passwort definieren

1. Menü **Projekt** öffnen
2. **Projekteinstellungen** selektieren
3. Im Dialogfenster **Projekteinstellungen** die **Benutzer und Gruppen** auswählen
4. Unter dem Tab **Benutzer** Owner selektieren
5. Schaltfläche **Bearbeiten...** aktivieren
6. Im Dialogfenster **Benutzer bearbeiten** altes Passwort, neues Passwort und Passwortbestätigung eingeben
7. Schaltfläche **OK** aktivieren

Für weitere Informationen zur Passwortvergabe wird auf die Online-Hilfe von IndraWorks-Standard verwiesen.

In der Benutzerverwaltung sind die Rechte auf Befehle, Benutzer und Gruppen, Objekttypen, Projektobjekte folgendermaßen vorgegeben, können aber von Mitgliedern der Gruppe Owner jederzeit geändert werden:

Rechtevergabe des Safety-Benutzerkonfiguration

- Die Gruppe **Owner** besitzt alle Rechte
- Die Gruppe **Safety** besitzt **nicht**
  - das Recht, eine **Safety Extended-POU** zu erzeugen
  - das Recht, eine **Safety Externe-POU** zu erzeugen
  - das Recht zum **Ausführen** des Befehls **Safety FUP: Return einfügen**
  - das Recht zum **Ausführen** des Befehls **Safety FUP: Sprung einfügen**
  - die Rechte für das **Ausführen** der **Befehle**, die die Benutzerverwaltung betreffen.
  - Recht zum **Bearbeiten** der **Benutzer und Gruppen**.
- Die Gruppe **Safety Extended.Level** besitzt **nicht**
  - die Rechte für das **Ausführen** der **Befehle**, die die Benutzerverwaltung betreffen
  - Recht zum **Bearbeiten** der **Benutzer und Gruppen**
- Die Gruppe **Everyone** besitzt **nicht**
  - die Rechte zum Ausführen der**Befehle** der Kategorie **Safety**
  - die **Rechte** zum **Erzeugen** der Safety-Objekttypen
  - die Rechte zum **Ausführen** der **Befehle**, die die Benutzerverwaltung betreffen
  - das Recht zum **Bearbeiten** der **Benutzer und Gruppen**

Hinweise für die Entwicklung einer Sicherheitsapplikation mit dem Safety-Benutzerkonfiguration

Zum Einfügen einer **Safety Extended-POU** muss der Entwickler Mitglied der Benutzergruppe **Safety.ExtendedLevel** sein.

*Nach Einfügen einer Sicherheitssteuerung, bzw. des Safety-Applikationsobjekts sind die Änderungsrechte für das Safety-Applikationsobjekt explizit in dem Dialog **Eigenschaften** des Safety-Applikationsobjekts unter dem Tab **Zugriffskontrolle** wie folgt zu konfigurieren:*

- **+** für Safety
- **+** für Safety.ExtendedLevel
- **-** für Everyone

Nach Einfügen einer Safety Extended-POU , muss der Entwickler im Eigenschaftendialog der Safety Extended-POU unter dem Tab **Zugriffskontrolle** der Benutzergruppe **Safety** das Recht zum **Bearbeiten** und **Entfernen** der Extended-POU explizit verbieten, falls die Extended-POU nur von Mitgliedern der Benutzergruppe **Safety.ExtendedLevel** bearbeitet und entfernt werden soll.

Genaue Vorgehensweise siehe Vorgehensweise bei Erstellung einer Sicherheitsapplikation auf Seite 34.

Anlegen eines neuen Projekts ohne Safety-Benutzerkonfiguration

Wird ein neues Projekt als **Leeres Projekt** angelegt, so muss der Projektleiter selbst eine geeignete Benutzerverwaltung für die Sicherheitsapplikation des Projekt erstellen. Für die genaue Vorgehensweise und detaillierte Informationen bei der Erstellung dieser Benutzerverwaltung im Projekt wird auf die Online-Hilfe von Standard-IndraWorks verwiesen.

Um die Auswahl der Safety-spezifischen Befehle und Objekttypen dem Entwickler zu erleichtern, ist bei der Rechtevergabe den Befehlen und den Objekttypen, die für den Safety-Entwickler relevant sind, **Safety** vorangestellt.



Wenn dem Entwickler das Recht zu einer bestimmenten Operation fehlt, kann er während der Operation das Login auf einen Benutzernamen mit mehr Rechten wechseln. (Aktivierung des Befehls **Benutzerverwaltung ▸ Benutzer anmelden...** des Menüs **Projekt**).

6.16.3 Admin-Passwort

Die Boot-Applikation kann mit einem Admin-Passwort (Administrationspasswort) vor nicht autorisiertem, schreibendem Zugriff geschützt werden. IndraWorks Safety Manager besitzt ein Default-Passwort.

Das Admin-Passwort wird im Editor **Safety Online Information** der Sicherheitssteuerung (Doppelklick auf Sicherheitssteuerung im Projektbaum) gesetzt und ist mit der Boot-Applikation der Sicherheitssteuerung verbunden.

Nach jedem Öffnen des Projekts muss sich der Anwender vor dem ersten schreibenden Zugriff auf die Boot-Applikation mit dem Admin-Passwort autorisieren. Es bleibt solange im Programmiersystem gültig gesetzt, bis das Projekt geschlossen wird.

Detaillierte Informationen siehe Schutz der Sicherheitssteuerung vor Schreibzugriff auf Seite 195.

6.16.4 Zugriffschutz bei Anbindung an Quellcodeverwaltung

Wird das Versionskontrollsystem VCS verwendet, so muss auch dort eine der Benutzerverwaltung im Projekt entsprechende Benutzerverwaltung eingerichtet werden.

7 Programmierung

7.1 Überblick Programmierung

Der IndraWorks Safety Manager unterstützt den Entwickler bei der Erstellung einer normenkonformen Sicherheitsapplikation durch

- Förderung guter Programmiertechniken
- Verbot von unsicheren Sprachmerkmalen
- Förderung der Code-Verständlichkeit
- erleichterte Testbarkeit
- Verfahren zur Code-Dokumentation

Die Programmierung einer Sicherheitsapplikation erfolgt in POUs, GVLs und dem Task-Objekt.

In den POUs wird der Programmcode in der IEC 61131-3-Sprache FUP (Funktionsplan) implementiert.

FUP zeichnet sich durch Eindeutigkeit, leichtes Erkennen von Programmierfehlern und klaren Datenfluss aus.

Die Bedienoberfläche und Handhabung des SafeLogic-FUP-Editors entspricht IndraWorks-Standard.



Die für SafeLogic-FUP spezifischen Befehle werden bei den jeweiligen Sprachelementen beschrieben.



Häufig verwendete Symbolik:

Eingabehilfe: das Symbol  , auch Ellipses genannt, kennzeichnet eine Schaltfläche, bei deren Aktivierung sich das Dialogfenster **Eingabehilfe** öffnet.

Sprachumfang der Sicherheitsprogrammierung mit IndraWorks Safety Manager

Der Sprachumfang von FUP wird entsprechend der in PLCopen definierten Sprachteilmengen Basic und Extended eingeschränkt.

Die entsprechende Auswahl für den Sprachumfang Basic oder Extended wird beim Neuanlegen einer POU (Programm oder Funktionsbaustein) durch den Entwickler festgelegt (siehe "Hinzufügen einer POU" auf Seite 61).

Programmierlevel Basic

Im Basic Level kann durch Verknüpfung der bereits zertifizierten Bausteine der PLCopen-Bibliothek (**SafetyPLCopen**) und der Standard-Bibliothek (**SafetyStandard**) mit relativ geringem Aufwand eine Sicherheitsapplikation implementiert und anschließend verifiziert werden.



Basic Level erlaubt keine Zuweisung auf eine WORD-Variable. Bei Safety-PLCopen-FBs ist daher eine Zuweisung des "DiagCode" nicht möglich.

Programmierlevel Extended

Extended Level bietet dem Entwickler zusätzliche Operatoren (boolesche, mathematische, und andere) und bedingte Sprünge/Returns, um umfangreichere Sicherheitsapplikationen zu erstellen. Sie erfordern im Anschluss an die Entwicklung einen entsprechend aufwendigeren Verifikationsprozess.



Bei der Verwendung von PLCopen-Bausteinen läuft die Anlage nach einem Fehlerzustand (Kommunikationsfehler) erst nach Betätigen von Reset wieder an.

Werden keine PLCopen-Bausteine verwendet, so muss dieses Verhalten durch die Applikation selbst implementiert werden.

7.2 Programmierrichtlinien

7.2.1 Übersicht

- Automatisch geprüfte Programmierrichtlinien (formal und sprachbezogen) (siehe Automatisch geprüfte Programmierrichtlinien auf Seite 82)
- Optionale automatisch geprüfte Programmierrichtlinien (siehe Automatische Prüfung der Programmierrichtlinien auf Seite 169)
- Manuell zu prüfende Programmierrichtlinien (siehe Manuelle Prüfung der Programmierrichtlinien auf Seite 170)
- Diese Programmierregeln entsprechen den "General Rules for Safety-Related Function Blocks" der PLCopen. Sie gelten für PLCopen-konforme Funktionsbausteine (siehe Allgemeine Regeln für sicherheitsgerichtete Funktionsbausteine auf Seite 293).
- Empfehlungen zur Dokumentierung des Codes (siehe Empfehlungen zur Dokumentierung des Codes auf Seite 85)

Beim Übersetzungslauf werden die Programmierrichtlinien der einzelnen Sprachelemente überprüft und bei Missachtung Fehler bzw. Warnungen ausgegeben.

Erläuterungen zu diesem Thema befinden sich in Safety-Fehlermeldungen auf Seite 635.

7.2.2 Automatisch geprüfte Programmierrichtlinien

Automatische geprüfte Programmierrichtlinien

Folgende auf der PLCopen basierende Programmierrichtlinien werden von SafeLogic automatisch geprüft:

- Sprachteilmenge der Programmierlevel Basic und Extended
- Verknüpfungsregeln für SAFEBOOL-Daten und analoge Verknüpfungsregeln für die anderen SAFExxx-Datentypen

Zusätzliche automatisch geprüfte formale Programmierrichtlinien

- Deklaration von Datenvariablen nur mit explizitem Initialwert

  (Ausnahme: FB-Instanzen und External-Deklarationen ohne Initialwert)
- Lesezugriff auf Ausgänge einer FB-Instanz nur in oder nach ihrem Aufruf
- Zuweisung an Ausgangsvariablen nur an einer Stelle in der Applikation
- Keine lokale Variable mit gleichem Namen wie eine globale Variable

Optional automatisch geprüfte Programmierrichtlinien

Zusätzlich können durch den Anwender Einstellungen für die Überprüfung formaler Programmierrichtlinien vorgenommen werden. Diese sind im Dialog **Eigenschaften** in der Registerkarte **Safety** des Safety-Applikationsobjekts einstellbar.

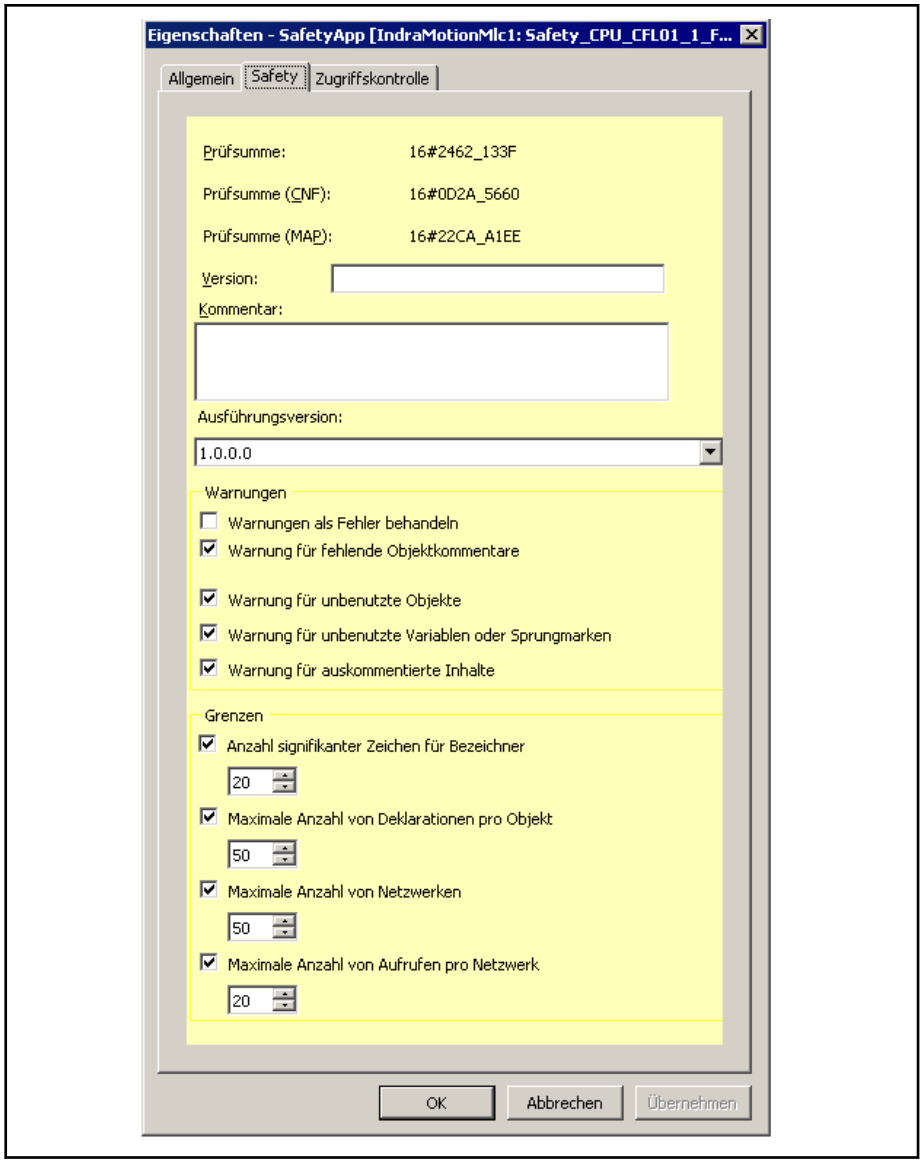


Abb. 7-1: Eigenschaften des Safety-Applikationsobjekts, Registerkarte 'Safety'

Warnungen:

- **Warnungen als Fehler behandeln:** Treten beim Übersetzen der Sicherheitsapplikation Warnungen auf, werden diese wie Fehler behandelt, was zur Folge hat, dass ein Download auf die Sicherheitssteuerung nicht möglich ist
- **Warnung für fehlende Objektkommentare:** Objektkommentare für POUs und die Sicherheitsapplikation

  (weitere Details zur Kommentierung siehe Kommentierung auf Seite 170)
- **Warnung für unbenutzte Objekte:** Keine Objekte, die nicht verwendet werden

  Das bedeutet:

  – jedes Programm der Sicherheitsapplikation wird in der Safety Task aufgerufen

  – jeder Funktionsbaustein der Sicherheitsapplikation wird verwendet

d. h. er wird in einer GVL oder einem Programm instanziiert, oder er wird in einem FB instanziiert, der selbst verwendet wird
  - aus jeder GVL wird eine Variable verwendet
  - aus jedem logischen Gerät wird eine der impliziten Variablen verwendet
- **Warnung für unbenutzte Variablen oder Sprungmarken:** Keine Variablen, die nicht verwendet werden

  Das bedeutet:
  - jede FB-Instanz hat einen Aufruf
  - jede Konstante, jeder Eingang, jede Eingangsvariable hat einen Lesezugriff
  - jede globale Variable, jede lokale Variable hat einen Schreib- und einen Lesezugriff
  - jeder Ausgang und jede Ausgangsvariable hat einen Schreibzugriff

  Im Extended Level: Keine Sprungmarken, die nicht verwendet werden

  Das bedeutet:
  - jede definierte Sprungmarke ist Ziel eines bedingten Sprungs
- **Warnung für auskommentierte Inhalte:** Keine Objektinhalte, die auskommentiert sind
- **Anzahl signifikanter Zeichen für Bezeichner:** Begrenzung der Anzahl signifikanter Zeichen von Bezeichnern
- **Maximale Anzahl von Deklarationen pro Objekt:** Begrenzung der Anzahl der Deklarationen in einem Objekt (POU, GVL)
- **Maximale Anzahl von Netzwerken:** Begrenzung der Anzahl der Netzwerke in einer POU (Ausnahme: wiederverwendete validierte POUs)
- **Maximale Anzahl von Aufrufen pro Netzwerk:** Begrenzung der Anzahl der Aufrufe in einem Netzwerk (Ausnahme: wiederverwendete validierte POUs)

7.2.3 Manuell zu prüfende Programmierrichtlinien

Siehe "Verifikation", Manuelle Prüfung der Programmierrichtlinien auf Seite 170.

7.2.4 Allgemeine Regeln für sicherheitsgerichtete Funktionsbausteine

Diese Programmierregeln entsprechen den "General Rules for Safety-Related Function Blocks" der PLCopen. Sie gelten für PLCopen-konforme Funktionsbausteine (siehe Allgemeine Regeln für sicherheitsgerichtete Funktionsbausteine auf Seite 293).

7.2.5 Defensives Programmieren

Verschiedene Prüfungen von Signalen sind gefordert. Prüfungen für E/A-Geräte und für Querkommunikation sind in Kap. 7.6 "E/A-Zugriff" auf Seite 113 und Kap. 7.5.4 "Safety Netzwerkvariablenliste (NVL)" auf Seite 91 aufgeführt.

Plausibilität



Plausibilitätsprüfungen zur defensiven Programmierung sollten programmiert werden (empfohlen in ISO 13649). Zum Beispiel sollte geprüft werden, ob Kombinationen von Eingangssignalen, von Signalen und Zuständen/Zeiten eine physikalisch unmögliche Situation darstellen.

Es muss keine Überwachung von Datenintegrität (gefordert von ISO 13849 und IEC 62061) und Kontroll- und Datenfluss (gefordert von IEC 62061) programmiert werden. Dies erfolgt durch das System.

Basic Level

Im Basic Level muss keine Grenzwertprüfung (Gefordert von IEC 62061) programmiert werden. Diese erfolgt in den vordefinierten Funktionsbausteinen.

Grenzwerte

Extended-Level: Für POUs, die numerische Werte verarbeiten gilt: "Vernünftige" Grenzwertprüfungen von Eingangssignalen und Eingangsvariablen (VAR_INPUT) müssen programmiert werden (gefordert von IEC 62061).

7.2.6 Empfehlungen zur Dokumentierung des Codes

Unabhängig von der zum Zertifizieren zugrunde gelegten Norm wird die Einhaltung folgender Richtlinien empfohlen:

- Die Dokumentation muss vollständig, verfügbar, lesbar und verständlich sein.
- Alle Änderungen über alle Lebenszyklen müssen dokumentiert werden.
- *Die Projektdokumentation muss enthalten*
  - juristische Person (Firma)
  - Funktionsbeschreibung der Anforderungen an das Projekt
  - E/A-Beschreibung
  - Version der verwendeten Funktionsbausteinbibliothek
- *Jedes Programm, jeder Funktionsbaustein muss über folgende Informationen verfügen:*
  - Autor
  - Datum der Erstellung
  - Datum der Freigabe
  - Version
  - Versionshistorie
  - ausreichende Kommentierung der Netzwerke
  - ausreichende Kommentierung der Deklarationszeilen

Kommentierung

IndraWorks Safety Manager bietet folgende Möglichkeiten, um die Sicherheitsapplikation, ihre Objekte und deren Inhalte zu kommentieren:

- Kommentarfeld zu jeder POU der Sicherheitsapplikation (Dialog **Eigenschaften** der POU)
- Kommentarfeld zu jeder GVL der Sicherheitsapplikation (Dialog **Eigenschaften** der GVL)
- Kommentarfeld zur gesamten Sicherheitsapplikation (Dialog **Eigenschaften** von **Safety-App**)

  Hier kann die Quellcode-Information eingegeben werden: Firma, Autor, Beschreibung, Eingänge und Ausgänge und Konfigurationsmanagement-Geschichte.

- Kommentarfeld zu jedem logischen E/A (Dialog **Eigenschaften** des logischen E/As)
- Kommentarfeld zum Task-Objekt
- Kommentar zu jedem FUP-Netzwerk (optional)
- Netzwerktitel (optional)
- In POUs und GVLs: Kommentar zu jeder anwenderdefinierten Variablendeklaration (optional)

Die Anzeige von Netzwerktitel und Kommentar von Netzwerk und Variablen kann optional eingestellt werden (Dialog **Optionen** des Menüs **Tools**).

Es wird empfohlen, Netzwerktitel und Netzwerkkommentar anzuzeigen.

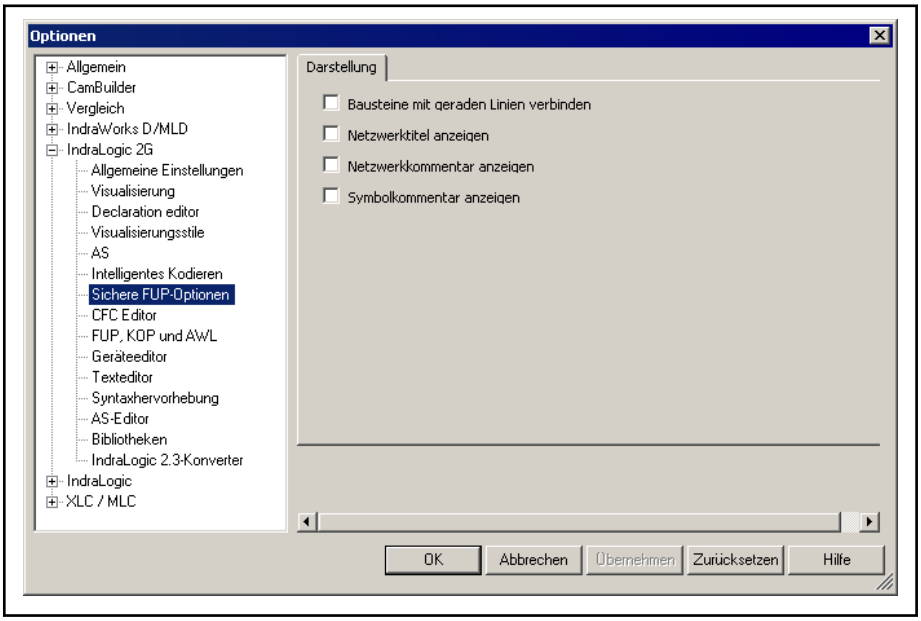


Abb. 7-2: Einstellungen zur Anzeige von Kommentaren und Titel der Netzwerke

Der Checker prüft optional, ob bei der Applikation und bei den POUs Kommentare vorhanden sind.

Ob diese oder die anderen Kommentare den Programmierrichtlinen genügen, muss manuell verifiziert werden.

Einstellungen zur Anzeige von Warnungen bei fehlenden Kommentar der Sicherheitsapplikation oder einer POU werden im **Eigenschaften**-Dialog der Sicherheitsapplikation vorgenommen.

7.3 Datenfluss sicherheitsgerichteter Signale

Der Datenfluss sicherheitsgerichteter Signale der FUP-Programmierung wird in SafeLogic folgendermaßen hervorgehoben:

- Literale und konstant deklarierte Variablen werden gelb hinterlegt
- SAFExxx-Variablen werden gelb hinterlegt
- Datenfluss von SAFE-Werten in SAFE-Variablen und in Eingänge von Operatoren und Bausteinen hinein wird durch dicke gelbe Linien dargestellt
- Funktionsbausteine werden gelb dargestellt, wenn sie mindestens einen SAFE-Ausgang haben

- *Operator-Aufrufboxen werden gelb ausgefüllt, wenn der Ausgang SAFE ist. Dies ist unter folgenden Bedingungen der Fall:*
  - bei Operator AND: der Ausgang ist SAFE, wenn mindestens 1 Eingang SAFE ist
  - bei allen anderen Operatoren, inklusiv Konvertierungen: der Ausgang ist SAFE, wenn alle Eingänge SAFE sind

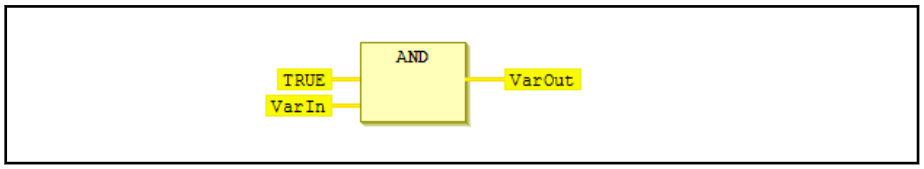


Abb. 7-3: Beispiel für sicheren Datenfluss: Operator AND mit Literal: TRUE, SAFE-Variablen: VarIn und VarOut, Operator AND

7.4 IEC 61131-3-Konformität

Zusammenfassung der Erweiterungen zur IEC 61131-3

In SafeLogic sind zur Programmierung von Sicherheitssteuerungen einige wenige Erweiterungen der IEC 61131-3 umgesetzt:

Bereich	Erweiterungen der IEC	Begründung
Datentypen	sicherheitsgerichtete Datentypen SAFEBOOL, SAFEINT, etc. für alle unterstützten IEC Datentypen	Für PLCopen: Datentyp zur Unterscheidung sicherer Signale gemäß PLCopen
generische Datentypen/ Funktionen	Jeder Datentyp SAFExxx wird neben X in der Hierarchie der generischen Datentypen eingereiht. Das heißt, die generischen Funktionen (AND, ADD, SEL, EQ, etc.), die u. a. auf Typ X definiert sind, sind analog auch auf Typ SAFEX definiert.	Für PLCopen: Generische Funktionen auf Typ X machen genauso Sinn auf Typ SAFExxx. Es müssen keinen neuen Funktionen "erfunden" werden.
	Zusätzliche Varianten der Funktion "AND", die nicht dem generischen Typschema entspricht: • AND: BOOL x SAFEBOOL → SAFEBOOL • AND: SAFEBOOL x BOOL → SAFEBOOL	Für PLCopen: "confirmation functionality" bzw. "enabeling function"
	Konvertierungsfunktionen A_TO_B sind generisch bzgl. der SAFE-Qualifikation: • A_TO_B : A → B • A_TO_B: SAFEA → SAFEB	Analogie zu generischen Funktionen: ADD auf INT und auf TIME wird als eine Funktion verstanden, weil das gleiche passiert; erspart dem Anwender, unterschiedliche Funktionsnamen hinzuschreiben (ADD_INT, ADD_TIME). Auch bei INT_TO_BOOL auf INT/BOOL wie auf SAFEINT/ SAFEBOOL passiert das Gleiche.

Bereich	Erweiterungen der IEC	Begründung
implizite Konvertierungen	Daten vom Typ SAFExxx können auf Variablen oder Eingänge vom Typ X zugewiesen werden ("SAFE-Polymorphie"). *U. a. ist auf diese Weise der Aufruf der generischen Funktion ADD mit einem INT- und einem SAFEINT-Wert möglich:* • SAFEBOOL → BOOL • SAFEINT → INT • SAFEDINT → DINT • SAFETIME → TIME • SAFEBYTE → BYTE • SAFEWORD → WORD • SAFEDWORD → DWORD	Für PLCopen: SAFExxx-Daten und xxx-Daten sind nicht andersartige Daten, sondern Daten mit unterschiedlicher Integrität. Daten höherer Integrität können verwendet werden, wo niedrigere Integrität ausreicht.
POUs	*POUs können qualifiziert werden, um ihren Sprachumfang oder ihre Verwendung einzuschränken:* • Level: Basic, Extended, External • Attribut **Einmaliger Aufruf** (siehe "Objekteigenschaften einer POU" auf Seite 62)	Level: zur expliziten Unterscheidung der PLCopen Programmierlevel. Einmaliger Aufruf: Kennzeichnet FBs, für welche die PLCopen-Regel "Einmaliger Aufruf" gilt.

Tab. 7-1: Sicherheitstechnische Erweiterungen zur IEC 61131-3

Bereich	Erweiterung	Begründung
implizite Konvertierung	*Daten vom Typ INT bzw. SAFEINT können auf Variablen oder Eingänge vom Typ DINT bzw. SAFEDINT zugewiesen werden ("INT-Polymorphie"):* • INT → DINT • SAFEINT → SAFEDINT	Der Wertebereich von INT bzw. SAFEINT ist enthalten im Wertebereich von DINT bzw. SAFEDINT.
FB-Typen	Statt den Standard-Funktionsbausteinen: SF_XXX, Varianten mit SAFE-Varianten der Ein/Ausgänge (bis auf Eingänge mit Reset-Semantik: RESET, LOAD bzw. IN)	Notwendig für Verarbeitung sicherer Signale mit Standard-FBs.

Tab. 7-2: Zusätzliche Erweiterungen in SafeLogic

7.5 Sprachelemente

7.5.1 Überblick über die Sprachelemente

Zur Implementierung des Programmcodes einer Sicherheitsapplikation dienen generell folgende Objekte und deren Elemente:

Objekte und deren Elemente:

- GVL, Globale Variablenlisten (siehe Globale Variablenliste (GVL) auf Seite 91),
- NVL, Netzwerkvariablenlisten, Querkommunikation (siehe Safety Netzwerkvariablenliste (NVL) auf Seite 91)
- POUs (siehe POUs auf Seite 96),
  - Netzwerke (siehe Überblick Netzwerke auf Seite 97),
  - Variablen (siehe Grundlegendes zu Variablen auf Seite 100),
  - Operatoren (siehe Variablen für Extended-POUs auf Seite 104),

  - Zuweisungen (siehe FUP-Zuweisungen auf Seite 110),
  - Sprünge/Returns (siehe Sprung/Return und Sprungmarke auf Seite 111),
  - FB-Aufrufe (siehe FB-Aufrufe auf Seite 112),
- Task (siehe Taskkonfiguration auf Seite 120),
  - Programmliste
- Logische E/As (siehe E/A-Zugriff auf Seite 113),
  - Mapping-Variablen
- Bausteine der Sicherheitsbibliotheken, die im Bibliotheksverwalter angezogen werden.

PLCopen **PLCopen, Technical Committee 5: Safety Software (Technical Specification), Part 1: Concepts and Function Blocks, Version 1.0 (2006-1)**	**Abweichung in Basic/Extended POUs**
Standard-FBs erlaubt	nur als Safety-Varianten vorhanden (SF_TON statt TON, etc)
Datentyp REAL/LREAL vorgesehen	Datentyp REAL/LREAL wird nicht unterstützt
(SAFE)TIME in Extended Level: nur als interne Variable oder konstanter Input für FBs	In Extended Level: keine Einschränkung bei Deklarationen, auch möglich als Datentyp für physikalische Ein- und Ausgänge
(SAFE)WORD in Extendend Level: nur als interne Variable oder als Output für Diagnosezwecke	In Extended Level und GVLs: keine Einschränkung bei Deklarationen, auch möglich als Datentyp für physikalische Ein- Und Ausgänge
kein (SAFE)BYTE, (SAFE)DWORD	In Extended Level und GVLs: keine Einschränkung bei Deklarationen, auch möglich als Datentyp für physikalische Ein- und Ausgänge
kein VAR_EXTERNAL in Basic Level	in Programmen: VAR_EXTERNAL erlaubt für Kanalvariablen und Stack-Instanzen VAR_EXTERNAL CONSTANT erlaubt für globale Konstanten
SAFEBOOL	SAFE-Varianten für alle unterstützten Datentypen
LD	nicht unterstützt
kein mehrfacher Aufruf der selben FB-Instanz	nur im Fall von FBs mit Call-Once-Qualifizierung (alle PLCopen-FBs fallen darunter); Implizite Call-Once "Vererbung": FB mit Call-Once-Instanz ist selbst Call-Once. Instanzen anderer FBs können mehrfach oder null mal aufgerufen werden.

PLCopen PLCopen, Technical Committee 5: Safety Software (Technical Specification), Part 1: Concepts and Function Blocks, Version 1.0 (2006-1)	Abweichung in Basic/Extended POUs
keine FB-Deklaration Features nach IEC 61131, 2003-1, Programmable controllers, Part 3: Programming languages, Tab.33: - kein VAR_EXTERNAL von globalen FBs	in Basic-Level Programmen: erlaubt für Stack-Instanzen in Extended Level Programmen: erlaubt für nicht-PLCopen FBs (genauer: für FBs ohne Call-Once-Qualifizierung)
keine FB-Declaration Features nach IEC 61131, 2003-1, Programmable controllers, Part 3: Programming languages, Tab.33: - kein VAR_EXTERNAL CONSTANT in einer FB-POU	erlaubt für globale Konstanten (VAR_GLOBAL CONSTANT)

Tab. 7-3: Abweichungen der Sprachelemente von PLCopen Safety

7.5.2 Regeln für Bezeichner von Safety-Objekten und -Variablen

- Namen für POUs, GVLs, E/A-Module, Variablen, Netzwerke (Labels), etc. sind Bezeichner (Identifier) im Sinne der IEC 61131-3. (Folge aus Buchstaben, Ziffern und Unterstrich, wobei das erste Zeichen keine Ziffer und das letzte Zeichen kein Unterstrich sein darf, und zwei Unterstriche nicht direkt aufeinander folgen dürfen.)
- Bezeichner, die mit Unterstrich beginnen, sind reserviert und dürfen in Basic/Extended POUs und GVLs nicht vorkommen.
- Die in der IEC 61131-3 (2003-1, Programmable controllers, Part 3: Programming languages - Annex C) aufgeführten IEC-Schlüsselwörter dürfen nicht als Bezeichner verwendet werden.
- Die über die IEC hinausgehenden Schlüsselwörter, die von IndraWorks-Standard reserviert sind, dürfen nicht als Bezeichner verwendet werden.
- *Als SafeLogic-spezifische Schlüsselwörter sind reserviert und dürfen nicht als Bezeichner verwendet werden:*
  - IOAPI, IOIN, IOOUT, SYSONLY
  - SAFEXXX, für alle elementaren Standard-Datentypnamen XXX
  - SF_FB, für alle Standard-Funktionsbausteintypen FB. FBs mit solchen Namen sind nur für externe FBs erlaubt
- Die Namen von Standard-Funktionen, die im Extended Level der PLCopen verboten sind, stehen nicht als Bezeichner zur Verfügung
- Variablennamen müssen innerhalb des Gültigkeitsbereichs (Scope) eindeutig sein

  Dies bedeutet im Einzelnen:
  - Keine zwei globalen Variablen dürfen den gleichen Namen haben
  - Keine zwei POU-weiten Variablen in der gleichen POU dürfen den gleichen Namen haben
  - Es darf keine POU-weite Variable den gleichen Namen haben wie eine globale Variable. Dies ist unabhängig davon, ob die globale Variable mit VAR_EXTERNAL in den Gültigkeitsbereich der POU importiert wird oder nicht

  - Eine Variable darf nicht gleich heißen, wie ein Objekt oder Bibliotheksbaustein der Applikation
- *Namen von Sprungmarken müssen eindeutig sein:*
  - Eine Sprungmarke darf nicht den gleichen Namen haben wie eine POU-weite oder globale Variable
- Das Präfix SF_ ist reserviert für PLCopen-konforme FBs und die Safety-Varianten der Standardfunktionsbausteine der IEC
- Unter den Objekten: POUs, GVLs, E/A-Module, Bibliotheksbausteine, die zur Sicherheitsapplikation gehören, dürfen keine zwei den gleichen Namen haben
- Kein Objekt (POU, GVL, E/A-Modul, Bibliotheksbaustein), das zur Sicherheitsapplikation gehört, darf den gleichen Namen haben, wie eine globale Variable der Applikation - mit einer Ausnahme: Implizite globale Variablen eines E/A-Moduls dürfen genau so heißen, wie das E/A-Modul selbst (aber nicht wie ein anderes E/A-Modul oder Objekt)

7.5.3 Globale Variablenliste (GVL)

Die GVL enthält Variablendeklarationen, die in sämtlichen Programmen der Sicherheitsapplikation verwendet werden können.

Funktionsbausteine (FBs) haben keinen Zugriff auf globale Variablen.

Um eine globale Variable in einem Programm verwenden zu können, muss sie dort nochmals als VAR_EXTERNAL deklariert werden.

Die VAR_EXTERNAL-Deklaration wird bei der Verwendung einer globalen Variablen in einem Programm automatisch eingefügt.



Globale Variablen dienen nicht der Programmübersicht und sollten sparsam verwendet werden.

Globale Variablen sind jedoch zum Austausch von Daten zwischen Programmen unerlässlich.

Informationen zur Erstellung von Variablendeklarationen siehe Variablendeklaration, Seite 73.

7.5.4 Safety Netzwerkvariablenliste (NVL)

Die NVL (Safety Netzwerkvariablenliste) enthält Variablendeklarationen, die zu oder von einer anderen Sicherheitssteuerung übertragen werden. Die Variablen der NVL können in sämtlichen Programmen der Sicherheitsapplikation verwendet werden können.

Der Zugriff auf Safety-Netzwerkvariablen erfolgt über die Objekte Safety-Netzwerkvariablenliste (Sender) und Safety-Netzwerkvariablenliste (Empfänger).

Siehe auch

- Objekt 'Safety Netzwerkvariablenliste (Sender)', Seite 663
- Objekt 'Safety Netzwerkvariablenliste (Empfänger)', Seite 665

Die Safety-Netzwerkvariablen können wie globale Variablen als VAR_EXTERNAL deklariert in der Sicherheitsapplikation verwendet werden.

Auf Senderseite sind Netzwerkvariablen Ausgänge der Applikation. Das heißt, ihnen muss genau einmal im Zyklus ein Wert zugewiesen werden.

Auf Empfängerseite sind Netzwerkvariablen Eingänge der Applikation. Das heißt, sie dürfen gelesen, aber nicht beschrieben werden.

Wenn die Werte erfolgreich übertragen werden, enthalten die Netzwerkvariablen auf Empfängerseite die Werte der verbundenen Netzwerkvariablen auf Senderseite.

Eine NVL-Sendeliste kann gleichzeitig auf mehrere Empfänger übertragen werden. Die Empfänger müssen sich beim Sender anmelden, um die Variablen empfangen zu können.

Da für jeden konfigurierten Empfänger eine eigene Verbindung angelegt wird, sollten Sie Anzahl der Empfänger auf ein Minimum reduzieren.

Die Verbindungsüberwachung von NVLs erfolgt über die Funktionsbausteine NetVarReceiver und NetVarSender. Die hierfür notwendigen Instanzen werden automatisch angelegt.

Funktionsbausteine (FBs) haben keinen Zugriff auf Netzwerkvariablen.

Um eine Netzwerkvariable in einem Programm verwenden zu können, muss sie dort nochmals als VAR_EXTERNAL deklariert werden.

Die VAR_EXTERNAL-Deklaration wird bei der Verwendung einer Netzwerkvariablen in einem Programm automatisch eingefügt.

Siehe auch:

- NetVarReceiver auf Seite 605
- NetVarSender auf Seite 603
- SafetyNetVar.library auf Seite 599



Die Kommunikation über Netzwerkvariablen belastet das Gesamtsystem. Sie sollte auf ein Minimum beschränkt werden.

Globale Variablen sind jedoch zum Austausch von Daten zwischen Programmen unerlässlich.

Informationen zur Erstellung von Variablendeklarationen siehe Variablendeklaration, Seite 73.

Datentypen, die über Netzwerkvariablen austauschbar sind:

- SAFEBOOL
- SAFEWORD
- SAFEINT

Einschränkungen bei der Update-Rate/Übertragungszeit

Die Werte der Safety-Netzwerkvariablen und Empfangsbestätigungen werden immer synchron am Ende eines Applikationszyklus (Output-Phase) verschickt und am Beginn eines Applikationszyklus (Eingangsphase) empfangen. Die Zykluszeiten der Applikationen von Sender und Empfänger und die Übertragungszeit beeinflussen die Aktualisierungsrate der Netzwerkvariablenwerte.

⚠ VORSICHT **Um zu vermeiden, dass zu kurze Signale nicht erkannt oder falsch übertragen werden, dürfen die Signale der Safety-Netzwerkvariablen bei der Übertragung nicht zu kurz sein.**

Sie müssen geeignete Maßnahmen ergreifen, um sicherzustellen, dass die Signale bei der Safety-Netzwerkvariablenliste (Sender) oder bei der Safety-Netzwerkvariablenliste (Empfänger) lange genug anliegen. Eine geeignete Maßnahme ist beispielsweise die Verwendung von Verzögerungsfunktionsbausteinen oder die Kombination von Ausgangswerten rücksetzgeschützter PLCopen-Funktionsbausteine.

- Wenn die Applikationen der Safety-Netzwerkvariablenliste (Sender) und der Safety-Netzwerkvariablenliste (Empfänger) die gleichen Zykluszeiten haben und die Übertragungszeit kleiner als die halbe Zykluszeit ist, muss eine Werteänderung in der Safety-Netzwerkvariablenliste (Sender) 4 Zyklen stehen bleiben, damit sie garantiert in der Safety-Netzwerkvariablenliste (Empfänger) ankommt.

  Übertragungsrate: 1 Werteänderung pro 4 Zyklen des NVL-Senders

- Wenn der Safety-Netzwerkvariablenlisten (Empfänger) eine relativ kleine Zykluszeit hat, darf eine erneute Werteänderung in der Safety-Netzwerkvariablenliste (Sender) nicht im gleichen Zyklus vorgenommen werden. Eine Werteänderung muss mindestens 2 Zyklen im NVL-Sender stehen bleiben.

  Dabei ist die **bestmögliche Übertragungsrate:** 1 Werteänderung pro 2 Zyklen des NVL-Senders. Diese bestmögliche Übertragungsrate wird erreicht, wenn die Zykluszeit des NVL-Senders größer ist als (2 x NVL-Empfängers + 2 x Übertragungszeit).

Bei Kommunikationsproblemen enthalten die Netzwerkvariablen auf Empfängerseite den **Ersatzwert** 0 bzw. False.

Die Empfängerapplikation kann über die Treiberinstanz <NVL-Name> des NVL-Treiberbausteins NetVarReceiver den Status der Verbindung kontrollieren und Kommunikationsfehler bestätigen.

Die Senderapplikation kann über die Treiberinstanz <NVL-Name>

des NVL-Treiberbausteins NetVarSender die Sendebereitschaft steuern bzw. Failsafe-Ersatzwerte aktivieren.

Regeln zur Vermeidung von gefährlichem Undersampling

Beispielregeln für SAFEBOOL-Netzwerkvariablen

- **Regel Wiederanlaufsperre:** Veröffentlichen Sie Netzwerkvariablen, die auf FALSE (nach Failsafe-Prinzip ein gefahranzeigendes oder reaktionsanforderndes Signal) stehen bleiben, bis der Operator die Abwesenheit der Gefahr und die Rücknahme der Sicherheitsreaktion, bzw. das Wiederanlaufen durch eine Signalflanke betätigt hat.
- **Regel Verriegelter Sensor:** Veröffentlichen Sie Variablen, die mit dem Eingangssignal von einem Sensor mit mechanischer Verriegelung, z.B. ein Nothalt-Befehlsgerät, belegt sind. Denn mit dem Rücksetzen der Verriegelung bestätigt der Operateur das Ende der Gefahr.
- **Regel Physikalischen Mindestdauer:** Veröffentlichen Sie Variablen, die eine Prozesswert darstellen (z.B. Schutztür geöffnet), wenn dieser physikalisch bedingt erst ab einer gewissen Mindestdauer (Zeit zum Passieren der Schutztür) eine gefährliche Situation darstellt, und wenn diese Mindestdauer länger als die Watchdog-Zeit der Verbindung zum Sensor plus der längsten Watchdog-Zeit aller Empfänger ist.

Beispielregeln für SAFEINT-Netzwerkvariablen

- **Regel Entscheidung im Sender:** Veröffentlichen Sie analoge Prozesswerte (Geschwindigkeit, Position, u.ä.) nicht zu dem Zweck, dass Empfänger die Werte überwachen und gegebenenfalls eine Sicherheitsfunktion auslösen. Vielmehr überwachen Sie diese analogen Prozesswerte selbst und melden Sie das Erreichen bestimmter Werte an die Empfänger. Das heißt, veröffentlichen Sie statt dessen digitale Netzwerkvariablen, von denen die Empfänger ihre Reaktion abhängig machen können.

  Bei diesen Variablen kann Undersampling mit den Regeln für SAFEBOOL-Netzwerkvariablen vermieden werden.
- **Regel Extremwertermittlung im Sender:** Veröffentlichen Sie zu überwachende analoge Prozesswerte (Geschwindigkeit, Position, Temperatur, etc.) nicht direkt. Bestimmen Sie vielmehr deren größten und ggf. kleinsten erreichten Werte und veröffentlichen Sie nur diese Werte.

  Empfänger können dann selbständig gegen Überschreitungen von Grenzen (nach oben oder unten) prüfen und Sicherheitsfunktionen auslösen, ohne Extremwerte wegen Undersampling zu verpassen. Sehen Sie den Neustart der Extremwertermittlung durch den Operateur nach einer Sicherheitsreaktion des Empfängers wegen Grenzüberschreitung vor. Dann kann der Empfänger mit aktuellen Prozesswerten neu beginnen.
- **Regel Nonreaktive Signale:** Veröffentlichen Sie Werte, die Empfängern nicht zur Erkennung von Gefahren dienen, d.h. keiner ihrer Werte stellt ein nach dem Failsafe-Prinzip gefahranzeigendes oder reaktionsanforderndes Signal dar. Es gibt also keinen bestimmten Wert der Netzwerkvariable, der im Empfänger eine Sicherheitsfunktion auslösen würde, und damit keinen Wert, dessen Überspringen (Undersampling) das Auslassen einer erforderlichen Reaktion bewirken würde.

Austausch von Informationen innerhalb eines Projektes und über Projektgrenzen

Anlegen einer Safety-Netzwerkvariablenliste (Sender) in der Sender-Sicherheitssteuerung und einer Safety-Netzwerkvariablenliste (Empfänger) in der Empfänger-Sicherheitssteuerung

Voraussetzung: Ihr Projekt enthält 2 Sicherheitssteuerungen mit Sicherheitsapplikationen den SafetyApp1 und SafetyApp2 das Projekt ist geöffnet.

1. Selektieren Sie in Ihrer Sender-Sicherheitssteuerung das Safety-Applikationsobjekt SafetyApp1 und wählen im Kontextmenü den Befehl **Objekt hinzufügen ▸ Safety-Netzwerkvariablenliste (Sender)**.

   ⇒ Der Dialog Safety-Netzwerkvariablenliste (Sender) hinzufügen öffnet sich.
2. Geben Sie eine Namen und einen Kommentar für Ihre Safety-Netzwerkvariablenliste (Sender) ein und klicken Sie auf Hinzufügen.

   ⇒ Die Safety-Netzwerkvariablenliste (Sender) wird unter der Sicherheitsapplikation hinzugefügt und im Editor geöffnet.

   Die Eingabefelder Safety Adresse dieser Variablenliste und Maximalanzahl von Empfängern sind mit Standardwerten belegt, die Sie nach Bedarf verändern können.
3. Wählen Sie im Projektexplorer das gerade neu hinzugefügte Objekt Safety-Netzwerkvariablenliste (Sender) aus und wählen Sie im Kontextmenü den Befehl **Eigenschaften**.
4. Geben Sie im Dialog Eigenschaften, Registerkarte Safety eine Objektversionsnummer im Eingabefeld Version ein und klicken Sie auf OK.

5. Wählen Sie im Deklarationsteil des Editors der Safety-Netzwerkvariablenliste (Sender) den Kontextmenübefehl **Variablendeklaration** einfügen.

   ⇒ Der Dialog Safety-Variable deklarieren erscheint.

6. Deklarieren Sie Ihre Netzwerkvariable und klicken Sie auf die Schaltfläche OK.

7. Selektieren Sie in Ihrer Empfänger-Sicherheitssteuerung das Safety-Applikationsobjekt SafetyApp2 und wählen Sie den Kontextmenübefehl **Objekt hinzufügen ▸ Safety-Netzwerkvariablenliste (Empfänger)**.

   ⇒ Der Dialog Safety-Netzwerkvariablenliste (Empfänger) hinzufügen öffnet sich.

8. Geben Sie einen Namen und einen Kommentar für Ihre Safety-Netzwerkvariablenliste (Empfänger) ein klicken Sie auf Hinzufügen.

   ⇒ Die Safety-Netzwerkvariablenliste (Empfänger) wird unter der Sicherheitsapplikation hinzugefügt und im Editor geöffnet.

9. Wählen Sie in der Auswahlliste Zugeordneter Sender die Safety-Netzwerkvariablenliste (Sender), in der Sie im vorherigen Schritt die Netzwerkvariable deklariert haben.

   ⇒ IndraLogic/ SafeLogic zeigt in der Registerkarte Safety-Konfiguration im Feld Safety-Adresse des Senders und die Objektversion des Senders die Informationen des zugeordneten Senders und in der Variablenliste die Safety Netzwerkvariable an, die Sie in der Sender-NVL deklariert haben.

10. Übersetzen Sie beide Applikationen mit dem Befehl **Erstellen ▸ Übersetzen**.

    ⇒ Im Meldungsfenster (**Ansicht ▸ Meldungen**) sehen Sie, ob Sie den Safety-Netzwerkvariablenaustausch fehlerfrei konfiguriert haben.

Verfahren zur Querkommunikation zwischen Projekten

Voraussetzung: Aus einem bestehenden Projekt (Projekt1) mit einer Sicherheitssteuerung sollen einige Variablen als Netzwerkvariablen veröffentlicht werden. Diese Variablen sollen in der Sicherheitssteuerung eines anderen Projekts (Projekt2) verwendet werden.

1. Selektieren Sie im Projektexplorer in der Sender-Sicherheitssteuerung von Projekt1 das Safety-Applikationsobjekt SafetyApp und wählen Sie im Kontextmenü den Befehl **Objekt hinzufügen ▸ Safety-Netzwerkvariablenliste (Sender)**.

2. Fügen Sie der Safety-Netzwerkvariablenliste (Sender) die Variablen hinzu, die an die Empfänger veröffentlicht werden sollen.

3. Passen Sie gegebenenfalls die Einstellungen der Safety-Netzwerkvariablenliste (Sender) in den Registerkarten Safety-Konfiguration und SPS-Netzwerk an.

4. Passen Sie Ihre Sicherheitsapplikation von Projekt1 derart an, dass die Netzwerkvariablen mit den gewünschten internen Variablenwerten beschrieben werden.

5. Exportieren Sie Projekt1 mit den Objekten Safety-Netzwerkvariablenliste (Sender) und Safety Task mit dem Befehl **Projekt ▸ Exportieren** unter dem Namen `Projekt1.export`

6. Öffnen Sie das Projekt Projekt2 und wählen Sie den Befehl **Projekt ▸ Importieren** und importieren Sie `Projekt1.export` als eigene Steuerung ins Projekt2. Wählen hierzu Sie im Dialog Import die Steuerung und die Objekte Safety-Netzwerkvariablenliste (Sender) und Safety Task aus.

7. Fügen Sie der Sicherheitsapplikation des Empfängerprojekts Projekt2 ein Objekt Safety-Netzwerkvariablenliste (Empfänger) hinzu.
8. Öffnen Sie die Safety-Netzwerkvariablenliste (Empfänger) und wählen Sie in der Auswahlliste Zugeordneter Sender den zuvor importierten Sender aus.

   ⇒ IndraLogic/ SafeLogic zeigt im Objekt Safety-Netzwerkvariablenliste (Sender), Registerkarte Safety- Konfiguration in den Feldern Safety-Adresse und Objektversion des Senders die Informationen des zugeordneten Senders.

   In der Variablenliste werden Safety-Netzwerkvariablen angezeigt, die Sie in der Safety-Netzwerkvariablenliste (Sender) von Projekt1 deklariert haben.
9. Passen Sie gegebenenfalls die Einstellungen der Safety-Netzwerkvariablenliste (Empfänger) in den Registerkarten Safety-Konfiguration und SPS-Netzwerk an.
10. Übersetzen Sie die Sicherheitsapplikation mit der Safety-Netzwerkvariablenliste (Empfänger) mit dem Befehl **Erstellen ▸ Übersetzen**.

    ⇒ Im Meldungsfenster (**Ansicht ▸ Meldungen**) sehen Sie, ob Sie den Safety-Netzwerkvariablenaustausch fehlerfrei konfiguriert haben.

7.5.5 POUs

Der Programmcode einer Sicherheitsapplikation wird in POUs erstellt. (Hinzufügen einer neuen POU siehe Kap. 6.8 "Programmorganisationseinheiten (POUs)" auf Seite 60).

Eine Sicherheitsapplikation muss mindestens ein nicht auskommentiertes Programm enthalten.

Der Editor einer POU besteht aus einem Variablendeklarationsteil, in dem die Variablen deklariert werden und einem Implementierungsfenster, in dem der Programmcode erstellt wird.

Im Implementierungsfenster wird der Programmcode in aufeinander folgenden Netzwerken erstellt, die in aufsteigender Reihenfolge abgearbeitet werden.

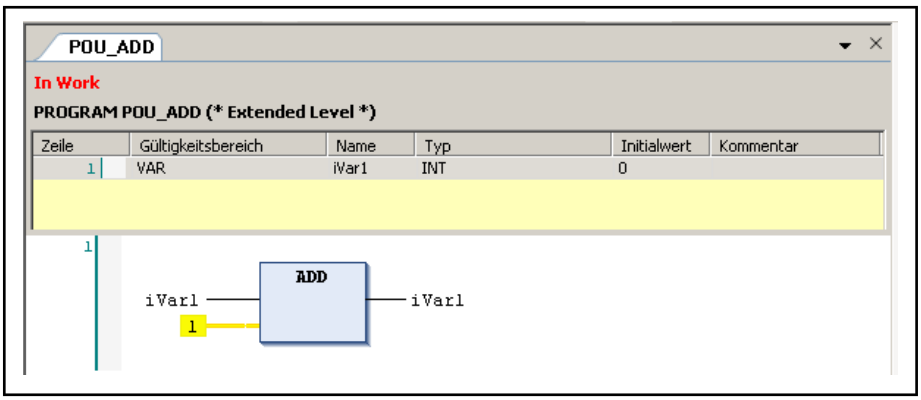


Abb. 7-4: Beispiel: POU-Editor: Deklarationsteil mit einer Variablendeklaration und Implementierungsteil mit einem Netzwerk

Details zum POU-Editor siehe Kap. "Grundlegendes zu Variablen" auf Seite 100.

☞ Zur Aktivierung von Befehlen bei der Programmierung im FUP stehen dem Entwickler verschiedene Bedienfunktionen zur Verfügung. Bei Verfahren und konkreten Beispielen wird in diesem Handbuch meist die Bedienung über das Kontextmenü beschrieben.

Befehle, die im aktuellen Kontextmenü nicht aufgelistet oder nicht aktivierbar sind, können an der momentanen Cursor-Position auch nicht ausgeführt werden.

Standardbefehle

Es können sowohl im Implementierungsteil, als auch im Variablendeklarationsteil des FUP-Editors folgende Standard-Befehle von IndraWorks ausgeführt werden:

- Kopieren
- Löschen
- Ausschneiden
- Einfügen
- Rückgängig machen
- Wiederherstellen

Des Weiteren werden folgende IndraWorks-Befehle unterstützt:

- Gehe zur Definition
- Querverweise ausgeben
- Eingabehilfe
- Suchen & Ersetzen
- Alles selektieren
- Lesezeichen

Die Befehle verhalten sich wie in Standard-IndraWorks.

☞ Für detaillierte und grundlegende Informationen wird auf die Online-Hilfe von Standard-IndraWorks verwiesen.

7.5.6 Netzwerke

Überblick Netzwerke

Programmiereinheiten in FUP sind in aufsteigend durchnummerierte Netzwerke untergliedert, die grafisch dargestellte Elemente, wie Operanden, Bausteinaufrufe, Zuweisungen, Sprünge oder Sprungmarken enthalten können.

Die Netzwerke werden in aufsteigender Reihenfolge abgearbeitet.

Innerhalb eines Netzwerks muss der Programmcode eine Baumstruktur aufweisen, dies bedeutet: keine Parallelverschaltung, keine Aufsplittung, keine expliziten Feedbackloops (Rückkopplungen).

Die logischen Elemente eines Netzwerks werden durch Linien zu einer mit der Wurzel rechts liegenden baumartigen Formation (kurz Baum) verbunden, wobei die Boxen in diesem Baum als Knoten fungieren.

Ausnahme:

Mehrfachzuweisungen sowie Sprung- und Return-Anweisungen fächern diesen Baum in entgegengesetzter Richtung auf. Die Elemente werden vom Editor automatisch, entsprechend ihrer Einfügeposition verbunden, eine freie Platzierung ist nicht möglich.

Grundlegende Funktionalitäten des FUP-Editors

Im IndraWorks Safety Manager werden folgende in IndraWorks-Standard unterstützten FUP-Sonderoperatoren nicht unterstützt:

- Flankenerkennung bei Zuweisungen (auf Eingänge und Variablen)
- Set/Reset
- EN/ENO-Parameter



Für detaillierte und grundlegende Informationen zur Bedienung des FUP-Editors wird auf die Online Hilfe von IndraWorks-Standard verwiesen.

Befehle für FUP-Netzwerke

Befehle für Netzwerke (im Kontextmenü aktivierbar)

- **Netzwerk einfügen**,
- **Netzwerk einfügen (unterhalb),**
- **Kommentierung ein/aus,**
- **Bausteinaufruf einfügen** (siehe Operatoren auf Seite 106 und FB-Aufrufe auf Seite 112),
- **Leeren Baustein einfügen** (siehe Operatoren auf Seite 106 und FB-Aufrufe auf Seite 112),
- **Bausteineingang einfügen** (siehe Erweiterbare Operatoren auf Seite 106),
- **Zuweisung einfügen** (siehe FUP-Zuweisungen auf Seite 110),
- **Sprungmarke einfügen** (siehe Sprung/Return und Sprungmarke auf Seite 111),
- **Sprung einfügen** (siehe Sprung/Return und Sprungmarke auf Seite 111),
- **Return einfügen** (siehe Sprung/Return und Sprungmarke auf Seite 111),
- **Weiterverschaltung festlegen** (siehe Weiterverschaltung festlegen auf Seite 113),
- **Nicht verwendete FB-Aufruf-Parameter entfernen** (siehe Nicht verwendete FB-Aufruf-Parameter entfernen auf Seite 113),
- **Parameter aktualisieren** (siehe FB-Aufrufe auf Seite 112).

Netzwerke einfügen

Durch Aktivierung des Kontextmenü-Befehls **Netzwerk einfügen** wird ein Netzwerk im Implementierungsteil des Editors eingefügt. Existieren bereits Netzwerke, wird das Netzwerk vor dem aktuellen Netzwerk eingefügt.

Der Kontextmenü-Befehl **Netzwerk einfügen unterhalb** bewirkt, dass ein Netzwerk unterhalb des aktuellen Netzwerks eingefügt wird.

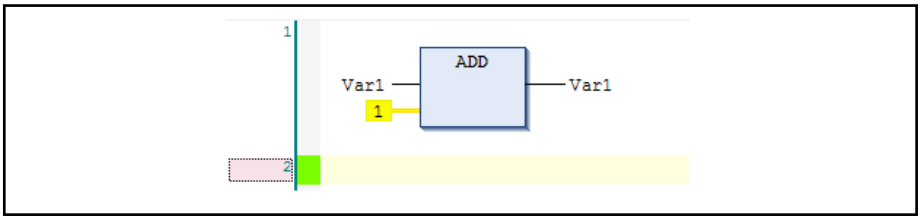


Abb. 7-5: Beispiel: Unterhalb von Netzwerk 1 eingefügtes Netzwerk 2

Kommentierung ein/aus

Durch den Befehl **Kommentierung ein/aus** wird ein Netzwerk auskommentiert bzw. in den normalen Zustand gebracht. Bei Auskommentierung werden die im Netzwerk enthaltenen Elemente ignoriert und als inaktiv dargestellt.

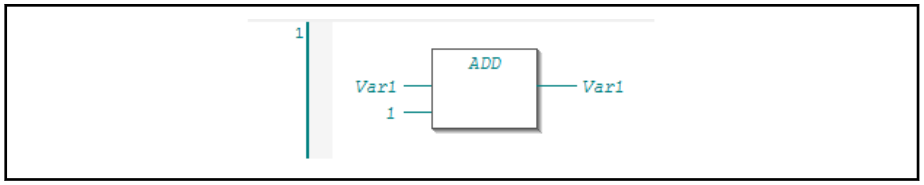


Abb. 7-6: Beispiel: Auskommentiertes Netzwerk

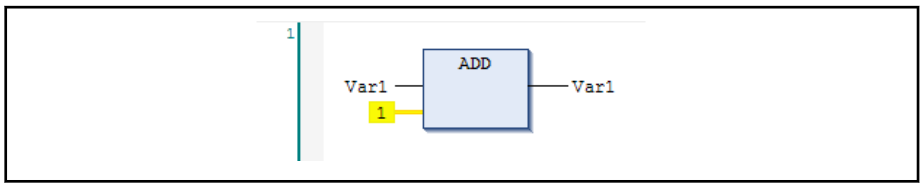


Abb. 7-7: Beispiel: Normaler Zustand des Netzwerks

Netzwerktitel und Netzwerkkommentar

Jedem Netzwerk kann ein Titel und ein Kommentar hinzugefügt werden, wenn die entsprechenden FUP-Optionen aktiviert sind.

Aktivierung der sicheren FUP-Optionen

1. Im Menü **Extras** den Dialog **Optionen...** öffnen
2. Im Dialog **Optionen** den Dialog **Sichere FUP-Optionen** aktivieren
3. Die Optionen **Netzwerktitel anzeigen** und **Netzwerkkommentar anzeigen** auswählen
4. Schaltfläche **OK** aktivieren

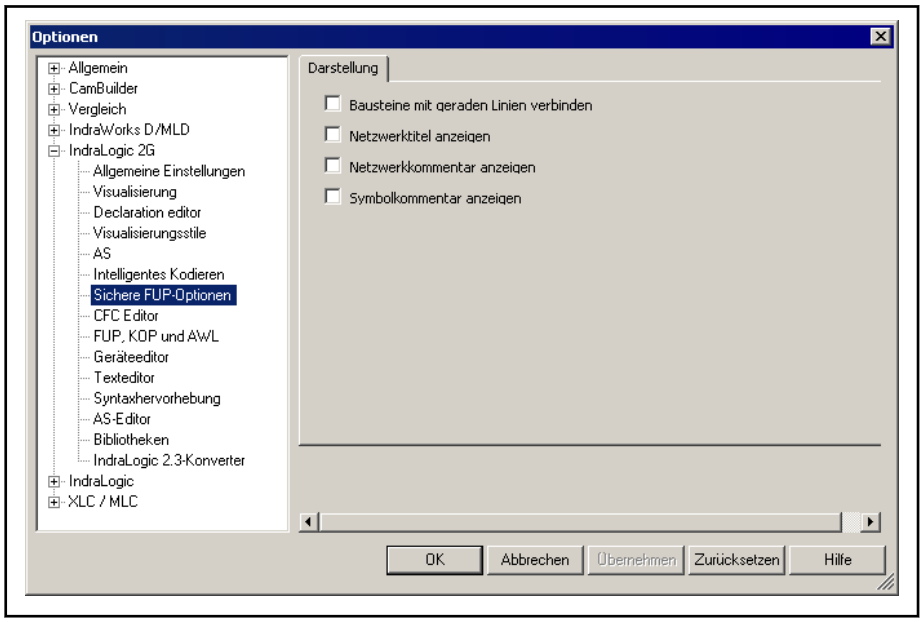


Abb. 7-8: Dialog Sichere FUP-Optionen

Ein Netzwerktitel kann direkt in der ersten Zeile des Netzwerks, der Netzwerkkommentar kann in der zweiten Zeile des Netzwerks editiert werden.

In beiden Fällen muss die entsprechende Zeile zuerst selektiert werden.

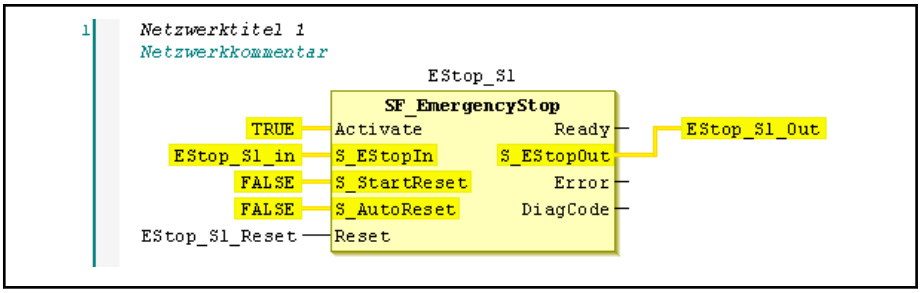


Abb. 7-9: Netzwerk mit Titel und Kommentar

Grundlegendes zu Variablen

Die Programmierung mit SafeLogic erfolgt gemäß IEC 61131-3 mit Hilfe von Variablen. Im Programmcode erfolgt der Zugriff auf Variable über Namen (symbolische Variablen), die entsprechend deklariert werden müssen. Deklarierte Variablen können im Programm-Code gelesen und zugewiesen werden.

Details zu den Namen von Variablen siehe Regeln für Bezeichner von Safety-Objekten und -Variablen auf Seite 90.

Die für die Safety-Programmierung benötigten Variablen werden im Deklarationseditor deklariert und bearbeitet. Dieser Editor ist der Deklarationsteil des POU-Editors. Ausgestattet mit den für die GVL (Globale Variablenliste) notwendigen Besonderheiten, wird dieser Editor auch für die Deklaration und Bearbeitung von globalen Variablen verwendet.

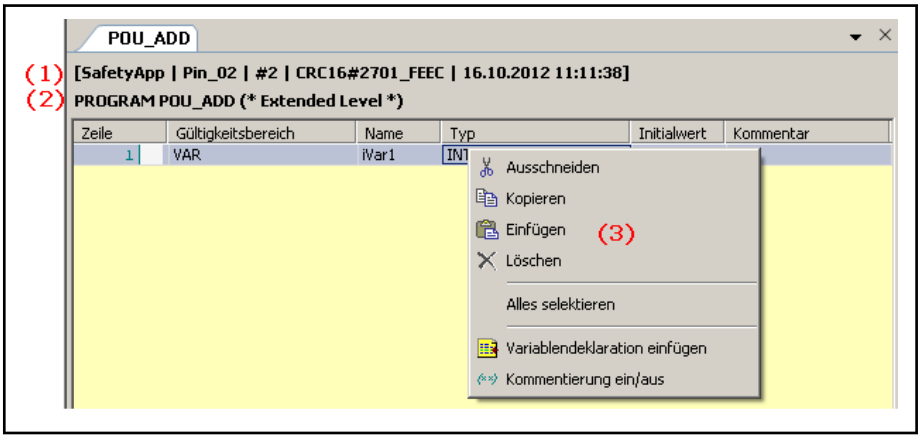


Abb. 7-10: Deklarationsteil des POU-Editors mit geöffnetem Kontextmenü

Aufbau des Deklarationsteils

- **(1):** Pin-Informationen des übergeordneten Safety-Applikationsobjekts **SafetyApp**
- **(2):** POU-Typ (**PROGRAM**), Name der POU mit Programmierlevel (**POU_Add (*Extended Level*)**)
- **(3):** Kontextmenü

☞ Die Verwendung von anderen, als die nachfolgend aufgeführten Datentypen der IEC, ist nicht erlaubt.

Sie stehen auch nicht als freie Bezeichner zur Verfügung, und ihre Namen bleiben reservierte Schlüsselwörter und können nicht benutzerdefiniert verwendet werden.

Die nachstehend aufgeführten, in SafeLogic automatisch geprüften Programmierrichtlinien werden bei den jeweiligen Sprachelementen konkretisiert.

Änderungsmarkierungen im Deklarationsteil des Editors

Zur besseren Übersicht werden neu editierte, geänderte und gelöschte Variablendeklaration farblich markiert:

Es werden nur die Felder, welche von der zuletzt ausgeführten Editieroperation betroffen sind, hervorgehoben. Es ist immer die Markierung der zuletzt durchgeführten Aktion sichtbar. Mit dem Schließen des Objekts (POU oder GVL) werden alle Markierungen aufgehoben.

Markierungen im Deklarationsteil

- grün: neu hinzugekommene Felder
- rot: geänderte Felder
- <del 1> in der Spalte **Zeile** der gelöschten Variable
- die Zeile mit der Änderungsmarkierung ist links mit einem roten Balken gekennzeichnet

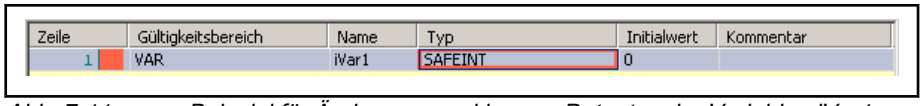


Abb. 7-11: Beispiel für Änderungsmarkierung: Datentyp der Variablen iVar1 wurde geändert

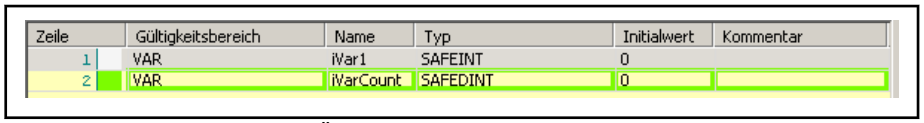


Abb. 7-12: Beispiel für Änderungsmarkierung: Neue Variable iVarCount wurde eingefügt

Änderungsmarkierungen im Implementierungsteil des Editors

Nach jeder Editieroperation werden die Unterschiede zur vorherigen Version farblich markiert.

Es ist immer die Markierung der zuletzt durchgeführten Aktion sichtbar:

- **grün:** neu hinzugekommene Netzwerke oder Elemente
- **rot:** Änderung an einem bestehenden Netzwerk/Element bzw. das Netzwerk mit der Änderung wird rot markiert
- **blau:** Löschmarkierung für gelöschtes Netzwerk oder Element

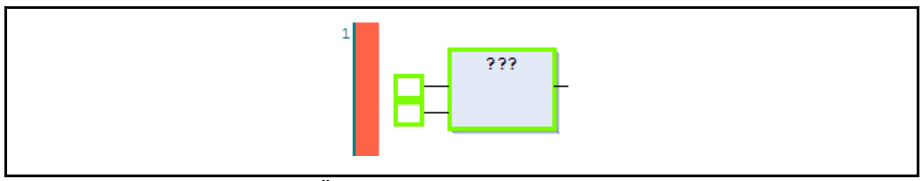


Abb. 7-13: Beispiel für Änderungsmarkierung: Neu eingefügter Bausteinaufruf

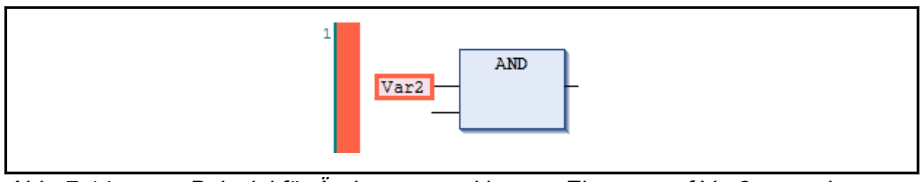


Abb. 7-14: Beispiel für Änderungsmarkierung: Eingang auf Var2 zugewiesen

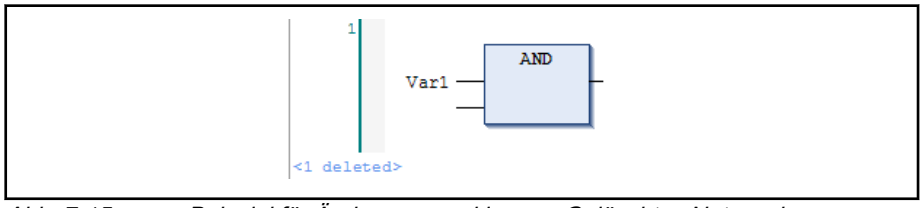


Abb. 7-15: Beispiel für Änderungsmarkierung: Gelöschtes Netzwerk

Mit dem Schließen der POU werden alle Markierungen aufgehoben.

Arten von Deklarationen

Mögliche Deklarationen im Deklarationsteil einer POU:

- IEC-Schlüsselwort **VAR** kennzeichnet Deklaration normaler interner Variablen (POU-weite Variable)
- IEC-Schlüsselwort **VAR_INPUT** kennzeichnet Deklarationen von Eingangsvariablen (POU-weite Variable)
- IEC-Schlüsselwort **VAR_OUTPUT** kennzeichnet Deklarationen von Ausgangsvariablen (POU-weite Variable)
- IEC-Schlüsselwort **VAR_EXTERNAL** kennzeichnet Deklarationen von mit VAR_GLOBAL in der Applikation (in GVL oder implizit im Logischen E/A) bereits deklarierter globaler Variablen, um sie in der POU benutzbar zu machen

  In Programmen auf dem Programmierlevel Basic sind External-Deklarationen nach PLCopen nicht erlaubt.

Modifizierer

CONSTANT

- zur Deklaration symbolischer Konstanten
- bei VAR_EXTERNAL CONSTANT zur Deklaration konstanter globaler Variablen
- kann auf VAR und VAR_GLOBAL angewandt werden, wobei die Variable keine FB-Instanz sein darf

Variablennamen

Variablennamen müssen innerhalb des Gültigkeitsbereichs (Scope) eindeutig sein.

Literalkonstanten

Es stehen folgende Literalkonstanten zur Verfügung:

- Ganzzahl-Literal mit optionaler Typ-Annotation (INT oder DINT) und verschiedenen Zahlenbasen
- Boolesche Literale: TRUE, FALSE
- Bitstring-Literal für BYTE, WORD, DWORD und verschiedenen Zahlenbasen

  BYTE- und DWORD-Literale stehen nur in implizitem Code und externen FBs zur Verfügung
- TIME-Literale

Das Format der Literalkonstanten muss dem in IndraWorks-Standard üblichen Format entsprechen.

Datentypen

Grundlegendes zu Datentypen siehe Datentypen auf Seite 74.

Bei Zuweisungen von Nicht-SAFE-Werten auf SAFE-Variablen oder SAFE-Eingängen werden die Variable bzw. der Eingang im Editor dunkelrot markiert.

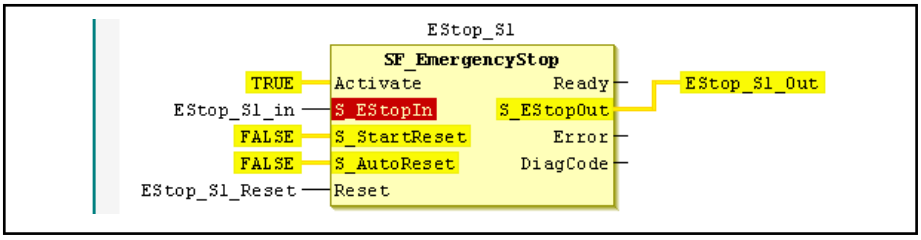


Abb. 7-16: Beispiel: Nicht-SAVE-Variable EStop_S1_in am SAVE-Eingang S_EStopIn

Variablen für Basic-POUs

Schlüsselwörter der Variablendeklaration für Basic Level

Variablen für Programme, Basic Level

- VAR
- VAR CONSTANT
- VAR_EXTERNAL für Kanalvariablen und Stack-Instanzen
- VAR_EXTERNAL CONSTANT nur erlaubt von VAR_GLOBAL CONSTANT

Variablen für Funktionsbausteine, Basic Level

- VAR
- VAR CONSTANT
- VAR_INPUT
- VAR_OUTPUT

Bedeutung der Schlüsselwörter

- **VAR:** Deklaration normaler interner Variablen, POU-weite Variable
- **VAR_INPUT:** Deklaration von Eingangsvariablen
- **VAR_OUTPUT:** Deklaration von Ausgangsvariablen
- **CONSTANT:** Modifizierer, dient zur Deklaration von symbolischen Konstanten

☞ In Basic- und Extended-POUs ist die Deklaration von VAR_IN_OUT-Variablen nicht möglich.

Datentypen Basic Level

Für die Implementierung einer POU im Programmierlevel Basic stehen dem Entwickler folgende Datentypen zur Verfügung:

- **BOOL**
- **INT**: nur als konstanter Eingang einer FB-Instanz
- **DINT**: nur als konstanter Eingang einer FB-Instanz
- **WORD**: nur als Ausgang für Diagnosezwecke
- **TIME**: nur als konstanter Eingang einer FB-Instanz
- **SAFEBOOL**
- **SAFEINT**: nur als konstanter Eingang einer FB-Instanz
- **SAFEDINT**: nur als konstanter Eingang einer FB-Instanz
- **SAFEWORD**: nur als konstanter Eingang einer FB-Instanz
- **SAFETIME**: nur als konstanter Eingang einer FB-Instanz

Der Datentyp **REAL/ LREAL** steht nicht zur Verfügung.

Detailinformationen zu Datentypen siehe Datentypen, Seite 74.

Funktionsbausteintypen Basic Level

Funktionsbausteine des Basic Level sind in nachstehenden Bibliotheken enthalten:

- SafetyPLCopen.library, Seite 292
- SafetyStandard.library, Seite 229

Im Basic-Level zur Verfügung stehende Bausteine der Bibliothek SafetyStandard:

- SF_CTU auf Seite 234, Vorwärtszähler,
- SF_CTD auf Seite 231, Rückwärtszähler,
- SF_CTUD auf Seite 232, Vorwärts- und Rückwärtszähler,

- SF_TON auf Seite 236, Timer mit Einschaltverzögerung,
- SF_TOF auf Seite 235, Timer mit Ausschaltverzögerung,
- SF_TP auf Seite 237, Timer, der einen Pulsgeber realisiert,

☞ Bistabile FBs und Flanken-FBs sind im Basic Level nicht erlaubt.

Einschränkungen für Funktionsbausteintypen

- Es darf zu keiner direkten oder indirekten Rekursion von FBs kommen.
- Die Variable darf nicht als konstant deklariert werden.
- Instanzen normaler FB-Typen können nur als globale Variablen und interne Variablen vorkommen (Eingangsvariablen, Ausgangsvariablen und logische E/As können nur von einem Basistyp sein).

Variablen für Extended-POUs

Schlüsselwörter der Variablendeklarationen für POUs, Extended Level

Variablen für Programme, Extended Level

- VAR
- VAR CONSTANT
- VAR_EXTERNAL
- VAR_EXTERNAL CONSTANT

Variablen für Funktionsbausteine, Extended Level

- VAR
- VAR CONSTANT
- VAR_INPUT
- VAR_OUTPUT

Bedeutung der Schlüsselwörter:

- **VAR:** Deklaration normaler interner Variablen, POU-weite Variable
- **VAR_INPUT:** Deklaration von Eingangsvariablen
- **VAR_OUTPUT:** Deklaration von Ausgangsvariablen
- **VAR_EXTERNAL:** Deklaration von mit VAR_GLOBAL in der Applikation bereits deklarierter globaler Variablen, um sie in der POU benutzbar zu machen.

  Globale Variablen, die den Modifizierer CONSTANT haben, müssen als VAR_EXTERNAL CONSTANT deklariert werden.
- **CONSTANT:** Modifizierer, dient zur Deklaration von symbolischen Konstanten.

Für die Deklaration als VAR_EXTERNAL und VAR_EXTERNAL CONSTANT stehen bereits existierende Variablen der Kategorie "Globale Variablen" zur Verfügung:

- Globale Variablen des GVL-Objekts der Sicherheitsapplikation.

  Wird eine Variable des GVL-Objekts der Sicherheitsapplikation im Implementierungsteil verwendet, so wird sie automatisch im Deklarationsteil als externe Variable deklariert.

  Nach IEC ist es explizit verboten, in einer POU globale Variablen zu verwenden, ohne sie als "External" zu deklarieren.
- Mapping-Variablen der logischen E/As (Logische Austauschgeräte und sichere Feldgeräte)

☞ In Basic- und Extended-POUs ist die Deklaration von VAR_IN_OUT-Variablen nicht möglich.

Datentypen Extended Level

Für die Implementierung einer POU im Programmierlevel Extended stehen dem Entwickler folgende Datentypen zur Verfügung:

- **BOOL**
- **BYTE**: für den Austausch kodierter Informationen (Statuscode, Diagnosecode, Steuercode) zwischen vordefinierten Funktionsbausteinen und mit der Umgebung
- **DINT**
- **DWORD**: für den Austausch kodierter Informationen (Statuscode, Diagnosecode, Steuercode) zwischen vordefinierten Funktionsbausteinen und mit der Umgebung
- **INT**
- **TIME**: erlaubt als konstanter Eingangsparameter und für lokale Variablen

  Nicht zulässig sind VAR_EXTERNAL-Deklarationen globaler Variablen vom Typ SAFETIME
- **WORD**: für den Austausch kodierter Informationen (Statuscode, Diagnosecode, Steuercode) zwischen vordefinierten Funktionsbausteinen und mit der Umgebung
- **SAFEBOOL**
- **SAFEBYTE**: für den Austausch kodierter Informationen (Statuscode, Diagnosecode, Steuercode) zwischen vordefinierten Funktionsbausteinen und mit der Umgebung
- **SAFEINT**
- **SAFEDINT**
- **SAFEDWORD**: für den Austausch kodierter Informationen (Statuscode, Diagnosecode, Steuercode) zwischen vordefinierten Funktionsbausteinen und mit der Umgebung
- **SAFEWORD**: für den Austausch kodierter Informationen (Statuscode, Diagnosecode, Steuercode) zwischen vordefinierten Funktionsbausteinen und mit der Umgebung
- **SAFETIME**: erlaubt als konstanter Eingangsparameter und für lokale Variablen

  Nicht zulässig: VAR_EXTERNAL-Deklarationen globaler Variablen vom Typ SAFETIME, wenn sie weder symbolische Konstanten noch importierte logische E/As sind
- **Nicht zulässig:** SAFE(WORD)-Eingänge (VAR_INPUT) und VAR_EXTERNAL-Deklarationen von normalen globalen (SAFE)WORD-Variablen

Die Datentypen BYTE, SAFEBYTE, DWORD und SAFEDWORD stehen nur für die Schnittstellen externer FBs und zum Import logischer E/As zur Verfügung.

Der Datentyp REAL/ LREAL steht nicht zur Verfügung.

Detailinformationen zu Datentypen siehe Datentypen, Seite 74.

Funktionsbausteintypen Extended Level

Im Extended Level können selbst definierte Funktionsbausteine und alle Bausteine der Safety-Bibliotheken genutzt werden.

Für Funktionsbausteine gelten folgende Einschränkungen:

- Es darf zu keiner direkten oder indirekten Rekursion von FBs kommen
- Die Variable kann nicht als konstant deklariert werden.
- Instanzen normaler FB-Typen können nur als globale Variablen und interne Variablen vorkommen, (Eingangsvariablen, Ausgangsvariablen und logische E/As können nur von einem Basistyp sein)

Operatoren

Als Operatoren werden die eingebauten Standardoperatoren und -funktionen, die eine Teilmenge der Standardoperatoren und -funktionen der IEC darstellen, bezeichnet.

Benutzerdefinierte Funktionen können in der Safety-Programmierung nicht erstellt werden.

Die Operatoren haben die gleiche Semantik wie in Standard-IndraWorks. Sie können sowohl mit SAFExxx als auch mit Standard-Datentypen beschaltet werden.

Einfügen von Operatoren

Das Einfügen der Operatoren erfolgt wie im Standard-FUP von IndraWorks.

Die Einfügeposition innerhalb des Netzwerks siehe Einfügeposition auf Seite 112.

Erweiterbare Operatoren

Um zusätzliche Eingänge erweiterbare Operatoren

- AND
- OR
- ADD (nur im Extended Level)
- MUL (nur im Extended Level)
- MUX (nur im Extended Level)

Die nach IEC ebenfalls erweiterungsfähigen Operatoren EQ, LT, LE, GT, GE lassen sich (wie in IndraWorks-Standard) nur mit zwei Bausteineingängen verwenden.

Der Befehl **Bausteineingang Einfügen** fügt einer erweiterbaren Operatorbox einen weiteren Eingang hinzu.

Operatoren im Basic Level

Boolesche Operatoren

- AND (2 Eingänge)
- AND (3 Eingänge)
- OR (2 Eingänge)
- OR (3 Eingänge)

Alle Operanden des OR müssen SAFEBOOL sein.

Operatoren im Extended Level

Boolesche Operatoren

- AND (2 Eingänge)
- AND (3 Eingänge)
- OR (2 Eingänge)
- OR (3 Eingänge)
- XOR
- NOT

HINWEIS

Der unachtsame Gebrauch der Operatoren XOR und NOT kann zum Verlust der Failsafe-Eigenschaft von SAFExxx-Variablen führen, es wird dabei keine Checker-Warnung erzeugt.

Die Operatoren XOR und NOT können die Failsafe-Eigenschaft einer SAFExxx-Variablen negieren, so dass die SAFE-Variable ihre Failsafe Eigenschaft verliert, also "ausfallunsicher" wird.

Dies kann zu unbeabsichtigtem Anlaufen der Anlage führen.

Programmierregel:

Die SAFExxx-Ausgänge von NOT und XOR müssen ermittelt werden. Anschließend muss sichergestellt werden, dass diese NOT/XOR-Ausgänge nicht auf Ausgänge (E/As) durchgeschaltet werden.

Mathematische Operatoren

- ADD (2 Eingänge)
- ADD (3 Eingänge)
- SUB
- MUL - erweiterbar
- DIV - Laufzeitfehler bei Division durch Null (siehe weiter unten)

Das Ergebnis einer Verknüpfung von Werten mit ADD, SUB, MUL, DIV kann eine Zahl ergeben, die außerhalb des Wertebereichs des Datentyps liegt. Kommt es zur Laufzeit zu einer Wertebereichsverletzung stellt das nach IEC 61131-3 [N1.1.3-Kap.2.5.1.5.2] bzw. [N1.1.3-Kap.2.5.1.5.6] einen Fehler im Programm dar. Dieser Fehler wird von der SafeLogic nicht diagnostiziert! Stattdessen wird die Applikation auf folgende Weise fortgesetzt:

DIV: Wenn die Division der Eingangswerte keine Ganzzahl ergibt, wird CPU-abhängig mit der nächstgrößeren oder nächstkleineren Ganzzahl weiter gerechnet.

ADD, SUB, MUL auf DINT/SAFEDINT-Werten: Wenn die mathematische Operation eine Zahl N außerhalb des Wertebereichs von DINT ergibt, wird mit folgender Zahl innerhalb des DINT-Wertebereichs weiter gerechnet: Bei positivem N: die erste Zahl im DINT-Wertebereich, die durch wiederholte Subtraktion von 4294967296 (2^{32}) erreicht wird. Bei negativem N: die erste Zahl im DINT-Wertebereich, die durch wiederholtes Addieren von 4294967296 (2^{32}) erreicht wird.

ADD, SUB, MUL auf INT/SAFEINT-Werten: Wenn die mathematische Operation eine Zahl N außerhalb des INT-Wertebereichs ergibt, dann liegt diese immer noch im DINT- Wertebereich und in anschließenden Operatoren wird mit dieser Zahl N‘ weiter gerechnet: Dabei werden mathematische Operatoren nach den Regeln für DINT/SAFEDINT-Werte angewendet. Konvertierungen INT_TO_xxx von N' führen zu einem Laufzeitfehler. Und bei Zuweisung von N‘ auf eine INT-Variable wird folgender Wert innerhalb des INT-Wertebereich in die Variable gespeichert: Bei positivem N': die erste Zahl im INT-Wertebereich, die durch wiederholte Subtraktion von 65536 (2^{16}) erreicht wird. Bei negativem N‘: die erste Zahl im INT-Wertebereich, die durch wiederholtes Addieren von 65536 (2^{16}) erreicht wird.

⚠ VORSICHT **Wertebereichsverletzung**

Der unachtsame Gebrauch der Operatoren ADD, SUB, MUL, DIV kann zu Wertebereichsverletzungen führen, die nicht bemerkt werden und ein unerwartetes Verhalten der Applikation nach sich ziehen. Werte dürfen nur mit ADD, SUB, MUL, DIV verknüpft werden, wenn eine Bereichsverletzung entweder ausgeschlossen werden kann (Applikationslogik, Grenzwertprüfungen, o.ä.), oder sie in der Applikation erkannt und der Ergebniswert nicht verwendet wird (sondern z.B. mittels SEL durch einen passenden Wert ersetzt wird).

Weitere mathematische Operatoren

- EQ
- NE
- LT
- LE
- GT
- GE

Andere Operatoren

- SEL
- MUX - erweiterbar, Laufzeitfehler bei ungültigem ersten Input (siehe unten)

Konvertierungen

- BOOL_TO_INT
- BOOL_TO_DINT
- BOOL_TO_TIME
- BOOL_TO_WORD
- BYTE_TO_INT
- BYTE_TO_DINT
- BYTE_TO_TIME
- BYTE_TO_WORD
- DINT _TO_BOOL
- DINT_TO_BYTE - Prüfung des Wertebereichs (siehe unten)
- DINT_TO_INT - Prüfung des Wertebereichs (siehe unten)
- DINT_TO_TIME - Prüfung des Wertebereichs (siehe unten)
- DINT_TO_WORD - Prüfung des Wertebereichs (siehe unten)
- DINT_TO_DWORD
- DWORD_TO_DINT
- DWORD_TO_TIME
- INT_TO_BOOL
- INT_TO_BYTE - Prüfung des Wertebereichs (siehe unten)
- INT_TO_DINT
- INT_TO_DWORD
- INT_TO_TIME - Prüfung des Wertebereichs (siehe unten)
- INT_TO_WORD
- TIME_TO_BOOL

- TIME_TO_BYTE - Prüfung des Wertebereichs (siehe unten)
- TIME_TO_INT - Prüfung des Wertebereichs (siehe unten)
- TIME_TO_DINT - Prüfung des Wertebereichs (siehe unten)
- TIME_TO_WORD - Prüfung des Wertebereichs (siehe unten)
- TIME_TO_DWORD - Prüfung des Wertebereichs (siehe unten)
- WORD_TO_BOOL
- WORD_TO_BYTE - Prüfung des Wertebereichs (siehe unten)
- WORD_TO_DINT
- WORD_TO_INT
- WORD_TO_TIME
- WORD_TO_DWORD

Laufzeitfehler bei Bereichsüberschreitungen der Operatoren im Extended Level

Die Sicherheitssteuerung reagiert bei den im Folgenden aufgeführten Bereichsüberschreitungen mit einem **Laufzeitfehler**, wodurch die Applikation gestoppt und ein Logbucheintrag erzeugt wird.

Level	Sprachelement	Laufzeitfehler bei
Extended	DIV	Division durch 0
Extended	MUX	Aufruf mit erstem Input mit negativem Wert oder mit Wert N größer als die Anzahl der Eingänge minus 1, z. B. MUX(2, 16#8000, 16#8001)
Extended	DINT_TO_INT, TIME_TO_DINT, TIME_TO_INT, DINT_TO_TIME, INT_TO_TIME, DINT_TO_WORD, TIME_TO_WORD, DINT_TO_BYTE, INT_TO_BYTE, TIME_TO_BYTE, WORD_TO_BYTE	Ausgangswert ist nicht im Wertebereich des Zieltyps: Bei Konvertierung zwischen zwei ANY_MAGNITUDE-Typen (INT, DINT, TIME) muss der numerische Ausgangswert im Wertebereich des Zieltyps liegen (wobei TIME-Werte als Anzahl von Millisekunden gerechnet werden). Bei Konvertierung von/zu Bitstring-Typen (BYTE, WORD, DWORD) muss das Bitmuster des Ausgangswerts ein Bitmuster der Zieltyps sein. Beispiele: DINT_TO_INT(16#0000FFFF), weil 2^{16}-1 kein INT-Wert ist, ebenso DINT_TO_TIME(-1), weil es keine negative TIME-Werte gibt TIME_TO_DINT(t#365d), weil 365 Tage = 3,153,600,000ms = 16#BBF81E00 ist, und damit größer als die größte DINT-Zahl 2^{31}-1 = 16#7FFFFFFF INT_TO_BYTE(-1), da BYTE nur 0 bis 255 umfasst, WORD_TO_BYTE(0xFFFF), da BYTE nur bis 0xFF geht.

Das Standardverhalten von SEL/MUX, dass bei zu großem Eingangswert der maximale Wert und bei negativem Eingangswert der Wert 0 ausgewählt wird, muss in der Sicherheitsapplikation programmiert werden.

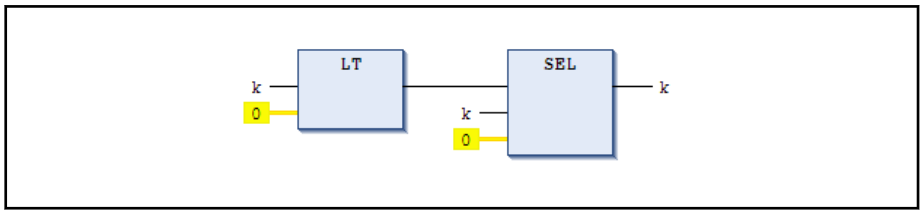


Abb. 7-17: Programmierung des Standardverhalten von SEL: für k<0

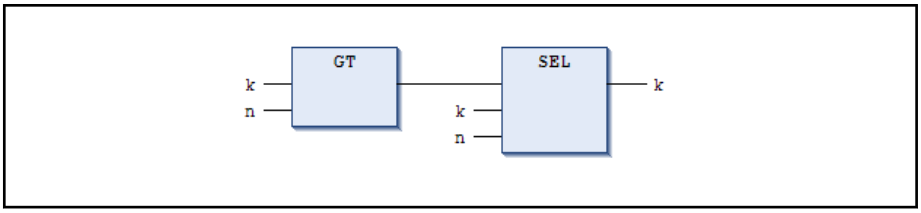


Abb. 7-18: Programmierung des Standardverhaltens von SEL für k>Max, n...Maximalwert

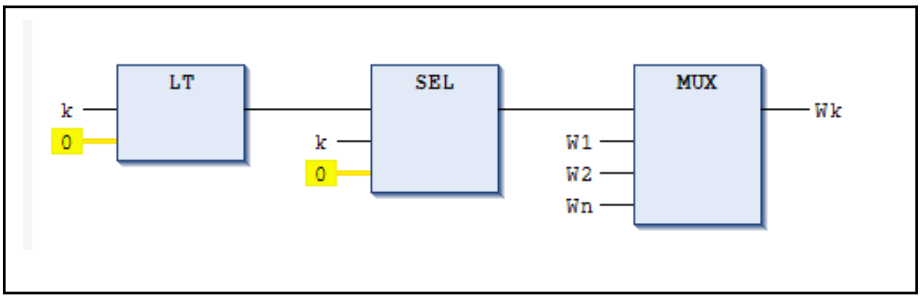


Abb. 7-19: Programmierung des Standardverhaltens von MUX für k<0

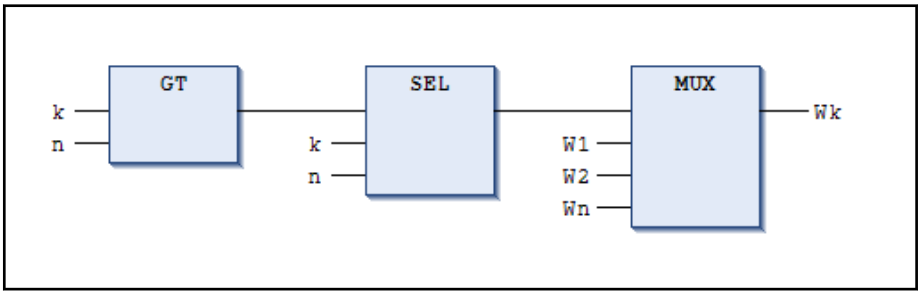


Abb. 7-20: Programmierung des Standardverhaltens von MUX für k>n

FUP-Zuweisungen

Eine Zuweisung ist ein FUP-Element, das in einem Netzwerk den ankommenden Signalfluss aufnimmt und in einem Operanden speichert, d. h. in die Variable schreibt. Für eine Zuweisung ist immer eine Variable notwendig. Zuweisungen können nur am Ausgang einer Box eingefügt werden.

Das Einfügen von Zuweisungen erfolgt wie im Standard-FUP von IndraWorks.

Beispiele für Zuweisungen

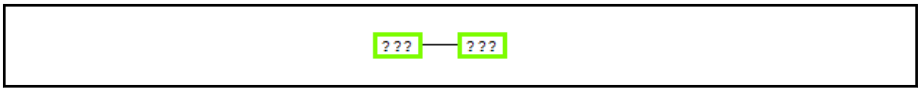


Abb. 7-21: Beispiel: In leerem Netzwerk eingefügte Zuweisung

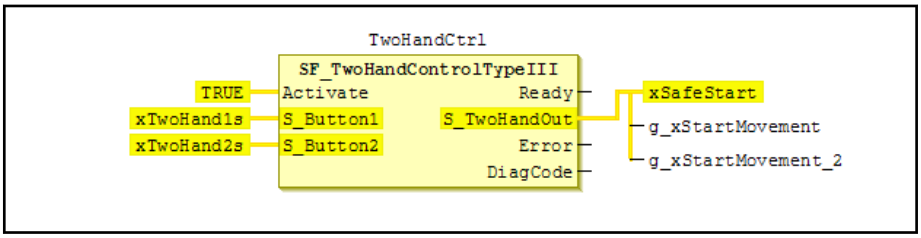


Abb. 7-22: Beispiel: Mehrfachzuweisung durch Einfügen einer neuen Zuweisung an bestehender Zuweisung

Bei Zuweisungen von Nicht-SAFE Werten auf SAFE-Variablen oder SAFE-Eingänge werden die Variablen bzw. Eingänge rot markiert.

Sprung/Return und Sprungmarke

Durch einen bedingten Sprung wird die sequentielle Abarbeitungsreihenfolge der POU unterbrochen. Wenn die Sprungbedingung TRUE ist, wird auf ein mit der Sprungmarke gekennzeichnetes Netzwerk gesprungen.

Mit einer bedingten Return-Anweisung wird die Abarbeitungsfolge des Bausteins unterbrochen. Wenn die Returnbedingung erfüllt ist, wird der Baustein verlassen.

☞ Sprünge und Returns sind nur als bedingte Vorwärtssprünge und bedingte Returns erlaubt.

Sprünge und Returns sind nur im Programmierlevel Extended möglich.

Im Programmierlevel Basic sind generell keine Sprünge/Returns erlaubt.

Sprung/Return

- Bedinge Vorwärtssprünge und Returns sind nur am Netzwerkende erlaubt (dabei bei Mehrfachzuweisungen nach der letzten Zuweisung)
- Als Sprungziel muss ein Netzwerk mit Sprungmarke innerhalb der gleichen POU existieren
- Das Sprungziel-Netzwerk muss hinter dem Netzwerk mit dem Sprung liegen
- Das Sprungziel darf sich in keinem auskommentierten Netzwerk befinden
- Die Bedingung eines Sprung/Return muss eine boolesche Bedingung sein

☞ Bedingte Vorwärtssprünge und Returns sind im Sinne von PLCopen "Sicherheitssoftware" nur als Fehleraussprung und zur Implementierung von Zustandsmaschinen erlaubt.

☞ Returns dürfen gemäß den PLCopen-Regeln für "Sicherheitssoftware" nur als Fehleraussprung verwendet werden.

HINWEIS

Der unachtsame Gebrauch von bedingten Sprüngen und Returns kann zum Verlust der Failsafe-Eigenschaft von SAFExxx-Variablen führen. Es wird dadurch keine Warnung durch den Safety-Checker erzeugt.

Sprünge über Zuweisungen an sichere Ausgänge sind erlaubt.

Ein bedingter Sprung, der von einem unsicheren Wert abhängt und als Sprungziel eine Zuweisung auf eine SAFExxx Variable hat, erlaubt es einem unsicheren Eingang, eine sicheren Ausgang zu beeinflussen.

Hierfür gilt folgende Regel:

- Alle Zuweisungen auf SAFExxx Variable, die Ziele von bedingten Sprüngen sind, die von unsicheren Variablen abhängen, müssen ermittelt werden.

  Es muss dafür gesorgt werden, dass die Sicherheit der Maschine in allen Fällen gewährleistet ist.

Sprungmarke

Sprungmarken sind Zieladressen für Sprünge und können nur am Anfang eines Netzwerks mit dem Kontextmenü-Befehl **Sprungmarke einfügen** eingefügt werden.

Eine Sprungmarke zwischen dem Aufruf des FBs und dem Auslesen eines Ausgangs dieses FBs ist nicht erlaubt.

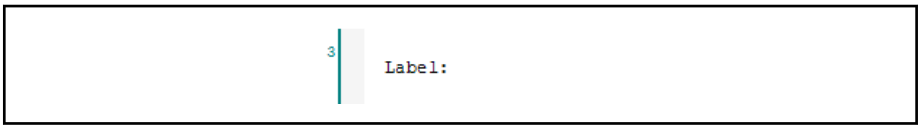


Abb. 7-23: Beispiel: Eingefügte Sprungmarke

FB-Aufrufe

Funktionsweise von FB-Aufrufen

Beim Aufruf des Funktionsbausteins werden die Formalparameter mit den aktuellen Werten der Eingangsvariablen oder Literalen versorgt, bzw. den Ausgängen zugewiesen.

Die Formalparameter und ihre zugewiesenen Variablen/Konstanten müssen im Datentyp übereinstimmen.

Der Instanzname wird im Deklarationsteil des Editors als Variable mit Datentyp = Bausteinname deklariert.

FB-Instanzen können nur auf ihre eigenen POU-weiten Daten und Parameter zugreifen.

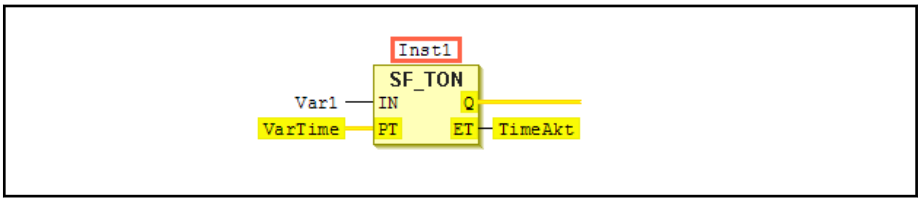


Abb. 7-24: Beispiel: Instanz Inst1 des Baussteins SF_TON mit Formalparametern IN, PT, Q, ET

In SafeLogic stehen dem Entwickler neben den selbst erstellten Funktionsbausteinen, die Bausteine der Sicherheitsbibliotheken "SafetyPLCopen", "SafetyStandard" und gegebenenfalls weiterer Sicherheitsbibliotheken zur Verfügung.

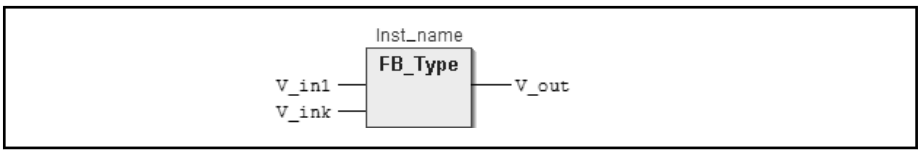


Abb. 7-25: Beispiel: FB-Aufruf

Bausteinaufruf einfügen, Leeren Baustein einfügen

Das Einfügen von Funktionsbausteinen erfolgt wie im Standard-FUP von IndraWorks.

Einfügeposition

An welcher Stelle des Netzwerks der ausgewählte Baustein eingefügt wird, hängt davon ab, wo das Kommando ausgeführt wird:

Kommando wird ausgeführt

- in einem Netzwerk, so dass eine Verknüpfung zweier Bausteine mit sicheren Ein- bzw. Ausgängen entsteht

  Es werden automatisch immer die ersten SAFE-Eingänge bzw. SAFE-Ausgänge miteinander verknüpft.

- **In einem leeren Netzwerk:**

  Für Funktionsbausteinaufrufe werden die Ein- und Ausgänge entsprechend der Signatur des Objekts zusätzlich beschriftet. Die Instanznamen von Funktionsbausteinen werden über der Box angezeigt.

- **An einem Hauptausgang einer anderen Box:**

  Der erste Eingang der neuen Box wird mit dem Hauptausgang der bestehenden Box verbunden. Das Einfügen einer Box an einem Nicht-Hauptausgang einer anderen Box ist nicht möglich.

- **An einem Eingang einer anderen Box:**

  Die neue Box wird zwischen dem eingehenden Signalfluss und dem Eingang der bestehenden Box eingefügt

Löschen der Ein- und Ausgänge

Die Ein- und Ausgänge einer Box, die einen Funktionsbausteinaufruf darstellen, können mit dem Befehl **Löschen** des Kontextmenüs gelöscht werden.

Dabei werden die an diesem Ein- oder Ausgang anliegenden Elemente auch gelöscht.

Wird der Hauptausgang eines Bausteinaufrufs, der sich innerhalb eines Netzwerks befindet, gelöscht, wird im Falle eines Netzwerkes der gesamte Baum links von diesem Bausteinausgang (also inklusive des Bausteins selbst) entfernt.

Ist der Baustein das (rechte) Ende von einem Netzwerk, kann auch der letzte Ausgang (Hauptausgang) entfernt werden, der Baustein bleibt dann erhalten und besitzt keine Ausgänge.

Weiterverschaltung festlegen

Der Befehl **Weiterverschaltung festlegen** legt fest, welcher Ausgang einer Box mit mehreren Ausgängen für die Weiterverschaltung (Hauptausgang) benutzt werden soll.

Jede Box mit Ausgängen kann nur einen einzigen Hauptausgang haben.

Andere Boxen können mit dieser Box nur an deren Hauptausgang verbunden werden. Das Kommando kann nur am Ausgang einer Box mit mehreren Ausgängen ausgeführt werden.

Beim Umdefinieren des Hauptausganges werden mit dem alten Hauptausgang verbundene Elemente automatisch auf den neuen Hauptausgang gesetzt.

Mit dem alten Nebenausgang verbundene Zuweisungsziele werden automatisch auf den neuen Nebenausgang gesetzt.

Parameter aktualisieren

Der Befehl **Parameter Aktualisieren** aktualisiert die Ein- und Ausgänge eines Funktionsbausteinaufrufs. Dabei werden die Parameter der Box mit den in ihrer Schnittstelle definierten verglichen.

Kommt ein neuer Ein- oder Ausgang hinzu, wird dieser mit der leeren Variablen beschaltet.

Fällt ein Ein- oder Ausgang weg, wird dieser zusammen mit allen beschalteten Elementen gelöscht. Die Zuordnung erfolgt dabei über die Namen der Ein- bzw. Ausgänge.

Nicht verwendete FB-Aufruf-Parameter entfernen

Der Befehl **Nicht verwendete FB-Aufruf-Parameter entfernen** entfernt alle Ein- und Ausgangsanschlüsse des Aufrufs, welche leer belegt sind. Die minimale vom Baustein geforderte Anzahl von Eingängen bleibt bestehen.

7.6 E/A-Zugriff

Der E/A-Zugriff erfolgt ausschließlich über die logischen E/As.

Jede Mapping-Variable kann wie eine globale Variable als VAR_EXTERNAL deklariert und im Programm verwendet werden.

Details siehe Logische E/As - Überblick, Seite 47.

7.6.1 Anbindung der Sicherheitsapplikation an 1oo1- und 1oo2-Eingabegeräte

⚠ VORSICHT **Bei der Verwendung eines sicheren Eingangssignals ist die Auswertlogik des Eingangsgeräts zu beachten!**

Zwei mögliche Auswertlogiken werden unterschieden:

- 1oo1
- 1oo2

1oo1-Auswertung bedeutet, dass nur ein Sensor vorhanden ist, der bei erwarteter Funktionsweise einen SAFEBOOL-Wert generiert.

1oo2-Auswertung bedeutet, dass einer von zwei Sensoren wie erwartet funktionieren muss, um den SAFEBOOL-Wert zu generieren. Wenn ein Sensor nicht das erwartete Verhalten zeigt, wird der Wert auf den fail-safe Wert gesetzt.

Bei der 1oo1-Auswertung wird die Fehlererkennung für jeden Kanal separat durchgeführt. In diesem Fall können zusätzliche Maßnahmen auf Ebene der Sicherheitsapplikation und bei der peripheren Schaltung notwendig sein.

Es gilt allgemein für PROFIsafe (d.h. PROFIBUS sowie PROFINET):

Der Wert des F-Parameters "F_SIL" zeigt die Auswertlogik an:

- "SIL2" bedeutet 1oo1-Auswertung
- "SIL3" bedeutet 1oo2-Auswertung (Im Fall einfacher Eingangsklemmen muss das Anschließen von Sensoren abhängig von diesem Parameter erfolgen.

Einzelheiten sind der Gerätedokumentation zu entnehmen).

Bei anderen Eingangsgeräten muss deren Auswertlogik der Gerätedokumentation entnommen werden.

Zugriff auf 1oo2 Eingangssignale

Die Sicherheitsapplikation wird wie folgt mit Safety Eingabegeräte mit 1oo2-Auswertung verbunden:

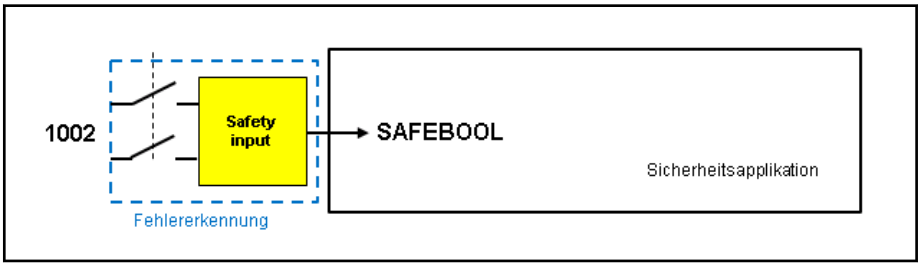


Abb. 7-26: Verbindung der Sicherheitsapplikation mit Safety Eingabegerät mit 1oo2-Auswertung

Einige PLCopen-Funktionsbausteine, wie z.B. SF_GuardMonitoring, können sowohl mit Eingangsgeräten mit 1oo2-Auswertung (siehe folgende Abbildung) als auch mit Eingangsgeräten mit 1oo1-Auswertung (Abb. 7-31 "1oo1-Auswertung, SF_GuardMonitoring" auf Seite 116) verbunden werden.

Siehe auch:

- SF_GuardMonitoring, Seite 341

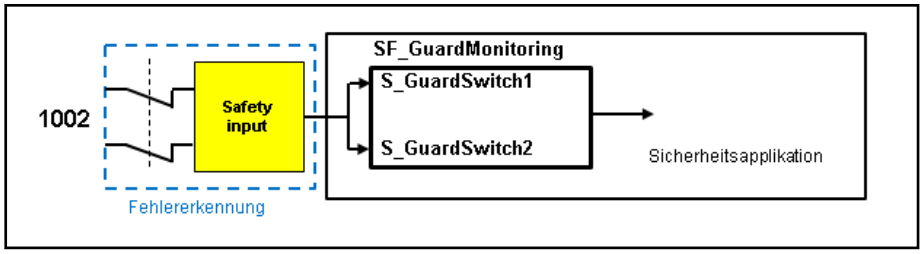


Abb. 7-27: 1oo2-Auswertung, SF_GuardMonitoring

Zugriff auf 1oo1 Eingangssignale

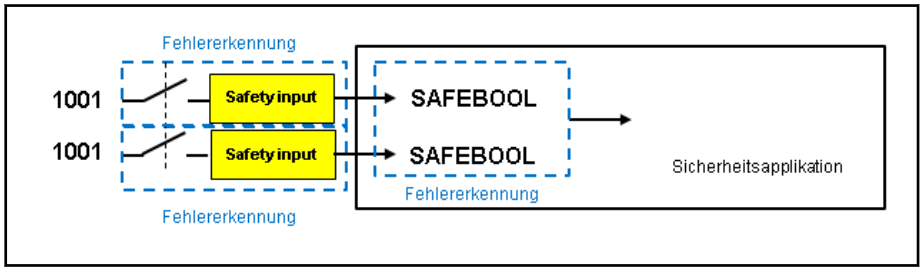


Abb. 7-28: Verbindung der Sicherheitsapplikation mit Safety Eingabegerät mit 1oo1-Auswertung

Die Schalter können aus jeder Kombination aus normal geschlossen (NC) und normal offen (NO) bestehen, welche die Anforderungen und die Fähigkeit zur Fehlererkennung des Safety- Eingangs berücksichtigen.

In Abb. 7-28 "Verbindung der Sicherheitsapplikation mit Safety Eingabegerät mit 1oo1-Auswertung" auf Seite 115 muss in der Sicherheitsapplikation eine zusätzliche Fehlererkennung implementiert werden. Diese wird entweder im signal-verarbeitenden Funktionsbaustein, oder separat über den Funktionsbaustein SF_Equivalent (Abb. 7-30 "1oo1-Auswertung: Fehlererkennung mit SF_Equivalent" auf Seite 116) oder den Funktionsbaustein SF_Antivalent (Abb. 7-29 "1oo1-Auswertung: Fehlererkennung mit SF_Antivalent" auf Seite 115) implementiert.

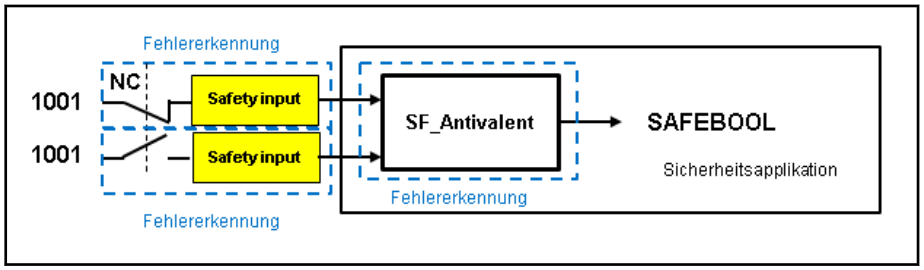


Abb. 7-29: 1oo1-Auswertung: Fehlererkennung mit SF_Antivalent

Bei Verwendung des Funktionsbaustein SF_Antivalent ist der NCSchalter mit einem spezifischen Funktionsbaustein-Eingang zu verbinden. Für die Kombination von NC-Schalter mit NO-Schalter implementiert SF_Antivalent die notwendige Fehlererkennung. Die Schalter müssen dazu mit den entsprechenden Eingängen (S_ChannelNC und S_ChannelNO) verbunden werden.

Siehe auch:

- SF_Antivalent, Seite 300
- SF_Equivalent, Seite 327

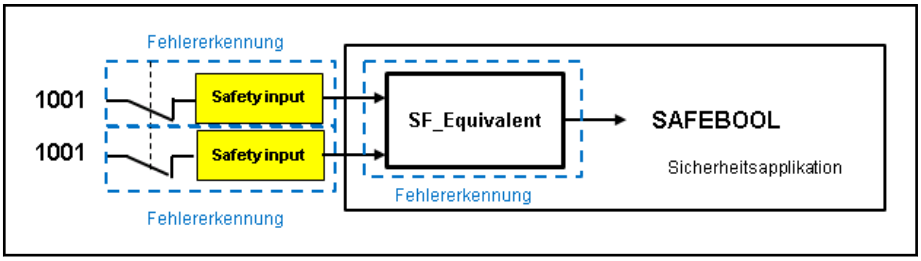


Abb. 7-30: 1oo1-Auswertung: Fehlererkennung mit SF_Equivalent

Für die Kombination zweier NC-Schalter oder zweier NO-Schalter implementiert SF_Equivalent die notwendige Fehlererkennung. Der SF_Equivalent verarbeitet zwei SAFEBOOL-Eingänge und überwacht, dass sich das Signal innerhalb einer spezifizierten Diskrepanzzeit äquivalent ändert.

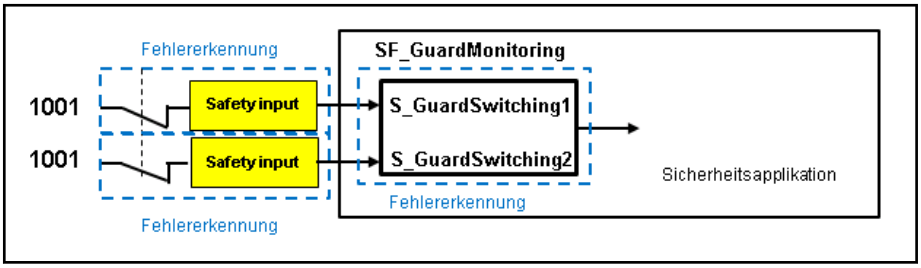


Abb. 7-31: 1oo1-Auswertung, SF_GuardMonitoring

Bei einigen PLCopen-Funktionsbausteinen , wie z.B. SF_Guard- Monitoring, ist die Fehlererkennung für die Kombination zweier NC- bzw. zweier NO-Schalter im Baustein selbst implementiert. D.h. die Schalter können ohne vorgeschaltetes SF_Equivalent direkt mit dem Baustein verbunden werden. (Wie solche Bausteine mit 1oo2-Eingangsgeräten verbunden werden können, zeigt Abb. 7-27 "1oo2-Auswertung, SF_GuardMonitoring" auf Seite 115).

Siehe auch:

- SF_GuardMonitoring, Seite 341

7.7 Querkommunikation mit Safety Netzwerkvariablen

Safety NetVars erfüllt die Funktion der sicheren Querkommunikation in SafeLogic. Es erlaubt die Konfiguration und den Betrieb eines sicheren Datenaustauschs zwischen SafeLogic-Sicherheitssteuerungen in einem Projekt.

Das Prinzip: Ein SafeLogic-Projekt enthält mehrere Hauptsteuerungen mit eingehängten Sicherheitssteuerungen.

Der sichere Datenaustausch erfolgt nach folgendem Prinzip:

Eine Sicherheitssteuerung veröffentlicht Daten durch einen Sender (Objekt vom Typ "Safety NVL (Sender)") und andere Sicherheitssteuerungen des Projekts können die Daten mit Hilfe eines Empfängers (Objekt vom Typ "Safety NVL (Empfänger)") lesen. Der Kommunikationsweg erfolgt über die Hauptsteuerung.

Der Zugriff auf Safety Netzwerkvariablen erfolgt über die Objekte Safety Netzwerkvariablenliste (Sender) und Safety Netzwerkvariablenliste (Empfänger).

Die Safety Netzwerkvariablen können wie globale Variablen als VAR_EXTERNAL deklariert in der Sicherheitsapplikation verwendet werden.

Auf Senderseite sind Netzwerkvariablen Ausgänge der Applikation. Das heißt, ihnen muss genau einmal im Zyklus ein Wert zugewiesen werden.

Auf Empfängerseite sind Netzwerkvariablen Eingänge der Applikation. Das heißt, sie dürfen gelesen, aber nicht beschrieben werden.

Wenn die Werte erfolgreich übertragen werden, enthalten die Netzwerkvariablen auf Empfängerseite die Werte der verbundenen Netzwerkvariablen auf Senderseite.

☞ Netzwerkvariablen verwenden standardmäßig UDP Broadcasts im Maschinennetzwerk.

Um die Netzlast zu reduzieren, können die IP-Adressen der gegenüberliegenden Standardsteuerung fix eingestellt werden.

Siehe

- Objekt 'Safety Netzwerkvariablenliste (Sender), Registerkarte 'SPS Netzwerk', Seite 664
- Objekt 'Safety Netzwerkvariablenliste (Empfänger), Registerkarte 'SPS Netzwerk', Seite 667

☞ Wenn Netzwerkvariablen auf Senderseite veröffentlicht werden, hat jeder Bearbeiter des Projekts automatisch das Recht, die Verwendung dieser Variablen in der Empfängerapplikation zu konfigurieren, wenn er allgemein das Benutzerrecht hat, die Verwendung von Variablen zu konfigurieren.

☞ Wenn ein Safety NVL-Empfänger mehrere Variablensätze des gleichen Safety NVL-Senders abonniert, ist weder garantiert, dass die Werte in verschiedenen Variablensätzen aus dem gleichen Zyklus des Safety NVL-Senders stammen (Zykluskonsistenz), noch, dass bei erfolgreicher Übertragung der Werte eines Variablensatzes auch die Werte der anderen Variablensätze übertragen wurden (Kommunikationskonsistenz).

Wenn mehrere Safety NVL-Empfänger den gleichen Variablensatz abonnieren, ist weder garantiert, dass jeder von ihnen die Werte aus dem gleichen Zyklus des Safety NVL-Senders erhält (Zykluskonsistenz), noch, dass bei erfolgreicher Übertragung der Werte in einen Receiver auch die anderen Receiver die Werte empfangen haben (Kommunikationskonsistenz).

Um eine Sender-Steuerung bei zukünftigen Erweiterungen der Maschine mit weiteren Empfängersteuerungen nicht ändern und aus diesem Grund neu abnehmen lassen zu müssen, kann das Empfängerlimit vorsorglich auf die Empfängeranzahl N im Maximalausbau (höher als die aktuelle Anzahl Empfänger in der Maschine) eingestellt werden und der Anwender kann sich im Vorhinein überzeugen (z.B. ausprobieren), dass die Senderapplikation auch im Maximalausbau mit N laufenden Variablenverbindungen zu N Empfängern ihre Zykluszeit nicht überschreitet.

Überwachung der Kommunikation

Bei Kommunikationsproblemen enthalten die Netzwerkvariablen auf Empfängerseite den Ersatzwert 0 bzw. FALSE.

Die Empfängerapplikation kann über die Treiberinstanz <NVLName> des NVL-Treiberbausteins "NetVarReceiver" den Status der Verbindung kontrollieren und Kommunikationsfehler bestätigen.

Die Senderapplikation kann über die Treiberinstanz <NVL-Name> des NVL-Treiberbausteins "NetVarSender" die Sendebereitschaft steuern bzw. Failsafe Ersatzwerte aktivieren.

Eine Übersicht zu den Treiberbausteinen finden Sie in SafetyNetVar.library, Seite 599, und Allgemeines zu Treiberbausteinen finden Sie in Feldbusse und Netzwerkvariablen, Allgemeines, Seite 573.

Einschränkungen bei der Sampling-Rate/Übertragungszeit

Die Werte der Safety Netzwerkvariablen und Empfangsbestätigungen werden immer synchron am Ende eines Applikationszyklus (Ausgangsphase) verschickt und am Beginn eines Applikationszyklus (Eingangsphase) empfangen. Die Zykluszeiten der Applikationen von Sender und Empfänger und die Übertragungszeit beeinflussen die Sampling-Rate der Netzwerkvariablenwerte.

Wenn eine Netzwerkvariable im Sender kurzzeitig von Wert A auf Wert D wechselt und wieder zurück auf A, kann es sein, dass der Wert D nie an den Empfänger übertragen wird (Undersampling). In dem Fall, dass der Wert D dieser Variable im aktuellen Zustand der Applikation eine Situation darstellt, die die Auslösung einer Sicherheitsfunktion bedarf, kann es zu einer Gefährdung kommen.

⚠ VORSICHT **Undersampling**

Um zu vermeiden, dass zu kurze Signale nicht erkannt oder falsch übertragen werden, dürfen die Signale der Safety Netzwerkvariablen bei der Übertragung nicht zu kurz sein. Der Anwender muss auf Senderseite (analog zu einem Sensor bzw. Eingangsmodul) sicherstellen, dass ein gefahranzeigendes Signal D bzw. ein Signal D, das eine sichere Reaktion anfordert (Demand), im Sender

- entweder so lange ansteht, wie die Gefahr besteht bzw. die Reaktion erforderlich ist
- oder mindestens so lange ansteht, wie die größte Watchdog-Zeit aller Verbindungen zu diesem Signal, d.h., dass es lange genug ansteht, um von allen Empfänger-Steuerungen noch rechtzeitig verarbeitet zu werden.

Eine geeignete Maßnahme ist beispielsweise die Verwendung von Verzögerungsfunktionsbausteinen oder Ausgangswerten rücksetzgeschützter PLCopen-Funktionsbausteine.

- Wenn die Applikationen der Safety Netzwerkvariablenliste (Sender) und der Safety Netzwerkvariablenliste (Empfänger) die gleichen Zykluszeiten haben und die Übertragungszeit kleiner als die halbe Zykluszeit ist, muss eine Werteänderung in der Safety Netzwerkvariablenliste (Sender) 4 Zyklen stehen bleiben, damit sie garantiert in der Safety Netzwerkvariablenliste (Empfänger) ankommt.

  Sampling-Rate: 1 Werteänderung pro 4 Zyklen des NVL-Senders
- Wenn der Safety Netzwerkvariablenlisten (Empfänger) eine relativ kleine Zykluszeit hat, darf eine erneute Werteänderung in der Safety Netzwerkvariablenliste (Sender) nicht sofort im nächsten Zyklus vorgenommen werden. Eine Werteänderung muss mindestens 2 Zyklen im NVL-Sender stehen bleiben. Dabei ist die bestmögliche Sampling-Rate: 1 Werteänderung pro 2 Zyklen des NVL-Senders. Diese bestmögliche Sampling- Rate wird erreicht, wenn die Zykluszeit des NVL-Senders größer ist als

  (2 x Zykluszeit NVL-Empfängers + 2 x Übertragungszeit).

Regeln zur Vermeidung von gefährlichem Undersampling

Beispielregeln für SAFEBOOL-Netzwerkvariablen

- **Regel "Wiederanlaufsperre":** Veröffentlichen Sie Netzwerkvariablen, die auf FALSE (nach Failsafe-Prinzip ein gefahranzeigendes oder reaktionsanforderndes Signal) stehen bleiben, bis der Operateur durch eine Signalflanke die Abwesenheit der Gefahr und die Rücknahme der Sicherheitsreaktion, bzw. das Wiederanlaufen durch eine Signalflanke bestätigt hat.
- **Regel "Verriegelter Sensor"**: Veröffentlichen Sie Variablen, die mit dem Eingangssignal von einem Sensor mit mechanischer Verriegelung, z.B. ein Nothalt-Befehlsgerät, belegt sind. Denn mit dem Rücksetzen der Verriegelung bestätigt der Operateur das Ende der Gefahr.
- **Regel "Physikalische Mindestdauer"**: Veröffentlichen Sie Variablen, die eine Prozesswert darstellen (z.B. Schutztür geöffnet), wenn dieser physikalisch bedingt erst ab einer gewissen Mindestdauer (Zeit zum Passieren der Schutztür) eine gefährliche Situation darstellt, und wenn diese Mindestdauer länger als die Watchdog-Zeit der Verbindung zum Sensor plus der längsten Watchdog-Zeit aller Empfänger ist.

Beispielregeln für SAFEINT-Netzwerkvariablen

- **Regel "Entscheidung im Sender"**: Veröffentlichen Sie analoge Prozesswerte (z.B. Geschwindigkeit, Position) nicht zu dem Zweck, dass Empfänger die Werte überwachen und gegebenenfalls eine Sicherheitsfunktion auslösen. Vielmehr überwachen Sie diese analogen Prozesswerte selbst und melden Sie das Erreichen bestimmter Werte an die Empfänger. Das heißt, veröffentlichen Sie statt dessen digitale Netzwerkvariablen, von denen die Empfänger ihre Reaktion abhängig machen können. Bei diesen Variablen kann Undersampling mit den Regeln für SAFEBOOL Netzwerkvariablen vermieden werden.
- **Regel "Extremwertermittlung im Sender"**: Veröffentlichen Sie zu überwachende analoge Prozesswerte (Geschwindigkeit, Position, Temperatur, etc.) nicht direkt. Bestimmen Sie vielmehr deren größten und ggf. kleinsten erreichten Werte und veröffentlichen Sie nur diese Werte. Empfänger können dann selbständig gegen Überschreitungen von Grenzen (nach oben oder unten) prüfen und Sicherheitsfunktionen auslösen, ohne Extremwerte wegen Undersampling zu verpassen. Sehen Sie den Neustart der Extremwertermittlung durch den Operateur nach einer Sicherheitsreaktion des Empfängers wegen Grenzüberschreitung vor. Dann kann der Empfänger mit aktuellen Prozesswerten neu beginnen.
- **Regel "Nonreaktive Signale"**: Veröffentlichen Sie Werte, die Empfängern nicht zur Erkennung von Gefahren dienen, d.h. keiner ihrer Werte stellt ein gefahranzeigendes oder reaktionsanforderndes Signal dar. Es gibt also keinen bestimmten Wert der Netzwerkvariable, der im Empfänger eine Sicherheitsfunktion auslösen würde, und damit keinen Wert, dessen Überspringen (Undersampling) das Auslassen einer erforderlichen Reaktion bewirken würde.

Erkennung von Undersampling (Sender und Empfänger)

Wenn Undersampling nicht durch Vermeidungsmaßnahmen ausgeschlossen wird, dann sollten Maßnahmen zur Erkennung von Undersampling und die Reaktion darauf implementiert werden.

Damit der Empfänger den Verlust einer kurzzeitigen Signaländerung erkennen kann, muss auf Senderseite entsprechende Vorarbeit geleistet werden.

Eine mögliche Umsetzung sieht wie folgt aus:

Senderseite

- Jeder Sender-NVL Nk mit Netzwerkvariablen `SO_xk` eine Variable `SO_changecount_k` vom Datentyp `SAFEINT` hinzufügen.
- Für jede Sender_NVL Nk ein Zählprogramm Pk anlgegen mit lokalen Merkervariablen `old_xi` für jede Netzwerkvariable `SO_xi`.
- Im Zählprogramm Pk durch Vergleich `SO_xi` gegen `old_xi` feststellen, ob sich der Wert irgendeiner Variablen geändert hat.
- Wenn sich der Wert einer Variablen geändert hat, dann `SO_changecount_k` um 1 erhöhen.
- Die Werte von `SO_xi` in `old_xi` merken.
- Pk in die Taskliste einfügen.

Empfängerseite

- Erkennung:

  Auf der Empfängerseite wird der gesendete aktuelle Zähler `S_changecount` der Safety Netzwerkvariablenliste (Empfänger) mit dem Zähler verglichen, der 1 Zyklus früher gesendet wurde. Wenn die beiden Zähler gleich sind, oder der aktuell gesendete um 1 höher ist, bedeutet dies, dass kein Signal verloren gegangen ist.

- Reaktion:

  Wenn der Zähler um mehr als 1 höher ist, ist ein Signal verloren gegangen. Die Reaktion könnte darin bestehen, alle empfangenen Netzwerkvariablen auf Failsafe- Werte zu setzen. Da die Applikation aber nicht die Variablen in der Empfänger-NVL überschreiben darf, müsste man für so eine Reaktion die Netzwerkvariablen in der NVL in `SI_raw_xi` umbenennen, eine GVL mit globalen Variablen (oder im gleichen Programm lokale Variablen) mit Namen `SI_xi` anlegen. Im Gutfall werden die Werte `SI_raw_xi` nach `SI_xi` kopiert, im Undersampling-Fall werden Failsafe-Werte in `SI_xi` kopiert.

7.8 Taskkonfiguration

Die Aufrufreihenfolge der Programme und die Zykluszeit werden im Task-Objekt, das in der Sicherheitsapplikation vorhanden sein muss, festgelegt.

Die Programmliste, die mindestens einen Aufrufeintrag enthalten muss, wird immer nach Ablauf der Zykluszeit von Beginn an verarbeitet.

Die Überschreitung der Zykluszeit bei Abarbeitung der Programmliste führt zu einem Laufzeitfehler. In diesem Fall wird ein Logbucheintrag erzeugt.

Übersteigt die tatsächliche Zykluszeit dauerhaft 80% der parametrierten Zykluszeit, so führt dies ebenfalls zu einem Laufzeitfehler.

Die gemessene Zykluszeit kann über den Steuerungsstatus ausgelesen werden (siehe Diagnose der Kommunikation auf Seite 202).

Genauere Informationen zum Task-Objekt siehe Safety Task auf Seite 63.

7.9 Programmierbeispiel

Als IndraWorks Safety Manager-Programmierbeispiel für ein Programm im "Extended Level" wird das Beispiel **2-Hand-Kontrolle mit EDM** des Dokuments "PLCopen - Technical Committee 5 Safety Software Technical Specification Part 2: User Examples Version 1.01 - Official Release" verwendet und als IndraWorks Safety Manager-FUP-Implementierung dargestellt.

Für weitere Beispiele zur Programmierung im Basic-Level wird auf das Dokument "PLCopen - Technical Committee 5 Safety Software Technical Specification Part 2: User Examples Version 1.01 - Official Release" verwiesen.

Die Programmierung im "Extended Level" erfolgt, um Diagnoseinformationen von den DiagCode-Ausgängen der Safety-PLCopen-Funktionsbausteine WORD-Variablen zuweisen zu können. Dies ist in der aktuellen Implementation auf dem "Basic-Level" nicht möglich.

Beschreibung der Sicherheitsfunktionen

Folgende Sicherheitsfunktionen werden in diesem Beispiel verwendet:

- Bei Betätigen des Not-HaltTasters müssen alle gefahrbringenden Bewegungen gestoppt werden (über Kap. 16.4.4 "SF_EmergencyStop" auf Seite 314)

  Not-Halt hat die höchste Priorität.

  Nach Lösen des E-Stop-Druckknopfs wird ein Reset über S0-Reset benötigt.

- Der Sicherheitsausgang wird durch Drücken beider Drucktaster der Zwei-Hand-Kontrolle aktiviert

  Das Lösen irgendeines der Zwei-Hand-Drucktaster deaktiviert den Sicherheits-Ausgang und stoppt die gefahrbringende Bewegung über die Schaltvorrichtungen K1 und K2 (über Kap. 16.4.17 "SF_TwoHandControlTypeII" auf Seite 412)

- Der Grundzustand und der Betriebszustand der verbundenen Schaltvorrichtungen werden überwacht. Falls ein Fehler entdeckt wird, kann der Sicherheitsausgang nicht funktionsbereit werden. (über Kap. 16.4.3 "SF_EDM" auf Seite 306)

- Nach Anschalten der Sicherheits- oder der funktionalen Anwendung, oder nach einer Not-Halt Bedingung muss die Zwei-Hand-Kontrolle gelöst und wieder betätigt werden, um den Sicherheits-Ausgang wieder zu aktivieren (über Kap. 16.4.14 "SF_OutControl" auf Seite 389).

  Um dies für den funktionalen Neustart zu gewährleisten, ist das Prozess-Signal der funktionalen Applikation mit dem Activate-Eingang des Zwei-Hand-Kontrolle Bausteins THC_S2_S3 verbunden.

  Wenn der Anwendungsprozess wiedergestartet wird, während die Zwei-Hand-Kontrolle aktiviert ist, geht der Baustein in Status 16#C003.

  Dieser Status signalisiert den Fehler, dass beide Taster bei Aktivierung gedrückt sind, ein Neustart wird verhindert.

In diesem Beispiel existiert nur ein Betriebszustand.

EDM_PRG

In Work

PROGRAM EDM_PRG (* Extended Level *)

Zeile	Gültigkeitsbereich	Name	Typ	Initialwert	Kommentar
1	VAR	EStop_S1	SF_EmergencyStop		Instanz des FBs SF_EmergencyStop
2	VAR	S1_S_EStopIn	SAFEBOOL	FALSE	Not-Halt-Taster S1
3	VAR_EXTERNAL	Reset	BOOL		Reset von Standardsteuerung
4	VAR	S_EStopOut	SAFEBOOL	FALSE	
5	VAR	Error_EStop_S1	BOOL	FALSE	Fehlerflag von EStop_S1
6	VAR	Diag_EStop_S1	WORD	0	Diagnosecode für EStop_S1, 16#8..., 16#Cxxx bei Fehler in EStop_S1
7	VAR_EXTERNAL	Process	BOOL		Freigabe der Bewegung durch den Prozess von Standardsteuerung
8	VAR	S2_S_Switch1	SAFEBOOL	FALSE	Schalter S2 verbunden mit Druckknopf 1 der Zwei-Hand-Kontrolle
9	VAR	S3_S_Switch2	SAFEBOOL	FALSE	Schalter S3 verbunden mit Druckknopf 2 der Zwei-Hand-Kontrolle
10	VAR	S_TwoHandOut	SAFEBOOL	FALSE	
11	VAR	THC_S2_S3	SF_TwoHandControlTypeII		Instanz des FBs SF_TwoHandControlTypeII
12	VAR	Error_THC_S2_S3	BOOL	FALSE	Fehlerflag von TCH_S2_S3
13	VAR	Diag_THC_S2_S3	WORD	0	Diagnosecode für THC_S2_S3, 16...6#Cxxx bei Fehler in THC_S2_S3
14	VAR	OC_K1_K2	SF_OutControl		Instanz des FBs SF_OutControl
15	VAR	EDM_K1_K2	SF_EDM		Instanz des FBs SF_EDM
16	VAR	Error_OC_K1_K2	BOOL	FALSE	Fehlerflag von OC_K1_K2
17	VAR	Diag_OC_K1_K2	WORD	0	Diagnosecode für OC_K1_K2, 16#..., 16#Cxxx bei Fehler in OC_K1_K2
18	VAR	K1_S_EDM1	SAFEBOOL	FALSE	Rückmeldung externes Gerät K1
19	VAR	K2_S_EDM2	SAFEBOOL	FALSE	Rückmeldung externes Gerät K2
20	VAR	Error_EDM_K1_K2	BOOL	FALSE	Fehlerflag von EDM_K1_K2
21	VAR	Diag_EDM_K1_K2	WORD	0	Diagnosecode für EDM_K1_K2, 16#...6#Cxxx bei Fehler in EDM_K1_K2
22	VAR	S_EDM_Out_EDM_K1_K2	BOOL	FALSE	Freigabesignal für Sicherheitssteuerung (Sichere Antriebe...)
23	VAR_EXTERNAL	S_EDM_Out_EDM_K1_K2_unsafe	BOOL		Freigabesignal für Standardsteuerung (unsafe)

Abb. 7-32: Deklaration für Programmierbeispiel: Zwei-Hand-Kontrolle mit EDM

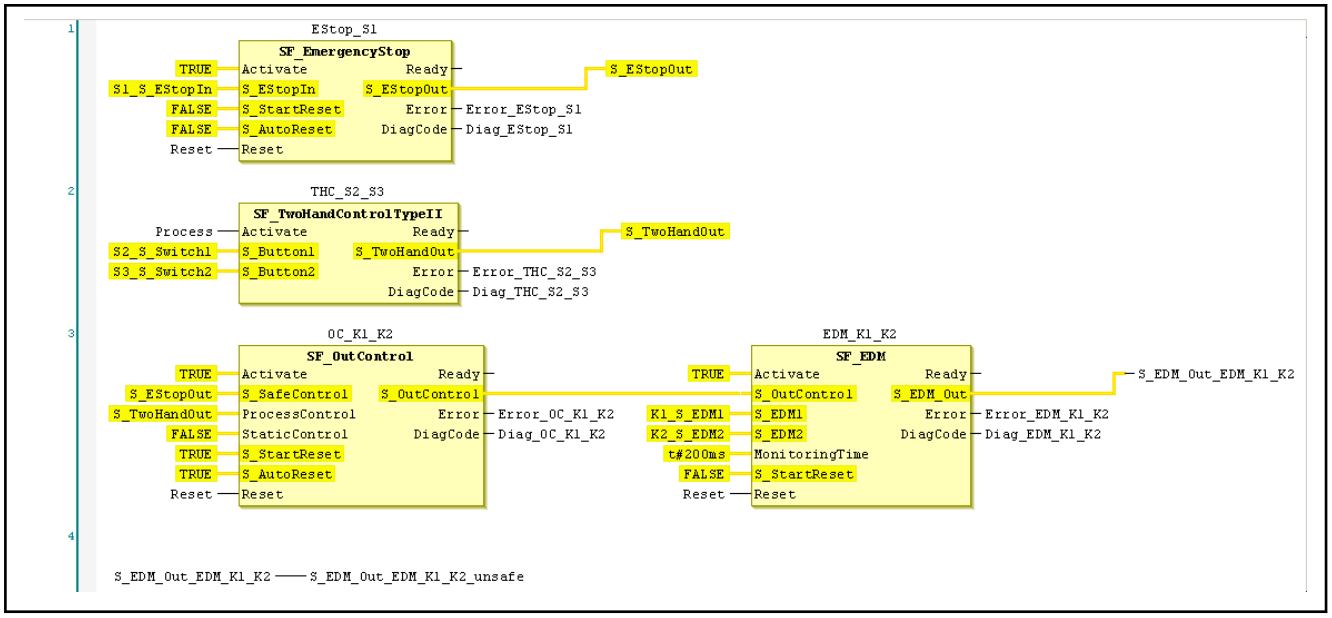


Abb. 7-33: Implementation zum Beispiel: Zwei-Hand-Kontrolle mit EDM

Name	Datentyp	Beschreibung
S1_S_EStopIn	SAFEBOOL	Not-Halt-Taster S1
S2_S_Switch1	SAFEBOOL	Schalter S2 verbunden mit Druckknopf 1 der Zwei-Hand-Kontrolle
S3_S_Switch2	SAFEBOOL	Schalter S3 verbunden mit Druckknopf 2 der Zwei-Hand-Kontrolle
K1_S_EDM1	SAFEBOOL	Rückmeldung externes Gerät K1
K2_S_EDM2	SAFEBOOL	Rückmeldung externes Gerät K2
Reset	BOOL	Reset von Standardsteuerung
Process	BOOL	Freigabe der Bewegung durch den Prozess von Standardsteuerung

Tab. 7-4: Eingänge:

Name	Datentyp	Beschreibung
S_EDM_Out_EDM_K1_K2	SAFEBOOL	Freigabesignal für Sicherheitssteuerung (Sichere Antriebe...)
S_EDM_Out_EDM_K1_K2_unsafe	BOOL	Freigabesignal für Standardsteuerung (nicht sicher)
Error_EStop_S1	BOOL	Fehlerflag von EStop_S1
Error_THC_S2_S3	BOOL	Fehlerflag von TCH_S2_S3
Error_OC_K1_K2	BOOL	Fehlerflag von OC_K1_K2
Error_EDM_K1_K2	BOOL	Fehlerflag von EDM_K1_K2
Diag_EStop_S1	WORD	Diagnosecode für EStop_S1, 16#8xxx: Regulärer Betrieb, 16#Cxxx bei Fehler in EStop_S1
Diag_THC_S2_S3	WORD	Diagnosecode für THC_S2_S3, 16#8xxx: Regulärer Betrieb, 16#Cxxx bei Fehler in THC_S2_S3

Name	Datentyp	Beschreibung
Diag_OC_K1_K2	WORD	Diagnosecode für OC_K1_K2, 16#8xxx: Regulärer Betrieb, 16#Cxxx bei Fehler in OC_K1_K2
Diag_EDM_K1_K2	WORD	Diagnosecode für EDM_K1_K2, 16#8xxx: Regulärer Betrieb, 16#Cxxx bei Fehler in EDM_K1_K2

Tab. 7-5: Ausgänge:

Zusätzliche Anmerkungen

Dieses Beispiel kann auch mit Kap. 16.4.18 "SF_TwoHandControlTypeIII" auf Seite 417 verwendet werden.

Der Eingang von **Activate** wurde der Einfachheit halber auf TRUE gesetzt. Dies kann in der Anwendung durch eine Variable ersetzt werden.

Baustein	Eingang	Konstanter Wert	Beschreibung
EStop_S1	S_StartReset	FALSE	Kein automatischer Reset, wenn die S-SPS gestartet ist
	S_AutoReset	FALSE	Kein automatischer Reset, Reset/Bestätigung durch Entwickler notwendig
OC_K1_K2	S_StartReset	TRUE	Automatischer Reset ist erlaubt, wenn die S-SPS gestartet ist
	S_AutoReset	TRUE	Automatischer Reset, kein Reset/Bestätigung durch Entwickler notwendig
	StaticControl	FALSE	Eine dynamische Änderung des Signals Appl_Control (steigende Flanke) wird gefordert nach Bausteinaktivierung oder eine getriggerte Sicherheitsfunktion (S_SafeControl auf FALSE)
EDM_K1_K2	S_StartReset	FALSE	Kein automatischer Reset, wenn die S-SPS gestartet ist
	MonitoringTime	T#200ms	Die maximale Antwortzeit der beiden Rückmeldesignale S_EDM1 und S_EDM2

Tab. 7-6: Informationen zu verwendeten Bausteinparametern

8 Online-Betrieb der Sicherheitssteuerung

8.1 Eingeloggter Zustand, Einführung

Analog zu IndraWorks-Standard unterstützt auch SafeLogic einen eingeloggten Zustand / Online-Modus.

Die Funktionalitäten des Online-Modus dienen dem Debuggen und der Diagnose bei der Entwicklung und der Verifikation einer Sicherheitsapplikation.

Die Online-Befehle können auf dem selektierten Geräteobjekt, den Geräte-Editoren und auf dem selektierten bzw. aktiven Applikationsobjekt ausgeführt werden. Die Online-Funktionalität der SafeLogic unterscheidet sich nicht grundsätzlich von der Online-Funktionalität von Standard SPS-Programmen.

Nachstehend werden die Besonderheiten der Online-Funktionalität von SafeLogic beschrieben.

Begriffserklärungen

- **Eingeloggt:** Im eingeloggten Zustand ist das Programmiersystem mit der Sicherheitssteuerung verbunden. Dies bedeutet, dass entweder eine Verbindung zur aktuellen, auf der Sicherheitssteuerung geladenen Applikation oder zur Bootapplikation (BA) auf der Sicherheitssteuerung besteht.

  Im eingeloggten Zustand können Debug-Befehle ausgeführt werden
- **Ausgeloggt:** Im ausgeloggten Zustand ist das Programmiersystem nicht mit der Steuerung verbunden
- **Sicherer Betrieb:** Zustand, in dem die Bootapplikation ungestört läuft
- **Debug-Betrieb (unsicherer Betrieb):** Zustand, in dem entweder statt der Bootapplikation eine Downloadapplikation läuft oder in dem der Lauf der Bootapplikation durch Debug-Befehle beeinflusst wurde, bzw. beeinflusst werden kann (siehe Kap. 8.6.2 "Debug-Modus und organisatorische Sicherheit" auf Seite 143).
- **Downloadapplikation:** Aktuelle, fehlerfrei übersetzte Sicherheitsapplikation, die beim Einloggen auf die Steuerung geladen wird

  Die Downloadapplikation ist nach dem Ausloggen nicht mehr auf der Steuerung.
- **Bootapplikation:** Eine Bootapplikation wird im eingeloggten Zustand aus der Downloadapplikation erzeugt und auf der Sicherheitssteuerung abgelegt

  Die Bootapplikation bleibt nach dem Ausloggen auf der Steuerung und startet nach dem Neustarten der Sicherheitssteuerung.
- **bestätigte Verbindung:** Beim Verbinden mit der Sicherheitssteuerung wird sichergestellt, dass die korrekte Steuerung angesprochen wird. Hierfür muss beim Einloggen der Bestätigungstaster an der Sicherheitssteuerung betätigt werden.
- **Fernzugang:** Der Fernzugang ermöglicht das Einloggen in die Sicherheitssteuerung zu Diagnosezwecken. Es sind keinerlei Änderungen am Programmablauf möglich. Der Bestätigungstaster muss nicht betätigt werden.

⚠ GEFAHR	**Der Entwickler ist dafür verantwortlich, dass die Sicherheit der Anlage/Maschine während des ganzen Zeitraums, in dem sich die Sicherheitssteuerung im Debug-Betrieb befindet, gewährleistet ist, z. B. durch Absperrungen um die Maschine (organisatorische Maßnahme).**

8.2 Verbindung zur Sicherheitssteuerung

Voraussetzungen für die Verbindung zur Sicherheitssteuerung:

- Für den vollen Zugriff sind die Voraussetzungen eine Netzwerkverbindung und die Verbindungsbestätigung.

  Vor jeder Verbindung zu einer Sicherheitssteuerung für den vollen Zugriff wird diese mittels einer eigenen Verbindungs-Id auf Konsistenz zwischen dem Geräteobjekt und der Sicherheitssteuerung geprüft. Ist die Verbindungs-Id noch nicht vorhanden oder inkonsistent, so muss der Anwender die "neue" Verbindung mit einem Druck auf den Bestätigungstaster an der Sicherheitssteuerung bestätigen. Mit der Verbindungsbestätigung bestätigt der Anwender, dass die Netzwerkverbindung ihn mit der richtigen Steuerung verbunden hat.
- Für den Fernzugriff sind die Voraussetzungen eine Netzwerkverbindung, das Fernpasswort und die Freischaltung des Fernzugriffs. Fernzugang hat seinen Anwendungsfall im Betrieb und wird in Kap. 13 "Betrieb und Administration der Steuerung" auf Seite 191 beschrieben (siehe Kap. 13.3 "Diagnose von Fehlern im Betrieb" auf Seite 197).



Für Entwicklung, Erzeugung und Debuggen von Sicherheitsapplikationen wird volle Online-Funktionalität zwischen IndraWorks und der Sicherheitssteuerung benötigt. Diese erfordert eine bestätigte Verbindung zu der vom Entwickler bestimmten Sicherheitssteuerung innerhalb des Steuerungsnetzwerks.



Fernzugang ist für die Verifikation und die Abnahme nicht qualifiziert.

Netzwerkverbindung und Verbindungsbestätigung mit Vergabe des Gerätenamens

1. Die Kommunikationseinstellungen für die Sicherheitssteuerung werden über die übergeordnete Standardsteuerung eingestellt.
2. Befehl **Einloggen** des Menüs **Debuggen** aktivieren
3. Im erscheinenden Dialog „Verbindung zur Sicherheitssteuerung“ die Verbindungsart „Bestätigte Verbindung“ auswählen.
4.



Der Gerätename der Sicherheitssteuerung wird im Dialog angezeigt. Er kann im Dialog **Gerätename ändern** geändert werden.

Die im Dialog **Verbindung zur Sicherheitssteuerung** beschriebene Aktion an der Sicherheitssteuerung (z. B. Taster auf der Steuerung drücken) durchführen.

Dieser Zustand wird durch die Sicherheitssteuerung durch blinken einer roten und einer grünen LED signalisiert.

Im Dialog "Verbindung zur Sicherheitssteuerung" den Gerätenamen ggf. ändern (Abb. 8-1 "Dialog Verbindung zur Sicherheitssteuerung" auf Seite 127).

Dabei sind die Hinweise zur Vergabe des Knotenamens zu beachten (Gerätename auf Seite 128).

5. Schaltfläche **OK** aktivieren
6. Je nachdem, ob sich keine Applikation, eine unveränderte oder eine veränderte Bootapplikation auf der Sicherheitssteuerung befinden, erscheinen entsprechende Dialoge, die zu prüfen und ggf. zu bestätigen sind
7. Im Dialog **Autorisierung** das gesetzte Administrationspasswort (Admin-Passwort) eingeben. (Default ist leer)

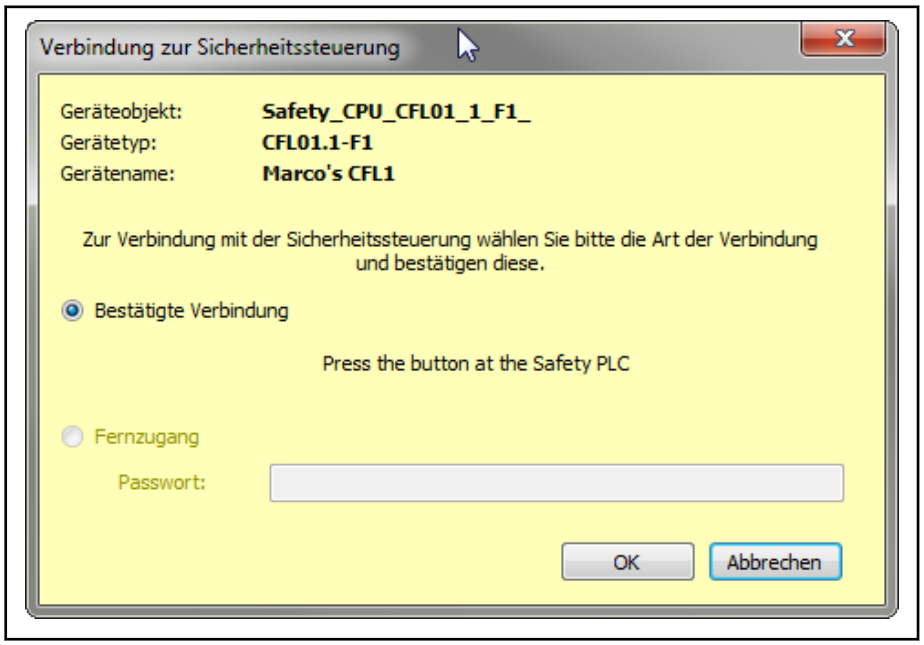


Abb. 8-1: Dialog "Verbindung zur Sicherheitssteuerung"

Verbindungsbestätigung

Die Verbindungsbestätigung erfolgt durch Drücken des Bestätigungstasters an der Sicherheitssteuerung.

Die Verbindung zur Sicherheitssteuerung erfolgt immer unter einem Benutzernamen. Ist der Entwickler nicht als Benutzer eingeloggt, so wird der aktuelle Benutzername des Betriebssystems (Windows) verwendet.

Drücken des Bestätigungstasters an der Sicherheitssteuerung

Bei Verbindung zur Steuerung muss einmal pro Sitzung direkt an der Sicherheitssteuerung der Bestätigungstaster gedrückt werden. Der Entwickler wird in dem Dialog **Verbindung zur Sicherheitssteuerung** zur Bestätigung der Verbindung durch Drücken des Bestätigungstasters an der Sicherheitssteuerung aufgefordert (Abb. 8-1 "Dialog Verbindung zur Sicherheitssteuerung" auf Seite 127).

Ohne diese Aktion an der Sicherheitssteuerung kann keine Online-Verbindung zur Sicherheitssteuerung hergestellt werden.

Während die Sicherheitssteuerung auf die Betätigung des Tasters wartet, blinken eine rote und eine grüne LED. Auf diese Weise wird überprüft, ob die Verbindung mit der richtigen Steuerung aufgebaut wurde. Sobald der Taster gedrückt wurde, wechseln die LEDs wieder in ihren vorherigen Zustand.

Erfolgt die Bestätigung durch den Entwickler nicht innerhalb der Wartezeit, so wird der entsprechende Dialog erneut geöffnet und der Entwickler kann die Aktion an der Sicherheitssteuerung wiederholen.

☞ Befinden sich in Projekten mit mehreren Sicherheitssteuerungen mehrere Sicherheitssteuerungen im Online-Modus, dann erscheint vor der Ausführung von Online-Befehlen ein Nachfrage-Dialog, auf welche Steuerung der Befehl gehen soll, so muss der Anwender den angezeigten Gerätenamen verifizieren, indem er den angezeigten Gerätenamen gegen die Erwartungshaltung prüft und im Fehlerfall den Online-Befehl abbricht.

Mit der Verbindungsbestätigung vergibt der Anwender der Steuerung einen Namen (Gerätenamen).

Gerätename

Der Gerätename ist ein vom Entwickler vergebener Name für eine Steuerung in seinem Netzwerk.

Durch ihn identifiziert sie die Steuerung bei Schreibbefehlen (siehe Debug-Modus und organisatorische Sicherheit auf Seite 143) und in Info-Anzeigen (siehe Informationen zu Firmware und Bootapplikation auf Seite 200).

Der Name der Steuerung (Gerätename) wird nicht im Projekt, sondern auf der Sicherheitssteuerung gespeichert.

⚠ GEFAHR **Es müssen eindeutige Gerätenamen für die Sicherheitssteuerungen im Netzwerk vergeben werden.**

Der Entwickler ist stets dafür verantwortlich, dass jeder Befehl an die richtige Steuerung geht.

Änderung des Gerätenamens der Sicherheitssteuerung

1. Führen Sie im Kontextmenü des Safety-Funktionsmoduls den Menüpunkt **Gerätenamen ändern** aus.
   → Der Dialog **Verbindung zur Sicherheitssteuerung** wird geöffnet.
2. Geben Sie im Dialog **Verbindung zur Sicherheitssteuerung** einen neuen, eindeutigen Namen für das Gerät ein (siehe Abb. 8-1 "Dialog Verbindung zur Sicherheitssteuerung" auf Seite 127)
3. Führen Sie die, im Dialog **Verbindung zur Sicherheitssteuerung** beschriebene Aktion an der Sicherheitssteuerung durch.
4. Bestätigen Sie den Dialog mit **OK**

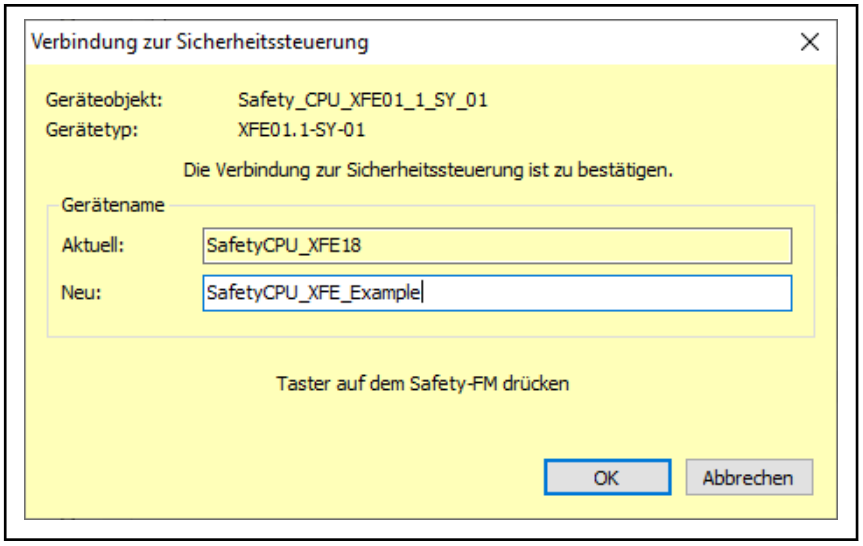


Abb. 8-2: Dialog "Gerätename ändern"

8.3 Einloggen auf der Steuerung und Wechsel in den Debug-Betrieb

Einloggen

Mit dem Befehl **Einloggen**, Menüpunkt Debuggen oder Kontextmenü der Applikation, wird eine Online-Verbindung zwischen der Applikation, für die der Befehl ausgeführt wird (aktive Applikation) und der Steuerung aufgebaut. Der Entwickler kann sich aus dem Projekt heraus nur auf die Applikation (Bootapplikation) auf der Steuerung einloggen, wenn die Applikation im Projekt und die Applikation auf der Steuerung übereinstimmen.

Deshalb findet vor der Ausführung des Einloggens ein Vergleich zwischen der Applikation im Projekt und der auf der Sicherheitssteuerung laufenden Applikation statt:

- Befindet sich keine Applikation auf der Steuerung, so wird der Entwickler in einem Dialog gefragt, ob die Applikation auf die Sicherheitssteuerung geladen werden soll (siehe Abb. 8-3 "Dialog beim Einloggen auf eine leere Steuerung" auf Seite 131). Wird die Applikation auf die Sicherheitssteuerung geladen (Download), so befindet sich die Sicherheitssteuerung im **Debug-Betrieb**. Die Applikation ist im Online-Modus.
- Haben die beiden Applikationen unterschiedliche Namen (d. h. es läuft eine andere Applikation auf der Sicherheitssteuerung), so wird in einem Dialog gefragt, ob die Applikation auf der Sicherheitssteuerung beendet und die aktuelle Applikation auf die Sicherheitssteuerung geladen werden soll . Wird diese aktuelle Applikation auf die Sicherheitssteuerung geladen (Download), so befindet sich die Sicherheitssteuerung im **Debug-Betrieb**. Die Applikation ist im Online-Modus.
- Wurde die aktuelle Applikation geändert, wird in einem Dialog gefragt, ob die Applikation auf der Sicherheitssteuerung beendet und die aktuelle Applikation auf die Sicherheitssteuerung geladen werden soll (siehe Abb. 8-4 "Dialog beim Einloggen auf Steuerung mit geänderter Applikation" auf Seite 131). Wird diese aktuelle Applikation auf die Sicherheitssteuerung geladen (Download), so befindet sich die Sicherheitssteuerung im **Debug-Betrieb**. Die Applikation ist im Online-Modus.
- Sind die aktuelle Applikation und die Bootapplikation auf der Sicherheitssteuerung identisch und haben den gleichen Pin, erfolgt die Meldung, dass der Login auf eine gepinnte Applikation erfolgt. Die Sicher-

heitssteuerung befindet sich im **sicheren Betrieb**. Die Applikation ist im Online-Modus.

- Sind die aktuelle Applikation und die Bootapplikation auf der Sicherheitssteuerung identisch und nicht gepinnt, wird ohne Dialog eingeloggt. Die Sicherheitssteuerung befindet sich im **sicheren Betrieb**. Die Applikation ist im Online-Modus.

In den Fällen, die zum Download der Applikation und in den Wechsel in den Debug-Betrieb führen, wird der Entwickler angewiesen, die Sicherheit organisatorisch herzustellen.



Wenn die aktuell auf der Sicherheitssteuerung laufende Applikation durch diese Aktion beendet wird, wird auch der sichere Betriebszustand der Steuerung beendet und die Ausgänge der Sicherheitssteuerung gehen sofort in den sicheren Zustand. Dies führt typischerweise einen harten Stopp der Maschine herbei.

⚠ GEFAHR

Der Entwickler ist dafür verantwortlich, dass die Sicherheit der Anlage/Maschine während des ganzen Zeitraums, in dem sich die Sicherheitssteuerung im Debug-Betrieb befindet, gewährleistet ist.

Applikation auf die Steuerung laden

Der Dialog „Applikation auf die Steuerung laden“ bietet zwei Varianten zum Laden der Sicherheitsapplikation:

Voraussetzung: Sie haben sichergestellt, dass die organisatorische Sicherheit an der gesamten Anlage hergestellt ist.

- **Temporär laden**: Die Applikation wird auf die Sicherheitssteuerung geladen. Die sogenannte Downloadapplikation befindet sich temporär auf der Sicherheitssteuerung. Der Zustand der Sicherheitsteuerung ist: UNSICHER (DL)
- **Bootapplikation erzeugen**: Die Applikation wird auf die Sicherheitssteuerung geladen. Danach wird aus ihr automatisch eine Bootapplikation erzeugt und auf der Sicherheitssteuerung abgelegt. Es findet kein automatischer Start der Bootapplikation statt. Der Zustand der Sicherheitsteuerung ist: UNSICHER (BA).

Bei beiden Varianten befindet sich die Sicherheitssteuerung nach dem erfolgreichen Download im Debug-Betrieb. Die Applikation ist im eingeloggten Zustand / Online-Modus.

Erst mit einem **Neustart** der Bootapplikation wechselt die Sicherheitssteuerung in den sicheren Betrieb.



- Beim Verwendung der Variante **Bootapplikation erzeugen** ist es hilfreich, die Applikation vorher zu pinnen.
- Der Befehl **Bootapplikation erzeugen** im Menü **Debuggen** wird deaktiviert, nachdem für die Downloadapplikation eine Bootapplikation erzeugt wurde.
- Das gleiche Ergebnis wie mit der Variante **Bootapplikation erzeugen** wird erreicht, wenn im Anschluss an die Variante **Temporär laden** im Menü **Debuggen** der Befehl **Bootapplikation erzeugen** manuell ausgeführt wird.

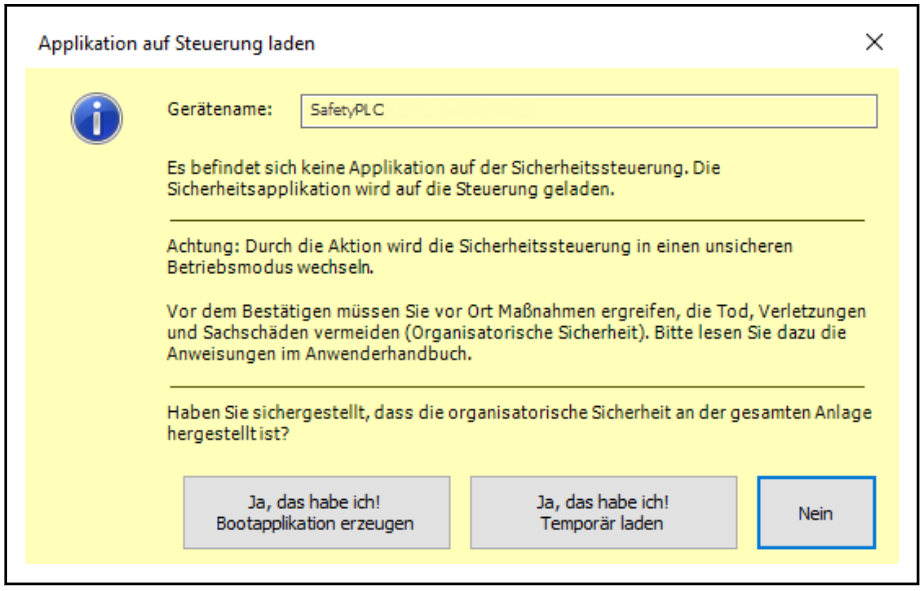


Abb. 8-3: Dialog beim Einloggen auf eine "leere" Steuerung

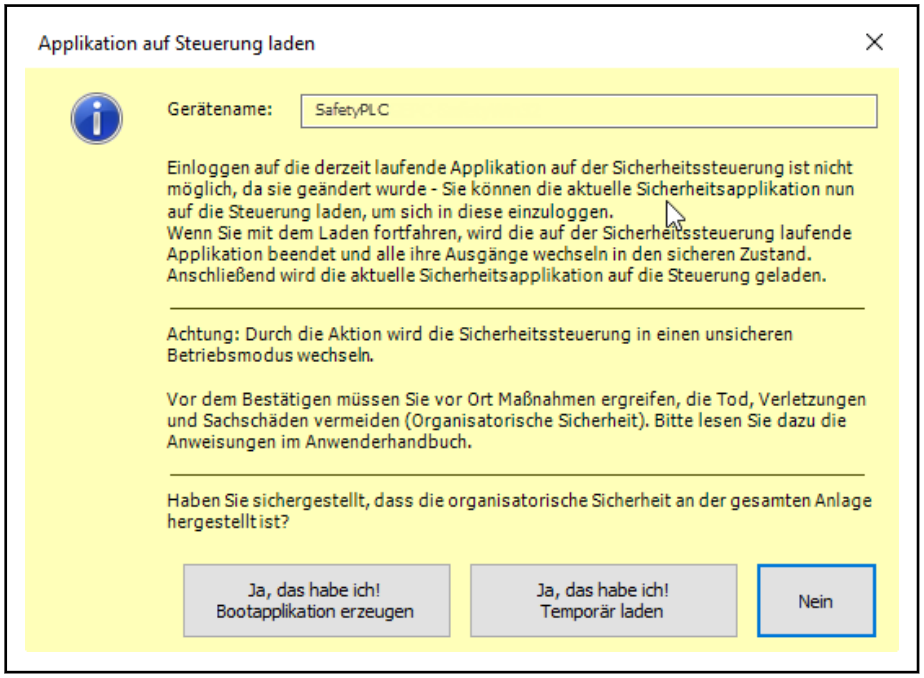


Abb. 8-4: Dialog beim Einloggen auf Steuerung mit geänderter Applikation

☞ Applikationen mit gleichem Namen werden als unterschiedlich betrachtet, wenn die Liste der Objekte nicht identisch ist oder wenn wenigstens eine Prüfsumme der Objekte unterschiedlich ist oder wenn die Pin-Prüfsumme unterschiedlich ist. Siehe Safety-Applikationsobjekt Kap. 6.6 "Sicherheitsapplikationsobjekt SafetyApp" auf Seite 41.

Download

Mit einem Download wird eine Sicherheitsapplikation temporär auf die Steuerung geladen. Ein Download erfolgt durch Aktivierung des Befehls **Einloggen** des Menüs **Debuggen**.

Vor dem Einloggen wird das Projekt auf Korrektheit überprüft. Ist das Projekt fehlerhaft, so ist ein Einloggen auf die Steuerung nicht möglich.

Solange sich eine temporär auf die Sicherheitssteuerung geladene Applikation auf der Sicherheitssteuerung befindet, ist die Sicherheitssteuerung im Debug-Betrieb!

Ausloggen

Der Debuggen-Befehl **Ausloggen** beendet die bestehende Online-Verbindung zur Applikation.

Es gibt folgende Varianten:

- Der Entwickler ist auf die laufende Bootapplikation eingeloggt und diese ist im sicheren Betrieb:
  - die Online-Verbindung wird beendet
  - die Bootapplikation läuft weiter
- Der Entwickler ist auf die Applikation eingeloggt, die sich im Debug-Betrieb (Downloadapplikation oder Bootapplikation) und auf der Sicherheitssteuerung ist eine Bootapplikation hinterlegt:
  - Applikation wird entladen
  - Der Entwickler wird in einem Dialog gefragt, ob die hinterlegte Bootapplikation geladen und gestartet werden soll (siehe Abb. 8-5 "Ausloggen bei unterschiedlicher Bootapplikation und Downloadapplikation" auf Seite 132).

    In diesem Fall wird die Debug-Freischaltung beendet, die Sicherheitssteuerung geht in den sicheren Betrieb.
- Der Entwickler ist auf die Applikation eingeloggt (siehe Abb. 8-6 "Dialog beim Ausloggen mit Laden der Bootapplikation" auf Seite 133) und es befindet sich keine Bootapplikation auf der Sicherheitssteuerung:
  - Die Applikation wird entladen

Wird automatisch durch das LZS ausgeloggt (Fehlerfall), so wird die temporäre Downloadapplikation beendet und die Bootapplikation nicht automatisch gestartet.

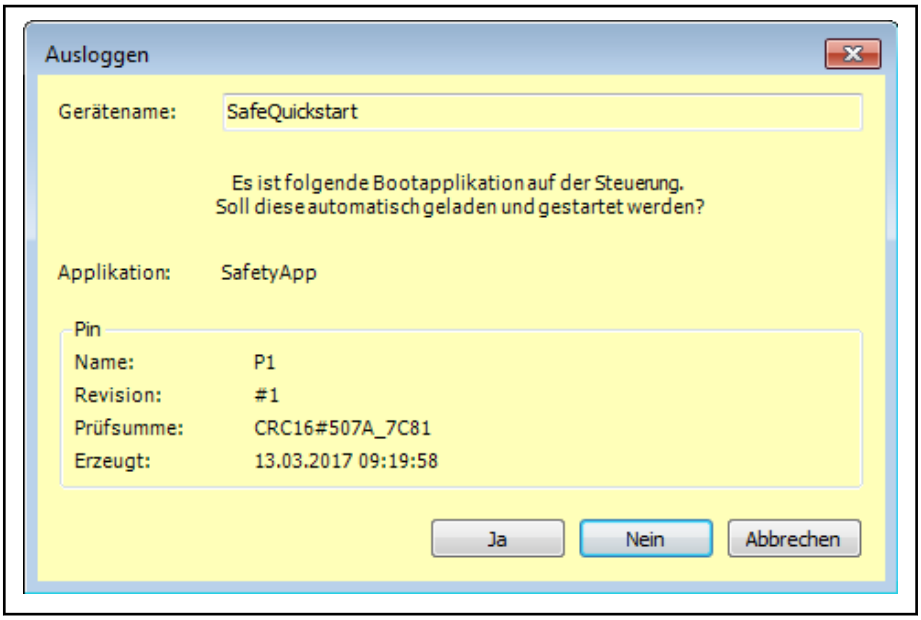


Abb. 8-5: Ausloggen bei unterschiedlicher Bootapplikation und Downloadapplikation

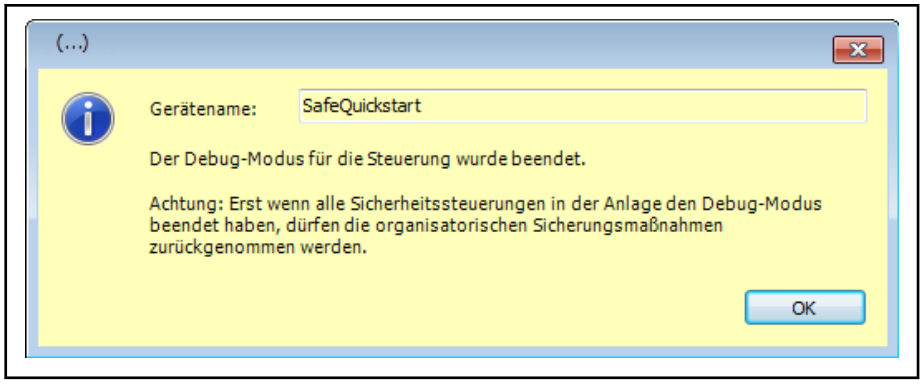


Abb. 8-6: Dialog beim Ausloggen mit Laden der Bootapplikation

8.4 Mehrfacher Download

Der Befehl "Mehrfacher Download" im Menü "Debuggen" bewirkt folgende Aktionen für die im IW-Projekt enthaltenen Applikationen und Sicherheitsapplikationen:

- Die syntaktische Überprüfung / Codegenerierung
- Das Laden auf die Standard- und Sicherheitssteuerungen
- Das Starten der Applikationen und Sicherheitsapplikationen

Voraussetzungen zum Mehrfachen Download

- Es wird eine bestätigte Verbindung zur Sicherheitssteuerung benötigt, der Fernzugang reicht nicht aus
- Das Admin-Passwort wird zur Autorisierung benötigt, falls vergeben

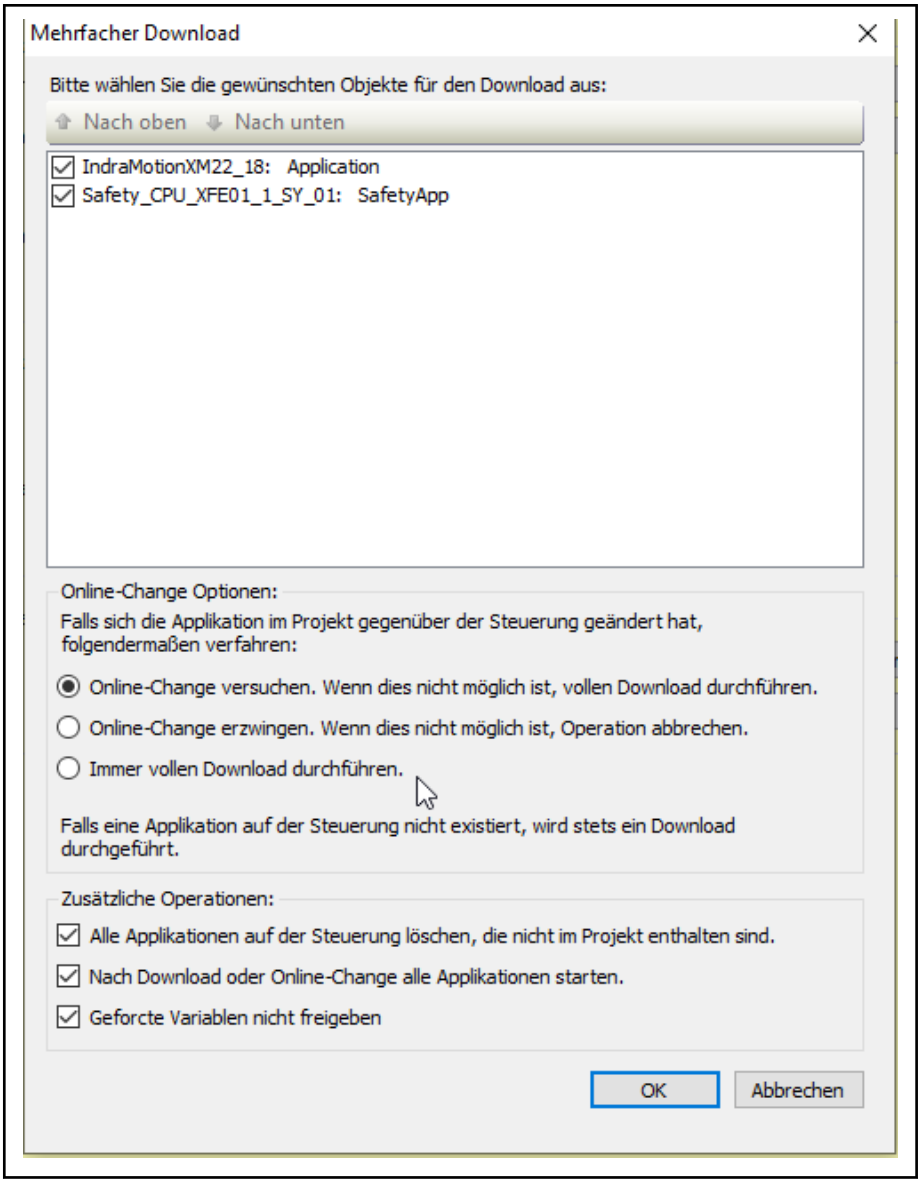


Abb. 8-7: Dialog "Mehrfacher Download"

Der Dialog "Mehrfacher Download" listet alle im IW-Projekt befindlichen Standard-Steuerungen / Applikationen und Sicherheitssteuerungen / Sicherheitsapplikationen auf.

Wählen Sie mittels der Checkbox die Steuerungen aus, die am Download teilnehmen sollen.

☞ Im Dialog "Mehrfacher Download" beziehen sich die "Online-Change Optionen" ausschließlich auf die Standardsteuerung.

Für die Sicherheitssteuerung gibt es nur die Möglichkeit, die Sicherheitsapplikation auf die Sicherheitssteuerung zu laden und daraus eine Bootapplikation zu erzeugen.

Wenn der Befehl "Mehrfacher Download" ausgeführt wird, muss der Anwender in jedem Online-Dialog auf den angezeigten Gerätenamen achten, damit er weiß, für welche Steuerung er den Dialog bestätigt oder von welcher Steuerung die Meldung im Dialog angezeigt wird.

Für jede in der Liste ausgewählte Sicherheitssteuerung / Sicherheitsapplikation werden folgende Schritte zum Zweck des Downloads durchlaufen:

- Syntaktische Überprüfung der Sicherheitsapplikation
- Aufbau einer bestätigten Verbindung zur Sicherheitssteuerung.

  Bei Bedarf erscheint der Dialog "Verbindung zur Sicherheitssteuerung".
- Prüfung, ob ein Download erforderlich ist:
  - Wenn die Sicherheitsapplikation bereits auf der Sicherheitssteuerung vorliegt, erfolgt kein Download.
    - Wenn die Sicherheitsapplikation gepinnt ist, erscheint der Dialog "Einloggen".
  - Wenn sich auf der Sicherheitssteuerung keine oder eine andere Sicherheitsapplikation befindet, wird der Dialog "Applikation auf Steuerung laden" geöffnet.
    - Wenn Sie sichergestellt haben, dass die organisatorische Sicherheit an der gesamten Anlage hergestellt ist, wählen Sie im Dialog "Bootapplikation erzeugen" aus.
- Der Dialog "Autorisierung" erfragt die Zustimmung für den schreibenden Zugriff.

Nach der Durchführung aller Downloads erfolgt für jede ausgewählte Sicherheitssteuerung / Sicherheitsapplikation die Erzeugung und der automatische Start der Bootapplikation mit folgenden Schritten:

- Der Dialog "Bootapplikation erzeugen" zeigt Informationen zur Bootapplikation an
- Meldung über die erfolgreiche Erzeugung der Bootapplikation
- Meldung, dass der Debug-Modus beendet wurde
- Abfrage, ob die Bootapplikation auf der Sicherheitssteuerung neu gestartet werden soll
- Meldung, dass die Bootapplikation gestartet wurde

Nach erfolgtem Download und Start der Applikationen erscheint der Dialog "Mehrfacher Download - Ergebnis", der den Status aller ausgewählter Applikationen und Sicherheitsapplikationen auflistet.

→ Die beteiligten Applikationen / Sicherheitsapplikationen befinden sich im ausgeloggten Zustand.



Der Gerätename der Sicherheitssteuerungen sollte so gewählt werden, dass der Anwender auch auf die zugehörige Standardsteuerung schließen kann.



Wird im Dialog "Applikation auf Steuerung laden" die Option "Temporär laden" ausgewählt, wird der Ladevorgang für die Sicherheitsapplikation abgebrochen.

Im Dialog "Mehrfacher Download - Ergebnis" erscheint eine entsprechende Fehlermeldung.

Wird im Dialog "Mehrfacher Download" die Option "Nach Download oder online-Change alle Applikation starten" deaktiviert, dann findet für die Sicherheitssteuerungen kein automatischer Start der heruntergeladenen Bootapplikation statt.

Bei einem späteren Einloggen in die Sicherheitssteuerung läuft dann die neue Bootapplikation noch nicht. Dem Anwender wird in diesem Fall die Option angeboten, die Sicherheitsapplikation aus dem IndraWorks-Projekt erneut in die Sicherheitssteuerung zu laden.

Workaround:

Nach dem Mehrfachen Download für jede Sicherheitssteuerung den Befehl "Neu starten" manuell ausführen oder die Sicherheitssteuerungen mittels Spannung Aus /Ein neu starten.

Weiterführende Informationen

- Die vollständige Beschreibung zum Befehl "Mehrfacher Download" ist in folgender Dokumentation enthalten:

  IndraLogic Programmiersystem

  DOK-IWORKS-IL2GPRO*V15-APRS

8.5 Erzeugen und Neustart der Bootapplikation

Erzeugen der Bootapplikation

Eine laufende Applikation wird durch Aktivierung des Befehls **Bootapplikation erzeugen** des Menüpunktes **Debuggen** als Bootapplikation erzeugt und auf der Sicherheitssteuerung abgelegt. Der Befehl steht nur zur Verfügung, wenn der Entwickler auf der Sicherheitssteuerung eingeloggt ist.

Die Erzeugung einer Bootapplikation kann nur nach expliziter Bestätigung durch den Entwickler ausgeführt werden. In dem Bestätigungsdialog wird der Status der Applikation bzw. Pin-Kennung angezeigt.

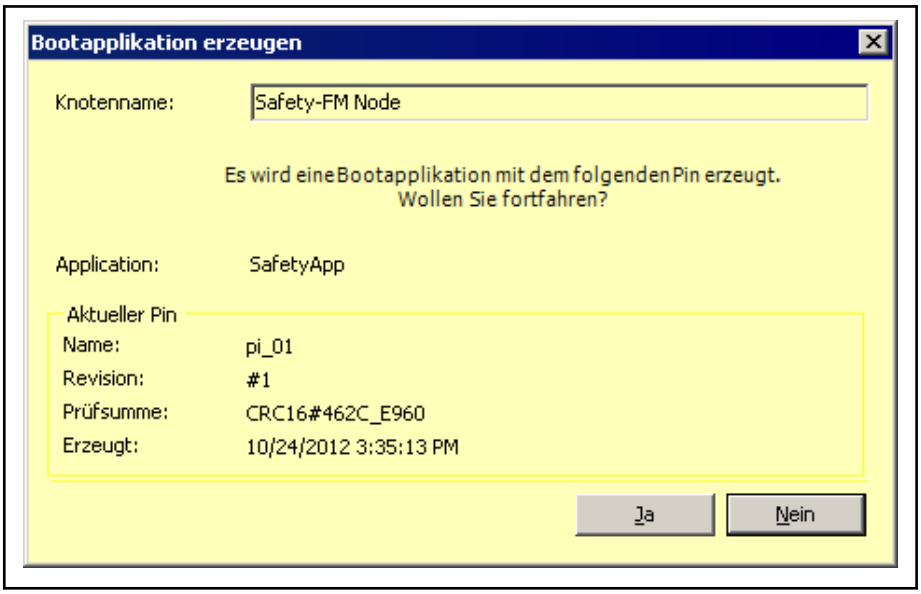


Abb. 8-8: Bootapplikation erzeugen

Mögliche Status der Applikation:

- Applikation wurde noch nicht gepinnt (In Work),
- Applikation ist nicht konsistent mit dem Pin (In Work),
- Applikation ist konsistent mit dem Pin,
- Applikation ist konsistent mit dem Pin, jedoch nicht mit dem Pin der zuletzt aus diesem Projekt erzeugten Bootapplikation (siehe Abb. 8-9 "Bootapplikation erzeugen bei geändertem Pin" auf Seite 137).

Wird in diesen Fällen nach Bestätigung durch den Entwickler eine Bootapplikation erzeugt, so bleibt die Downloadapplikation geladen und die Steuerung bleibt im Debug-Betrieb.

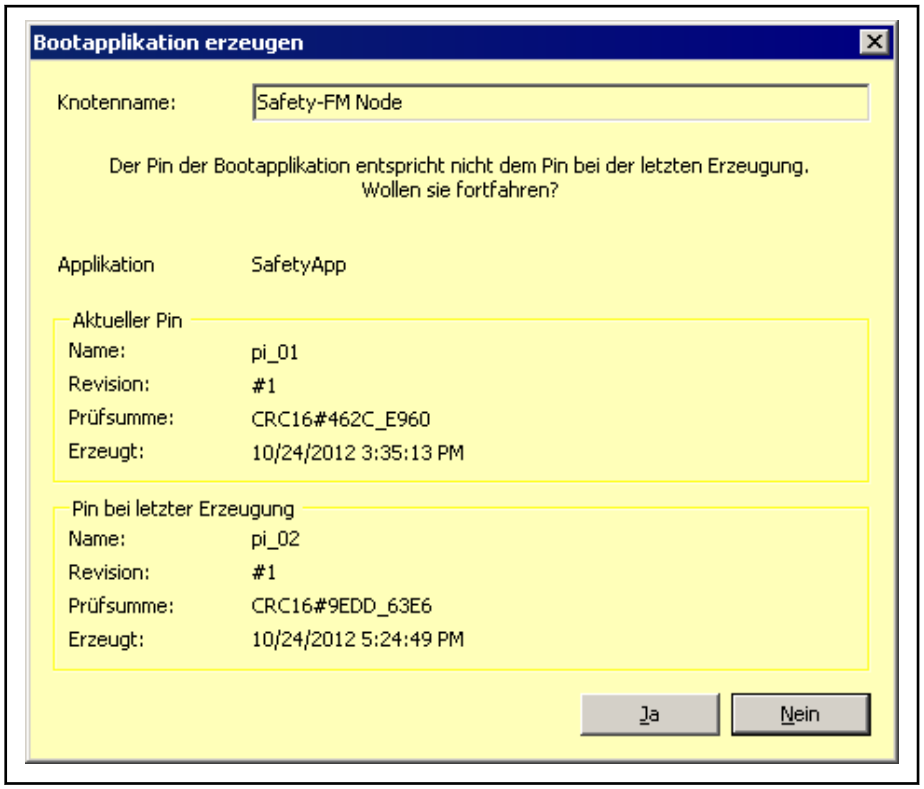


Abb. 8-9: Bootapplikation erzeugen bei geändertem Pin

⚠ GEFAHR

Konnte der Befehl "Bootapplikation erzeugen" nicht erfolgreich ausgeführt werden, so kann es nach Neustart zu einer Gefährdung in der Anlage kommen, weil diese unter Umständen noch mit der alten Applikation anläuft.

Der Service-Mitarbeiter muss auf die Rückmeldung warten, ob der Befehl erfolgreich ausgeführt wurde.

Bleibt die Rückmeldung aus, so ist die Steuerung so zu behandeln, als ob die Bootapplikation nach Neustart wieder gestartet werden kann.

Neustart der Bootapplikation

Eine Bootapplikation auf der Steuerung wird durch Aktivierung des Befehls **Neu starten** (Registerkarte **Safety Online Information** der Steuerung) oder automatisch nach Einschalten der Steuerung neu gestartet.

Neustart bedeutet nicht, dass die Anlage losläuft. Der Entwickler legt in der Sicherheitsapplikation fest, ob die PLCopen Bausteine und die sicheren Ausgangsmodule automatisch anlaufen (Auto-Reset) oder erst durch ein Standardsignal (Reset).

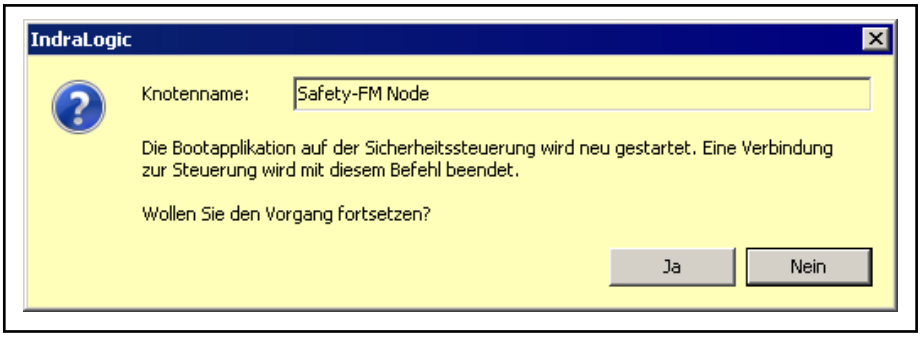
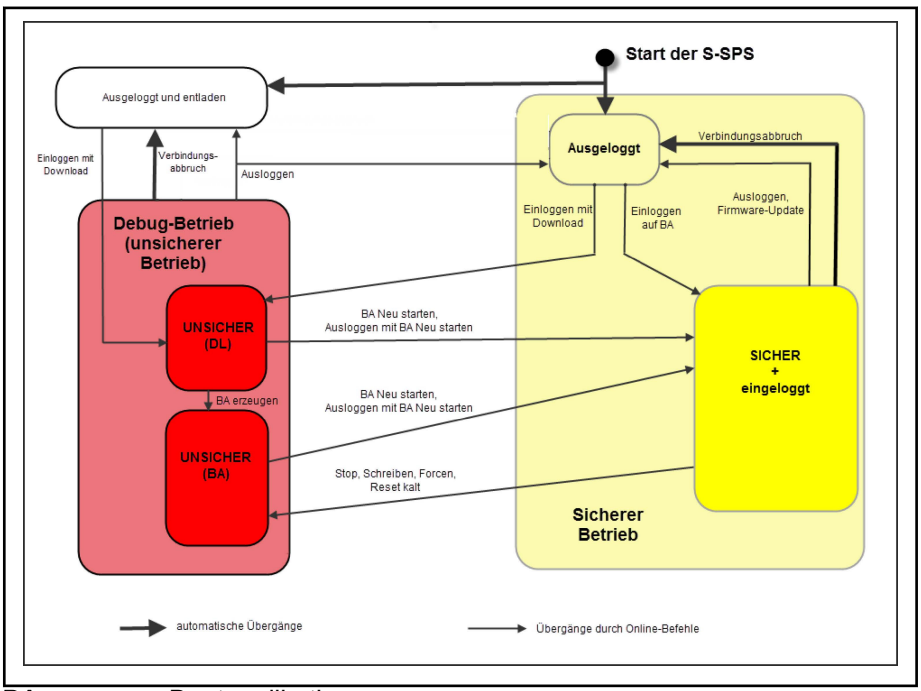


Abb. 8-10: Dialog beim Neustarten der Bootapplikation

8.6 Zustände einer Sicherheitssteuerung

8.6.1 Betriebszustände und Laden einer Sicherheitsapplikation



BA Bootapplikation
DL Downloadapplikation

Abb. 8-11: Betriebsmodi der Sicherheitssteuerung

Betriebszustände der Sicherheitssteuerung: sicher und unsicher

Die beiden möglichen Zustände einer mit SafeLogic programmierten Sicherheitssteuerung sind **sicher** und **unsicher**.

Als **sicherer Betrieb** wird der Modus der Sicherheitssteuerung bezeichnet, in dem eine Bootapplikation geladen ist und die Steuerung nicht im Debug-Modus betrieben wird.

Die Sicherheitssteuerung befindet sich so lange im sicheren Betrieb, wie die Bootapplikation läuft und der Entwickler nicht schreibend darauf zugreift. Sobald ein schreibender Zugriff stattfindet, wechselt die Steuerung in den Debug-Betrieb.

Die Steuerung bleibt auch im sicheren Zustand, wenn ein Login auf die Steuerung erfolgt und Variablenwerte in SafeLogic angezeigt werden. Erst ein schreibender Zugriff wie das Forcen eines Wertes veranlasst die Steuerung, in den Debug-Betrieb zu wechseln.



Der Zustand einer nicht geladenen Applikation ist zwar ebenfalls sicher, wird aber nicht als sicherer Betrieb bezeichnet.

Befindet sich eine Bootapplikation auf der Sicherheitssteuerung, so läuft diese Bootapplikation mit dem Start der Steuerung an, und die Steuerung befindet sich im sicheren Betriebszustand.

Wird die Steuerung dazu veranlasst vom sicheren in den unsicheren Zustand zu wechseln, so muss der Entwickler den Wechsel in den unsicheren Zustand bestätigen.

⚠ VORSICHT **Der angezeigte Gerätename muss geprüft werden**

Der Gerätename muss die richtige Steuerung identifizieren. Nur so ist sichergestellt, dass der Befehl an die richtige Steuerung geht.

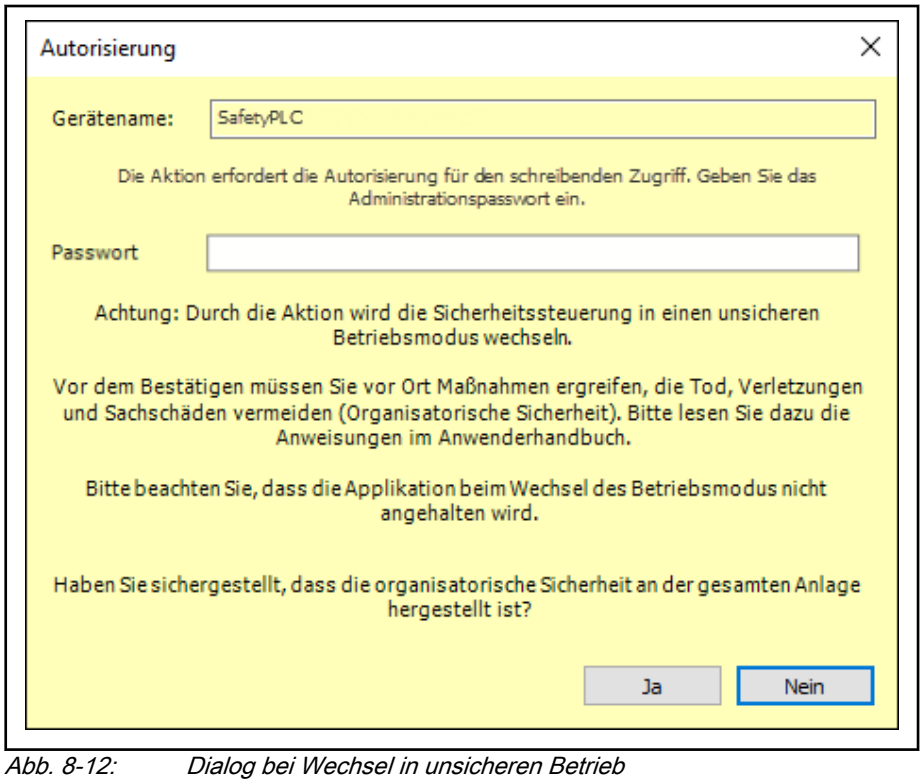


Abb. 8-12: Dialog bei Wechsel in unsicheren Betrieb

Hinweise siehe Kap. 8.6.2 "Debug-Modus und organisatorische Sicherheit" auf Seite 143

Laden einer Applikation

Wird die Applikation mit einem Download auf die Steuerung geladen (siehe Kap. 8.3 "Einloggen auf der Steuerung und Wechsel in den Debug-Betrieb" auf Seite 129), ist die Sicherheitssteuerung immer im unsicheren Zustand. In diesem Applikationszustand kann Debuggen und Start/Stop auf der Steuerung durchgeführt werden.

Zustandsanzeige der Sicherheitssteuerung im IndraWorks

Ob sich die Sicherheitssteuerung im sicheren oder unsicheren Zustand und die Applikation im Stop- oder Läuft-Zustand befindet, wird in der in der allgemeinen Statuszeile von IndraWorks am unteren Fensterrand angezeigt.

☞ Es wird der Status der aktiven Applikation angezeigt, unabhängig von den geöffneten Editoren.

Beispiele für im Online-Modus angezeigte Informationen in der untersten Statuszeile:

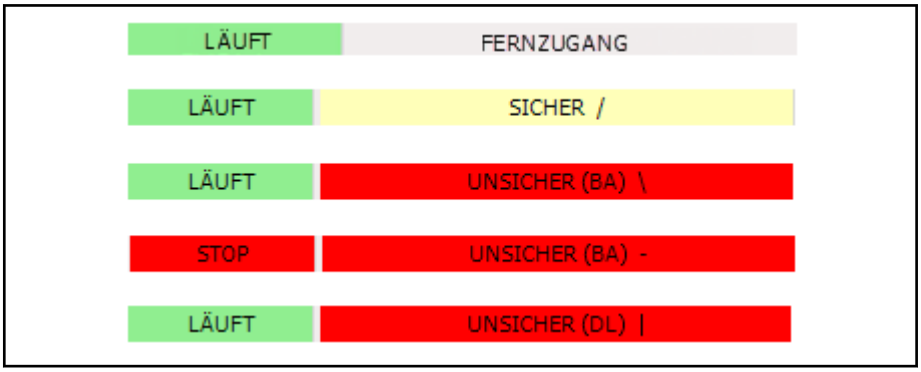


Abb. 8-13: links: Zustände der Sicherheitsapplikation, rechts: Zustände der Sicherheitssteuerung

Zustände der Sicherheitsapplikation in IndraWorks

- **LÄUFT**, grün hinterlegt
- **STOP**, rot hinterlegt: die Applikation pausiert
- **BEENDET**, rot hinterlegt: die Applikation wurde beendet aufgrund eines Laufzeitfehlers

Zustände der Sicherheitssteuerung:

- **FERNZUGANG**, grau hinterlegt, Zugriff auf Sicherheitssteuerung über Fernzugang
- **SICHER**, gelb hinterlegt, wenn die Bootapplikation läuft
- **UNSICHER (DL)**, rot hinterlegt: Downloadapplikation in Debug-Modus
- **UNSICHER (BA)**, rot hinterlegt: Bootapplikation in Debug-Modus
- **ENTLADEN**, grau hinterlegt: die laufende Applikation auf der Steuerung wurde entladen (kein Applikationsstatus mehr)
- **Force Aktiv**, wird angezeigt, falls momentan Werte geforct sind.
- **AUSNAHME** rot hinterlegt: zeigt unter speziellen Umständen, wenn Login bestehen bleibt, einen Systemfehler an (gewöhnlich erfolgt gleich ein Verbindungsabbruch).

„Force Aktiv“ wird zusätzlich zum Zustand der Sicherheitssteuerung angezeigt, falls momentan Werte geforct sind.

Der Zustand der Sicherheitsapplikation wird im eingeloggten Zustand auch im Projektbaum bei der aktiven Sicherheitsapplikation angezeigt.

☞ Außer bei Zustand "FERNZUGANG" wird hinter dem Zustand der Sicherheitssteuerung ein kreisender Balken **- \ | / -** angezeigt. Wenn er einfriert, kann sich der Status der Sicherheitssteuerung schon weiter verändert haben, ohne dass es angezeigt wurde.

Zustandsanzeige an der Sicherheitssteuerung

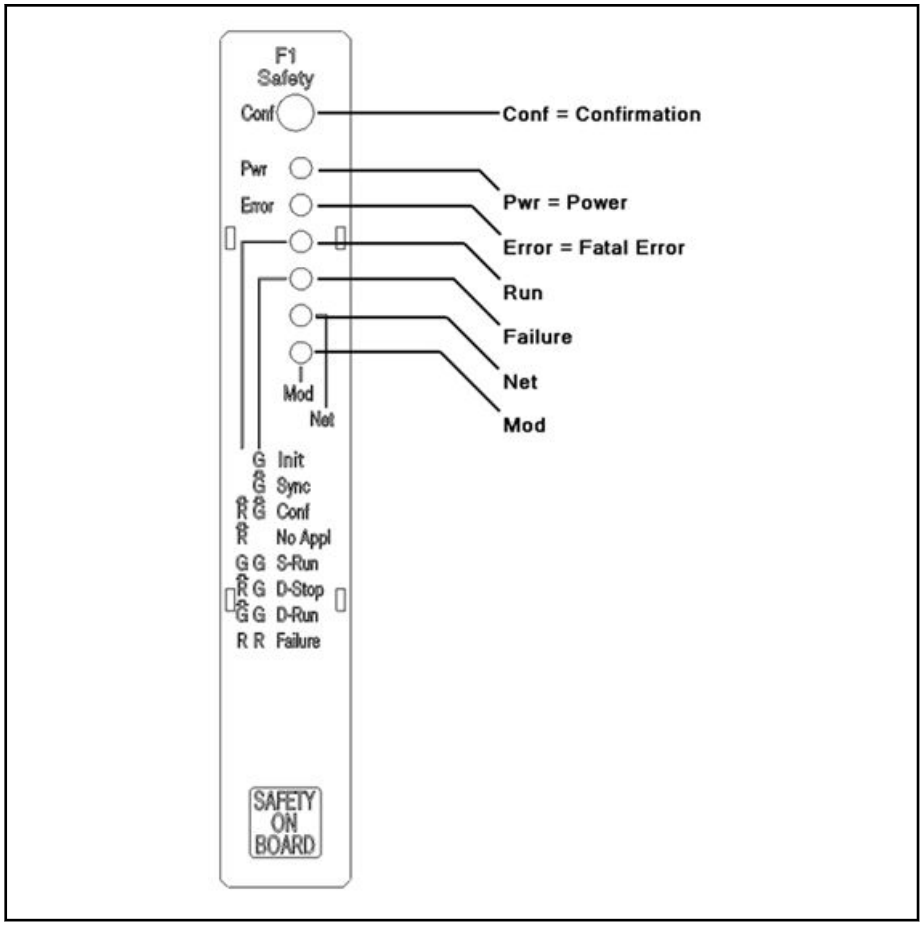


Abb. 8-14: Anzeige- und Bedienelemente

Die Bedeutung der Anzeige-LED sind nachfolgender Tabelle zu entnehmen:

LED	Farbe	Beschreibung
Pwr	Aus	Keine Spannungsversorgung durch die Standardsteuerung liegt an.
	Rot	Fehler: Die Spannungsüberwachung des Safety-Funktionsmoduls hat die sichere Logikspannung abgeschaltet.
	Grün	Alle internen Spannungen des Safety-Funktionsmoduls sind im zulässigen Bereich.
Error	Aus	Das Safety-Funktionsmodul ist betriebsbereit.
	Rot	Das Safety-Funktionsmodul ist nicht betriebsbereit. Diese LED leuchtet während der Initialisierungsphase. Leuchtet die LED nach der Initialisierung weiter (ca. 1 min) oder wird die LED im Betrieb dauerhaft eingeschaltet, haben integrierte Selbsttests einen "fatalen Fehler" (z. B. Hardwarefehler) festgestellt. Das Safety-Funktionsmodul ist dann in der Regel defekt und muss ausgetauscht werden. Kontaktieren Sie den Bosch Rexroth Service.

Tab. 8-1: LED zur Betriebsbereitschaft

LED "Run" (Farbe, Status)	LED "Failure" (Farbe, Status)	Betriebszustand des Safety-Funktionsmoduls
Aus	Aus	Keine Spannungsversorgung vorhanden oder Hardware-Reset ausgelöst
Aus	Grün	Initialisierungsphase 1. Teil (ca. 30 s) läuft
Rot	Aus	Initialisierungsphase 2. Teil (ca. 10 s) läuft
Rot, blinkend	Aus	Bereit zum Download Im Safety-Funktionsmodul ist keine gültige Applikation oder ein Download-Vorgang ist noch nicht abgeschlossen.
Rot	Grün, blinkend	Aufforderung zur Betätigung des Tasters "Conf"
Rot	Grün	SAFETY STOP
Grün	Grün	SAFETY RUN
Rot, blinkend	Grün	DEBUG STOP
Grün, blinkend	Grün	DEBUG RUN oder SERVICE RUN
Rot	Rot	FAILURE STOP, Ausnahmezustand des Safety-Funktionsmoduls aufgrund eines internen Fehlers. Die unterschiedlichen Anzeigen dienen der Fehlereingrenzung.
Rot	Rot, blinkend	
Rot, blinkend	Rot, blinkend	
Rot, blinkend	Grün, blinkend	Firmware-Update läuft
Grün, blinkend	Rot, blinkend	Spezieller Betriebsmode für Bosch Rexroth - Service-Personal

Tab. 8-2: Anzeige des Betriebszustands des Safety-Funktionsmoduls

LED	Farbe, Status	Beschreibung
Mod	Aus	Keine Spannungsversorgung vorhanden
	Grün, blinkend	Standby-Zustand
	Grün	Normalbetrieb
	Rot, blinkend	Beseitigbarer Fehler
	Rot	Nicht beseitigbarer Fehler
	Rot/Grün, blinkend	Selbsttest oder Konfiguration
Net	Aus	Keine Spannungsversorgung vorhanden
	Grün, blinkend	Keine gültige Busverbindung
	Grün	Busverbindung OK
	Rot, blinkend	Time-Out
	Rot	Kritischer Verbindungsfehler
	Rot/Grün, blinkend	Kommunikationsfehler

Tab. 8-3: Spezielle Anzeigen für CIP Safety on sercos



Die Status-LEDs der Sicherheitssteuerung zeigen keine gesicherten Informationen an.



Der Betriebszustand der Standardsteuerung (RUN, STOP) hat keinen Einfluss auf den Betriebszustand der Sicherheitssteuerung.



Der Betriebszustand der Sicherheitssteuerung hat keinen Einfluss auf den Betriebszustand der Standardsteuerung.



Eine zur Standardsteuerung unterschiedliche Konfiguration hat weder Einfluss auf den Betriebszustand der Sicherheitssteuerung noch auf den Betriebszustand der Standardsteuerung.

8.6.2 Debug-Modus und organisatorische Sicherheit

Wechsel in den Debug-Betrieb

Folgende Befehle führen zum Wechsel in den Debug-Betrieb:

Befehle des Menüpunktes Debuggen:

- **Reset Kalt**
- **Start**
- **Stop**

  (siehe Kap. 8.7.5 "Debuggen-Befehle: Start, Stop, Reset kalt " auf Seite 150)
- **Werte schreiben**
- **Werte forcen**
- **Forcen für alle Werte aufheben**

  (siehe Kap. 8.7.4 "Debuggen-Befehle: Schreiben/Forcen" auf Seite 148)

Der Befehl **Start** des Menüs **Debuggen** kann nur im Debug-Modus ausgeführt werden, wenn die Applikation im Stop-Zustand ist.

⚠ GEFAHR **Beim Debug-Betrieb bestehen Einschränkungen in der Sicherheit der Anlage.**

Vor jedem Wechsel vom sicheren in den unsicheren Betriebs der Sicherheitssteuerung muss der Anwender sicherstellen, dass die organisatorische Sicherheit der Anlage sichergestellt ist.

⚠ GEFAHR

Während des Debug-Betriebs befindet sich die Sicherheitssteuerung immer im unsicheren Zustand!

Dabei liegt es in der Verantwortung des Entwicklers, dass die Sicherheit der Anlage durch organisatorische Maßnahmen sichergestellt wird.

Diese Maßnahmen müssen vor Freischaltung der Debug-Dienste beginnen und solange aufrechterhalten bleiben, bis eines der folgenden Ereignisse eintritt:

- Nach einem Online-Befehl wird die Zurücksetzung der Debug-Freischaltung rückgemeldet
- Im Programmiersystem wird der sichere Status laufend dynamisiert angezeigt
- Reset der Sicherheitssteuerung
- Wenn im Netzwerk, Safety Netzwerkvariablen für die Querkommunikation verwendet werden, müssen die Maßnahmen so lange bestehen bleiben, bis auch die letzte Sicherheitssteuerung den Debugbetrieb beendet hat.

  Die Rückmeldung von einer Sicherheitssteuerung genügt nicht!

Schreiben auf mehrere SPS

⚠ GEFAHR

Bei Multi-SPS-Projekten muss der Entwickler für die Freigabe von Debug-Diensten, bzw. für die Befehlsausführung in einem Dialog den Gerätenamen bestätigen, um sicher zu stellen, dass der Befehl an die richtige Steuerung geht.

Unterbrechung der Kommunikation im Debug-Betrieb

Wird während des Debug-Betriebs die Kommunikation zur Sicherheitssteuerung unterbrochen, so wird die Sicherheitsapplikation in der Sicherheitssteuerung entladen. Dann ist keine Sicherheitsapplikation mehr aktiv. Ist ein Bootprojekt vorhanden, so wird dieses erst beim nächsten Hochlauf der Sicherheitssteuerung gestartet.

☞ Die Timeout-Zeit für die Kommunikation beträgt 200 Sekunden. Während dieser Zeit ist die Sicherheitsapplikation noch aktiv.

Der unsichere Betrieb (Debugmodus)

Der unsichere Betrieb dient zur Ausführung einer neuen, ungeprüften Applikation und zum schreibenden Eingriff in eine laufende Applikation (Debugging).

☞ Ungeprüfte Applikationen oder Eingriffe in laufende Applikationen können folgende Gefahren hervorrufen:

- Gefahr durch nicht erwarteten Anlauf von Maschinen und Maschinenteilen
- Gefahr durch nicht implementierte oder gestörte Sicherheitsfunktionen
- Gefahr durch Erprobung ungesicherter Anlagenteile
- Gefahr möglicherweise auch an verbundenen Maschinenteilen, wenn die Steuerung mit diesen kommuniziert (Querkommunikation)
- Gefahr durch Zutritt von Personen in den Gefahrenbereich

Die organisatorische Sicherheit muss gewährleistet sein!

Organisatorische Sicherheit

⚠ GEFAHR **Im unsicheren Betrieb muss die Sicherheit der Maschine durch organisatorische Maßnahmen hergestellt werden:**

- Es muss sichergestellt sein, dass sich keine Person im Gefährdungsbereich befindet
- Der gesamte Bereich um die Anlage herum sollte weiträumig gegen den Zutritt Unbefugter gesichert werden
- Der Zutritt darf nur bei Stillstand der Anlage durch den für den jeweiligen Bereich Verantwortlichen gewährt werden
- An möglichen Zutrittswegen zum Gefahrenbereich sind Warnschilder anzubringen
- Gegebenenfalls zusätzliche technische Schutzeinrichtungen zum Stillsetzen im Notfall oder Ausschalten im Notfall (bei elektrischen Gefährdungen)
- Nicht funktionsfähige Schutzeinrichtungen müssen gekennzeichnet werden, um eine Verwechslung mit wirksamen Schutzeinrichtungen auszuschließen
- Ungesicherte Anlagenteile dürfen nur erprobt werden, wenn sich alle gefährlichen Abläufe im Sichtfeld des Personals befinden
- Netzwerke müssen physikalisch abgetrennt werden, oder alle Komponenten im Netzwerk müssen ebenfalls organisatorisch abgesichert werden

☞ Netzwerke, in der Applikationen und Maschinen mit Querkommunikation entwickelt werden, müssen von Netzwerken des Betriebs physikalisch getrennt sein, um eine Beeinflussung zuverlässig ausschießen zu können.

Rückkehr in den sicheren Zustand

Die Rückkehr einer Sicherheitssteuerung vom unsicheren in den sicheren Zustand kann erfolgen durch

- Neustarten der Bootapplikation (Befehl **Neu starten** der Registerkarte **Safety Online Information** der Sicherheitssteuerung, siehe "Neustart der Bootapplikation" auf Seite 137)
- Ausloggen aus der Sicherheitssteuerung (Befehl **Ausloggen** des Menüs **Debuggen**, siehe "Ausloggen" auf Seite 132)
- Ausschalten und anschließendes Einschalten der Sicherheitssteuerung

- Urlöschen der Bootapplikation (Befehl **Urlöschen** der Registerkarte **Safety Online Information** der Sicherheitssteuerung, siehe "Urlöschen" auf Seite 227)
- Löschen der Bootapplikation (Befehl **Löschen** der Registerkarte **Safety Online Information** der Sicherheitssteuerung, siehe Kap. 13.4.2 "Bootapplikation löschen" auf Seite 210)

8.6.3 Beenden der Applikation

Die laufende Sicherheitsapplikation kann durch verschiedene Operationen im sicheren Betrieb und im Debug-Betrieb beendet werden:

- Einloggen mit Download und Wechsel in Debug-Betrieb (Einloggen auf der Steuerung und Wechsel in den Debug-Betrieb auf Seite 129)
- Neustart der Bootapplikation (Erzeugen und Neustart der Bootapplikation auf Seite 136
- Urlöschen
- Firmware Update
- Laufzeitfehler in der Applikation

In all diesen Fällen wird noch ein letztes Ausgangsabbild erzeugt, in dem alle Ausgangskanäle (digital und analog) von sicheren Feldgeräten gemäß Protokoll auf fail-safe Werte gesetzt und im Protokoll als fail-safe markiert sind. Alle Ausgangskanäle (digital und analog) von nicht-sicheren Feldgeräte sind in diesem Ausgangsabbild genullt.

8.7 Debuggen und Diagnose

8.7.1 Monitoring

Monitoring

Beim Monitoring im Online-Modus werden, solange ein Login auf die Applikation besteht, der Online-Status und die gemonitorten Werte sichtbarer Variablen zyklisch aus der Steuerung geholt und im Implementierungs- und Deklarationsteils des jeweiligen Objekts angezeigt.

Das Monitoring funktioniert sowohl im sicheren Betrieb wie im Debug-Betrieb (bei lesendem Zugriff).

HINWEIS

Der angezeigte Monitoringwert für eine Variable ist ein Wert, den diese Variable auf der verbundenen Steuerung hatte. Dieser Wert ist nicht unbedingt der aktuelle Wert, d. h. auf der Steuerung kann sich der Wert bereits wieder geändert haben.

Die Monitoringanzeige im Safety-Editor im Online-Modus ist nur zum Nachweis geeignet, dass ein bestimmter Wert bzw. Zustand einmal eingenommen wurde.

Alle in einem Safety-Editor im Online-Modus gleichzeitig angezeigten Werte haben in der angezeigten Kombination auch auf der Sicherheitssteuerung am Ende eines Applikationszyklus vorgelegen (zykluskonsistentes Monitoring). Deshalb kann die Monitoringanzeige als Hilfsfunktion eingesetzt werden, um die Zweigabdeckung von Tests anhand von Merker-Variablen zu verifizieren (siehe Nachweis der Zweigabdeckung auf Seite 180)

HINWEIS **Solange das Login besteht, dürfen die Objekte der Sicherheitsapplikation nicht geändert werden.**

Ob ein Objekt geändert ist, erkennt man bei einer gepinnten Applikation an der Anzeige "In Work" in der Online-Ansicht des Editors.

Das Monitoring der Variablen im Deklarationsfenster und im Implementierungsfenster der POE ist qualifiziert und für die Verifikation geeignet.

Überwachungsfenster

HINWEIS **Monitoring im Überwachungsfenster (Watchfenster) ist nicht qualifiziert.**

Es ist damit nicht für die Verifikation einer Sicherheitsapplikation geeignet!

Für Detailinformationen zu den Themen Überwachungsfenster wird auf die Online Hilfe von Standard-IndraWorks verwiesen.

Monitoring bei Laufzeitfehler der Applikation

Das Monitoring dient auch zur Diagnose von Laufzeitfehlern. Im Falle eines Ausführungsfehlers wird die Applikation beendet, jedoch nicht entladen. Der Entwickler kann sich auf die Applikation einloggen und die aktuellen Werte der Variablen zum Zeitpunkt des Ausführungsfehlers monitoren.

8.7.2 Ablaufkontrolle

Bei aktivierter Ablaufkontrolle werden im FUP-Implementierungsteil von POUs nicht mehr die Werte von Variablen am Ende des Zyklus dargestellt. Es werden nun die Werte von Variablen, die Ergebnisse von Aufrufen von Operatoren und die Operationen an der jeweiligen Abarbeitungsposition und zum jeweiligen Abarbeitungszeitpunkt im Implementierungsteil des Safety FUP-Editors angezeigt. In der Statuszeile wird dabei "Ablaufkontrolle aktiviert" angezeigt.

Die Ablaufkontrolle ist nicht qualifiziert und damit nicht für die Verifikation einer Sicherheitsapplikation geeignet.

Voraussetzung für die Funktionalität "Ablaufkontrolle" ist eine aktuelle Sicherheitssteuerung. Bei einer älteren Firmware-Version kann es notwendig sein, dass Sie die Firmware aktualisieren (siehe Firmware-Update auf Seite 212

8.7.3 Debug-Betrieb der Sicherheitssteuerung

Während des Debug-Betriebs (Debug-Modus) befindet sich die Sicherheitssteuerung immer im **unsicheren Betrieb**.

Debuggen kann auf der Sicherheitssteuerung auf der Download-Applikation und auf der Bootapplikation durchgeführt werden. Das Debuggen dient dem Erkennen und Finden von Fehlern der Sicherheitsapplikation.

Zum Debuggen der Sicherheitsapplikation muss der Entwickler aus dem Projekt heraus auf der Sicherheitssteuerung eingeloggt sein!

⚠ GEFAHR

Der Entwickler ist dafür verantwortlich, dass die Sicherheit der Anlage/Maschine während des ganzen Zeitraums, in dem sich die Sicherheitssteuerung im Debug-Betrieb (unsicheren Betrieb) befindet, gewährleistet ist.

⚠ GEFAHR

Wenn der Entwickler die Sicherheitsapplikation mit schreibenden Diensten debuggen möchte, während die Sicherheitssteuerung in der Maschine eingebaut ist, muss er an der Maschine/Anlage sicherstellen, dass sich keine Person im Gefährdungsbereich befindet.

Diese Maßnahmen müssen vor Freischaltung der Debug-Dienste beginnen und müssen aufrechterhalten werden.

- **bis nach einem Online-Befehl rückgemeldet wird, dass die Debugbefehl-Freischaltung zurückgesetzt wurde,**

  oder
- **bis in IndraWorks die Statusanzeige den Status "sicher" laufend dynamisiert anzeigt,**

  oder
- **bis zum Reset der Steuerung**

Wenn die Bootapplikation läuft, werden schreibende Debug-Dienste (inkl. Download einer temporären Applikation, aber ohne Stop) nur ausgeführt, wenn der Entwickler die Verwendung von Debug-Diensten (d. h. Debug-Betrieb) explizit freigeschaltet hat (siehe Kap. 8.6.2 "Debug-Modus und organisatorische Sicherheit" auf Seite 143).

Es ist dabei sichergestellt, dass eine erteilte Freigabe nur für eine Debug-Sitzung gültig ist und dass sie nach einem Logout nicht mehr gültig ist.

8.7.4 Debuggen-Befehle: Schreiben/Forcen

⚠ GEFAHR

Das Ausführen schreibender Debug-Dienste ändert das aktuelle Verhalten der gerade laufenden Applikation!

Befindet sich die Steuerung im sicheren Betrieb und die Befehle **Werte schreiben** bzw. **Werte forcen**, oder **Reset kalt** werden aktiviert, so geht die Steuerung in den Zustand **DEBUG (BA), d. h. in den Debug-Betrieb (unsicherer Betrieb)**

Schreiben und **Forcen** in SafeLogic funktionieren wie in IndraWorks-Standard. Es wird deshalb auch auf die Online-Hilfe von IndraWorks-Standard verwiesen.

☞ Während der Entwicklung der Sicherheitsapplikation können schreibende Debug-Dienste uneingeschränkt verwendet werden.

Schreibende Debug-Dienste dürfen nicht für funktionale Tests der Sicherheitsapplikation verwendet werden.

Ausnahme siehe Kap. 10.4.1 "Dynamische Verifikation und Validierung" auf Seite 177

Werte schreiben

Der Debuggen-Befehl **Werte schreiben** bewirkt, dass einmalig vor dem Applikationszyklus alle zum Schreiben vorbereiteten Werte der aktiven Sicherheitsapplikation aus allen Editoren von Variablendeklarationen und allen Überwachungsfenstern auf einmal in die Steuerung geschrieben werden.

Werte Forcen

Forcen funktioniert analog zum Schreiben, nur dass hier die zum Schreiben vorbereiteten Werte in der Steuerung hinterlegt werden.

Vor und nach jedem Zyklus werden die Werte der entsprechenden Variablen mit den Werten aus der Schreibliste überschrieben (geforct), bis das Forcen aufgehoben wird oder die Steuerung den Debug-Modus verlässt.

Angezeigt in Zustand als **Force aktiv** siehe "Zustandsanzeige der Sicherheitssteuerung im IndraWorks" auf Seite 140.

Da die Variablen nur am Anfang und Ende von jedem Zyklus auf den Forcewert gesetzt werden,

- können die Variablen während des Zyklus durch andere Werte überschrieben werden
- kann nur in die Ursprungswerte einer Berechnung eingegriffen werden und ist es nicht sinnvoll Zwischenvariablen zu forcen
- werden nur die Ausgänge der Applikation am Ende des Zyklus mit den Forcewerten überschrieben

Die Vorbereitung von Werten erfolgt wie in IndraWorks, siehe:

- Abb. 8-15 "Deklaration der POU mit Wert und Vorbereiteter Wert" auf Seite 149
- Abb. 8-16 "Wert vorbereiten" auf Seite 150
- Abb. 8-17 "Var1-Werte markiert, vorbereiteter Wert existiert" auf Seite 150

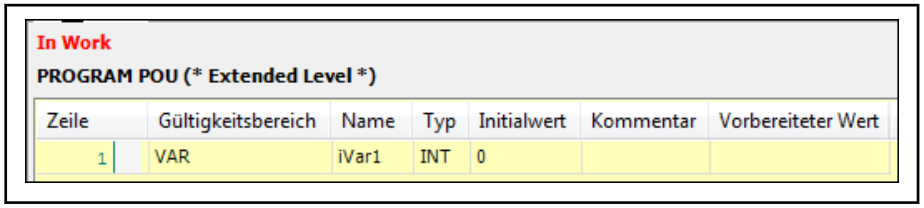


Abb. 8-15: Deklaration der POU mit Wert und Vorbereiteter Wert

Über den Dialog **Wert vorbereiten** kann bestehendes Forcen für diese Variable aufgehoben werden.

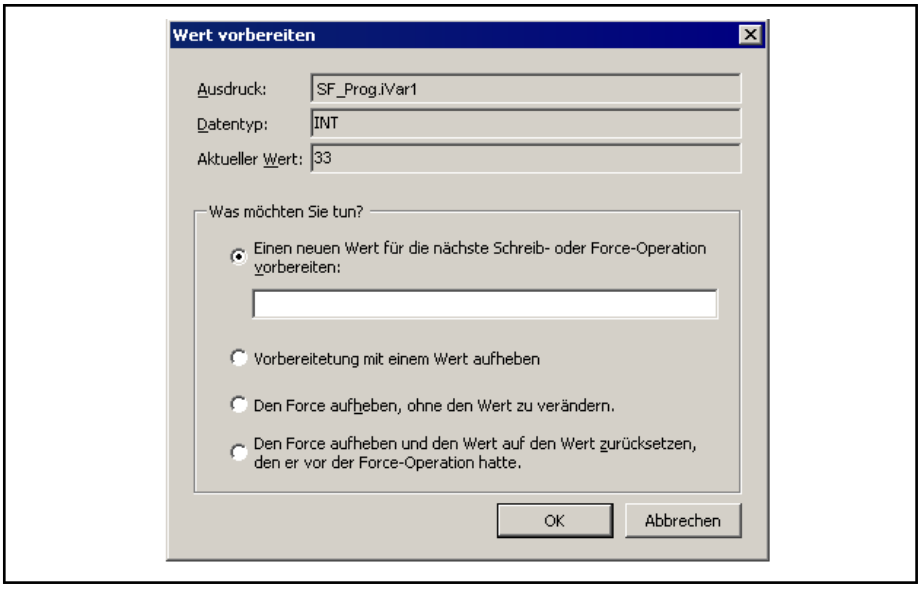


Abb. 8-16: Wert vorbereiten

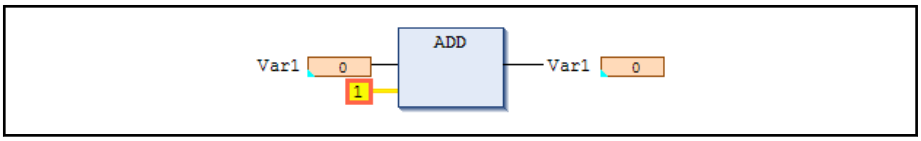


Abb. 8-17: Var1-Werte markiert, vorbereiteter Wert existiert

Forcen für alle Werte aufheben

Dieser Debuggen-Befehl beendet das Forcen in der Sicherheitssteuerung. In der Oberfläche werden die Markierungen der geforcten Variablen zurückgesetzt.

8.7.5 Debuggen-Befehle: Start, Stop, Reset kalt

Programmablaufkontrolle Start/ Stop und Applikation zurücksetzen

Die Befehle **Start** und **Stop**, Menüpunkt **Debuggen**, sind nur verfügbar, wenn der Entwickler aus dem Projekt heraus auf der Steuerung eingeloggt ist.

Der Befehl **Start** startet die aktive Applikation auf der Sicherheitssteuerung und setzt sie in den Zustand **RUN**.

Der Befehl **Stop** hält die aktive Applikation der Sicherheitssteuerung an, die Applikation wird nicht mehr durchlaufen, die Stacks der sicheren Protokolle werden weiterhin ausgeführt.

☞ Wird mit dem Befehl **Stop** die Bootapplikation, die auf der Sicherheitssteuerung im sicheren Betrieb läuft, angehalten, so geht die Sicherheitssteuerung in den Debug-Betrieb.

Im Zustand STOP werden alle Ausgangswerte an Feldgeräten (digital und analog, sicher und nicht-sicher) automatisch genullt und im Fall von sicheren Feldgeräten, sofern das Safety-Protokoll es unterstützt, werden die Failsafe-Werte aktiviert. Bei sicheren Ausgangsmodulen zwingt das die Maschine in den sicheren Zustand (das heißt in der Regel, dass sie die Maschine anhalten).

☞ Die Ausgangswerte eines gestoppten NVL-Senders bleiben auf dem letzten Wert vor dem Stopp stehen und werden weiterhin an seine NVL-Empfänger gesendet.

Und im Zustand STOP werden gültige Eingangswerte von Feldgeräten und NVL-Sendern weiterhin in die gemappten Variablen kopiert.

⚠ VORSICHT **Verhalten nach erneutem "Start"**

Mit "Start" gelangen sofort wieder die aktuellen Werte aus der Applikation an die Feldgeräte und die Maschine läuft gegenfalls weiter (je nach Zustand der Applikation). (Anders als bei einem Kommunikationsfehler, der in der Regel erst quittiert werden muss, bevor wieder Werte übertragen werden).

☞ Für die Debug-Befehle **Start** und **Stop** wird auf die Online-Hilfe von Standard-IndraWorks verwiesen.

Applikation zurücksetzen, Reset kalt

Der Befehl **Reset kalt**, Menüpunkt Debuggen, ist nur im Online-Betrieb verfügbar.

Durch Aktivierung dieses Befehls geht die Applikation in "STOP" bzw. bleibt in "STOP", wird zurückgesetzt und neu initialisiert (gleich der Initialisierung nach dem Laden der Applikation).

8.8 Online-Informationen aus der Sicherheitssteuerung

Die Sicherheitssteuerung stellt dem Anwender eine Reihe von Informationen zu Diagnose- und Debugzwecken zur Verfügung. Diese Informationen sind in einem Editor auf verschiedenen, über Tabs auswählbaren Registerkarten hinterlegt. Der Editor kann durch einen Doppelklick auf die Sicherheitssteuerung im Projektbaum oder über Selektion der Sicherheitssteuerung im Projektbaum und Aktivierung des Kontextmenübefehls **Objekt bearbeiten** geöffnet werden.

Der Editor enthält folgende Registerkarten:

- **Log**

  siehe Kap. 13.3.2 "Logbuch" auf Seite 198
- **Safety Online Information**

  siehe Kap. 13.3.3 "Informationen zu Firmware und Bootapplikation" auf Seite 200
- **Status**

  siehe Kap. 13.3.4 "Diagnose der Kommunikation" auf Seite 202
- **Information**

  siehe Kap. 13.3.5 "Informationen zur Hardware" auf Seite 204

8.9 Abstimmung mit der Standardsteuerung

☞ *Werden in der Sicherheitssteuerung E/As und Variablen der Standardsteuerung verwendet (als Logisches E/A-Objekt), so müssen sowohl*

- ein Download der Applikation auf die Standardsteuerung, als auch
- ein Download der Sicherheitsapplikation auf die Sicherheitssteuerung durchgeführt werden

Dies ist erforderlich, damit

- die physikalischen E/As und die GVLs für logischen Austausch der Standardsteuerung und
- die logischen E/As der Sicherheitssteuerung mit aktuellen Werten versorgt werden

Konfigurationsunterschiede

Sind die physikalischen Geräte und die Austauschvariablen anders konfiguriert als die entsprechenden Objekte der Sicherheitsapplikation, so wird dies mit dem Warnsymbol für Konfigurationsfehler von IndraWorks-Standard im Projektbaum bei der Sicherheitssteuerung angezeigt.

Die Konfigurationsunterschiede beziehen sich auf Anzahl, Id oder E/A-Größe der E/A-Module und Austauschvariablen für Standard- und Sicherheitssteuerung.

Für Bussysteme, bei denen die Übertragung der sicheren Konfiguration über die Standardkonfiguration erfolgt, wird die unterschiedliche Parametrierung eines E/A-Moduls als Konfigurationsunterschied angezeigt.

Ersatzwerte

Wird die Sicherheitsapplikation gestoppt, während die Standardapplikation läuft, so erhalten die physikalischen E/As keine aktuellen Werte der Sicherheitsapplikation. Ist die Standardapplikation in Stop, während die Sicherheitsapplikation läuft, so erhalten die logischen E/As keine aktuellen Werte der Standardsteuerung. In diesen Fällen werden Ersatzwerte verwendet:

Ersatzwerte für E/As

- Wenn die Sicherheitsapplikation durch den Entwickler gestoppt wurde, werden alle Ausgangskanäle (digital und analog, sicher und nicht-sicher) automatisch genullt und, im Fall von sicheren Feldgeräten, sofern das Safety-Protokoll es unterstützt, als Failsafe-Werte markiert.
- Wenn der Entwickler die Sicherheitsapplikation gestoppt hat, werden gültige Daten weiterhin von Eingangskanälen in die gemappten Variablen kopiert.
- Wenn bei laufender Applikation die E/A-Kommunikation aufgrund von Mismatch eingestellt wurde, werden die Eingangskanäle sicherer Feldgeräte gemäß Protokoll auf Failsafe-Werte gesetzt und in der Protokoll-API als Failsafe markiert.
- Wenn bei laufender Applikation die E/A-Kommunikation aufgrund von Mismatch eingestellt wurde, werden die Eingangskanäle nicht-sicherer Feldgeräte genullt.
- Wenn die Applikation beendet wird (z. B. durch Download einer neuen Applikation), wird noch ein letztes Ausgangsabbild erzeugt, in dem alle Ausgangskanäle (digital und analog) von sicheren Feldgeräten gemäß Protokoll auf Failsafe-Werte gesetzt und im Protokoll als Failsafe markiert sind.
- Wenn die Applikation beendet wird (z. B. durch Download einer neuen Applikation), wird noch ein letztes Ausgangsabbild erzeugt, in dem alle Ausgangskanäle (digital und analog) von nicht-sicheren Feldgeräte genullt sind.

Können, solange die Applikation nicht beendet ist, keine aktuellen Werte ausgetauscht werden, dann gilt für den Variablenaustausch:

Ersatzwerte für Austauschvariablen

- Hat der Entwickler die Sicherheitsapplikation gestoppt oder die Applikation der Standardsteuerung läuft nicht, werden Null-Werte auf die gemappten Lesevariablen (Austauschvariablen) der anderen Applikation geschrieben.

  Außer bei Austausch-Mismatch werden weiterhin die Werte der anderen Applikation (Variable**...Out**) in die eigenen Variablen (Variable **...In)** kopiert.
- Wenn der Variablenaustausch wegen Mismatch eingestellt wurde, werden Null-Werte auf die gemappten Lesevariablen (Logisches Austauschobjekt **... In**) der Sicherheitsapplikation geschrieben.

- Wenn die Sicherheitsapplikation beendet wird (z. B. durch Download einer neuen Applikation), werden zuvor alle Schreib-Variablen (Variable **... Out** der anderen Applikation noch einmal genullt.
- Wenn das Safety-FM eine Fehlerreaktion auslöst, werden Null-Werte auf die gemappten Lesevariablen (Logisches Austauschobjekt **... In**) der Sicherheitsapplikation geschrieben.

⚠ WARNUNG

Das Laden eines neuen SPS-Programms auf die Standardsteuerung unterbricht zeitweise die Kommunikation zwischen Sicherheitssteuerung und der sicheren Peripherie.

Es ist durch organisatorische Maßnahmen sicherzustellen, dass keine Gefährdung entsteht.

Der Betriebszustand der Standardsteuerung (START, STOP) hat keinen Einfluss auf den Datentransport von Nachrichten von und zur Sicherheitssteuerung.

8.10 Deaktivieren von sicheren Busteilnehmern

Für eine Serienmaschine sollen in einem Projekt alle Maschinenvarianten abgebildet werden. Je nach Maschinenvariante werden einzelne Maschinenmodule und somit auch E/A-Geräte deaktiviert. Dieses Vorgehen soll auch im Zusammenspiel mit einer aktiven Sicherheitssteuerung funktionieren, wobei das Safety-Programm nach der einmaligen Abnahme nicht mehr geändert werden soll.

Um dies zu erreichen, muss die E/A-Konfiguration der Maschine in die Sicherheitssteuerung übertragen und auf Gültigkeit geprüft werden. Dies muss zur Laufzeit erfolgen.

Die Sicherheit der Maschine muss trotz variabler E/A-Konfiguration weiterhin gewährleistet sein. Ein versehentliches Deaktivieren von für die Maschinensicherheit notwendigen sicheren E/As darf nicht möglich sein.

Maschinenaufbau

Die Maschinen sind modular aufgebaut. Sie werden entweder als Serienmaschinen in verschiedenen einmalig konfigurierten Ausbaustufen, oder als modulare Maschinen, die zur Laufzeit des Safety-Programms anpassbar sind, gebaut.

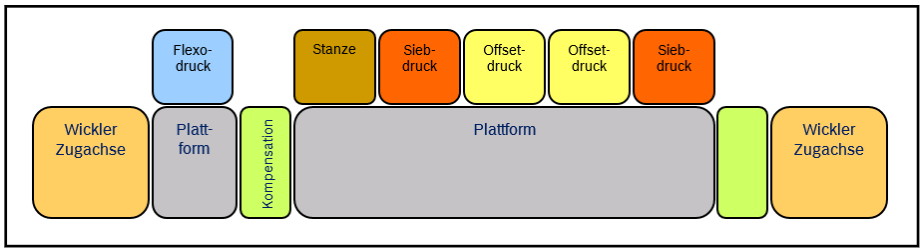


Abb. 8-18: Aufbau einer modularen Maschine

Für beide Typen gilt, dass es nur ein Safety-Programm für alle Konfigurationen geben soll. Dieses wird einmalig abgenommen und ist danach für alle Maschinenvarianten ohne Code-Änderung nutzbar.

Um dies zu ermöglichen, muss jedes Modul in einen einzeln abnehmbaren Code gekapselt werden. Dies erfolgt durch klar definierte Schnittstellen zwischen den Modulen und der Vermeidung von Zugriffen auf globale Variablen.

Hierdurch wird größtmögliche Rückwirkungsfreiheit erreicht. Gleichartige Module können hierbei gleiche Codeanteile als Instanz nutzen.

Die für die Absicherung der Maschine notwendigen sicheren E/As sind in der Regel modulweise zusammengefasst. Die sicheren E/As eines Moduls sind also nur vorhanden, wenn das Modul vorhanden ist. Das gleiche gilt für die Antriebe und die darin verbaute Sicherheitstechnik eines Maschinenmoduls. Die Zuordnung der Module ist über die eineindeutigen logischen Adressen der im Modul enthaltenen sicheren Teilnehmer möglich.

Grundprinzip

Alle sicheren Verbindungen werden überwacht. Die Sicherheitssteuerung kennt also zu jedem Zeitpunkt den Status der sicheren Verbindungen. Die Verbindungsinformation kann genutzt werden, um zu erkennen, ob ein Modul in der Maschine vorhanden ist. Weiterhin kann über die Verbindungsinformation geprüft werden, ob alle zu einem Maschinenmodul gehörenden sicheren Teilnehmer vorhanden sind und auch Verbindung zur Sicherheitssteuerung haben.

Sind Maschinenmodule nicht vorhanden, so werden sie auf der Standard-Steuerung deaktiviert. Dies kann sowohl Busteilnehmer, als auch einzelne Unterteilnehmer z.B. E/A-Scheiben an einem Buskoppler betreffen. Der Vorgang des Deaktivierens auf der Standard-Steuerung ist stark vom jeweiligen Feldbus abhängig und wird hier nicht beschrieben. Das Deaktivieren der Teilnehmer in der Standard-Steuerung wirkt sich natürlich auch auf sichere Busteilnehmer aus. Wird ein sicherer Busteilnehmer auf der Standard-Steuerung deaktiviert, so kann die Sicherheitssteuerung zu diesem Busteilnehmer keine sichere Verbindung aufbauen. In der Folge wird durch die Verbindungsüberwachung festgestellt, dass der sichere Teilnehmer fehlt und die Notaus-Kette ausgelöst. Weiterhin sind alle Eingänge des Teilnehmers im sicheren Zustand - also 0, was wieder zu Problemen in der sicheren Signalverarbeitung führen kann. Daher führt das Fehlen von sicheren Teilnehmern ohne zusätzliche Maßnahmen in der Regel zu einem Nothalt der Maschine.

Im Safety-Programm werden zunächst alle in der Maschine möglichen Module sowie die dafür notwendigen E/A-Anbindungen implementiert. Das entstandene Programm stellt somit den theoretisch maximalen Ausbau der Maschine dar. Unter Umständen kann das Programm sogar über den maximalen Ausbau der Maschine hinausgehen, da die Modulplätze der Maschine begrenzt sind, jedoch unterschiedlichste Module auf einen Modulplatz passen.

Die Module des Safety-Programms sind einzeln abzunehmen und zu testen. Daher sind hier die Modularisierung und eine klare Modulschnittstelle wichtig.

Wird die reale Maschine in Betrieb genommen, so bekommen nur die sicheren E/As der verbauten Module Verbindung. Alle anderen sicheren E/A-Verbindungen gehen in den Fehlerzustand, d.h. den sicheren Zustand. Alle Überwachungsbausteine der sicheren Verbindungen verfügen über den Ausgang "FV_Activated_S", der anzeigt, dass die Verbindung einen Fehler hat und deshalb nur sichere Daten (in der Regel 0) übertragen werden. Diese Information wird genutzt, um den Ausbauzustand der Maschine zu erkennen. Der Verbindungszustand zu den sicheren Teilnehmern kann auf Knopfdruck eingelernt werden. Er wird durch eine Prüfsumme gesichert als Sollkonfiguration auf der SD-Card der Safety-Steuerung abgelegt. Gleichzeitig wird ein Eintrag im Logbuch der Sicherheitssteuerung abgelegt. Beim nächsten Start der Maschine wird der abgespeicherte Verbindungssollzustand wieder geladen. Auch dies wird im Logbuch der Sicherheitssteuerung dokumentiert. Der Verbindungssollzustand kann dazu genutzt werden, nicht vorhandene Module zu überbrücken. So kann hiermit z.B. die Verbindungsüberwachung zu nicht vorhandenen Modulen gebrückt werden oder die Notaus-Kette an die tatsächliche Maschinenkonfiguration angepasst werden.



Die Programmierung der Überbrückung obliegt dem Maschinenbauer. Er muss die Korrektheit durch geeignete Maßnahmen prüfen.

8.10.1 Anwendung

Funktionsbausteine

Im Safety-Programm stehen zwei Funktionsbausteine zur Umsetzung der Funktionalität zur Verfügung:

- SF_IOConfigCntl auf Seite 266
- SF_IOConfig auf Seite 262

Beide Funktionsbausteine sind intern verbunden.

Der Funktionsbaustein SF_IOConfigCntl führt den Einlernvorgang durch und stellt die eingelernte Bussollkonfiguration zur Verfügung. Diese wird von den SF_IOConfig-Funktionsbausteininstanzen ausgegeben.

SF_IOConfigCntl

Der Funktionsbaustein SF_IOConfigCntl führt den Lernvorgang durch und gibt Statusinformationen zur aktiven Konfiguration aus. Er darf nur einmal im Safety-Programm instanziiert sein. An ihm werden der Schlüsselschalter zum Aktivieren des Einlernvorgangs und der Taster zur Übernahme der aktuellen Konfiguration angeschlossen.

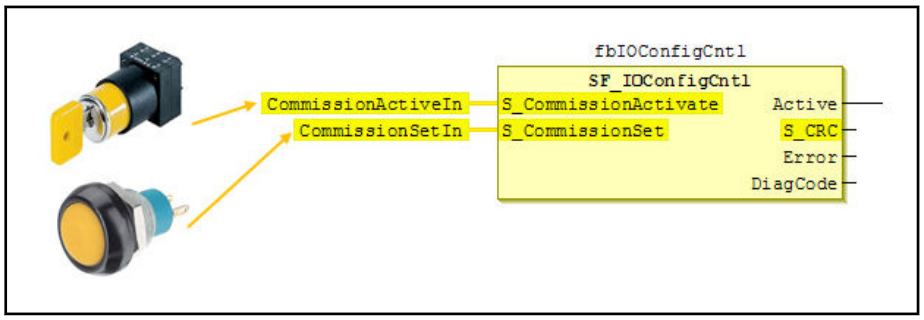


Abb. 8-19: Beschaltung des Schlüsselschalters und des Einlernknopfes

Der Ausgang "Active" wird True, wenn eine Konfiguration eingelernt wurde. In diesem Fall ist die Prüfsumme der eingelernten E/A-Konfiguration am Ausgang "S_CRC" gültig. Mit dieser Prüfsumme kann eine E/A-Konfiguration eindeutig identifiziert werden.

Der Ausgang "S_CRC" kann nur einen Wert von ungleich 0 annehmen, wenn sowohl der SF_IOConfigCntl als auch alle Instanzen des Funktionsbausteins SF_IOConfig fehlerfrei sind.

Die Ausgänge "Error" und "DiagCode" dienen der Diagnose von Fehlern während des Einlernvorgangs.

SF_IOConfig

Der Funktionsbaustein SF_IOConfig ist instanziierbar und beinhaltet je acht einlernbare Verbindungen. Er ermöglicht das Einlernen von sicheren Verbindungen auf Maschinenmodulebene. Über den Eingang "ID" wird die Kennung der Instanz eingelesen. In der Regel wird die ID mit der Kennung des Moduls gleichgesetzt. In einem Maschinenmodul kann bei Bedarf mehr als eine Instanz des Funktionsbausteins SF_IOConfig verwendet werden.

Die Eingänge "S_FV_act_x" werden direkt mit den "FV_Activated_S"-Ausgängen der Verbindungsüberwachungsbausteine (ProfisafeHost oder CSosOriginator) der am jeweiligen Maschinenmodul vorhandenen sicheren Geräte verbunden.

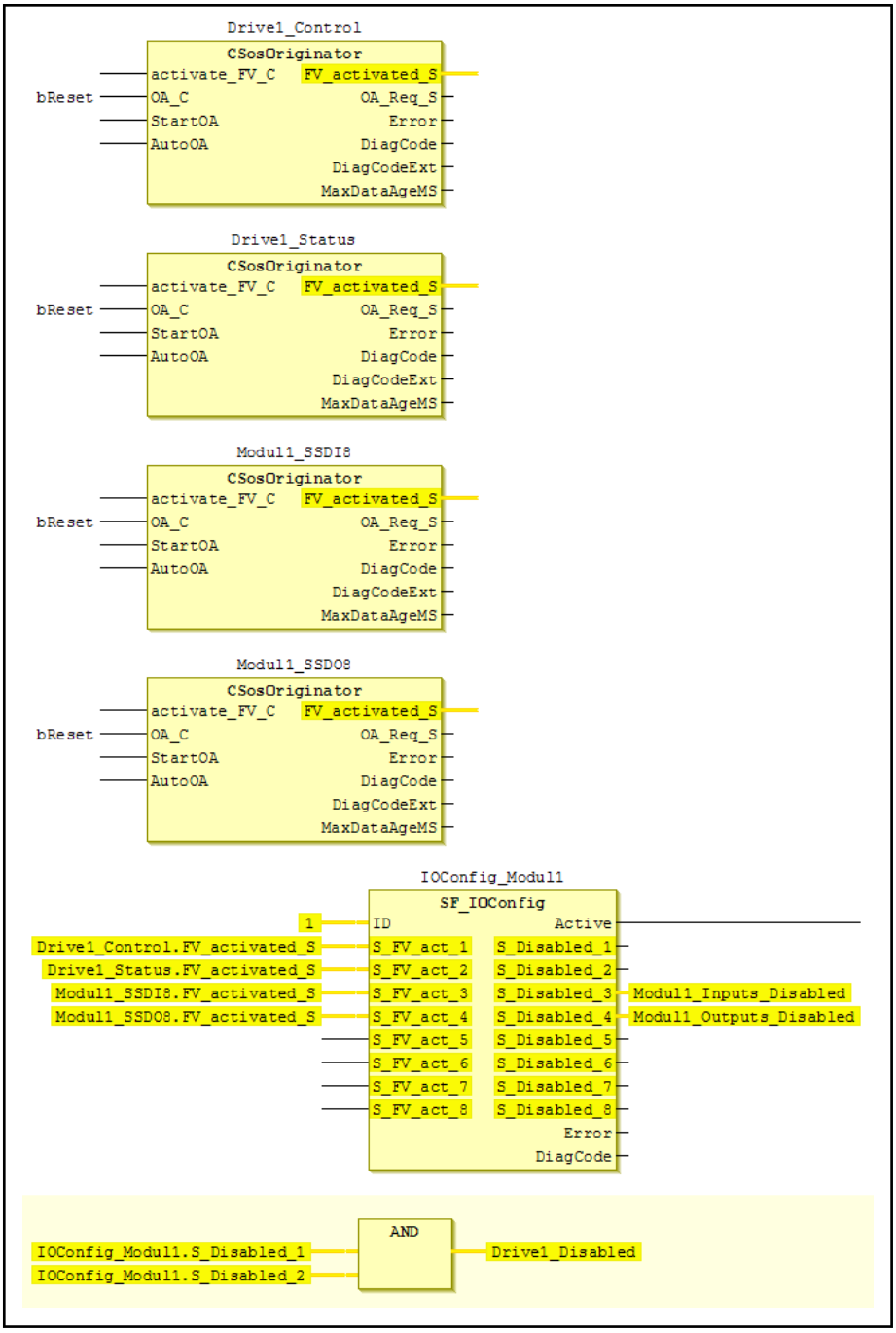


Abb. 8-20: Verschalten eines Funktionsbausteins SF_IOConfig

Der Ausgang "Active" zeigt an, dass eine eingelernte E/A-Konfiguration geladen wurde. Der eingelernte Zustand erscheint dann an den "S_Disabled_x"-Ausgängen.

Die Ausgänge "Error" und "DiagCode" dienen der Diagnose im Fehlerfall.

In Abb. 8-20 "Verschalten eines Funktionsbausteins SF_IOConfig" auf Seite 156 werden 1 CSos-Antrieb, ein SSDI8/4 (CSos-E/A mit 8 sicheren Eingängen) und ein SSDO8/3 (CSos-E/A mit 8 sicheren Ausgängen) verschaltet.

Der CSos-Antrieb hat zwei sichere Verbindungen und wird mit zwei CSosOriginator-Funktionsbausteinen (Drive1_Control und Drive1_Status) überwacht.

Die beiden CSos-E/As haben jeweils eine sichere Verbindung und deshalb jeweils einen CSosOriginator-Funktionsbaustein zur Überwachung.

Der Antrieb gilt als nicht vorhanden, wenn keine der beiden Überwachungsbausteine eine Verbindung erhält. Daher werden die beiden Signale "S_Disabled_1" und "S_Disabled_2" zu einem "Drive1_Disabled"-Signal UND-verknüpft.

Anwendung der Funktionsbausteine im Safety-Programm

Beim Deaktivieren von sicheren Busteilnehmern sind zwei Dinge zu beachten:

1. Die Verbindungsüberwachung über die ProfisafeHost- und CSosOriginator-Funktionsbausteine muss überbrückt werden.
2. Die Funktion der über den Busteilnehmer eingelesenen Signale muss gebrückt werden.

Überbrückung der Verbindungsüberwachung

Wird erkannt, dass eine sichere Verbindung unterbrochen ist, so wird in der Regel die Notaus-Kette unterbrochen. Hierfür wird das invertierte FV_activated_S-Signal der Verbindungsüberwachungsbausteine in die UND-Verknüpfung der Notaus-Kette integriert. Ist eine sichere Verbindung aufgrund eines nicht vorhandenen Maschinenmoduls deaktiviert, so muss auch die Verbindungsüberwachung überbrückt werden, da sonst die Notaus-Kette ausgelöst wird.

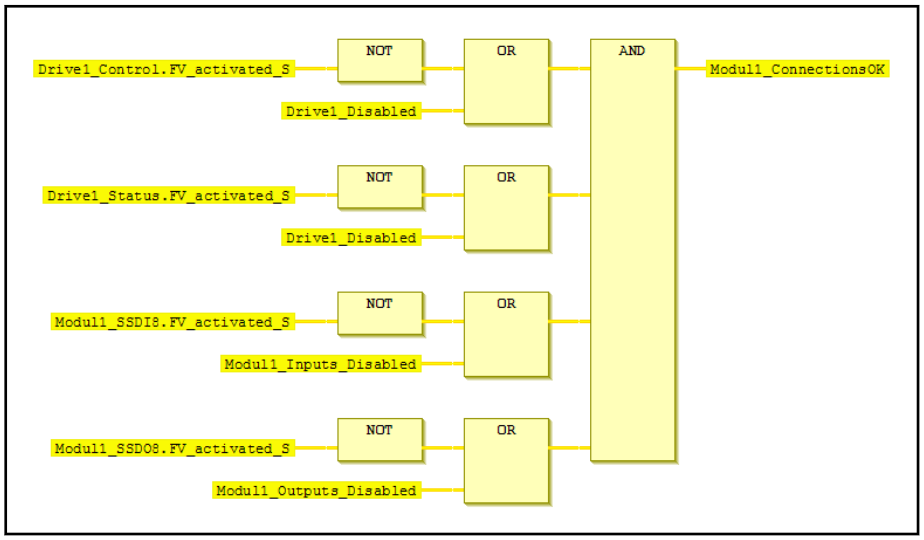


Abb. 8-21: Verbindungsüberwachung mit deaktivierten sicheren Busteilnehmern

In Abb. 8-21 "Verbindungsüberwachung mit deaktivierten sicheren Busteilnehmern" auf Seite 157 wird die Verbindung zu allen im Modul1 vorhandenen sicheren Teilnehmer überwacht. Die Information, noch Verbindung zu einem sicheren Teilnehmer besteht, wird aus den "FV_activated_S"-Ausgängen der Verbindungsüberwachungsbausteine ermittelt. Dieses Verbindungssignal wird über die "S_Disabled_x"-Ausgänge des SF_IOConfig-Funktionsbausteins überbrückt, so dass ein gewollt nicht vorhandener Teilnehmer keinen Verbindungsfehler auslöst.

Überbrückung der eingelesenen Signale

Ohne Verbindung fallen alle Eingangssignale des betroffenen sicheren Teilnehmers in den sicheren Zustand. Dieser ist als 0 bzw. False definiert.

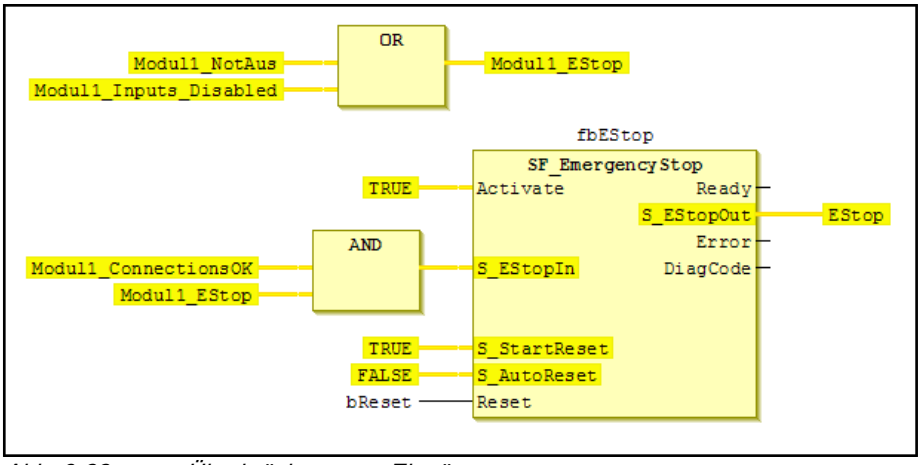


Abb. 8-22: Überbrückung von Eingängen

Abb. 8-22 "Überbrückung von Eingängen" auf Seite 158 zeigt beispielhaft die Überbrückung eines Notaus-Knopfes. Dieses Beispiel lässt sich auf andere Eingänge übertragen.

Diagnosen in der Standard-Steuerung

Auf der Standard-Steuerung sind die Prüfsumme der aktuellen Konfiguration sowie der Deaktivierungsstatus jeder Verbindung zugänglich. Die Information wird über eine Struktur abgebildet. Der Zugriff erfolgt über die globale Variable `SafetyData.IOConfig` (Diagnose auf der Standard-SPS-Steuerung auf Seite 205).

Zugriffspunkt	Beschreibung
SafetyData.IOConfig.CRC	Prüfsumme der aktuell geladenen Konfiguration. 0: keine gültige Konfiguration geladen
SafetyData.IOConfig.DisabledIOs[0..63].0..7	Array der aktuellen E/A-Konfiguration. Der Array-Index ist die ID des SF_IOConfig-Funktionsbausteins. Die einzelnen Verbindungen sind als Bits auswertbar. 0: normale Operation 1: Verbindung ist deaktiviert

Tab. 8-4: Zugriffspunkte auf die IOConfig-Daten aus der Standard-SPS

Über die Prüfsumme können bekannte Maschinenkonfigurationen verglichen werden. Der Zugriff auf die detaillierte E/A-Konfiguration dient der überprüfenden Darstellung in der HMI.

☞ SafetyData.IOConfig.DisabledIOs zeigt nicht sofort die eingelernten Daten. Es sind mehrere Aufrufe notwendig, bis alle Elemente gefüllt sind.

8.10.2 Einlernvorgang

Der Einlernvorgang ist während der Engineering- und der Inbetriebnahmephase einer Maschine identisch. Wird das Safety-Programm nicht aus der Boot-Applikation gestartet, so ist immer ein Einlernvorgang notwendig. Das Ergebnis des Einlernvorgangs wird nur auf der SD-Card der Safety-Steuerung gespeichert, wenn die Safety Boot-Applikation läuft.

1. Deaktivieren der nicht vorhandenen Teilnehmer auf der Standard-Steuerung.

2. Prüfen, dass alle an der Maschine vorhandenen sicheren Teilnehmer fehlerfrei laufen (FV_activated_S=FALSE).
3. Aktivieren des Einlernprozesses über den Schlüsselschalter
4. Bestätigen der aktuellen E/A-Konfiguration als neue Sollkonfiguration der Maschine
5. Prüfen der CRC (SafetyData.IOConfig.CRC) für eine bereits bekannte Konfiguration oder prüfen der deaktivierten sicheren Teilnehmer gegen die Sollkonfiguration der Standard-Steuerung (SafetyData.IOConfig.DisabledIOs[]).

   Bei Abweichung: Fehler korrigieren und Vorgang wiederholen.
6. Schlüsselschalter wieder in Ausgangsposition bringen

8.10.3 Anforderungen

HW-technische Maßnahmen

1. Die Maschine muss über einen sicheren Schlüsselschalter zum Einleiten des Einlernprozesses verfügen. Alternativ ist auch ein in einem durch einen Schlüssel gesicherten Schaltschrank verbauter sicherer Schalter möglich.
2. Die Maschine muss über einen sicheren Taster zum Auslösen des Einlernens verfügen.

Maßnahmen im Safety-Programm

1. Das Safety-Programm muss so modularisiert sein, dass die Maschinenmodule einzeln getestet werden können und sich nicht beeinflussen. [A]
2. Der Funktionsbaustein SF_IOConfigCntl muss genau einmal im Safety-Programm vorhanden sein. [L]
3. Der Schlüsselschalter und der Taster zum Einlernen der E/A-Konfiguration dürfen nicht SW-technisch überbrückt werden. Der Eingang "S_CommissionActivate" des Funktionsbausteins SF_IOConfigCntl muss direkt mit dem Schlüsselschalter und der Eingang "S_CommissionSet" direkt mit dem Taster zur Übernahme verbunden werden. [A]
4. Jedes deaktivierbare Maschinenmodul muss über mindestens eine Instanz des Funktionsbausteins SF_IOConfig verfügen. [A]
5. Die "FV_Activated_S"-Ausgänge der Verbindungsüberwachungsbausteine eines Maschinenmoduls müssen mit den "S_FV_act_x"-Eingängen der SF_IOConfig-Funktionsbausteine des Moduls verbunden werden. [A]
6. Jede Instanz des Funktionsbausteins SF_IOConfig muss eine eineindeutige ID haben. [L]
7. Der Eingang ID am Funktionsbaustein SF_IOConfig muss konstant sein. [L]
8. Es muss sichergestellt werden, dass im nicht eingelerntem Zustand von der Maschine keine Gefahr ausgeht (z. B. EStop aktiv, organisatorische Maßnahmen) [A]

[A] Durch Maschineninbetriebnehmer zu prüfen

[L] Wird durch Laufzeitsystem geprüft

Maßnahmen in der Standard-SPS

1. Die Prüfsumme der E/A-Konfiguration muss ohne Engineering-Tool ausgelesen werden können.

2. Die eingelernte E/A-Konfiguration des Safety-Programms muss gegen die E/A-Konfiguration des Standard-SPS Programms geprüft werden können.

8.10.4 Abnahmeprotokolle

Abnahme des Safety-Programms

Dieses Abnahmeprotokoll dient nur der Prüfung der Funktionalität zum Deaktivieren von sicheren Teilnehmern. Es ersetzt nicht die normale Abnahme des Safety-Programms, sondern erweitert diese um die im unten stehenden Abnahmeprotokoll durchzuführenden Tests.

Die Abnahme des Safety-Programms muss nach jeder Änderung erfolgen, wenn eine neue Pin vergeben wurde. Es dürfen nur gepinnte Applikationen an einer Kundenmaschine betrieben werden.

Bei Änderung am Code für ein Maschinenmodul genügt es, diesen Code abzunehmen. Die Überprüfung, welcher Code geändert wurde, kann über die Prüfsummen der einzelnen Programmteile erfolgen.

Bei Änderung am Umfang der E/A-Konfiguration (Hinzufügen oder Entfernen von sicheren Busteilnehmern im Maximalprojekt) muss auch das Safety-Programm neu übersetzt und abgenommen werden.

Das Abnahmeprotokoll befindet sich in Anhang A (Deaktivieren sicherer Teilnehmer im SafeLogic-Programm auf Seite 707).

Maschinenabnahme

Die Maschinenabnahme muss vor der Übergabe der Maschine an den Endkunden geschehen. Sie muss wiederholt werden, wenn Änderungen am Safety-Programm erfolgen.

Dieses Protokoll dient der Ermittlung von gültigen IOConfig-CRCs. Es ist als zusätzliche Kontrolle einer Maschinenkonfiguration zu sehen und ersetzt nicht die normale Maschinenabnahme.

Die Maschinenabnahme muss einmalig für jede neue Maschinenkonfiguration erfolgen, bei der die IOConfig-CRC noch nicht bekannt ist.

Ist eine Maschinenkonfiguration einmal als gültig abgenommen und die dazugehörige IOConfig-CRC ermittelt, so genügt die Prüfung auf die bekannte IOConfig-CRC im SPS-Programm der Standard-Steuerung.

Das Abnahmeprotokoll befindet sich in Anhang B (Maschinenabnahme Deaktivieren sicherer Teilnehmer auf Seite 708).

9 Pinnen

Vorbereitende Maßnahme für die Verifikation

Für die Verifikation der Sicherheitsapplikation muss der Entwickler vorbereitende Maßnahmen treffen. Ein wesentlicher Aspekt hierbei ist es, den zur Verifikation vorgesehenen Stand der Sicherheitsapplikation festzusetzen und somit sicherzustellen, dass immer nur genau dieser Stand der Sicherheitsapplikation für die Verifikation, Validierung und anschließende Abnahme verwendet wird.

Speziell hierfür bietet der IndraWorks Safety Manager die Funktion **Pinnen**.



Bevor die programmierte oder geänderte Sicherheitsapplikation verifiziert wird, ist sie neu zu pinnen.

Verifikation und Abnahme darf nur bei gepinnten Applikationen durchgeführt werden. Kein Objekt der Applikation darf den Status "In Work" aufweisen.

Die Prüfung auf "In Work" erfolgt bei den einzelnen Verifikationsschritten.



Eine gepinnte Applikation darf erst nach der Verifikation (siehe Software-Verifikation, page 167) als überprüft angesehen werden.

Was ist Pinnen

Pinnen bedeutet, dass ein Referenzpunkt auf den aktuellen Stand einer Sicherheitsapplikation gesetzt wird, der den konkreten Stand der Sicherheitsapplikation und der dazugehörenden Objekte identifiziert.

Mittels Pin ist es möglich, einen bestimmten Stand der Applikation im Projekt, eines Objekts im Editor und einer Bootapplikation auf der Sicherheitssteuerung zu identifizieren. Außerdem kann der Verifizierende - basierend auf dem Pin - jederzeit Änderungen in der Applikationsstruktur, im Inhalt ihrer Objekte und in den angezogenen Bibliotheksbausteinen erkennen.



Durch das Setzen eines Pins wird ein konkreter Stand identifizierbar gemacht, es wird dabei jedoch keine Kopie des konkreten Standes erzeugt!



Ein gepinntes Objekt kann durch Export/Import auf eine andere Applikation übertragen werden, ohne es erneut verifizieren zu müssen. Es muss lediglich die Objekt-Pin geprüft werden.

Vergleichsansicht

Die Pin-Funktionen sind im Editor des Applikationsobjekts zu finden. Dazu wird im Projektbaum das Safety-Applikationsobjekt selektiert und über den Kontextmenü-Befehl **Öffnen** geöffnet. Die Registerkarte **Objekte** zeigt die Vergleichsansicht, welche die Version und die Prüfsumme der Objekte des aktuellen Projekts und des gepinnten Projekts darstellt.

Eine genaue Beschreibung zu den Informationen und zur Objektliste der Vergleichsansicht befindet sich im Editor des Safety-Applikationsobjekts mit Vergleichsansicht, page 45).

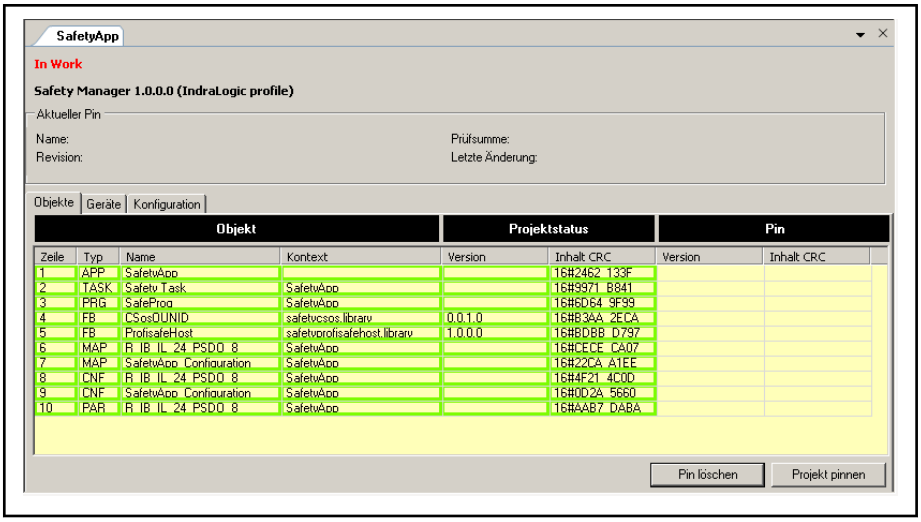


Abb. 9-1: Vergleichsansicht einer noch nicht gepinnten Sicherheitsapplikation

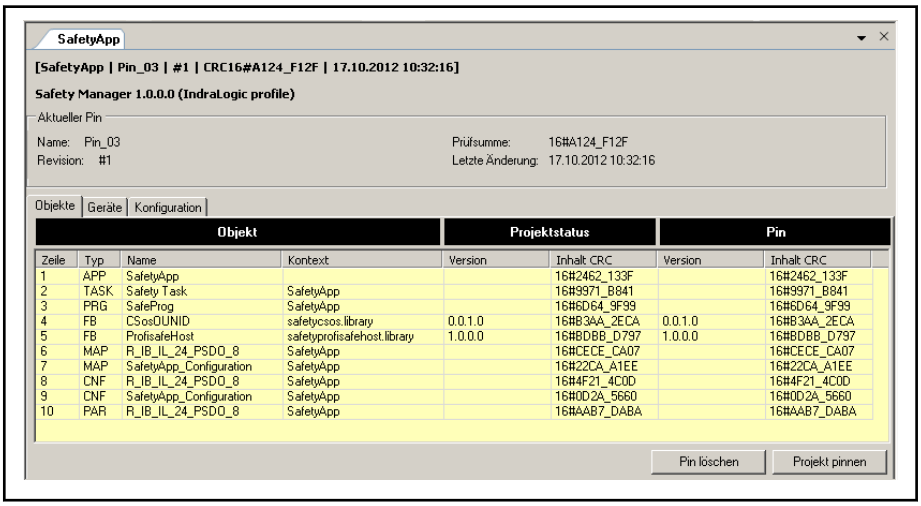


Abb. 9-2: Vergleichsansicht einer gepinnten Sicherheitsapplikation

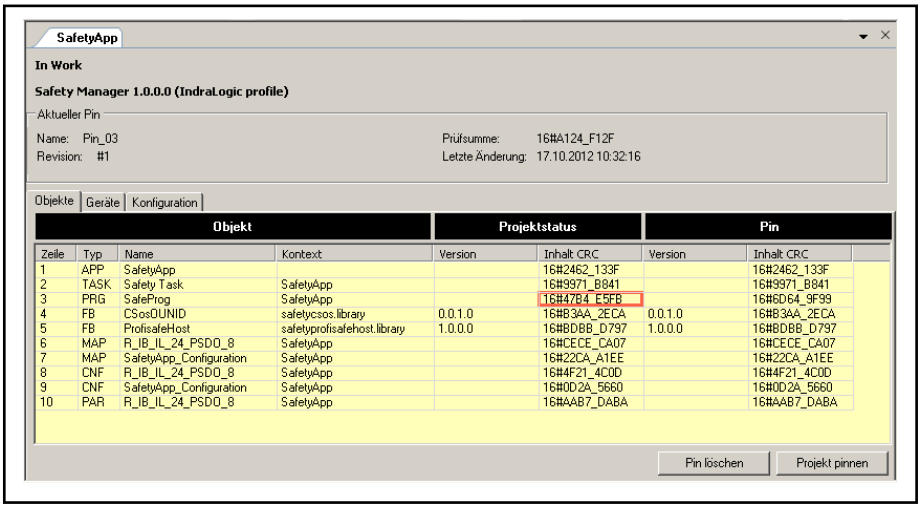


Abb. 9-3: Vergleichsansicht einer gepinnten Sicherheitsapplikation mit geänderter POU der aktuellen Applikation

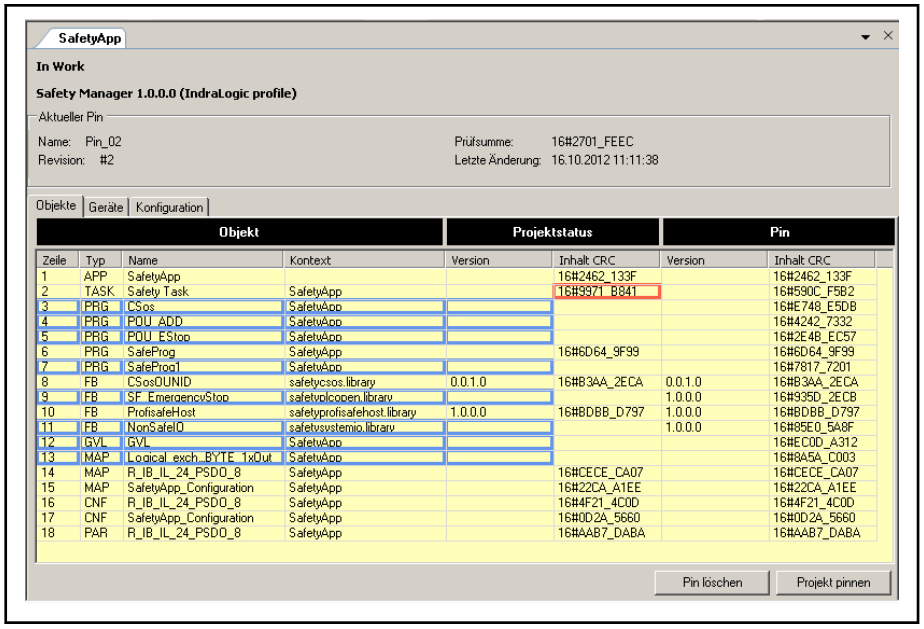


Abb. 9-4: Vergleichsansicht einer gepinnten Sicherheitsapplikation mit gelöschten POUs der aktuellen Applikation

Vergleichsansicht einer gepinnten Sicherheitsapplikation mit gelöschten POUs (blau). Durch das Löschen wurde die Safety Task geändert (rot).

Sicherheitsapplikation pinnen

Mit dem Befehl **Projekt pinnen** wird auf den aktuellen Ausführungsstand der aufgelisteten Objekte ein Pin gesetzt, wodurch nicht der Inhalt, sondern die Prüfsumme und die Version festgehalten (gemerkt) werden. Es kann ein Pin-Name eingegeben werden. Die Revisionsnummer des Pins wird mit jedem "neuen" Pinnen automatisch um 1 erhöht.

Durch den Befehl **Pin löschen** wird der aktuelle Pin gelöscht und alle Objekte befinden sich wieder **In Work**.

Der gepinnte Stand Sicherheitsapplikation umfasst:

- Umfang der Sicherheitsapplikation:
  - welche Safety-Objekte gehören zur Applikation
  - welche Bibliotheksbausteine benötigt die Applikation
- Ausführungsrelevanter Stand der Objekte und Bibliotheksbausteine im Applikationsumfang:
  - Code jedes Objekts der Applikation
  - Konfiguration und Geräteparameter jedes logischen E/A-Objekts der Applikation
  - Schnittstelle der externen Implementierung jedes verwendeten Bibliotheksbausteins
  - Versionsbezeichnungen der Objekte

Nicht zum ausführungsrelevanten Stand gehören die Kommentare der Objekte. Diese werden nicht mitgepinnt und können somit am Ende und während der Verifikation aktualisiert werden!

Der Verifizierende identifiziert einen gepinnten Stand durch eine Pin-Kennung, die an verschiedenen Stellen im Programmiersystem angezeigt wird.

Die Pin-Kennung enthält:

- Name
- Revisionszähler, der beim Pinnen automatisch um 1 erhöht wird

- Prüfsumme: ein CRC32 über den gepinnten Ausführungsstand

Zusätzlich wird die Zeit des Pinnens festgehalten. Sie ist jedoch nicht Bestandteil der Pin-Kennung.

Anzeige der Pin-Informationen und deren Abweichungen

Die Pin-Informationen zur Applikation zu einer Sicherheitsapplikation werden im Editor des Safety-Applikationsobjekts angezeigt.

Die Informationen zum Pin der Sicherheitsapplikation sind:

- **Name:** Name des Pins
- **Revision**
- **Prüfsumme:** Die Prüfsumme wird über die gesamte gepinnte Applikation erstellt
- **Letzte Änderung:** Zeitpunkt der Pin-Erzeugung

Zusätzlich wird in der Vergleichsansicht des Safety-Applikationsobjekts dargestellt, wie der aktuelle Projektstand vom aktuell gepinnten Stand der Applikation abweicht.

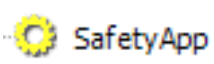
Folgende Abweichungen werden angezeigt:

- neue Objekte
- gelöschte Objekte
- bezüglich Code, Konfiguration oder Parameter modifizierte Objekte
- neu aus Bibliotheken angezogene Bausteine
- nicht mehr angezogene Bibliotheksbausteine
- bezüglich Schnittstelle oder Implementierungsversion abweichende Bibliotheksbausteine

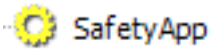
Abweichungen sind deutlich farblich gekennzeichnet, so dass sie der Verifizierende leicht erkennen kann:

- **grün:** Im Projekt neue Objekte oder Bibliotheksbausteine
- **rot:** Inhaltliche Änderung/Abweichung im Objekt bzw. Geräteparametersatz bzw. Bibliotheksbaustein
- **blau:** Im Projekt gelöschte oder nicht mehr verwendetete Objekte bzw. Bibliotheksbausteine

Pinnen in der Projekt- und in der Objektansicht

Ist die Sicherheitsapplikation gepinnt, so enthält die Vergleichsansicht die Pin-Informationen und im Projektbaum sind der Knotenpunkt 

und seine Kindobjekte mit dem Symbol  gekennzeichnet. Der Knotenpunkt **SafetyApp** gilt als gepinnt,  , wenn das Objekt selbst und alle seine Kindobjekte dem im Pin gemerkten Objektstand entsprechen.

Ist die Applikation noch nicht gepinnt oder der Pin wurde gelöscht, so erscheint in der obersten Zeile nur der Status **In Work** und im Projektbaum werden der Knotenpunkt 

und seine Kindobjekte nicht markiert. Befindet sich ein Kindobjekt der Sicherheitsapplikation **In Work**, so ist auch die Sicherheitsapplikation **In Work**.

Die Informationen zum Pin bzw. **In Work** werden in der Objektansicht und im Ausdruck des Projekts angezeigt.

Vorgehen bei der Verifikation



Das Setzen eines Pins erlaubt, ausführungsrelevante Änderungen während und nach der Verifikation zu erkennen. Mit dem Setzen eines Pins wird der Stand bzgl. der ausführungsrelevanten Teile für Verifikationstätigkeiten und Abnahme identifizierbar gemacht.



- Vor Verifikationstätigkeiten und Abnahme ist der Stand per Pin identifizierbar zu machen.
- Der Entwickler muss bei Reviews, Whitebox Testentwicklung und Testdurchführung prüfen, dass die vorliegende Applikation gepinnt wurde, dass die angezeigte Pin-Kennung den aktuell zu verifizierenden Applikationstand darstellt, und dass keine Abweichung des Projektstandes vom gepinnten Stand gemeldet wird.
- Der Entwickler muss vor Änderungen an einer abgenommenen oder in Verifikation befindlichen Applikation prüfen, dass im Ausgangsstand seiner Änderung die vorliegende Applikation gepinnt ist, die richtige Pin-Kennung angezeigt wird, und dass keine Abweichung des Projektstandes vom gepinnten Stand gemeldet wird.

10 Software-Verifikation

10.1 Software-Verifikation, Allgemeines

Als Voraussetzung muss der Anwender in früheren Entwicklungsphasen spezifiziert haben, was sein gewünschtes Verhalten der Software-Applikation ist (Software-Spezifikation) und was seine gewünschten Regeln von Programmierung und Kommentierung (Programmierrichtlinien) sind.

☞ Verifizieren Sie, dass die in diesem Dokument beschriebene Version von IndraWorks SafetyManager verwendet wird. Dazu aktivieren Sie in SafeLogic den Befehl „Safety-Versionsinformation anzeigen...“ der Kategorie "Hilfe".

Die angezeigte Safety Version muss für die auf dem Titelblatt / dem Hilfetitel genannte Dokumentenversion des Anwenderhandbuchs gültig sein.

Die Gültigkeit kann durch einen Abgleich mit der Revision List (Appendix) zum Zertifikat für SafeLogic auf der Internetseite fs-products.tuvasi.com überprüft werden.

☞ Verifizieren Sie, dass die von Ihrer Applikation verwendeten Versionen von Bibliotheksbausteinen mit den in diesem Dokument beschriebenen Versionen übereinstimmen. Sie ermitteln die Version eines verwendeten Bibliotheksbausteins in der Registerkarte "Objekte" des Editors des Safety Applikationsobjekts (Editor des Safety-Applikationsobjekts mit Vergleichsansicht, Seite 45

Wie sie lauten muss, steht in Versionsliste der Safety-Bausteine, Seite 623.

(Im Fall von Bibliotheksbausteinen anderer Hersteller muss gegen die in der entsprechenden Begleitdokumentation beschriebenen Version verglichen werden.)

☞ Das Bibliotheks-Repository und der Bibliotheksverwalter sind für den Nachweis der in der Sicherheitsapplikation verwendeten Bibliotheksbausteine bei der Verifikation und der Abnahme nicht geeignet. Der Nachweis der ausführungsrelevanten Stände muss über die Objektliste erfolgen, siehe Pinnen, Seite 161 (Software-Abnahme und Dokumentation, Seite 183).

Mit dem Prozess der Software-Verifikation wird nachgewiesen, dass die Spezifikation und die Programmierrichtlinien erfüllt und damit verifiziert sind. Die Software-Verifikation erfolgt durch die Kombination von statischen und dynamischen Verifikationsmethoden (toolbasierte Checks, Codereview, Tests).

Der Software-Verifikation folgt als weiterer Verifikationsschritt die Validierung der programmierten Sicherheitsfunktionen in der aufgebauten Maschine.

Beides sind Voraussetzungen für die Abnahme von Maschine und Sicherheitsapplikation.

☞ Verifikation und Abnahme darf nur auf gepinnten Sicherheitsapplikationen durchgeführt werden. Kein Objekt der Applikation darf den Status "In Work" aufweisen.

Der gelbe Datenfluss ist als Unterstützung für Verifikation, Validierung und Abnahme nicht geeignet.

Die IndraWorks-Standard-Projektansicht ist für die Verifikation und Abnahme einer Sicherheitsapplikation **nicht geeignet**. Zum Nachweis, welche Objekte zur Sicherheitsapplikation gehören muss die Vergleichsansicht verwendet werden (siehe Editor des Safety-Applikationsobjekts mit Vergleichsansicht, Seite 45).

Der IndraWorks-Standard-Projektvergleich ist für die Verifikation und Abnahme einer Sicherheitsapplikation **nicht geeignet**. Er kann lediglich als Hilfsfunktion verwendet werden, um die Vergleichseditoren geänderter Objekte zu öffnen.

Um den gesamten Prozess der Verifikation und Validierung zu vereinfachen und damit zu beschleunigen, stellt SafeLogic die bereits validierten und zertifizierten PLCopen-Bausteine und die Bausteine der SafetyStandard-Bibliothek zur Verfügung.

10.2 Anforderungen an Verifikation/Validierung

10.2.1 PL-e Sicherheitsapplikationen

Verifikation, Validierung und Abnahme für Sicherheitsapplikation nach PL-e

*Für Sicherheitsapplikationen mit **PL-e** sollen gemäß der Norm EN ISO 13849 Tätigkeiten zur Verifikation und Validierung durchgeführt werden, z. B.:*

- Verifikation durch Kontroll- und Datenflussanalyse
- Validierung der funktionalen Leistungen mit Black-Box-Tests
- Validierung des Leistungsverhaltens (z. B. zeitliches Leistungsverhaltens) mit Black-Box-Tests

Empfehlung:

- Testfallausführung auf Basis von Grenzwertanalysen
- E/A-Tests müssen die korrekte Verwendung der sicherheitsbezogenen Signale sicherstellen
- Nach Änderungen in der Sicherheitsapplikation muss mit Hilfe einer Einflussanalyse sichergestellt werden, dass die Spezifikation erfüllt ist

10.2.2 SIL3 Sicherheitsapplikationen

Verifikation, Validierung und Abnahme für Sicherheitsapplikation nach SIL3

Für Sicherheitsapplikationen nach SIL3 muss gemäß der Norm EN IEC 62061 die Software dahingegen geprüft werden, ob sie mit dem spezifizierten Entwurf, den Programmierrichtlinien und den Anforderungen der Sicherheitsplanung übereinstimmt.

EN IEC 62061 verweist dazu auf die Sicherheitsgrundnorm EN IEC 61508, wo die Zuverlässigkeitsstufen SIL1 bis SIL4 definiert sind.

Für Sicherheitsapplikationen nach SIL3 schließen Verifikation und Validierung strukturelle Tests der Anwendungssoftware als White-Box-Tests, funktionale Tests des Anwendungsprogramms als Black-Box-Tests und Schnittstellentests als Grey-Box-Tests (Wechselwirkungen mit Sicherheitssteuerung und anwendungsspezifischer Hardwarekonfiguration) ein. Es müssen gemäß der Norm Tätigkeiten für Verifikation und Validierung durchgeführt werden, wobei Testen die hauptsächlich verwendete Verifikationsmethode sein soll.

Dies beinhaltet unter anderem:

- Überprüfung der E/A-Konfiguration durch Überprüfung, Test oder Simulation.
- Geeignete funktionale Tests aller Softwaremodule durch Überprüfung, Test oder Simulation.
- *Geeignete Tests der Module mit*
  - Zweigabdeckung
  - Testen mit Grenzdaten
  - Überprüfung der Implementierung der Abläufe, einschließlich relevanter Synchronisierungsbedingungen

10.3 Statische Verifikation und Tests

10.3.1 Statische Verifikation

Die statische Verifikation umfasst die Überprüfung der Übereinstimmung mit

- den Codier- bzw. Programmierrichtlinien

  Diese Richtlinien werden zum Teil automatisch durch den Checker (siehe Automatische Prüfung der Programmierrichtlinien, Seite 169) und zum Teil durch Review überprüft (siehe Manuelle Prüfung der Programmierrichtlinien, Seite 170).
- den vorhandenen Spezifikationen

  Diese Überprüfung findet durch ein Review statt.
- der Planung des Gesamtsystems (Planung des Gesamtsystems, Seite 19)



Die gewünschten Check-Optionen können im Safety-Applikationsobjekt eingestellt werden (siehe Automatische Prüfung der Programmierrichtlinien, Seite 169).



Beim Review kann der Stand eines Objekts durch Vergleich der angezeigten Pin-Kennung gegen die geforderte Pin-Kennung verifiziert werden.



Bei Multi-SPS-Projekten muss während der ganzen statischen Verifikation durch Prüfen der Pin-Kennung in der Objektansicht jedes Objekts der Sicherheitsapplikation verifiziert werden, dass das Objekt zur richtigen Sicherheitssteuerung gehört.

10.3.2 Überprüfung der Programmierrichtlinien

Automatische Prüfung der Programmierrichtlinien

Einstellungen für die automatische Prüfung von Programmierrichtlinien



Die automatische Prüfung der Programmierrichtlinien findet explizit über den Befehl **Übersetzen** im Erstellen-Menü und automatisch bei jedem Erzeugen einer Downloadapplikation (Befehl **Einloggen**, Menüpunkt Debuggen oder Kontextmenü der Applikation) oder bei jedem Erzeugen einer Bootapplikation (Befehl **Bootapplikation erzeugen**, Menüpunkt Debuggen) statt.

Siehe auch Automatisch geprüfte Programmierrichtlinien, Seite 82.



Der Verifizierende muss prüfen, dass die in den Programmierrichtlinen gewünschten Checker-Optionen eingestellt sind, und dass darüber die Pin-Kennung des zu verifizierenden Applikationsstandes steht!

Alle in diesem Dialog ausgewählten optionalen Programmierregeln und optionalen Begrenzungen werden beim Übersetzen der Sicherheitsapplikation überprüft. Für jeden Verstoß wird eine Warnung im Meldungsfenster ausgegeben.

Es gibt zahlreiche weitere, nicht optionale Regeln, die in jedem Fall überprüft werden.

Diese sind bei den einzelnen Sprachelementen in Überblick Programmierung, Seite 81 und als Erläuterung bei den Safety-Fehlermeldungen, Seite 635 aufgeführt.

Manuelle Prüfung der Programmierrichtlinien

Es liegt in der Verantwortung und Entscheidung des Verifizierenden, manuelle Prüfung von Programmierrichtlinien vorzunehmen.

Der Checker prüft optional, ob bei der Applikation und bei den POUs Kommentare vorhanden sind. Ob diese oder die anderen Kommentare den gewählten Programmierrichtlinien genügen, muss manuell verifiziert werden.

Kommentierung

IndraWorks Safety Manager bietet mehrere Möglichkeiten, um die Sicherheitsapplikation zu kommentieren. Fehlende Objektkommentare ergeben optional eine Warnung.

Kommentierung der Safety-Objekte:

- Kommentar zu jeder POU der Sicherheitsapplikation (Registerkarte **Eigenschaften** der POU)
- Kommentar zu jeder GVL der Sicherheitsapplikation (Dialog **Eigenschaften** der GVL)
- Kommentarfeld zu jedem logischen E/A (Dialog **Eigenschaften** des logischen E/As)
- Kommentar zur Sicherheitsapplikation (Dialog **Eigenschaften** des Objekts **SafetyApp**)

  Hier kann die von der Norm geforderte Quellcode-Information eingegeben werden: Firma, Autor, Beschreibung, Eingänge und Ausgänge und Konfigurationsmanagement-Geschichte.
- Kommentar zum Task-Objekt

Zusätzlich zu den Objektkommentaren kann der Programmcode kommentiert werden. Fehlende Code-Kommentierung erzeugt keine Warnungen. Die Auswahl der optionalen Code-Kommentierung erfolgt im Dialog **Optionen** des Menüs **Extras**.

Optionale Kommentierung des Codes:

- **Netzwerktitel anzeigen**: Jedem Netzwerk des Programmcodes der POUs der Sicherheitsapplikationen eines Projekts kann ein Titel gegeben werden. Der Netzwerktitel wird in der ersten Zeile des FUP-Netzwerks editiert.
- **Netzwerkkommentar anzeigen:** Jedem Netzwerk des Programmcodes der POUs der Sicherheitsapplikationen eines Projekts kann ein Kommentar gegeben werden. Der Netzwerkkommentar wird in der zweiten Zeile des FUP-Netzwerks editiert.

(**Ausnahme:** Falls die Option **Netzwerktitel anzeigen** nicht gesetzt ist, wird der Netzwerkkommentar in der ersten Zeile editiert.)

- In POUs und GVLs: Kommentar zu jeder anwenderdefinierten Variablendeklaration (optional)
- **Symbolkommentar anzeigen** Die Variablenkommentare, die im Deklarationsfenster des FUP-Editors eingegeben wurden, werden im Implementierungsfenster oberhalb der Verwendungsstelle der Variablen angezeigt.

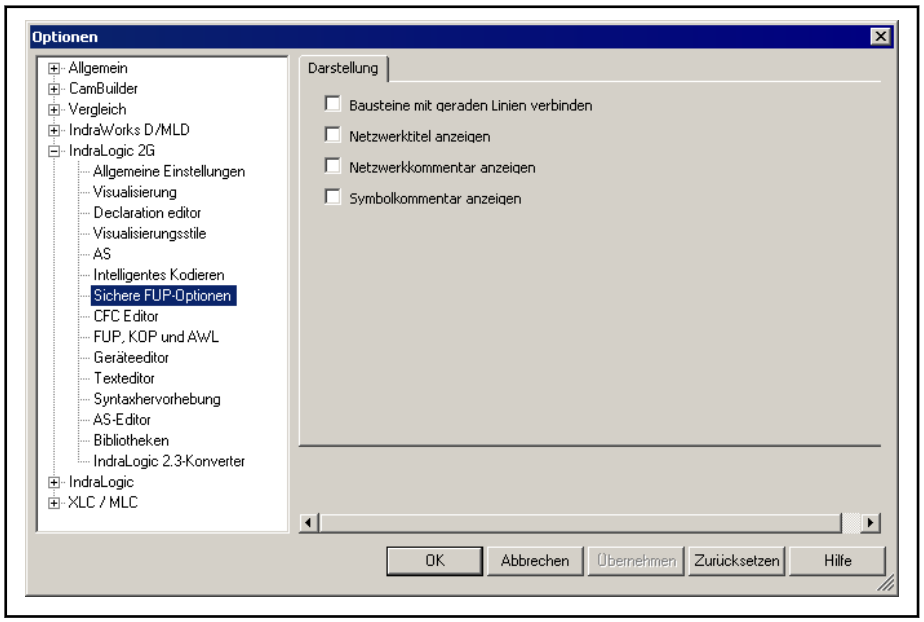


Abb. 10-1: Dialog Sichere FUP-Optionen

PLCopen-Regeln zu Sprüngen und Return

Bedingte Sprünge und Returns im Programmcode einer Sicherheitsapplikation unterliegen der manuellen Prüfung.

Bedingte Sprünge

Bedingte Sprünge sollten daraufhin geprüft werden, dass sie gemäß den PLCopen-Regeln nur für Zustandsmaschinen verwendet werden.



Es wird empfohlen, bei der Prüfung auf Zustandsmaschinen die Netzwerke, welche einen Zweig bilden, im Netzwerkkommentar zu markieren, um diese für den späteren Schritt "Zweigabdeckung" wieder zu finden.

Bedingte Returns

Bedingte Returns sollten daraufhin geprüft werden, dass sie gemäß den PLCopen-Regeln nur als Fehleraussprung verwendet werden.

Gefährliches Undersampling

Es sollte geprüft werden, dass gefährliches Undersampling mittels Techniken aus "Querkommunikation mit Netzwerkvariablen" (Querkommunikation mit Safety Netzwerkvariablen, Seite 116) oder mittels gleichwertiger Techniken vermieden wurde.

10.3.3 Verwendung von Bausteinen

Wenn die Dokumentation oder Spezifikation eines Bausteins Einschränkungen oder spezielle Anforderungen für der Verwendung bzw. Parametrierung des Bausteins enthält, so muss die Einhaltung dieser Einschränkungen bzw. Anforderungen bei der Verifikation manuell geprüft werden.

PLCopen-Bausteine bei Extended-Level-Programmierung

⚠ VORSICHT **TIME-Eingänge der PLCopen-Bausteine dürfen ihre Werte bei den Aufrufen nicht verändern.**

10.3.4 Testen gegen die Spezifikation

Prüfung gegen die Spezifikation

Die Überprüfung gegen die Spezifikation der Sicherheitsapplikation wird durch Prüfung der Schnittstellen und durch die Analyse des Kontroll- und Datenflusses durchgeführt.

Schnittstellen

Bei der statischen Verifikation gegen die Spezifikation muss geprüft werden, ob die im IndraWorks-Projekt verwendeten Gerätebeschreibungen der Feldgeräte den in der Hardware-Spezifikation der Maschine vorgesehenen Feldgeräte entsprechen.

Dabei sollten geprüft werden:

- Die richtige Beschreibung für die sicherheitsgerichtete Konfiguration
- Die richtige Beschreibung für die Geräteparametrierung

Der Verifizierende sollte dabei folgendermaßen vorgehen:

1. Jedes logische E/A, das in der gepinnten Objektliste im Applikations-Editor aufgelistet ist, öffnen.

   Es muss sichergestellt werden, dass alle Geräte geprüft werden.
2. Bei jedem logischen E/A die Pin-Kennung (**Mapping**-Seite), den Eintrag gegen die Maschine (Registerkarte **Information**) und Originator-Info (aktuelle IndraWorks Safety Manager-Version) überprüfen.
3. Bei jedem logischen E/A die Registerkarte **Sichere Konfiguration** (falls vorhanden) und Pin-Kennung, Device-Info und Originator-Info (aktuelle IndraWorks Safety Manager-Version) prüfen.

Globale Kontrollflussanalyse

Kontrollflussanalyse

Bei der Kontrollflussanalyse wird der Programmablauf, d. h. die Abarbeitungsreihenfolge eines Programms und seiner Funktionsbausteine überprüft.

☞ Die Kontrollflussanalyse wird von PLCopen zur Verifikation einer Sicherheitsapplikation gefordert.

In SafeLogic ist die Kontrollflussanalyse bereits dadurch vereinfacht, dass Sprünge nur im Extended-Level erlaubt werden und hier wiederum nur Vorwärtssprünge zulässig sind. Zusätzliche Kontrollflussanalyse für den Extended Level siehe Lokale Kontrollflussanalyse im Extended Level, Seite 174.

Für die Durchführung der Kontrollflussanalyse stehen die Funktionen **Querverweise ausgeben** und **Gehe zur Definition** zur Verfügung. Besonders geeignet ist hierzu die Funktion **Gehe zur Definition**.

☞ Für die Beschreibung der Funktion **Querverweisliste** wird auf die Online-Hilfe von IndraWorks-Standard verwiesen.

Gehe zur Definition kann auf Variablen und Bausteine ausgeführt werden. Bei deklarierten Variablen wird durch Ausführung des Befehls die entsprechende Stelle im Deklarationsteil angezeigt. Wird der Befehl auf einen Baustein ausgeführt, so wird dieser Baustein geöffnet.

☞ Die Befehle **Gehe zur Definition** und **Querverweise ausgeben** werden im Kontextmenü über das Untermenü **Symbol suchen** aktiviert.

Kontrollflussanalyse mit "Gehe zur Definition"

Der Verifizierende möchte

- von einer Task aus das aufgerufene Programm öffnen: Im Task-Editor muss die Zeile mit dem entsprechenden Programmaufruf selektiert und der Befehl **Gehe zur Definition** aktiviert werden
- von einer POU in einen aufgerufenen Funktionsbaustein wechseln: Die Funktionsbaustein-Box im Implementierungsteil der POU selektieren und den Befehl **Gehe zur Definition** aktivieren

Kontrollflussanalyse mit Querverweisliste

⚠ VORSICHT **Verwendung der Safety Querverweisliste**

Bei Verwendung der Safety Querverweisliste zur Kontrollflussanalyse oder zur Datenflussanalyse der Sicherheitsapplikation sind folgende Punkte zu beachten:

1. **Richtiges Namensformat**

   Im Feld "Name" dürfen nur "unqualifizierte" Bezeichner eingegeben werden. D.h. nach einer globalen Variable wird durch Eingabe von `<Variablenname>` gesucht, nicht durch Eingabe von `<GVL-Name>.<Variablenname>`.

   Nach FB-Eingängen und FB-Ausgängen wird durch Eingabe von `<Ein/Ausgangsname>` gesucht; die instanzbezogene Suche mittels `<FB Instanz-Name>.<Ein/Ausgangsname>` wird nicht unterstützt.

2. **Richtiger Abschluss der Eingabe / Start der Suche**

Nach Auswahl des Gültigkeitsbereichs und Eingabe des Bezeichners im Feld "Name" müssen diese Eingaben durch Betätigen der [Enter]-Taste abgeschlossen werden. Die [Enter]-Taste aktiviert die Auflistung aller Querverweise in der darunter liegenden Tabelle. Die Verwendung der Lupen-Schaltfläche sucht nur alle Verwendungsstellen im Gültigkeitsbereich "Aktive Applikation".

Der Verifizierende möchte

- zu einer POU (Programm oder Funktionsbaustein) alle Aufrufstellen finden: Querverweisliste öffnen (Menü **Ansicht**, Auswahl **Querverweisliste**) und in die Combobox **Name:** den Namen der POU eingeben und mit <Enter> abschließen.

  Es werden alle Verwendungsstellen der POU aufgelistet.

  Durch einen Doppelklick auf ein Element der Querverweisliste, kann dieses geöffnet und ggf. als Aufruf identifiziert werden.

- alle Aufrufe für eine FB-Instanz finden: Querverweisliste öffnen (Menü **Ansicht**, Auswahl **Querverweisliste**) und in die Combobox **Name:** den Namen der FB-Instanz eingeben und mit <Enter> abschließen.

  Es werden alle Verwendungsstellen der POU aufgelistet.

  Durch einen Doppelklick auf ein Element der Querverweisliste, kann dieses geöffnet und ggf. als Aufruf identifiziert werden.

☞ Die Deklaration einer POU selbst wird in der Querverweisliste nicht als Verwendungsstelle aufgeführt.

Lokale Kontrollflussanalyse im Extended Level

Bedingte Sprünge (nur Vorwärtssprünge erlaubt) und Returns stehen nur für POUs im Extended Level zur Verfügung.

Werden bedingte Sprünge und Returns verwendet, so muss für diese Extended-POUs eine zusätzliche Kontrollflussanalyse der Sprünge durchgeführt werden.

Für die Kontrollflussanalyse stehen folgende Funktionen zur Verfügung:

- Von einem bedingten Sprung zum Sprungziel wechseln: Sprung selektieren und den Befehl **Gehe zur Definition** aktivieren
- Von einem Sprungziel alle Sprünge finden, die zu diesem Sprungziel führen: Sprungziel selektieren und den Befehl **Querverweisliste** aktivieren. Durch einen Doppelklick auf einen in der Querverweisliste aufgelisteten Sprung kann an die entsprechende Stelle gewechselt werden

⚠ VORSICHT **Verwendung der Safety Querverweisliste**

Bei Verwendung der Safety Querverweisliste zur Kontrollflussanalyse oder zur Datenflussanalyse der Sicherheitsapplikation sind folgende Punkte zu beachten:

1. **Richtiges Namensformat**

   Im Feld "Name" dürfen nur "unqualifizierte" Bezeichner eingegeben werden. D.h. nach einer globalen Variable wird durch Eingabe von `<Variablenname>` gesucht, nicht durch Eingabe von `<GVL-Name>.<Variablenname>`.

   Nach FB-Eingängen und FB-Ausgängen wird durch Eingabe von `<Ein/Ausgangsname>` gesucht; die instanzbezogene Suche mittels `<FB Instanz-Name>.<Ein/Ausgangsname>` wird nicht unterstützt.

2.

Richtiger Abschluss der Eingabe / Start der Suche

Nach Auswahl des Gültigkeitsbereichs und Eingabe des Bezeichners im Feld "Name" müssen diese Eingaben durch Betätigen der [Enter]-Taste abgeschlossen werden. Die [Enter]-Taste aktiviert die Auflistung aller Querverweise in der darunter liegenden Tabelle. Die Verwendung der Lupen-Schaltfläche  sucht nur alle Verwendungsstellen im Gültigkeitsbereich "Aktive Applikation".

Datenflussanalyse

Datenflussanalyse

Bei der Datenflussanalyse wird untersucht und aufgezeigt, wo und wie Ein-/Ausgänge und Variablen in einem SafeLogic-Projekt verwendet werden.

Die von der PLCopen zur Verifikation geforderte Datenflussanalyse wird in SafeLogic erleichtert durch:

- Verwendung von FUP
- keine Sprünge und Returns im Basic Level
- Trennung von sicheren und unsicheren Signalen
- keine globalen Variablen in Funktionsbausteinen

Die Datenflussanalyse wird insbesondere mit Hilfe der Querverweisliste durchgeführt. Es werden in dieser Liste alle Positionen im Projekt oder optional in der aktuellen POU aufgeführt, an denen die Variable verwendet wird.

☞ Für die detaillierte Beschreibung der Funktion **Querverweise ausgeben** wird auf die Online-Hilfe von IndraWorks-Standard verwiesen.

☞ Für Querverweise in Sicherheitsapplikationen müssen Sie die Ansicht "Safety Querverweisliste" verwenden. (In der Ansicht "Querverweisliste" werden Sicherheitsapplikationen nicht angezeigt.)

☞ Die Befehle **Gehe zur Definition** und **Querverweise ausgeben** werden im Kontextmenü über das Untermenü **Symbol suchen** aktiviert.

⚠ VORSICHT **Verwendung der Safety Querverweisliste**

Bei Verwendung der Safety Querverweisliste zur Kontrollflussanalyse oder zur Datenflussanalyse der Sicherheitsapplikation sind folgende Punkte zu beachten:

1. **Richtiges Namensformat**

   Im Feld "Name" dürfen nur "unqualifizierte" Bezeichner eingegeben werden. D.h. nach einer globalen Variable wird durch Eingabe von `<Variablenname>` gesucht, nicht durch Eingabe von `<GVL-Name>.<Variablenname>`.

   Nach FB-Eingängen und FB-Ausgängen wird durch Eingabe von `<Ein/Ausgangsname>` gesucht; die instanzbezogene Suche mittels `<FB Instanz-Name>.<Ein/Ausgangsname>` wird nicht unterstützt.

2.

Richtiger Abschluss der Eingabe / Start der Suche

Nach Auswahl des Gültigkeitsbereichs und Eingabe des Bezeichners im Feld "Name" müssen diese Eingaben durch Betätigen der [Enter]-Taste abgeschlossen werden. Die [Enter]-Taste aktiviert die Auflistung aller Querverweise in der darunter liegenden Tabelle. Die Verwendung der Lupen-Schaltfläche  sucht nur alle Verwendungsstellen im Gültigkeitsbereich "Aktive Applikation".

Ermittlung, worauf ein Eingang wirkt

1. Die Variable ermitteln, worauf der Eingang wirkt:

   Die Variable ist in der Registerkarte **Mapping** des logischen E/As des entsprechenden physikalischen Gerätes, bzw. der GVL für logischen Austausch (Doppelklick auf logisches E/A im Projektbaum und Öffnen der Registerkarte **Mapping**) festgelegt.

2. Positionen der Variablen im Projekt ermitteln:

   Die Querverweisliste öffnen, in die Combobox **Name:** den Namen der Variablen eingeben und mit **<Enter>** abschließen. In einer Spalte der Querverweisliste wird die Zugriffsart auf die Variable aufgeführt. Durch einen Doppelklick auf eine Zeile der Querverweisliste gelangt der Verifizierende zu der entsprechenden Stelle im Implementierungsteil der POU bei Zugriffsart **Lesen** oder **Schreiben**, bzw. zu der entsprechenden Stelle im Deklarationsteil der POU bei der Zugriffsart **Deklaration**.

3. Den Datenfluss der Variablen innerhalb der POUs, in denen sie verwendet wird, ermitteln:

   Die Datenflusslinien der Variablen überprüfen

4. Ermitteln, auf welchen Ausgang die Variable zugewiesen wird:

   Mit Hilfe der Datenflusslinien und Querverweisliste

5. Ermitteln, ob die Variable anderen Variablen zugewiesen wird. In diesem Fall muss der Datenfluss auch für diese Variablen gemäß den Schritten 2 bis 5 überprüft werden.

Ermittlung, was auf einen Ausgang wirkt

1. Die Variable ermitteln, die auf den Ausgang wirkt:

   Die Variable ist in der Registerkarte **Mapping** des logischen E/As des entsprechenden physikalischen Gerätes, bzw. der GVL für logischen Austausch (Doppelklick auf logisches E/A im Projektbaum und Öffnen der Registerkarte **Mapping**) festgelegt.

2. Positionen der Variablen im Projekt ermitteln:

   Die Querverweisliste öffnen, in die Combobox **Name:** den Namen der Variablen eingeben und mit **<Enter>** abschließen.

   Es werden in der Querverweisliste alle Positionen im Projekt, oder optional in der aktuellen POU aufgeführt, an denen die Variable verwendet wird. In einer Spalte der Querverweisliste wird die Zugriffsart auf die Variable aufgeführt.

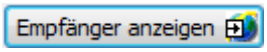
   Durch einen Doppelklick auf eine Zeile der Querverweisliste gelangt der Verifizierende zu der entsprechenden Stelle im Implementierungsteil der POU (bei Zugriffsart Lesen oder Schreiben) oder im Deklarationsteil der POU (bei Zugriffsart Deklaration).

3. Den Datenfluss der Variablen in den POUs, in denen sie verwendet wird, mit Hilfe der Datenflusslinien überprüfen.

4. Ermitteln, welche Eingänge auf die Variable wirken: Mit Hilfe der Datenflusslinien, Querverweisliste und Mapping-Registerkarte der logischen E/As

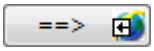
5. Ermitteln, ob der Variablen anderen Variablen zugewiesen werden. In diesem Fall muss der Datenfluss auch für diese Variablen gemäß den Schritten 2 bis 5 überprüft werden.

Ermittlung, worauf eine Sender-Variable wirkt

Im Editor der Safety Netzwerkvariablenliste (Sender) auf die Schaltfläche 

oder im Kontextmenü der Safety Netzwerkvariablenliste (Sender) im Projektexplorer den Befehl "Alle zugehörigen Empfänger anzeigen" wählen.

Ermittlung, was auf eine Empfänger-Variable wirkt

Im Editor der Safety Netzwerkvariablenliste (Empfänger) auf die Schaltfläche 

klicken oder im Kontextmenü der Safety Netzwerkvariablenliste (Empfänger) im Projektexplorer den Befehl "Gehe zu Sender" wählen.

Datenflussanalyse im Extended Level

Überprüfung der Failsafe-Eigenschaft der Logik bei Extended Level-Bausteinen

Bei **analogen Eingängen** von Extended Level-Bausteinen muss die Absicherung gegen das unbeabsichtigte Wiederanlaufen nach einem Fehlerzustand durch den Stack sichergestellt werden. Es muss geprüft werden, dass bei der Aktivierung des Failsafe-Werts 0 kein Ausgang in einen gefährlichen Zustand geht.

Z. B. führt Eingang=0 immer zu einem sicheren Ausgang oder der SInReady-Output des Eingang-Stacks fließt so in Logik ein, dass kein Ausgang in den gefährlichen Zustand geht.

Bei **digitalen Eingängen** ist die Prüfung meist einfacher, da in der Sicherheitstechnik FALSE als Failsafe-Wert global definiert ist.

Vorsicht ist bei Verwendung von NOT oder XOR geboten:

⚠ VORSICHT

Der unachtsame Gebrauch der Operatoren XOR und NOT kann zum Verlust der Failsafe-Eigenschaft von SAFExxx-Variablen führen, es wird dabei keine Checker-Warnung erzeugt.

Die Operatoren XOR und NOT können die Failsafe- Eigenschaft einer SAFExxx-Variablen negieren, sodass die SAFE-Variable ihre Failsafe Eigenschaft verliert, also "ausfallunsicher" wird. Dies kann zu unbeabsichtigtem Anlaufen der Anlage führen.

Programmierregel: Die SAFExxx-Ausgänge von NOT und XOR müssen ermittelt werden. Anschließend muss sichergestellt werden, dass diese NOT/ XOR-Ausgänge nicht auf Ausgänge (E/As) durchgeschaltet werden.

10.3.5 Verifikation gegen den Systemplan

Bei der Verifikation gegen den Systemplan prüft der Verifizierende, das die Safety-Adressen, Connection IDs und Watchdog-Zeiten konfiguriert sind wie geplant.

Der Verifizierende muss prüfen, dass die Empfänger-Netzwerkvariablenlisten mit den richtigen Sender-Netzwerkvariablenlisten verknüpft sind.

☞ Bei der Überprüfung, dass die NVL-Konfiguration die richtigen Sender- und Receiver-Netzwerkvariablenlisten miteinander verknüpft, müssen alle Safety-Adressen überprüft werden, die in der Netzwerkvariablenlisten (Empfänger) und Netzwerkvariablenlisten (Sender) angezeigt werden, bzw. für eine Steuerung oder Maschine dokumentiert sind.

10.4 Dynamische Verifikation und Tests

10.4.1 Dynamische Verifikation und Validierung

⚠ VORSICHT

Während der gesamten Verifikation liegt die Sicherheit in der Verantwortung des Verifizierenden!

Dies bedeutet, dass die Sicherheit organisatorisch auch dann hergestellt werden muss, wenn die Sicherheitssteuerung "sicherer Betrieb" meldet oder der Verifizierende nicht explizit zur Herstellung der organisatorischen Sicherheit aufgefordert wird.

Durch die Verwendung der bereits zertifizierten Bausteine der Bibliotheken PLCopen und SafetyStandard und der Programmiersprache FUP mit den Sprachteilmengen Basic und Extended wird die dynamische Verifikation wesentlich vereinfacht.

Die dynamische Verifikation ist die Überprüfung des Programms mit Hilfe von funktionalen Tests an der laufenden Applikation oder in einer speziell dafür entwickelten Testumgebung.

Dieser Teil der Verifikation ist mit White-Box- und Black-Box-Tests durchführbar.

☞ Vor der Testdurchführung muss der Verifizierende prüfen, dass die Bootapplikation auf der Sicherheitssteuerung die richtige Pin-Kennung hat. (Doppelklick auf Sicherheitssteuerung im Projektbaum und die Registerkarte **Safety Online Information** öffnen).

☞ Während des gesamten Testdesigns muss durch Prüfen der Pin-Kennung in der Objektansicht jedes Objekts der Sicherheitsapplikation verifiziert werden, dass das Objekt zur richtigen Sicherheitsapplikation gehört.

Monitoring von Variablen bei funktionalen Tests

☞ Das Monitoring der Variablen im Deklarationsfenster und Implementierungsfenster der Objekte ist qualifiziert und damit für die Verifikation einer IndraWorks Safety Manager-Sicherheitsapplikation geeignet.

⚠ VORSICHT

Wenn Ablaufkontrolle aktiviert ist, dann sind die im Implementierungsfenster angezeigten Werte nicht für die Verifikation einer Sicherheitsapplikation geeignet.

Dass es sich bei Werten im Implementierungsfenster um verifikationsgeeignete Monitoringwerte handelt, erkennen Sie an der Hintergrundfarbe: sie darf nicht grün sein!

☞ Monitoring im Überwachungsfenster ist nicht qualifiziert und damit **nicht** für Verifikation einer Sicherheitsapplikation geeignet!

⚠ VORSICHT

Der angezeigte Monitoringwert für eine Variable ist ein Wert, den diese Variable auf der verbundenen Steuerung hatte. Dieser Wert ist nicht unbedingt der aktuelle Wert, d.h. auf der Steuerung kann sich der Wert bereits wieder geändert haben. Die Monitoringanzeige im Safety-Editor im Online-Modus ist nur zum Nachweis geeignet, dass ein bestimmter Wert bzw. Zustand einmal eingenommen wurde. Alle in einem Safety-Editor im Online-Modus gleichzeitig angezeigten Werte haben in der angezeigten Kombination auch auf der Sicherheitssteuerung am Ende eines Applikationszyklus vorgelegen (zykluskonsistentes Monitoring). Deshalb kann die Monitoringanzeige als Hilfsfunktion eingesetzt werden, um die Zweigabdeckung von Tests anhand von Merker-Variablen zu verifizieren (siehe Nachweis der Zweigabdeckung, Seite 180)

⚠ VORSICHT

Verifizieren Sie, dass der angezeigte Werte für eine Variable, der für die Verifikation verwendet wird, der qualifizierte Wert ist, der am Ende eines Applikationszyklus vorgelegen hat (zykluskonsistentes Monitoring), und nicht der nicht qualifizierte Wert der Ablaufkontrolle.

Erkennen der qualifizierten Werte

- Alle angezeigten Werte im Deklarationsteil eines Editors sind Werte, die am Ende eines Applikationszyklus vorgelegen haben und sind damit qualifiziert.
- Im Implementierungsteil des FUP-Editors sind Werte, die am Ende eines Applikationszyklus vorgelegen haben, nicht grün hinterlegt. Dabei darf der Befehl **Debuggen ▸ Ablaufkontrolle** nicht aktiviert sein.

Die Funktionalität "Schreiben/Forcen" darf nur für Tests einzelner Funktionsbausteine und nur zur Erzwingung eines Zustandes eingesetzt werden, der über Black-Box nicht erreichbar ist, z. B. um den Zweig mit der Behandlung illegaler Werte zu testen.

Wenn es für die Testbeurteilung entscheidend ist, welcher Wert in die Anwendung geschrieben wurde, muss der Verifizierende über das qualifizierte Monitoring unter Berücksichtigung der Hinweise aus Monitoring, Seite 146, verifizieren, dass der richtige Wert in die Variable geschrieben wurde.

Variablen werden für das qualifizierte Monitoring bei funktionalen Tests nach folgender Vorgehensweise geschrieben oder geforct:

1. Befehl **Einloggen** der Kategorie **Debuggen**
2. Instanzauswahl durch Doppelklick auf das entsprechende Objekt Sicherheitsapplikation **SafetyApp**
3. Im Deklarationsteil der POU oder GVL durch Klick auf die Spalte **Vorbereiteter Wert** den Wert zum Schreiben vorbereiten
4. Befehl **Werte schreiben** oder **Werte forcen** der Kategorie **Debuggen** aktivieren
5. Die gemonitorten Variablenwerte im Implementierungs- oder Deklarationsfenster des entsprechenden Objekts anschauen

10.4.2 Online-Tests



Vor der Testdurchführung muss der Verifizierende prüfen, dass die Bootapplikation auf der Sicherheitssteuerung die richtige Pin-Kennung hat. (Doppelklick auf die Sicherheitssteuerung im Projektexplorer und die Registerkarte **Safety Online Information** öffnen).

Durch Aktivieren des Befehls **Neu starten** in der Registerkarte **Safety Online Information** der Sicherheitssteuerung wird die SPS in den Zustand für den nächsten Test gebracht, d. h. die Bootapplikation auf der Steuerung startet.

Grenzwertanalyse

Für Programmorganisationseinheiten (POUs) in Basic Level ist eine Überprüfung der Grenzwerte nicht notwendig, da hier nur boolesche Werte verarbeitet werden können.

Grenzwertanalyse für POUs im Extended Level siehe Online-Test im Extended Level, Seite 180.

10.4.3 Online-Test im Extended Level

Durch Aktivieren des Befehls **Neu starten** in der Registerkarte **Safety Online Information** der Sicherheitssteuerung wird die S-SPS in den Zustand für den nächsten Test gebracht, d. h. die Bootapplikation auf der Steuerung startet. Der Befehl **Neu starten** muss vor jedem Test aktiviert werden.

Nachweis der Zweigabdeckung

Bei der Programmierung mit SafeLogic können Zweige ausschließlich in Extended-POUs, durch die nur hier erlaubten Sprünge und Returns vorkommen.

Nur in POUs im Extended Level mit bedingten Sprüngen und Returns ist auch ein Nachweis der Zweigabdeckung durchzuführen.

Beispielhaftes Verfahren

1. Der zu verifizierende Programmcode steht zur Verfügung
2. Die Liste der Zweige der POU aus der Kontrollflussanalyse steht zur Verfügung
3. In der POU in jeden Zweig Z ein neues Netzwerk einfügen und darin einer (über Autodeklaration neu deklarierten) booleschen Variablen bZ (mit Initialwert FALSE) den Wert TRUE zuweisen
4. Für jeden Testfall für die POU: **Neu starten** der Bootapplikation, Testlauf, wenn Variable bZ = TRUE, den Zweig Z in der Zweigliste abhaken
5. Nach dem Durchlauf aller Tests, müssen alle Zweige in der Liste abgehakt sein

Die Merker-Variablen müssen nach dem Nachweis der Zweigabdeckung im Deklarations- und im Implementierungsfenster manuell entfernt werden.

Sie können auch auskommentiert werden, aber dann sollte die entsprechende Warnung deaktiviert werden.

☞ Funktionale Tests, die mit Instrumentierung (Merkern) durchgeführt wurden, müssen nach Entfernung der Instrumentierung oder Auskommentierung wiederholt werden!

☞ Die Anzeige von Variablen in einem Überwachungsfenster ist für die Verifikation einer Sicherheitsapplikation **nicht** geeignet!

Grenzwertanalyse

Sollen die Grenzwerte einer POU im Extended Level überprüft werden, so werden durch den bereitgestellten Sprachumfang die notwendigen Mittel zur Verfügung gestellt.

Beispielhaftes Verfahren

1. Die Grenzwerte aus der bereitstehenden Spezifikation oder dem Programmcode ermitteln
2. Test schreiben. Dabei sollte pro Grenzwert eine Instanz des Bausteins aufgerufen werden
3. Pro Testfall überprüfen, ob Ausgang des Funktionsbausteins der Erwartung entspricht
4. Pro Testfall einen **Neu starten** der Bootapplikation ausführen

10.5 Vollständiger Funktionstest der Applikation

Mithilfe von funktionalen Tests wird die Sicherheitsapplikation auf der Steuerung dahingegen getestet, ob die Ausgänge auf die Beschaltungen der Eingänge entsprechend der Spezifikation reagieren.

Es handelt sich hierbei um einen vollständigen Test des Input-Output-Verhaltens gegen die Spezifikation der Applikation.

Dabei kann die Applikation noch weitere zu testende Eingänge und Ausgänge haben.

Es muss nicht nur die Reaktion an den Ausgängen zu Feldgeräten getestet werden, sondern auch die Reaktion an versendeten Netzwerkvariablen und an Austauschvariablen zur Hauptsteuerung.

Auch für zukünftige Erweiterungen vorsorglich veröffentlichte Netzwerkvariablen ohne aktuelle Empfänger müssen getestet werden (vgl.Änderungen in Projekten mit Querkommunikation, Seite 224)!

Und die Reaktionen müssen nicht nur unter verschiedenen Beschaltungen der Eingänge von Feldgeräten getestet werden, sondern auch unter verschiedenen Beschaltungen der Netzwerkvariablen und Austauschvariablen, die die Applikation von Safety NVL-Sendern bzw. der Hauptsteuerung empfängt.

Sie können einen vollständigen Funktionstest der Applikation für jede Sicherheitsapplikation der Maschine separat durchführen.

Um die Sicherheitssteuerung in den Zustand für den nächsten Test zu bringen, muss ein Reset der Hardware durchgeführt werden.

Der Test kann entweder auf der endgültigen Sicherheitssteuerung durchgeführt werden oder auf einem baugleichen Modell wie die endgültige Sicherheitssteuerung, wenn anschließend die Bootapplikation auf die endgültige Sicherheitssteuerung transferiert werden, wie unter "Hardware-Tausch" beschrieben ist (Hardware-Tausch, Seite 213).

⚠ VORSICHT

Der Transfer darf nur so erfolgen, wie es unter "Hardeware-Tausch" beschrieben ist, nicht durch die Programmiersystemfunktion Bootapplikation erzeugen.

Wenn eine Bootapplikation mit "Bootapplikation erzeugen" erzeugt wird, dann muss dies Bootapplikation auf dieser Sicherheitssteuerung neu getestet werden.

10.6 Verifikation in der fertigen Maschine

⚠ VORSICHT

Die im Folgenden erläuterten **Prüfungen der Eindeutigkeit der Adressen** sind zwingend erforderlich, um mit SIL3-Sicherheit auszuschließen, dass durch einen Fehler auf der Kommunikationsstrecke eine Sicherheitssteuerung mit dem falschen sicheren Feldgerät verbunden werden könnte bzw. mehrere Sicherheitssteuerungen mit dem selben sicheren Feldgerät verbunden werden könnten.

In der fertigen, kompletten Maschine, in der die separat funktionsgetesteten Sicherheitsapplikationen installiert sind, ist zu verifizieren:

- Eindeutigkeit der Adressen:
  - Die an den sicheren Feldgeräten in der Maschine eingestellten Safety-Adressen sind maschinenweit eindeutig.
  - Die in allen Sicherheitsapplikationen der Maschine konfigurierten Safety-Adressen.
    - Das heißt bei PROFIsafe-Geräten sind "F_Source_Add" und "F_Dest_Add" eindeutig.
    - Bei CSos-Geräten sind die "SDID" in Verbindung mit der "SNN" maschinenweit eindeutig.
    - Die Safety-Adresse jeder Sender-Netzwerkvariablenliste und die "Connection ID" jeder Empfänger-Netzwerkvariablenliste sind maschinenweit eindeutig.
- **Richtige Gerätebeschreibungen**: Für jedes Feldgerät mit eingestellter Safety-Adresse X in der Maschine verwendet das entsprechende Geräteobjekt mit konfigurierter Safety-Adresse X im Projekt die passende Beschreibungsdatei (Registerkarte "Sichere Konfiguration"

  Diese Verifikation kann durch Testen gegen die Spezifikation, Schnittstellen ersetzt werden (Schnittstellen, Seite 172).
- **Richtige Prozesssignale**: An die Feldgeräte mit eingestellter Safety-Adresse X sind in der Maschine Sensoren und Aktoren so angeschlossen, dass sie zur Verwendung des Abbilds des Geräteobjekts mit konfigurierter Safety-Adresse X im Projekt passen.
- **Sicherheitsvalidierung**: Im Rahmen der Sicherheitsvalidierung der Maschine wird die Realisierung der spezifizierten Sicherheitsfunktionen durch die Gesamtheit der (kommunizierenden) Sicherheitssteuerungen in der Maschine verifiziert.

☞ Vor der Validierung muss sichergestellt werden, dass die Sicherheitsapplikation, die sich auf der Steuerung befindet, gepinnt ist, und dass der Pin und die Pin-Kennung mit der zu verifizierenden Sicherheitsapplikation übereinstimmt.

Bei der Validierung werden alle Sicherheitsfunktionen, die in der Sicherheitsapplikation programmiert worden sind, direkt an der Anlage getestet. Es wird getestet, ob die Sicherheitsfunktionen tatsächlich funktionieren und die Sicherheitsapplikation für die Anlage geeignet ist.

11 Software-Abnahme und Dokumentation

Nachstehend werden die Voraussetzungen, die erforderlichen Nachweise und die Funktionen zur Archivierung des Projekts und Ausdruck der Abnahmedokumentation beschrieben.

☞ Während des gesamten Prozesses von der Entwicklung bis zur Abnahme einer Sicherheitsapplikation muss von Anwenderseite dafür gesorgt werden, dass die richtigen Objektstände, die richtigen Bibliotheken und richtigen Gerätebeschreibungsdateien verwenden werden (siehe Pinnen, Seite 161).

☞ Der Anwender ist dafür verantwortlich, dass bei der Abnahme der Sicherheitsapplikation alle dafür relevanten Normen eingehalten werden.

☞ Für die Beurteilung der richtigen Verwendung der (externen) FB-Schnittstelle ist die zu dieser Steuerung gehörende Anwenderdokumentation für diesen FB in der nachgewiesenen Version heranzuziehen.

☞ Die IndraWorks-Standard-Projektansicht ist für die Verifikation und Abnahme einer Sicherheitsapplikation **nicht geeignet**. Zum Nachweis, welche Objekte zur Sicherheitsapplikation gehören muss die Vergleichsansicht verwendet werden (siehe Editor des Safety-Applikationsobjekts mit Vergleichsansicht, Seite 45).

☞ Der IndraWorks-Standard-Projektvergleich ist für die Verifikation und Abnahme einer Sicherheitsapplikation **nicht geeignet**. Er kann lediglich als Hilfsfunktion verwendet werden, um die Vergleichseditoren geänderter Objekte zu öffnen.

☞ Um den korrekten Hochlauf der Safety-Steuerung über die Bootapplikation vor der Abnahme zu überprüfen, ist die Safety-Steuerung aus- und wieder einzuschalten.

11.1 Voraussetzungen und Nachweise für die Abnahme

☞ Der Anwender muss vor der Abnahme seiner Sicherheitsapplikation verifizieren, dass die vorliegende Applikation, aus der tatsächlich die Bootapplikation erzeugt wird, gepinnt ist, die richtige Pin-Kennung angezeigt wird, und dass keine Abweichung des Projektstandes vom gepinnten Stand gemeldet wird.

☞ Für die Abnahme ist vom Anwender zu prüfen, ob im Projekt tatsächlich die zum Gerät in der Maschine passende Gerätebeschreibung zum Einsatz kommt.

☞ Die Dokumentation aller verwendeter Safety- Adressen und Connection IDs von Netzwerkvariablenlisten ist nötig, um die Sicherheitssteuerung bzw. Teilmaschine mit anderen Maschinenteilen zur einer Gesamtmaschine kombinieren zu können.

Voraussetzungen für die Abnahme einer Sicherheitsapplikation:

- Das gesamte Projekt muss auf einem Speichermedium abgelegt sein
- Das gesamte Projekt muss archiviert sein (inkl. aller Gerätebeschreibungsdateien), siehe Archivierung, Seite 185
- Das Projekt muss eine Benutzerverwaltung besitzen (eine vom Anwender eigens erstellte Benutzerverwaltung)
- Dem Anwender muss eine IndraWorks-Version zur Verfügung stehen, die das Format und die Ausführungsversion von IndraWorks unterstützt, mit der die Sicherheitsapplikation verifiziert wurde
- Alle Objekte müssen dem zur Abnahme zugrundeliegenden Referenzstand entsprechen (Nachweis der Pin-Kennung der Objekte)
- Die Sicherheitsapplikation muss mit der Sicherheitsapplikation, aus der das Bootprojekt auf der Steuerung erzeugt wurde, identisch sein. Dies ist sichergestellt, wenn die Pin-Information der Bootapplikation in der Registerkarte **Safety Online Information** der Sicherheitssteuerung (siehe Informationen zur Firmware und zur Bootapplikation, Seite 200) mit der Pin-Information der Sicherheitsapplikation (siehe Objektinformationen der Sicherheitsapplikation, (siehe Sicherheitsapplikationsobjekt SafetyApp, Seite 41) übereinstimmt.
- Die Informationen der Sicherheitsapplikation und der Geräte der Sicherheitsapplikation stehen als Ausdruck zur Verfügung

  Drucken siehe Ausdruck Abnahmedokumentation, Seite 186.

Objekte, bzw. Informationen, für die für die Abnahme einer Sicherheitsapplikation ein Nachweis erbracht werden muss:

- Verwendete Quellen (Safety-Applikationsobjekt im Projektbaum öffnen, Registerkarte **Objekte** auswählen)
- Wiederverwendete validierte Bausteine (Safety-Applikationsobjekt im Projektbaum öffnen, Registerkarte **Objekte** auswählen)
- Version der Gerätebeschreibungen (Safety-Applikationsobjekt im Projektbaum öffnen, Registerkarte **Geräte** auswählen)
- alle sicherheitsgerichteten Geräteparameter der parametrierbaren sicheren Feldgeräte (Registerkarte **Sichere Parametrierung** des logischen E/As des sicheren Feldgerätes
- Eindeutige Quelladresse jedes F-Hosts innerhalb einer Anlage/eines Netzwerks

  Siehe auch

  – PROFIsafe-spezifische Nachweise für die Abnahme, Seite 589

  – Safety NetVar-spezifische Nachweise für die Abnahme, Seite 611
- Die Quelladressen aller F-Devices eines Hosts müssen der Quelladresse ihres F-Hosts innerhalb einer Anlage/eines Netzwerks entsprechen (**F_Source_Add** in der Registerkarte **Sichere Konfiguration** des logischen E/As des sicheren Feldgeräts (F-Devices))
- Eindeutige Zieladresse aller F-Devices aller F-Hosts. (**F_Dest_Add** in der Registerkarte **Sichere Konfiguration** des logischen E/As des sicheren Feldgeräts (F-Devices))
- Die Netzwerkadresse (**SNN** in der Registerkarte **Sichere Konfiguration**) aller CSos-Targets müssen innerhalb einer Anlage/eines Netzwerks identisch sein
- Die Safety Device ID (**SDID** in der Registerkarte **Sichere Konfiguration**) jedes CSos-Targets müssen eindeutig sein

- IndraWorks-Version (Menü **Hilfe** ▸ **Safety Versionsinformation anzeigen...**), die bei Entwicklung und Testen der Sicherheitsapplikation verwendet wurden
- Version des installierten Safety Packages (Menü **Hilfe** ▸ **Safety Versionsinformation anzeigen...**)
- Ausführungsversion (in der Registerkarte **Safety Online Information** der Sicherheitssteuerung)
- Stand der Firmware, der auf der Steuerung läuft (in der Registerkarte **Safety Online Information** der Sicherheitssteuerung)
- Bei Multi-SPS Projekten: Nachweis, dass sich die Sicherheitsapplikation auf der richtigen Sicherheitssteuerung befindet (Verifizieren des Targetnamens in der Registerkarte **Safety Online Information** der Sicherheitssteuerung)

☞ Für den Nachweis, welche Bibliotheken oder Bibliotheksbausteine in welchen Versionen verwendet werden ist der Bibliotheksmanager nicht geeignet. Für diesen Nachweis muss die Vergleichsansicht des Safety-Applikationsobjekts verwendet werden (siehe Editor des Safety-Applikationsobjekts mit Vergleichsansicht, Seite 45).

11.2 Funktionen für die Abnahme

11.2.1 Archivierung

Mit Hilfe der Archivierung werden sämtliche zu einer Sicherheitsapplikation gehörenden Objekte in einer einzigen Datei zusammengefasst.

Diese Datei kann als Datensicherung abgelegt und im Bedarfsfall wieder entpackt werden.

Das Projekt wird während der Archivierung automatisch gespeichert.

☞ Das Archivieren des Projekts für die Abnahme der Sicherheitsapplikation kann über die Funktionalität der Projektarchivierung erfolgen. Dies erfolgt über das Menü mit dem Befehl **Projekt** ▸ **Archivieren...**

*Folgende Informationen müssen im Dialog **Projekt archivieren** zur Archivierung der Sicherheitsapplikation ausgewählt werden:*

- Standardbibliotheken und Programmkompilat ins Archiv aufnehmen

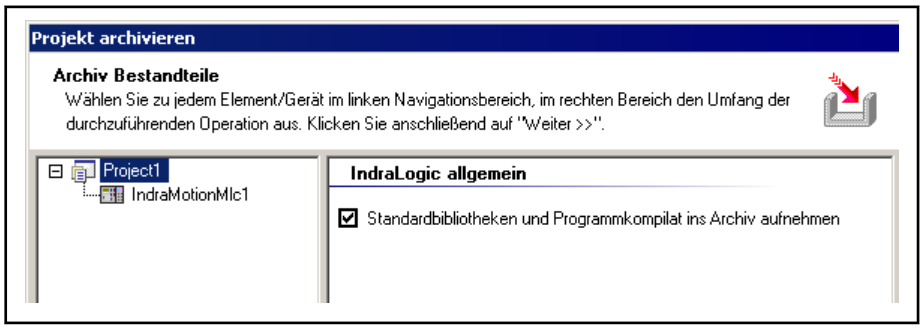


Abb. 11-1: Dialog 'Projekt archivieren'

Ins Projektarchiv mit aufgenommen werden durch korrekt erfolgte Selektion im Auswahldialog:

- Alle Sicherheitsapplikationen mit Kanalmappings und Austauschlisten

- Alle Feldbuskonfigurationen und Geräteparametrierungen
- Alle verwendeten Bibliotheken

11.2.2 Ausdruck Abnahmedokumentation

Zum Ausdruck der Abnahmedokumentation wird in SafeLogic der Menübefehl **Datei ▸ Drucken...** verwendet.

Es wird dabei ein Ausdruck aller markierten Objekte, inklusiv der Objektinformationen, Prüfsumme und Pin-Information (Pin-Kennung) erstellt.

Im Ausdruck werden alle markierten Objekte, inklusive Objektinformationen, Prüfsumme und Pin-Information (AppPin-Kennung) dargestellt. Mit den Objekte-Ansichten (Objektinhalt) wird auch zu jedem Objekt die Pin-Kennung und falls das Objekt nicht gepinnt ist die Information **In Work** mit ausgedruckt.

Der Ausdruck enthält optional zusätzlich ein Deckblatt mit den Informationen: Datei, Datum und Profil sowie ein Inhaltsverzeichnis.

1. Für den Ausdruck der Abnahmedokumentation sind im Dialog "Projekt- und Gerätedaten drucken" die Sicherheitssteuerung und alle ihre Unterknotenpunkte und Objekte markiert.
2. Vor Aktivierung des Befehls zum Ausdruck müssen alle Fenster der für den Ausdruck ausgewählten Objekte geschlossen werden.

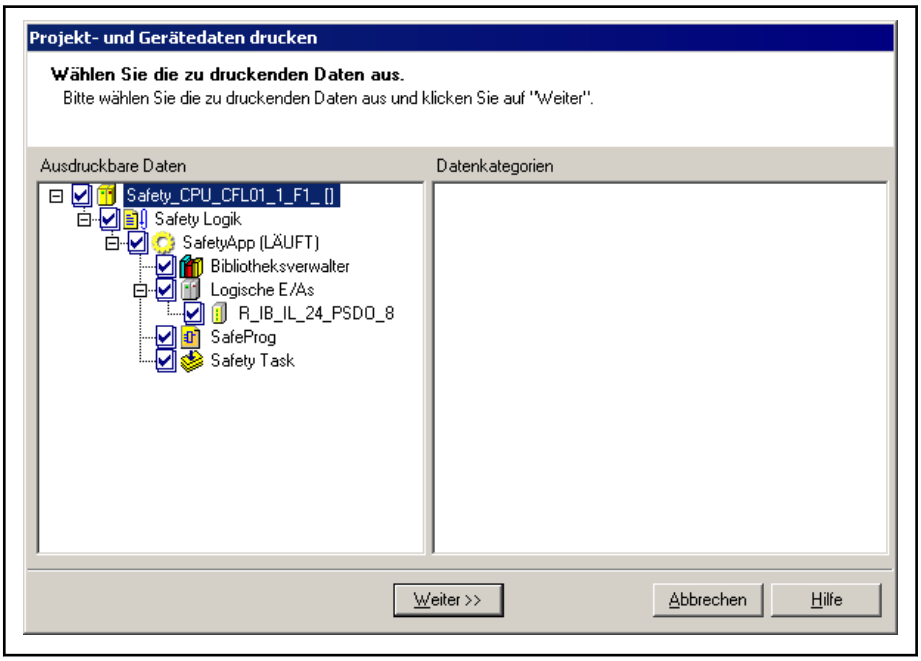


Abb. 11-2: Dialog "Projekt- und Gerätedaten drucken"

Folgende Nachweise sind nicht in dem durch den Befehl **Projekt ▸ Drucken...** erstellten Ausdruck enthalten und müssen deshalb extra erzeugt werden:

- Stand der Firmware, die auf der Steuerung läuft (Registerkarte **Safety Online Information** der Sicherheitssteuerung)

Detailinformationen zum Drucken eines Projekts siehe Online-Hilfe von IndraWorks-Standard.

12 Software-Aktualisierung

12.1 Überblick Versionierung

In diesem Kapitel werden die mögliche Auswirkungen der Installation einer neuen Geräte-, Firmware- oder IndraWorks-Version auf ein bereits verifiziertes bzw. abgenommenes Projekt und Folgen der Installation eines nicht für IndraWorks Safety Manager freigegebenen Installationspaketes beschrieben.

12.2 Aktualisieren der Geräteversion

⚠ VORSICHT

Ein geänderter Inhalt einer Gerätebeschreibung kann dazu führen, dass die Sicherheitsapplikation nicht mehr gepinnt, sondern wieder In Work ist.

Die in der Vergleichsansicht des Safety-Applikationsobjekts aufgelisteten, von der geänderten Gerätebeschreibung betroffenen Safety-Objekte mit geänderter Prüfsumme müssen einer Einflussanalyse unterzogen und anschließend neu gepinnt und verifiziert, und gegebenenfalls neu abgenommen werden.

Steht eine neue Version einer Gerätebeschreibung zur Verfügung, so sollte sich der Service-Mitarbeiter vor dem Installieren der neuen Gerätebeschreibungsversion genau darüber informieren, um welche der möglichen Varianten einer neuen Version es sich handelt und welche Auswirkungen dies auf das Projekt und damit auch auf die Verifikation und die Abnahme haben wird:

- Das neue Gerät ist kompatibel zum bisherigen Gerät:

  Nach Installieren der neuen Gerätebeschreibung im Geräte-Repository und Aktualisieren (über Kontextmenu im Projektexplorer) des Gerätes, treten beim Übersetzen der Sicherheitsapplikation keine Fehler auf. Es müssen keine weiteren Tätigkeiten ausgeführt werden.

- Es handelt sich um ein Update eines Safety-Gerätes:

  Das Update kann geänderte Geräte-Parameter oder eine geänderte E/A-Struktur enthalten, die Änderungen an Safety-Objekten der Sicherheitsapplikation nach sich ziehen.

  Die in der Vergleichsansicht des Safety-Applikationsobjekts aufgelisteten, von der geänderten Gerätebeschreibung betroffenen Safety-Objekte mit geänderter Prüfsumme müssen einer Einflussanalyse unterzogen und anschließend neu gepinnt und verifiziert werden.

- Es handelt sich um ein Update eines Standard-Feldgeräts:
  - Werden keine E/A-Daten des Standard-Feldgerätes über die logischen E/As in die Sicherheitsapplikation gemappt, ist in der Sicherheitsapplikation nichts zu tun. Die Sicherheitsapplikation ist nicht betroffen.

    Die Vergleichsansicht der Sicherheitsapplikation muss geprüft werden: Sie bleibt unverändert.
  - Werden E/A-Daten des Standard-Feldgeräts über logischen E/As in die Sicherheitsapplikation gemappt, so kann das Update Änderungen an Safety-Objekten der Sicherheitsapplikation nach sich ziehen.

Die in der Vergleichsansicht des Safety-Applikationsobjekts aufgelisteten, von der geänderten Gerätebeschreibung betroffenen Safety-Objekte mit geänderter Prüfsumme müssen einer Einflussanalyse unterzogen und anschließend neu gepinnt und verifiziert werden.

☞ Die Detailinformationen zu den installierten Gerätebeschreibungen sind auf der Karteikarte **Informationen** des Dialogs des jeweiligen Gerätes. Hierzu muss das entsprechende Gerät selektiert und im Kontextmenü der Menüpunkt **Öffnen...** aktiviert werden.

12.3 Aktualisieren der Firmware, die Ausführungsversion

Ein Update der Firmware führt in der Regel nicht zum Verlust der Abnahme Ihrer Sicherheitsapplikation.

Nachstehend wird beschrieben, unter welchen Umständen Sie Ihre Abnahme verlieren und wann eine bereits abgenommene Sicherheitsapplikation Ihre Abnahme behält.

Wie geschieht ein Firmware-Update

Expliziter Firmaware-Update: Aktivieren der Schaltfläche **Firmware-Update** in der Registerkarte **Safety Online Information** der Sicherheitssteuerung.

Impliziter Firmware-Update: Verwendung einer neueren Sicherheitssteuerung mit einem anderen Firmware-Stand als dem, mit dem die Sicherheitsapplikation abgenommen wurde.

Firmwareversion und Ausführungsversion

Welcher Firmwarestand sich auf Ihrer Hardware befindet können sie mit Hilfe der Registerkarte **Safety Online Information** auslesen. Im untersten Ausgabefeld wird der Firmwarestand ausgegeben. Der Firmwarestand hat keine direkte Auswirkung auf die Bearbeitung Ihrer Sicherheitsapplikation. Änderungen in der Abarbeitung der Sicherungsapplikation werden getrennt von der Firmwareversion in der sogenannten **Ausführungsversion** sowie in den Versionen der Bibliotheken angezeigt. Die Ausführungsversion findet sich im Eigenschaften-Dialog der Sicherheitsapplikation in der Registerkarte **Safety**. Gleiche Ausführungsversion bei gleicher Sicherheitsapplikation garantiert eine identische Bearbeitung. Eine Abnahme der Sicherheitsapplikation bleibt erhalten.

☞ Die Ausführungsversion der Bootapplikation auf der Steuerung ist im Online-Zustand in der Registerkarte **Safety Online Information** des Editors der Sicherheitssteuerung ersichtlich.

Firmware und Gerätebeschreibung

Alle für IndraWorks relevanten Informationen zur Firmware und Hardware eines Gerätes sind in der sogenannten Gerätebeschreibung hinterlegt (siehe Standard-IndraWorks). Passt die Gerätebeschreibung nicht zum Gerät, mit dem sie verbunden wird, erscheint eine Fehlermeldung wie in IndraWorks-Standard. Ein Verbinden mit der falschen Gerätebeschreibung ist somit unmöglich. In der Gerätebeschreibung ist auch hinterlegt, welche Ausführungsversionen das Gerät unterstützt und welche Versionen der Bibliotheken sich auf dem Gerät befinden.

Ausführungsversion ändern

In der Regel wird ein neues Gerät alte Ausführungsversionen unterstützen:

- Durch Beibehalten der Ausführungsversion bleibt das Laufzeitverhalten Ihrer Sicherheitsapplikation erhalten.
- Wird eine Ausführungsversion allerdings nicht mehr unterstützt, so sollte die neueste Version eingestellt werden. In diesem Fall verliert die Sicherheitsapplikation ihren Pin.

☞ Änderung der Ausführungsversion:

Die Ausführungsversion kann im Dialog **Eigenschaften** des Applikationsobjekts unter dem Tab **Safety** ausgewählt werden.

Änderung einer Bibliotheksversion:

Durch Aktualisieren der Gerätebeschreibung kann es passieren, dass neuere Bibliotheksversionen angezogen werden.

Dies erfolgt automatisch so, dass die eingestellten Bibliotheken immer zum verwendeten Gerät passen.

Wenn sich Bibliotheksbausteine geändert haben, haben diese eine andere Prüfsumme und der Anwender muss seine Applikation neu pinnen, verifizieren und abnehmen.

⚠ VORSICHT **Ein Update der Gerätebeschreibung der Sicherheitsapplikation kann zum Verlust der Abnahme der Sicherheitsapplikation führen.**

Informieren Sie sich bei Ihrem Gerätehersteller, ob ein Update problemlos möglich ist.

Verwenden einer anderen Bibliotheksversion ändert unter Umständen den Pin eines verwendeten Bibliotheksbausteins. Ob dies der Fall ist, ist der Vergleichsansicht der Sicherheitsapplikation zu entnehmen.

In diesem Fall muss die Applikation neu gepinnt, verifiziert und abgenommen werden.

Mit der Gerätebeschreibung der Sicherheitssteuerung werden automatisch die richtigen Bibliotheken zu dieser Sicherheitssteuerung in der richtigen Version angezogen.

Der Bibliotheksverwalter dient lediglich der Übersicht über die vorhandenen Bibliotheken und Ihre Bausteine.

Es wird empfohlen, Bibliotheken nicht manuell zu entfernen und hinzuzufügen.

12.4 Aktualisieren der IndraWorks-Version

Für IndraWorks Safety Manager sind 2 Versionen von Bedeutung:

- Die IndraWorks-Version
- Die Version der IndraWorks-Erweiterung "Safety-Manager"

Beide Informationen sind im Menü **Hilfe ▸ Safety-Versionsinformation anzeigen...** zu finden.

IndraWorks und IndraWorks Safety Manager werden so weiterentwickelt , dass eine Softwareaktualisierung ohne Verlust der Abnahme einer Sicherheitsapplikation möglich ist.

☞ Neuere Versionen von IndraWorks und IndraWorks Safety Manager werden stets gemeinsam auf einem Datenträger bereitgestellt und können auch nur gemeinsam installiert werden.

13 Betrieb und Administration der Steuerung

Im Betrieb der Maschine ist die Sicherheit durch die abgenommenen Sicherheitsfunktionen gegeben. Im Betrieb läuft die Maschine für sich oder wird gewartet (offline), oder eine IndraWorks-Instanz ist mit einer Steuerung der Maschine verbunden, zum Beobachten oder zur Administration (online). Dieses Kapitel beschreibt Maßnahmen und Verfahren, die während des Betriebs der Maschine (offline und online) beachtet werden müssen, unter anderem um die Sicherheitsfunktionen zu erhalten.

13.1 IT-Sicherheit im Betrieb

13.1.1 Schutzmaßnahmen für die Sicherheitssteuerung

Ohne Security kein Safety!

⚠ VORSICHT

Das Konzept zur Sicherheit der Maschine geht von bestimmten betrieblichen Rahmenbedingungen und Risiken aus.

Das Betriebskonzept der Maschine muss die IT-Sicherheit (Security) der "Standard"-Steuerungseinrichtungen gewährleisten. Ansonsten könnten Angreifer die Maschine außerhalb dieser Rahmenbedingungen laufen lassen, so dass zusätzliche oder vergrößerte Risiken bestehen, welche die abgenommenen Sicherheitsfunktionen nicht (mehr) gewährleisten (Safety).

⚠ VORSICHT

Das Betriebskonzept der Maschine muss die ITSicherheit (Security) der Sicherheitssteuerung gewährleisten.

Ansonsten könnten Angreifer die Sicherheitsfunktionen so beeinflussen, dass sie selbst bei unveränderten betrieblichen Rahmenbedingungen keine funktionale Sicherheit (Safety) mehr bieten.

Ansonsten könnten Angreifer die Sicherheitsfunktionen so beeinflussen, dass die funktionale Sicherheit (Safety) selbst bei unveränderten betrieblichen Rahmenbedingungen nicht mehr gewähleistet ist.

Für den Einsatz im Betrieb sollte der Zugriffsschutz der Sicherheitssteuerung eingerichtet werden. Insbesondere sollten eventuell während der Applikationsentwicklung verwendete Passwörter (siehe Kap. 13.1.2 "Schutz der Sicherheitssteuerung vor Schreibzugriff" auf Seite 195) durch Passwörter für den Betrieb ersetzt werden. Bei Bedarf kann auch ein lesender Fernzugriff ohne Bestätigung ermöglicht werden.

Während des Betriebs sollten Zugriffsschutz und Fernzugriffsmöglichkeit zu geeigneten Zeitpunkten neu bewertet und gegebenenfalls angepasst werden.

Um den Zugriffsschutz der Hauptsteuerung und der Sicherheitssteuerung zu gewährleisten, müssen die im Folgenden aufgeführten Hinweise beachtet und die Maßnahmen durchgeführt werden.

☞ Auf jeder Ebene kann eine Verletzung der Schutzmechanismen gegen unerlaubte Zugriffe von Außen eine Bedrohung für das Sicherheitssystem darstellen!

Überblick über die Schutzebenen:

- Barriere 1: Zugriffsschutz für die Maschine
- Barriere 2: Zugriffsschutz für die Hauptsteuerung

- Barriere 3: Grundsätzlicher Selbstschutz der Sicherheitssteuerung
  - Barriere 3a. Identifizierung der Sicherheitssteuerung
  - Barriere 3b. Fernpasswort
- Barriere 4: Operationsabhängiger Schutz der Sicherheitssteuerung
  - Barriere 4a. Administrationspasswort
  - Barriere 4b. Benutzerverwaltung im Projekt

Barriere 1: Zugriffsschutz für die Maschine

Das Steuerungssystem einer Maschine kann aus mehreren Steuerungen bestehen. Allgemein gilt, dass ein zwischen ihnen bestehendes Netzwerk (Maschinennetzwerk) geschützt werden sollte.

Wenn das Maschinennetzwerk für Safety Netzwerkvariablen verwendet wird (Topologie T3), reicht eine Zugriffschutz auf jeder einzelnen Steuerung nicht mehr aus. Es kann immer noch die sichere Querkommunikation gestört werden oder es können sogar falsche Daten eingeschmuggelt werden.

Aufgrund von Sicherheitsanforderungen (vgl. Kap. 17.4 "SafetyNetVar.library" auf Seite 599) **muss** dann das Netzwerk mit allen teilnehmenden Steuerungen **physikalisch** von der Außenwelt getrennt sein. Damit ergibt sich für den Zugriffsschutz automatisch das höchste Schutzniveau.

Wenn keine Sicherheitssteuerung in der Maschine Safety Netzwerkvariablen verwendet (Topologien T1 oder T2) und das Maschinennetzwerk mit der Außenwelt verbunden werden soll, ist ein Gateway zu verwenden:

- Es wird als eigenes Netzwerk ausgelegt, das nur über ein Gateway mit der Außenwelt (dem Firmennetz) verbunden ist.
- Auf diesem Gateway den Zugriffsschutz auf das Steuerungsnetzwerk der Maschine einrichten.
- Gateways sollten die Kommunikationsprotokolle mit der Außenwelt auf das absolut Notwendige einschränken, und z.B. keine Safety Netzwerkvariablen-Kommunikation von und zu der Maschine durchlassen.

Barriere 2: Zugriffsschutz für die Hauptsteuerung

⚠ VORSICHT **Zugriffsmöglichkeiten auf Steuerung prüfen!**

Steuerungen sollten nicht vom Internet oder nicht vertrauenswürdigen Netzen aus zugreifbar sein!

Im Speziellen dürfen die Programmier-Ports der Steuerung unter keinen Umständen ungeschützt aus dem Internet zugreifbar sein (meist UDP-Ports 1740..1743 und TCP-Ports 1217 + 11740 bzw. die steuerungsspezifischen Ports)!

Wenn ein Zugriff aus dem Internet dennoch ermöglicht werden muss (z.B. für Fernzugriff auf die Sicherheitssteuerung, vgl. DOK-IWORKS-REMOTE**V14-APRS-DE-E R911382627), dann muss zwingend ein sicheres Verfahren gewählt werden, um sich mit der Steuerung zu verbinden (z.B. VPN).



Um das Risiko von Datensicherheitsverletzungen zu minimieren, empfehlen wir die folgenden organisatorischen und technischen Maßnahmen für das System, auf dem Ihre Applikationen laufen:

Vermeiden Sie soweit als möglich, die SPS und Steuerungsnetzwerke offenen Netzwerken und dem Internet auszusetzen. Verwenden Sie zum Schutz zusätzliche Sicherungsschichten wie ein VPN für Remote-Zugriffe und installieren Sie Firewall- Mechanismen. Beschränken Sie den Zugriff auf autorisierte Personen, ändern Sie vorhandene Standard-Passwörter bei der ersten Inbetriebnahme und auch weiterhin regelmäßig. Wenn Sie trotz allem Ihre Web-Visualisierung veröffentlichen möchten, wird dringend empfohlen, sie zumindest mit einem einfachen Passwortschutz zu versehen, um zu verhindern, dass jemand über Internet auf Ihre Steuerungsfunktionalität zugreifen kann.

Die IT-Security-Analyse sollte prüfen, dass über Angriffe auf das Maschinennetzwerk und die Standardsteuerungen keine Risiken in der Maschine hervorgerufen werden können, die außerhalb dessen liegen, wofür das Sicherheitssteuerungssystem ausgelegt wurde.

Barriere 3: Grundsätzlicher Selbstschutz der Sicherheitssteuerung

Jede Onlineverbindung zur Sicherheitssteuerung unterliegt einer grundsätzlichen Schutzmaßnahme.

Die zwei Anwendungsfälle einer Onlineverbindung (einerseits Entwicklung und Administration vor Ort, andererseits Diagnose aus der Ferne) werden durch zwei verschiedene Barrieren geschützt.

Barriere 3a. Identifizierung der Sicherheitssteuerung

Jeder Zugriff auf die Sicherheitssteuerung zum Zweck der Entwicklung oder Administration erfordert zuvor mindestens einmal die Identifizierung der Sicherheitssteuerung.

Folgende Maßnahme aus dem Security Level 1 der IEC 62443 ist implementiert:

Das Einloggen auf die falsche Steuerung wird durch Drücken des Bestätigungstaster auf der Sicherheitssteuerung vermieden.

Um sich erfolgreich auf die Sicherheitssteuerung einzuloggen, muss die oben genannte Aktionen durchgeführt werden (siehe "Verbindungsbestätigung" auf Seite 127). Diese Maßnahmen können als relativ sicher eingestuft werden, da das Drücken eines Knopfes manuell und direkt an der Sicherheitssteuerung durchgeführt werden muss und die Eingabe der Seriennummer Insiderwissen über die Maschine erfordert.

Barriere 3b. Fernpasswort

Zum Zweck der Diagnose (vgl. DOK-IWORKS-REMOTE**V14-APRS-DE-E R911382627 ist auch ein Zugriff auf die Sicherheitssteuerung ohne physikalischen Zugriff auf die Sicherheitssteuerung (Betätigen des Tasters auf der Sicherheitssteuerung) möglich. Dieser Zugriff ist durch folgende Maßnahmen geschützt:

1. Er wird von der Steuerung nur akzeptiert, wenn der Fernzugriff zuvor durch einen Anwender vor Ort unter Identifikation der Steuerung (Barriere 3.1a) und Angabe des Administrationspassworts (Barriere 3.2a) freigeschalten (erlaubt) wurde.
2. Er erfordert das Fernpasswort.
3. Er ist auf lesenden Zugriff beschränkt.

Befehle, die bei Fernzugang möglich sind:

- "Einloggen"
- "Ausloggen"
- "Aktualisieren" , Schaltfläche in der Registerkarte „Safety Online Information“ der Sicherheitssteuerung
- Logbuch anzeigen und abspeichern, Schaltflächen in der Registerkarte „Log“ der Sicherheitssteuerung

☞ Um einen unerlaubten Zugriff aus dem Netz auf die Sicherheitssteuerung zu verhindern, sollte ein möglichst "starkes" Passwort gewählt werden.

Barriere 4: Operationsabhängiger Schutz der Sicherheitssteuerung

Auch wenn grundsätzlich der Zugriff auf die Sicherheitssteuerung per Identifizierung oder Fernpasswort erlangt wurde, können die dann möglichen Operationen beschränkt sein.

Barriere 4a. Administrationspasswort Jeder Eingriff in den Ablauf und jede Änderung an der identifizierten Sicherheitssteuerung über Online-Dienste setzen die Identifikation der Sicherheitssteuerung (Barriere 3a) voraus und erfordern das Administrationspasswort.

Die Bootapplikation kann mit dem Administrationspasswort (Admin-Passwort) vor nicht autorisiertem schreibendem Zugriff geschützt werden (siehe "Barriere 3: Grundsätzlicher Selbstschutz der Sicherheitssteuerung" auf Seite 193).

Befehle, die durch das Administrationspasswort geschützt sind:

- "Einloggen"
- "Bootapplikation erzeugen"

  "Bootapplikation löschen"
- "Bootapplikation neu starten"
- "Administrationspasswort setzen"
- "Firmware aktualisieren"
- "Fernzugang konfigurieren"
- "Reset Kalt"
- "STOP"
- "START"
- "Werte schreiben"
- "Werte forcen"
- "Forcen für alle Werte aufheben"
- "Gerätenamen ändern"

☞ Um eine Manipulation einer einmal erfolgreich identifizierten Sicherheitssteuerung zu verhindern, sollte ein möglichst "starkes" Passwort gewählt werden.

Barriere 4b. Benutzerverwaltung im Projekt

Jeder Zugriff auf die Inhalte der laufenden Applikation (Variablen lesen, schreiben, stoppen etc.) erfordert ein Projekt mit dem gleichen Stand der Applikation wie auf der Steuerung. Über die Benutzerverwaltung des Projekts kann ein Schutz vor unerlaubten Zugriffsmöglichkeiten erreicht werden.

Es können in der Projektbenutzerverwaltung Rechte so vergeben werden, dass Safety-Objekte eines Projekts nur von bestimmten Benutzergruppen (z.B. "Safety-Entwickler") erstellt bzw. modifiziert werden können.

Jeder Teil einer Sicherheitsapplikation eines IndraWorks-Projekts kann vor nicht autorisiertem Zugriff durch entsprechende Einstellungen in der Benutzerverwaltung geschützt werden.

Um den Zugriffsschutz der Sicherheitsapplikation zu gewährleisten, muss der Anwender für jedes Projekt eine entsprechende Benutzerverwaltung einrichten oder die integrierte Safety Benutzerkonfiguration (siehe Kap. 6.16.2 "Benutzerverwaltung im Projekt" auf Seite 78) verwenden.

Ohne Benutzerverwaltung im Projekt besitzt jede Person, die an dem Projekt mitarbeitet, alle Rechte an den Standard- und Sicherheitsapplikationen im Projekt!

13.1.2 Schutz der Sicherheitssteuerung vor Schreibzugriff

Administrationspasswort setzen

Mit Hilfe des Administrationspassworts kann die Sicherheitssteuerung vor nicht-autorisiertem schreibendem Zugriff geschützt werden.

Um eine Manipulation einer einmal erfolgreich identifizierten Sicherheitssteuerung zu verhindern, sollte ein möglichst "starkes" Passwort gewählt werden.

Die Schaltfläche für den Befehl **Admin-Passwort setzen** befindet sich im Editor der Sicherheitssteuerung in der Registerkarte **Safety Online Information**. Diese kann nach Aktivierung des Befehls **Öffnen** des Kontextmenüs der Sicherheitssteuerung und Anklicken des Tabs **Safety Online Information** geöffnet werden.

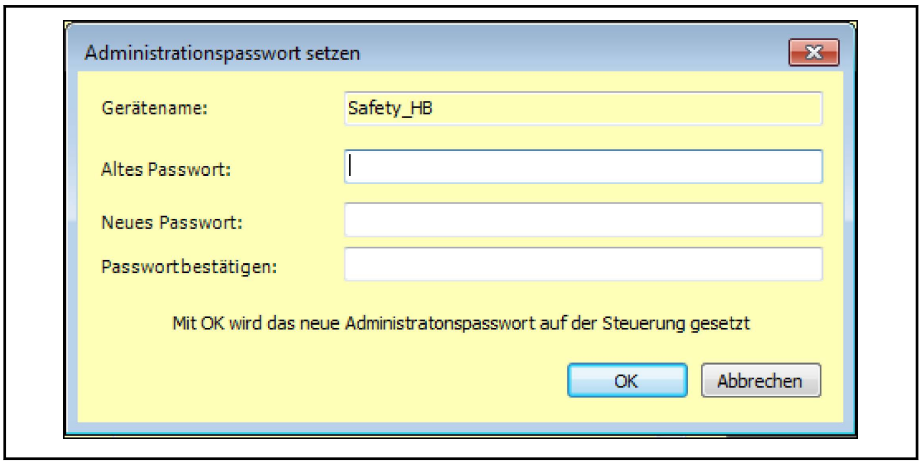


Abb. 13-1: Dialog 'Administrationspasswort setzen'

Bei einem Austausch der SPS wird das Admin-Passwort mit der Sicherheitsapplikation auf die neue Steuerung übertragen.

Bei der Serienfertigung einer Sicherheitssteuerung mit Bootapplikation wird das Admin-Passwort mit kopiert, was zur Folge hat, dass alle Serienmodelle das gleiche Admin-Passwort haben!

Um das Admin-Passwort ändern zu können, muss sich der Service- Mitarbeiter zuerst mit dem bereits existierenden Admin-Passwort autorisieren!

Admin-Passwort Abfrage

Das Admin-Passwort wird vor der Ausführung des ersten schreibenden Online-Befehls nach Öffnen des Projekts, bevor also die Steuerung in den nicht-sicheren Zustand geht, abgefragt.

Ein eingegebenes Passwort bleibt in SafeLogic bis zum Schließen des Projekts aktiv und muss nicht erneut eingegeben werden.

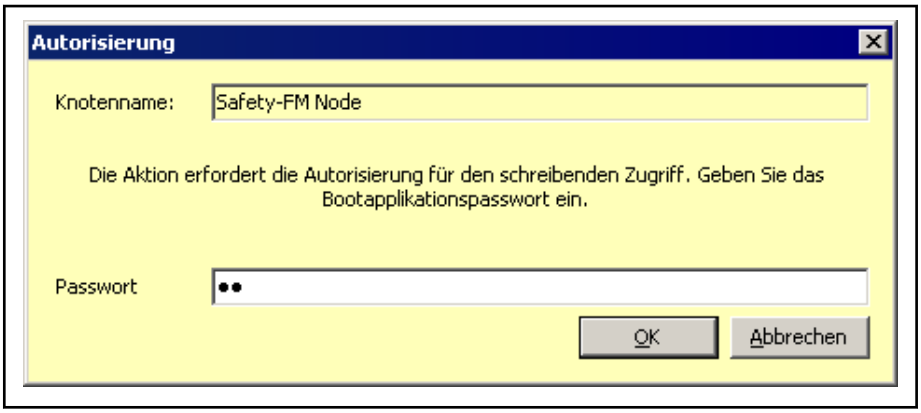


Abb. 13-2: Dialog zur Admin-Passwort Abfrage

13.1.3 Schutz der Sicherheitssteuerung vor Fernzugriff

Fernzugang konfigurieren

Anfänglich ist kein Fernzugang möglich. Er muss erst freigeschalten werden. Dabei muss ein Fernpasswort vergeben werden. Mit Hilfe des Fernpassworts kann die Sicherheitssteuerung vor nicht autorisiertem Fernzugriff (nur lesend) geschützt werden.

☞ Um die Ausspähung einer über die Hauptsteuerung erreichten Sicherheitssteuerung zu verhindern, sollte der Fernzugang nur freigeschalten werden, wenn und solange er für den Betrieb benötigt wird, und es sollte dabei ein möglichst starkes Passwort vergeben werden.

Die Schaltfläche für den Befehl "Fernzugang konfigurieren" befindet sich im Editor der Sicherheitssteuerung in der Registerkarte "Safety Online Information". Über den Dialog kann der Fernzugang freigeschalten werden und wieder blockiert werden, sowie das Fernpasswort jederzeit geändert werden.

Um über den Fernzugang auf die Sicherheitssteuerung zugreifen zu können, wird das Fernpasswort benötigt. Zusätzlich muss auch die ganze Kommunikationsstrecke von außen bis in die Sicherheitssteuerung für den Fernzugang möglich sein. Für den Kommunikationsweg muss ein sicheres Verfahren gewählt werden, um sich mit der Steuerung zu verbinden, zum Beispiel VPN, siehe auch Kap. 13.1.1 "Schutzmaßnahmen für die Sicherheitssteuerung" auf Seite 191.

13.1.4 Beobachtung Security-relevanter Ereignisse

Änderungen an den Schutzmaßnahmen lassen sich wie folgt nachvollziehen: Für jede Passwortänderung erfolgt ein Logbucheintrag.

Ob und wann ein Fernzugriff erfolgt ist, lässt sich wie folgt nachvollziehen: Für jeden Fernzugriff erfolgt ein Logbucheintrag.

13.2 Beobachtung von Fehlern im Betrieb

13.2.1 Beobachtung der Kommunikationsfehler-Häufigkeit

Safety Netzwerkvariablen in der Maschine

⚠ VORSICHT **Restfehlerrate**

Von der Treiberinstanz NetVarReceiver gemeldete Kommunikationsfehler dürfen nicht öfter als 1 Mal in 5 Stunden auftreten, damit die Restfehlerrate pro Stunde für die sicherheitsbezogenen Signale unter dem SIL3-Grenzwert von 10^{-9} bleibt.

13.2.2 Anwenderverhalten bei undefiniertem oder sicherheitsrelevantem Fehler

☞ Der Anwender ist dazu verpflichtet, sicherheitsrelevante Fehler unverzüglich an den jeweiligen Gerätehersteller zu melden!

☞ Das Auftreten eines undefinierten Fehlers muss unverzüglich an den Gerätehersteller gemeldet werden!

13.3 Diagnose von Fehlern im Betrieb

Eine Diagnose der Sicherheitsfunktion kann auf vier Ebenen erfolgen:

- Applikative Diagnose: Funktionen zur Diagnose, die in die Sicherheitsapplikation und Hauptapplikation einprogrammiert wurden, sollten die typischen Diagnosefälle abdecken. Zum Beispiel könnte eine Visualisierung auf der Hauptsteuerung die Fehlercodes aus Treiberbausteinen und PLCopen Funktionsbausteine auswerten (wenn diese per Austauschvariablen in die Hauptsteuerung übertragen werden).
- Diagnose mit Standard IndraWorks-Mitteln:

  Die Verbindung zur Sicherheitsteuerung und die von der Sicherheitssteuerung mit verwendeten Feldbusse und Netzwerkvariablen können bei bestehender Onlineverbindung mit der Hauptsteuerung über die Status-Icons im Gerätebaum und die Registerkarte "Status" des Geräteeditors der Hauptsteuerung oder Sicherheitssteuerung diagnostiziert werden.
- Diagnose mit IndraWorks im sicheren Betrieb: nur lesend, entweder vor Ort mit bestätigter Verbindung (Kap. 8.2 "Verbindung zur Sicherheitssteuerung" auf Seite 126) oder mit Fernzugriff (Kap. 13.3.1 "Verbindung zur Sicherheitssteuerung für Fernzugang" auf Seite 198).

  Möglich sind:
  - Registerkarte „Log“ der Sicherheitssteuerung (siehe Kap. 13.3.2 "Logbuch" auf Seite 198)
  - Registerkarte „Safety Online Information“ der Sicherheitssteuerung (siehe Kap. 13.3.3 "Informationen zu Firmware und Bootapplikation" auf Seite 200)
  - Registerkarte „Status“ der Sicherheitssteuerung (siehe Kap. 13.3.4 "Diagnose der Kommunikation" auf Seite 202)
  - Nach dem Einloggen: Monitoren in Editoren und Überwachungsliste (siehe Kap. 8.7.1 "Monitoring" auf Seite 146), die Ablaufkontrolle ist bei Fernzugriff nicht möglich.

- Nur bei bestätigter Verbindung möglich:

  Debuggen mit IndraWorks: Wenn die genannten Möglichkeiten zur Diagnose nicht ausreichen, besteht als letzte Möglichkeit, die Applikation durch Eingriff in den Ablauf zu debuggen (siehe Kap. 8 "Online-Betrieb der Sicherheitssteuerung" auf Seite 125). Diese stellt eine Form der Wartung dar, bei welcher der sichere Betrieb temporär verlassen wird und die Maschine organisatorisch zu sichern ist (siehe Kap. 8.6.2 "Debug-Modus und organisatorische Sicherheit" auf Seite 143.

13.3.1 Verbindung zur Sicherheitssteuerung für Fernzugang

Die Voraussetzungen für den Fernzugriff sind eine Netzwerkverbindung, die "SafetyManager Lite"-Lizenz (siehe Kap. 5.3 "IndraWorks SafetyManager-Lizenz" auf Seite 29), das Fernpasswort und die Freischaltung des Fernzugriffs. Während des Fernzugriffs ist nur eingeschränkter Zugriff (lesender Zugriff) auf die Sicherheitssteuerung möglich.

1. Befehl "Einloggen" des Menüs "Debuggen" aktivieren
2. Im erscheinenenden Dialog "Verbindung zur Sicherheitssteuerung" die Verbindungsart "Fernzugang" auswählen.

☞ Die Option ist nur verfügbar, wenn der Fernzugang für die Steuerung freigeschalten wurde (siehe Kap. 13.1.3 "Schutz der Sicherheitssteuerung vor Fernzugriff" auf Seite 196)

3. Das Fernpasswort eingeben.
4. Schaltfläche „OK“ aktivieren.

☞ Fernzugang ist für die Verifikation und die Abnahme nicht qualifiziert.

13.3.2 Logbuch

Logbuch

☞ Die Logbuch-Ansicht von SafeLogic entspricht dem Logbuch von Standard-IndraWorks.

Für weitere Detailinformationen wird deshalb auf die IndraWorks-Onlinehilfe verwiesen.

Das Logbuch wird in der Registerkarte **Log** der Sicherheitssteuerung angezeigt und dient als Protokoll und zur Diagnose von Laufzeitfehlern der Applikation und Systemfehlern. Es kann helfen, die Ursache eines Fehlers in der SPS oder in der Applikation zu finden.

☞ Das Logbuch ist für die Verifikation, also für den Nachweis der Fehlerfreiheit und den Nachweis der Erfüllung der Anforderungen nicht zulässig!

In SafeLogic stehen zwei Logbücher zur Verfügung:

- Gerätelogbuch
- Applikations-Logbuch

Das Gerätelogbuch gehört zum Gerät und es ist für Einträge gedacht, die das Gerät betreffen, z. B. Systemfehler, Erzeugen neuer Bootapplikationen.

Das Applikations-Logbuch gehört zur IEC-Applikation und es ist für Einträge gedacht, die die Applikation betreffen, z. B. Laufzeitfehler, Fehler beim Laden der Bootapplikation, Fehler bei der Online-Kommunikation.

Das Applikations-Logbuch ist das Default-Logbuch der Sicherheitssteuerung. Das Geräte-Logbuch steht erst dann zur Verfügung, wenn das Default-Logbuch (Applikations-Logbuch) geladen ist.

Durch Betätigen der Schaltfläche  werden alle zur Verfügung stehenden Logbücher (Geräte- und Applikations-Logbuch) zyklisch von der Steuerung geladen und können in Fenster **Logger** selektiert werden.

Die in den Logbüchern angezeigten Informationen sind folgendermaßen strukturiert:

- Gewichtung (Information, Warnung, Fehler, Ausnahme)
- Zeitstempel
- Fehlerbeschreibung
- Komponente (Erzeuger)

Bei Einträgen von Änderungen aufgrund von Online-Zugriffen wird der Name des Entwicklers mit eingetragen.

Da es keine Benutzerverwaltung auf der Steuerung gibt, wird dabei der Name des Entwicklers aus der Benutzerverwaltung des Programmiersystems verwendet.

Wenn der Entwickler im Programmiersystem nicht als ein spezifischer Benutzer, sondern als einer der vordefinierten Benutzer angemeldet ist, wird statt dessen der Name des Anwenders aus der Benutzerverwaltung von Windows verwendet.

Die Logbücher können als XML-Datei exportiert und importiert werden.

Erzeugung der Logbucheinträge

Logbucheinträge werden insbesondere dann erstellt, wenn die Bootapplikation im Offline-Betrieb aus irgendeinem Grund nicht geladen werden kann und ein Systemfehler auftritt.

Ein Systemfehler wird auch dann erzeugt, wenn bei dem Versuch des Laufzeitsystems, die Bootapplikation beim Hochfahren zu laden, das Logbuch nicht beschrieben werden kann.

☞ Zeigt das Logbuch die Ursache eines Systemfehlers nicht an, so kann der Grund für diesen Systemfehler sein, dass das Logbuch vom sicherheitsgerichteten Laufzeitsystem nicht beschrieben werden konnte.

☞ Bei Fehlern der Hardware konnten eventuell nicht alle Einträge ins Logbuch geschrieben werden.

Ein Logbucheintrag wird erstellt,

- immer wenn das Laufzeitsystem im Offline-Betrieb (d. h. beim Booten ohne Online-Verbindung) gemäß einer Sicherheitsanforderung auf einen Fehler in der Applikation mit dem Abbruch des Ladens reagiert
- immer wenn das Laufzeitsystem im Offline-Betrieb auf einen Fehler der Applikation mit dem Beenden der Ausführung der Applikation reagiert
- bei Vertauschung der Bootapplikation gegenüber dem letzten Bootladen bzw. der letzten Erzeugung einer Bootapplikation. Dies wird durch das Laufzeitsystem erkannt
- bei verfälschter oder unpassender Ausführungsversion des Laufzeitsystems
- Protokollierung von Änderungen der Bootapplikation

- Protokollierung Firmware-Update
- Protokollierung der Erzeugung neuer Bootapplikationen
- Protokollierung von Passwortänderungen
- Protokollierung von Fernzugriffen
- Protokollierung des Tauschs der Sicherheitssteuerung
- Protokollierung des Tauschs der SD-Karte

Logbucheintrag bei Laufzeitfehler

Wird die Applikation wegen eines Laufzeitfehlers beendet, so enthält der erzeugte Logbucheintrag folgende Informationen:

- Name der fehlerhaften POU,
- Nummer des fehlerhaften Netzwerks der POU,
- bei einer FB-POU: FB-Instanz, in welcher sich der Fehler zeigte

13.3.3 Informationen zu Firmware und Bootapplikation

Informationen zur Firmware und zur Bootapplikation

Die Informationen zur geladenen Bootapplikation und zur Firmware und werden in der Registerkarte **Safety Online Information** der Sicherheitssteuerung angezeigt.

☞ Bei Verwendung der Registerkarte **Safety Online Information** für Verifikationstätigkeiten oder Analysen muss anhand des Gerätenamens geprüft werden, dass die Informationen von der gewünschten Steuerung stammen und dass die Informationen über eine bestätigte Verbindung geholt wurden ("aktuelle Verbindung: Identität des Geräts bestätigt").

☞ Durch Aktivierung des Befehls **Aktualisieren** werden alle aktuellen Informationen für die Bootapplikation und Firmware mit aktuellen Werten von der angezeigten Geräteinstanz angezeigt. Es wird dem Entwickler gemeldet, wenn die von der Steuerung hochgelesenen Informationen korrupt sind.

Bei bestätigter Verbindung zur Sicherheitssteuerung können Passwörter, Bootapplikation und Firmware der Sicherheitssteuerung verwaltet werden.

Ist IndraWorks mit keiner Steuerung verbunden, sind die Textausgabefelder leer.

Befehle und Textausgaben:

- Befehl **Aktualisieren**: Damit die aktuellen Informationen zur Bootapplikation in der Steuerung ausgegeben werden, muss zunächst dieser Befehl ausgeführt werden.
- **Gerätename**: Der Name der Sicherheitssteuerung, auf der eingeloggt ist.

Informationen und Befehle zum Zugang zum Gerät:

- **Akutelle Verbindung**: Zeigt an, ob die angezeigten Informationen von einem bestätigten Gerät oder aus einem Fernzugang stammen.
- **Fernzugang**: Zeigt an, ob der Fernzugang auf die Steuerung erlaubt ist.
- Befehl **Admin-Passwort**: Setzt ein Passwort für alle schreibenden Zugriffe auf die Steuerung. Dies ist initial leer. Schreibende Zugriffe auf die Sicherheitssteuerung (z.B. Projekt laden) müssen mit diesem Passwort bestätigt werden. (siehe Kap. 13.1.2 "Schutz der Sicherheitssteuerung vor Schreibzugriff" auf Seite 195)

- Befehl **Fernzugang konfigurieren**: Öffnet einen Dialog, der dazu dient den Fernzugang für die Sicherheitssteuerung zu erlauben, das Fernpasswort zu ändern, oder den Fernzugang zu verbieten (siehe Kap. 13.1.3 "Schutz der Sicherheitssteuerung vor Fernzugriff" auf Seite 196).

Informationen und Befehle zur Bootapplikation: (Diese Information kann nur ausgegeben werden, wenn auf der Steuerung ein Bootprojekt hinterlegt ist.)

Zur Bootapplikation werden folgende Daten angezeigt:

- **Name** Name des Safety-Applikationsobjekts
- **Kommentar**
- **Ausführungsversion**
- **Erzeugt** Zeitpunkt der Erzeugung der Bootapplikation
- Befehl **Neu starten**: Entlädt die aktuelle Applikation, lädt und startet das Bootprojekt. (siehe Kap. 13.4.1 "Bootapplikation starten" auf Seite 210)
- Befehl **Löschen**: Löscht das Bootprojekt in der Sicherheitssteuerung und entlädt die laufende Applikation. (siehe Kap. 13.4.2 "Bootapplikation löschen" auf Seite 210)
- **Bootapplikation-Pin** Informationen zum Pin der Bootapplikation (siehe "Sicherheitsapplikation pinnen" auf Seite 163)

☞ Wurde das Bootprojekt zu einer nicht gepinnten oder nach dem Pinnen veränderten Bootapplikation erzeugt, sind die Ausgabefelder leer.

  - **Name**
  - **Revision**
  - **Prüfsumme**
  - **Erzeugt**

Genauere Informationen siehe "Safety Online Information" auf Seite 40.

Information zur Sicherheitssteuerung

- **Firmware**: Aus der Sicherheitssteuerung wird die genaue Bezeichnung der Firmware ausgelesen.
- Befehl **Firmware-Update**: Lädt ein (anderes) Firmware-Release aus einer Datei auf die Steuerung, um eine dort bereits vorhandene Firmware zu ersetzen.

  siehe Kap. 13.6.2 "Firmware-Update" auf Seite 212
- Befehl **Urlöschen**: Erlaubt das Entladen der Applikation, Löschen der Bootapplikation und Rücksetzen des Passworts. Dieser Befehl kann ohne Passwort ausgeführt werden und erlaubt das Rücksetzen der Steuerung, auch wenn ein Passwort vergessen wurde.

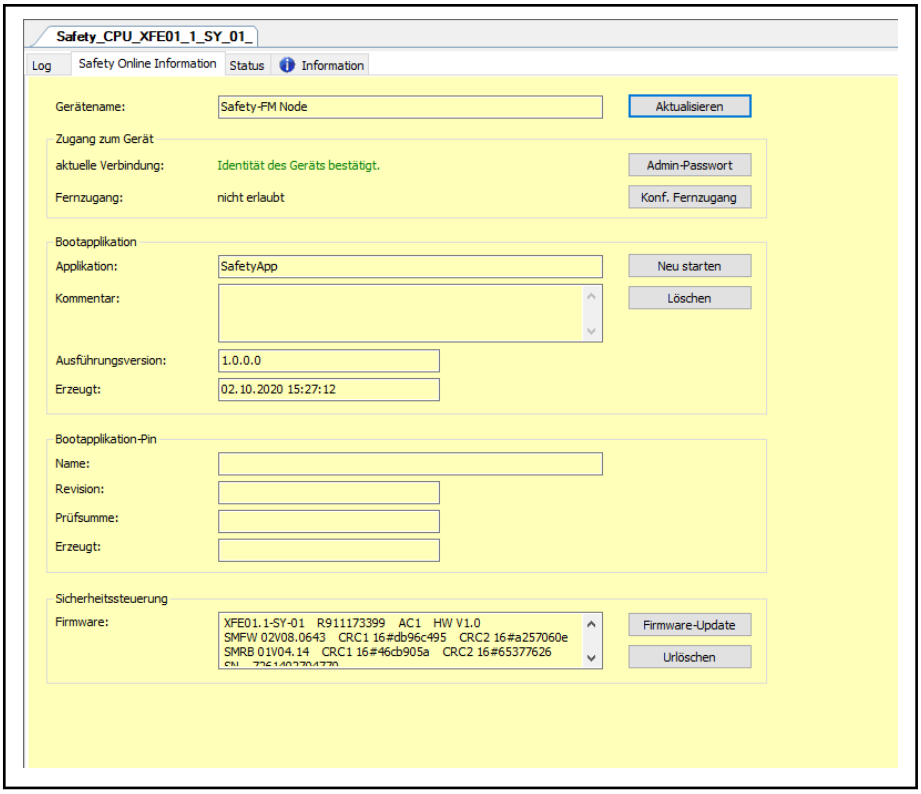


Abb. 13-3: Registerkarte 'Safety Online Information'

Im Feld Firmware werden die Hardwareinformationen des SafeLogic-Funktionsmoduls mitsamt der Hardware-Version, sowie die Firmware mit Prüfsummen angezeigt.

13.3.4 Diagnose der Kommunikation

In der Registerkarte **Status** werden die Diagnosemeldungen der Sicherheitssteuerung und die Diagnosemeldungen über den Verbindungsstatus der Sicherheitssteuerung zu übergeordneten bzw. unterlagerten Geräten angezeigt.

Die Registerkarte steht nur im Online-Modus der Sicherheits-SPS oder der Standardsteuerung zur Verfügung.

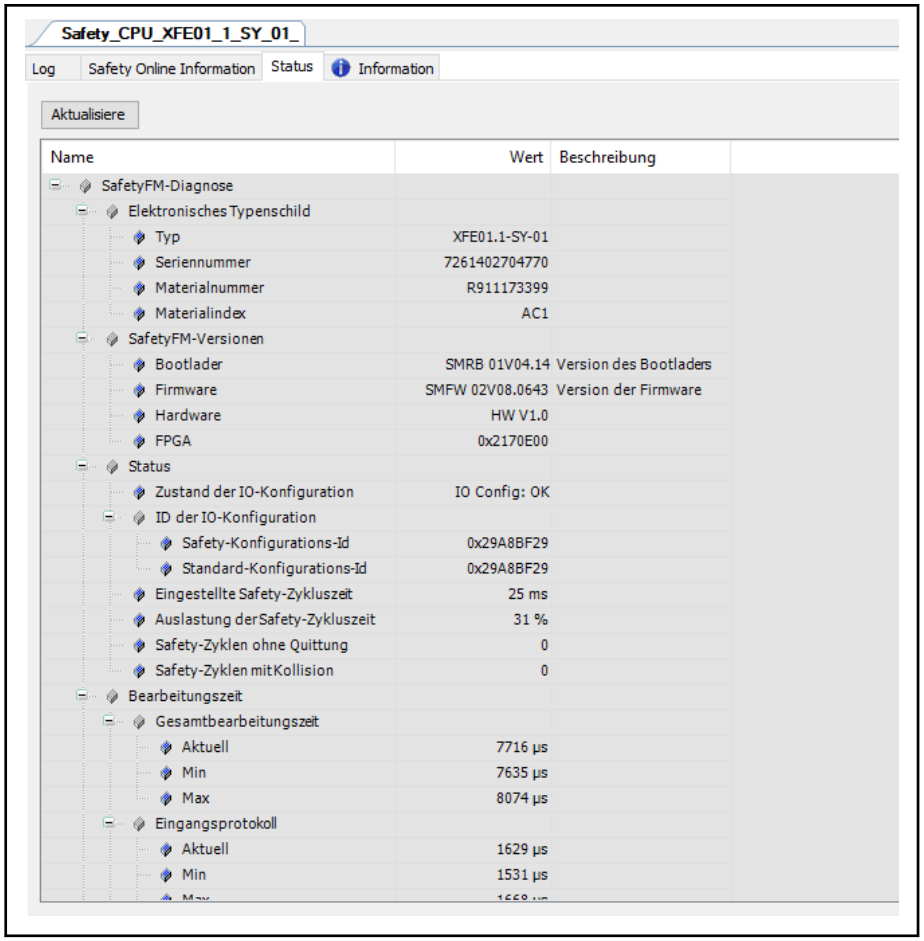


Abb. 13-4: Registerkarte 'Status'

Dieser Dialog zeigt Informationen zum internen Modulstatus an. Mit ihm kann sowohl die Hardware- und Firmware-Version als auch die Auslastung der Safety-Steuerung überprüft werden.

Der Dialog gliedert sich in die folgenden Abschnitte:

- **Elektronisches Typenschild**: Zeigt alle relevanten Hardware-Informationen
- **SafetyFM-Version**: Zeigt alle relevanten Informationen über die Firmware der Sicherheitssteuerung.
  - **Bootloader**: Version der Basis-Firmware. Diese kann nicht geändert werden.
  - **Firmware**: Aktive Safety-Firmware. Diese kann bei Bedarf über IndraWorks aktualisiert werden.
  - **Hardware**: interne Kennung der HW-Version
  - **FPGA**: Version des FPGAs zum Zugriff auf den Funktionsmodulbus. Ein Wert von 0x0 deutet auf ein veraltetes Modul hin.
- **Status**: Liefert einen Überblick über die aktive Safety-Applikation
  - **E/A-Konfiguration**: Zeigt die Prüfsummen über die E/A-Konfiguration der Standard- und der Safety-Steuerung. Nur wenn beide Prüfsummen gleich sind, können die sicheren Feldbusse bedient werden und Variablen zwischen Standard- und Safety-Steuerung ausgetauscht werden.

  - **Safety-Zykluszeit**: Zeigt die aktive Zykluszeit des geladenen Safety-Programms.
  - **Auslastung**: Gibt Auskunft über die aufgewendete Rechenzeit pro Safety-Zyklus.
  - **Safety-Zyklen ohne Quittung und Safety-Zyklen mit Kollision**: Dies sind Diagnoseanzeigen. Sie sollten immer 0 sein.
- **Bearbeitungszeit**: Min, Max und aktueller Wert für:
  - **Gesamtbearbeitungszeit**: Messung über den gesamten Safety-Zyklus, also Eingangsdaten-, SPS- und Ausgangsdatenverarbeitung
  - **Eingangsprotokoll**: Eingangsdatenverarbeitung (hauptsächlich PROFIsafe und CIP)
  - **Applikationsprogramm**: Laufzeit des Anwender SPS-Programms
  - **Ausgangsprotokoll**: Ausgangsdatenverarbeitung (hauptsächlich PROFIsafe und CIP)
- **Systemfehler**: Interne Fehlercodes der Firmware. Im Falle eines kritischen Fehlers werden hier die Fehlercodes ausgegeben, mit der sich die Ursache für den Ausfall zurückverfolgen lässt. Im Normalfall haben alle Einträge dieser Kategorie den Wert 0.
- **Temperatur**: Gibt die gemessene Temperatur der beiden Safety-Prozessoren an. Diese sollte sich innerhalb der angegebenen Systemgrenzen bewegen um einen sicheren Betrieb der Hardware zu gewährleisten.

Meldungen bzgl. des Verbindungsstatus der Sicherheitssteuerung zu übergeordneten bzw. unterlagerten Geräten, werden im Feld für das entsprechende Modul angezeigt.

Es werden folgende Meldungen angezeigt:

- unterschiedliche Konfiguration der E/A-Module bzgl. Anzahl, Id oder E/A-Größe auf Standard und Safety,
- unterschiedliche Konfiguration eines E/A-Moduls bezüglich der Protokollparametrierung auf Standard und Safety.

  Diese Anzeige erfolgt nur für Bussysteme, bei denen die Übertragung der sicheren Konfiguration über die Standardkonfiguration erfolgt (z. B. Profisafe)

13.3.5 Informationen zur Hardware

Informationen zur Hardware

Die Informationen zur Hardware werden im Editor der Sicherheitssteuerung bzw. der logischen E/As (der physikalischen Feldgeräte) unter dem Tab **Information** angezeigt.

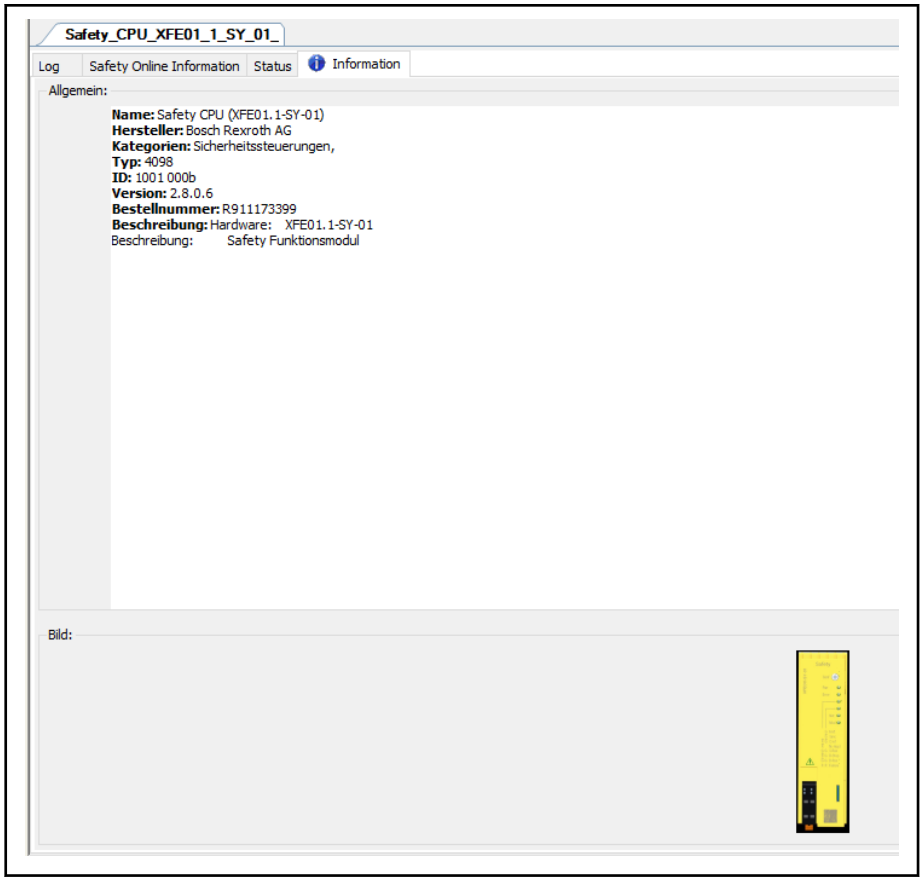


Abb. 13-5: Registerkarte 'Information'

13.3.6 Diagnose auf der Standard-SPS-Steuerung

Für viele Anwendungen ist es notwendig, dass die Standardsteuerung lesenden Zugriff auf die sicheren E/As der Safety-Steuerung hat. Weiterhin werden für den Maschinenbetrieb weitere Status-Informationen aus der Safety-Steuerung benötigt.

Diese Informationen werden über die globale Variable `SafetyData` auf der Standard-SPS zur Verfügung gestellt. Diese Variable ist strukturiert aufgebaut. Sie wird automatisch beim Übersetzen der Standard-SPS-Applikation angelegt und aktualisiert.

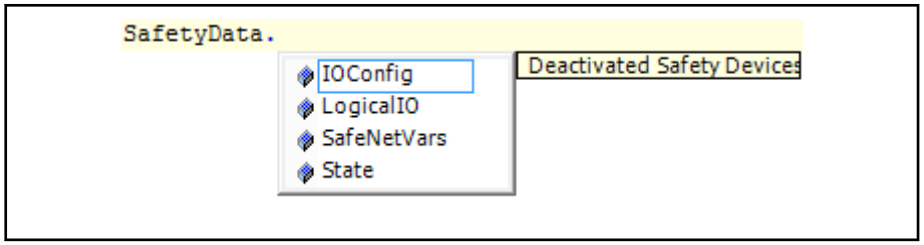


Abb. 13-6: Elemente der globalen Variable SafetyData

Die globale Variable `SafetyData` ermöglicht den Zugriff auf die folgenden Diagnoseinformationen:

- `SafetyFM` - Statusinformationen der Safety-Steuerung:

  Diese Informationen sind größtenteils identisch zu den in "Diagnose der Kommunikation" dargestellten Informationen (Diagnose der Kommunikation auf Seite 202).

- `LogicalIO` - Zugriff auf die sicheren Verbindungen:

Dieser Eintrag ermöglicht den Zugriff auf die sicheren Verbindungen. Er bildet die Geräte unterhalb des Knotens "Logische E/As" der Safety-Steuerung hierarchisch ab. Neben dem Zugriff auf die übertragenen Variablen stehen zusätzlich Verbindungsinformationen zur Verfügung.

- `SafeNetVars` - Zugriff auf die sicheren Netzwerkvariablen:

  Dieser Eintrag stellt alle der sicheren Steuerung bekannten sicheren Netzwerkvariablenlisten zur Verfügung (Safety Netzwerkvariablenliste (NVL) auf Seite 91). Bei Senderlisten ist jede konfigurierte Verbindung einzeln sichtbar. Neben dem Zugriff auf die übertragenen sicheren Variablen wird zusätzlich der Verbindungsstatus angezeigt.

- `IoConfig` - Deaktivierte sichere Teilnehmer:

  Dieser Eintrag bildet die Funktionalität zum Einlernen deaktivierter sicherer Teilnehmer ab (Deaktivieren von sicheren Busteilnehmern auf Seite 153). Neben der Prüfsummer (CRC) der aktuell eingelernten Konfiguration ist auch der Zugriff auf die Deaktivierungsbits möglich.

Laufzeitverhalten

Die globale Variable SafetyData erzeugt keine zusätzliche Rechenlast in der Standard-SPS. Erst ein Zugriff auf einen Eintrag führt zur Bereitstellung der benötigten Daten und somit zu Rechenlast. Einträge, auf die nicht zugegriffen wird, erzeugen keine Rechenlast.

Die Einträge unterhalb der globalen Variable `SafetyData` haben die folgenden Eigenschaften:

- Alle Einträge ermöglichen grundsätzlich nur lesenden Zugriff.
- Die gelesenen Informationen sind als nicht sichere Variablen zu betrachten.
- Die Informationen werden auch aktualisiert, wenn die Standard-SPS steht.
- Die Daten können mit dem SPS-Trace aufgezeichnet werden.
- Die Abtastung der Signale erfolgt zum Zugriffszeitpunkt des Standard SPS-Programm bzw. des Monitorings im Debugger. Sie sind somit nicht zwingend synchron.

SafetyFM

Diese Informationen sind größtenteils identisch zu den in "Diagnose der Kommunikation " dargestellten Informationen (Diagnose der Kommunikation auf Seite 202).

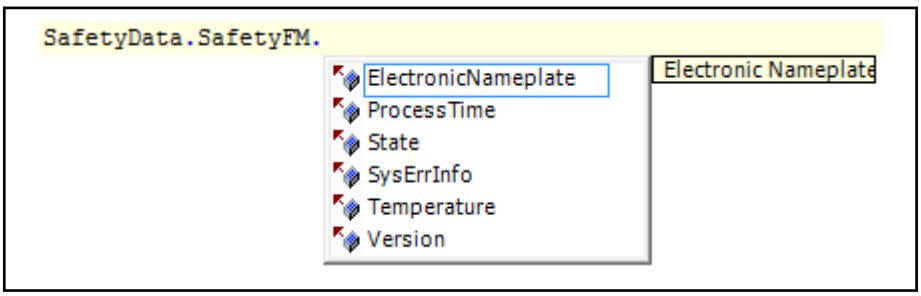


Abb. 13-7: Unterpunkte von SafetyData.SafetyFM

Jeder Unterpunkt enthält weitere Einträge. Für die Verwendung im SPS-Programm sind die folgenden Einträge relevant:

- `SafetyData.SafetyFM.State.PlcState`: Status der Safety-SPS

  Mögliche Werte:

  – `INIT_RB`, `INIT`: Safety-Steuerung bootet

  – `FAILURE_STOP-RB`: Fehler beim Booten der Safety-Steuerung

  - `INIT_WAIT_HOST`: Safety-Steuerung hat Boot-Vorgang beendet und wartet auf die Standard-Steuerung
  - `NO_APPL`: Keine SafeLogic-Applikation geladen
  - `DEBUG_STOP`: Die SafeLogic-Applikation befindet sich im Debug-Modus im Zustand STOP.
  - `DEBUG_RUN`: Die SafeLogic-Applikation befindet sich im Debug-Modus im Zustand RUN.
  - `SAFE_RUN`: Es wurde eine SafeLogic-Bootapplikation geladen. Sie befindet sich im Zustand RUN.
  - `IDLE_HALT`: Die Safety-Steuerung wartet auf einen Reboot.
  - `FAILURE_STOP`: Die Systemfehlereaktion wurde ausgelöst. Weitere Informationen unter `SafetyData.SafetyFM.SysErrInfo`

- `SafetyData.SafetyFM.State.IOConfigIDState`: Status der ConfigID

  Mögliche Werte:

  - `IOCONFIGID_OK`: Die ConfigID der Standard- und der Safety-Steuerung sind identisch. Ein Datenaustausch zu den Feldbusgeräten ist möglich.
  - `IOCONFIGID_ERR`: Die ConfigID der Standard- und der Safety-Steuerung sind verschieden. Ein Datenaustausch zu den Feldbusgeräten ist nicht möglich. Die Applikation der Safety- oder der Standard-Steuerung neu laden.
  - `IOCONFIGID_IDLE`: Keine SafeLogic-Applikation geladen.

LogicalIO Dieser Eintrag ermöglicht den Zugriff auf die sicheren Verbindungen. Er bildet die Geräte unterhalb des Knotens "Logische E/As" der Safety-Steuerung hierarchisch ab. Neben dem Zugriff auf die übertragenen Variablen stehen zusätzlich Verbindungsinformationen zur Verfügung.

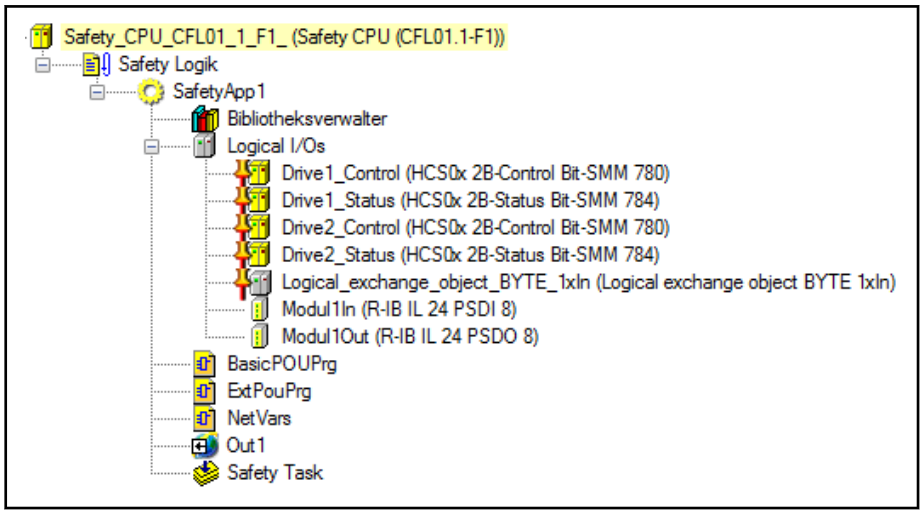


Abb. 13-8: Beispiel für logische E/As unterhalb der Safety-Steuerung

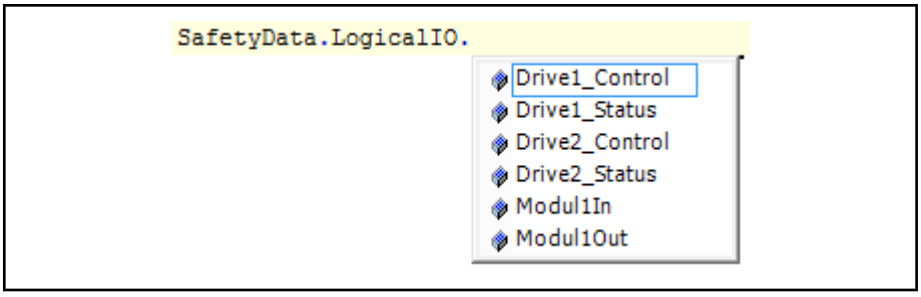


Abb. 13-9: Resultierende Abbildung unter SafetyData.LogicalIO

Unterhalb jedes logischen E/As gibt es weitere Unterpunkte. Hier kann auf alle unterhalb des logischen E/As definierten Variablen zugegriffen werden. Die E/A-Informationen stehen auch noch einmal als Word oder Byte (abhängig von der Gerätebeschreibungsdatei) zur Verfügung. Weiterhin wird unter dem Eintrag `_ConnState` der Zustand der sicheren Verbindung abgelegt.

Der Eintrag `_ConnState` kann die folgenden Zustände annehmen:

- `NoDevice`: Das Gerät ist im Moment am Feldbus nicht verfügbar oder die ConfigIDs der beiden Steuerungen sind verschieden.
- `NoConnection`: Es besteht keine sichere Verbindung zum Gerät
- `FailsafeValues`: Es besteht eine sichere Verbindung. Aufgrund eines vorher aufgetretenen Fehlers werden jedoch sichere Werte (0) übertragen.
- `OK`: Die sichere Verbindung läuft fehlerfrei. Es werden sichere Prozessdaten übertragen.

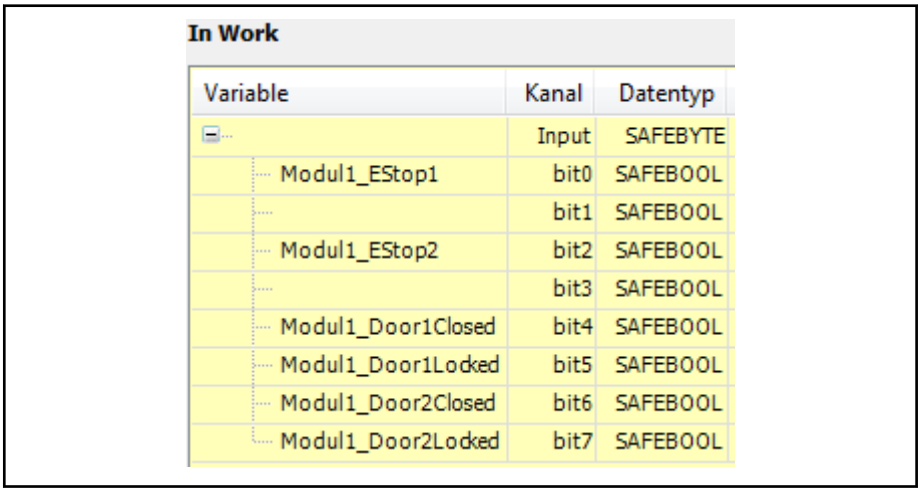


Abb. 13-10: Variablendefinition eines logischen E/As (in diesem Beispiel Modul1In)

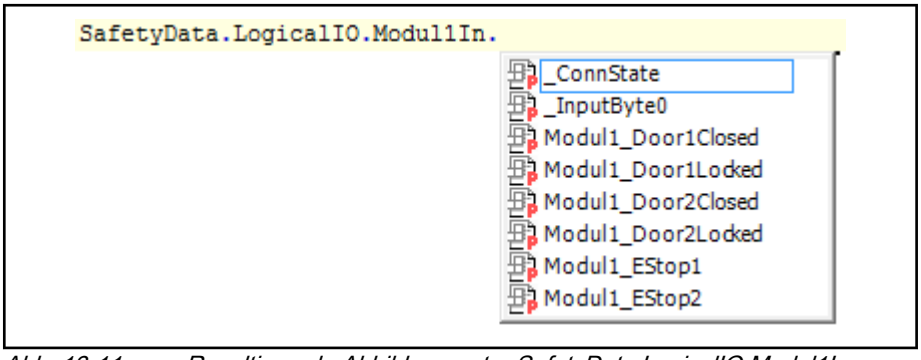


Abb. 13-11: Resultierende Abbildung unter SafetyData.LogicalIO.Modul1In

SafeNetVars

Dieser Eintrag stellt alle der sicheren Steuerung bekannten sicheren Netzwerkvariablenlisten zur Verfügung (Safety Netzwerkvariablenliste (NVL) auf Seite 91). Bei Senderlisten ist jede konfigurierte Verbindung einzeln sichtbar. Neben dem Zugriff auf die übertragenen sicheren Variablen wird zusätzlich der Verbindungsstatus angezeigt.

IOConfig

Dieser Eintrag bildet die Funktionalität zum Einlernen deaktivierter sicherer Teilnehmer ab (Deaktivieren von sicheren Busteilnehmern auf Seite 153). Neben der Prüfsumme (CRC) der aktuell eingelernten Konfiguration ist auch der Zugriff auf die Deaktivierungsbits möglich.

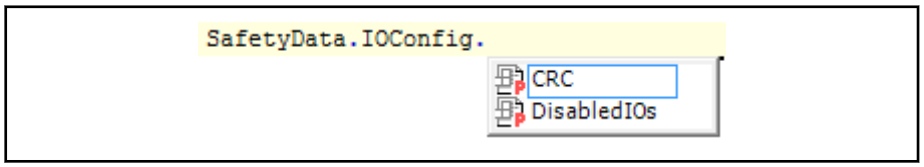


Abb. 13-12: Unterpunkte von SafetyData.IOConfig

Der Eintrag `CRC` liefert die Prüfsumme der aktuell geladenen sicheren Sollkonfiguration. Ist `CRC` 0, so ist keine Sollkonfiguration geladen.

Der Eintrag `DisabledIOs` ist ein Array. Es bildet die Instanzen des SF_IOConfig-Funktionsbausteins ab. Der Array-Index ist hierbei der Eingang "ID" am SF_IOConfig-Funktionsbaustein.

☞ Die Elemente von `SafetyData.IOConfig.DisabledIOs` werden systembedingt nicht sofort aktualisiert.

Es sind mehrere Aufrufe notwendig, bevor die Daten an allen Array-Elementen anstehen.

13.4 Administration mit IndraWorks

Im laufenden Betrieb einer SafeLogic-Sicherheitsapplikation sind für den Service-Mitarbeiter die folgenden Befehle und Aktionen relevant.

Identifikation der Sicherheitsapplikation

Eine Sicherheitsapplikation in der Steuerung wird über die Pin-Kennung der Bootapplikation identifiziert (in der Registerkarte **Safety Online Information** der Sicherheitssteuerung). Diese Pin-Kennung muss mit der Pin-Kennung der Sicherheitsapplikation in SafeLogic (in der Registerkarte **Objekte** des Safety-Applikationsobjekts) übereinstimmen.

Vorgehen im Betrieb

☞ Vor Einbau eines neuen Geräts muss seine Konfiguration gelöscht werden (siehe Bootapplikation löschen, Seite 210).

⚠ VORSICHT

Der Systemintegrierende in der Maschinenproduktion muss beim Download einer abgenommenen Bootapplikation auf die Steuerung prüfen, dass die Erzeugung einer gepinnten Bootapplikation ohne Abweichungen und mit der richtigen Pin-Kennung gemeldet wird.

Wird die Erzeugung einer Bootapplikation eines **In Work**-Standes gemeldet, so muss der Vorgang abgebrochen werden.

Wird vor Ort eine abgenommene Bootapplikation zum Software-Update auf die Steuerung gespielt, muss der Service-Mitarbeiter prüfen, ob eine gepinnte Bootapplikation ohne Abweichungen und mit der richtigen Pin-Kennung gemeldet wird.

Der Service-Mitarbeiter muss bei Übertragung einer abgenommenen Applikation durch ein Speichermedium auf eine andere Steuerung verifizieren, dass die Bootapplikation auf der "neuen" Steuerung, die richtige Pin-Kennung hat (in der Registerkarte **Safety Online Information** der Sicherheitssteuerung).

Hochlauf der Sicherheitsapplikation

Die Sicherheitssteuerung wartet beim Hochlauf bis die Standardsteuerung die Uhrzeit und die Projektinformationen abgleicht. In diesem Zustand leuchten die Pwr-LED grün, die Error-LED rot sowie die Grüne LED für Init. Für diesen Zustand existiert keine zeitliche Überwachung.

13.4.1 Bootapplikation starten

Neustart der Bootapplikation

Eine Bootapplikation auf der Steuerung wird durch Aktivierung des Befehls **Neu starten** (Seite **Safety Online Information** der Steuerung) oder automatisch nach Einschalten der Steuerung neu gestartet.

Neustart bedeutet nicht automatisch, dass die Anlage in jedem Fall losläuft.

Der Entwickler legt in der Applikation fest, ob die PLCopen-Bausteine und die sicheren Ausgangsmodule automatisch anlaufen (AutoReset) oder erst durch ein Standardsignal (Reset).

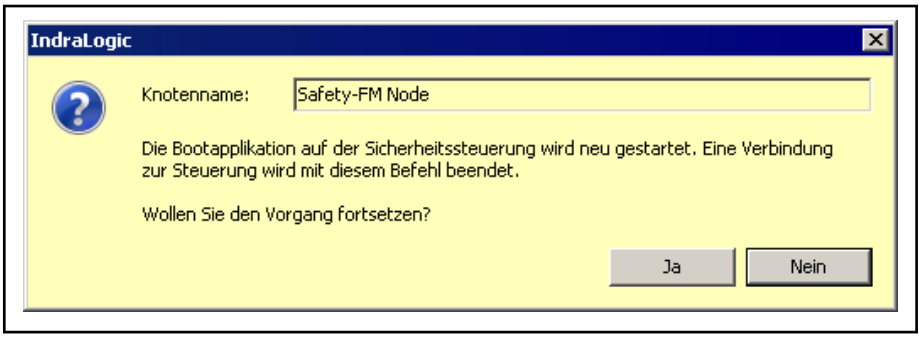


Abb. 13-13: Dialog "Neu starten" der Bootapplikation

Bei Fehlererkennung während des Ladens der neuen Bootapplikation wird der Ladevorgang abgebrochen und es wird ein Logbucheintrag erzeugt.

13.4.2 Bootapplikation löschen

Mit der Schaltfläche **Löschen** der Registerkarte **Safety Online Information** der Sicherheitssteuerung wird die aktuelle Applikation entladen und die Bootapplikation auf der Sicherheitssteuerung gelöscht. Das Löschen der Bootapplikation ist nur nach bereits erfolgter Autorisierung durch das Admin-Passwort möglich.

Eine bestehende Online-Verbindung zur Sicherheitssteuerung wird nach Bestätigung durch den Service-Mitarbeiter durch die Ausführung des Befehls beendet.

Dieser Befehl ist nur aktivierbar, wenn der Befehl **Aktualisieren** mindestens einmal ausgeführt wurde und auf der Steuerung BA-Informationen verfügbar waren.

Die Schaltfläche **Aktualisieren** befindet sich in der Registerkarte **Safety Online Information**.

Der Service-Mitarbeiter erhält immer eine Rückmeldung, ob der Befehl erfolgreich ausgeführt wurde oder nicht.

⚠ VORSICHT

Wird der Befehl Löschen der Bootapplikation nicht vollständig ausgeführt, so kann es nach Neustart zu einer Gefährdung in der Anlage kommen, weil diese noch mit der alten Applikation anläuft.

Der Service-Mitarbeiter muss auf die Rückmeldung warten, ob der Befehl erfolgreich ausgeführt wurde. Bleibt diese aus, so ist die Steuerung so zu behandeln, als ob die Bootapplikation nach Neustart wieder starten kann.

☞ Der Service-Mitarbeiter kann sich zusätzlich in der Anzeige der Applikationsinformationen zur Applikation auf der Steuerung in der Registerkarte **Safety Online Information** der Sicherheitssteuerung vergewissern, dass sich keine Bootapplikation mehr auf der Steuerung befindet.

Anzeige der Applikationsinformation

1. Sicherheitssteuerung im Projektexplorer auswählen
2. Befehl **Öffnen** des Kontextmenüs aktivieren
3. Registerkarte **Safety Online Information** öffnen
4. Schaltfläche **Aktualisieren** betätigen
5. Überprüfen, ob keine Bootapplikation mehr existiert

☞ Das Admin-Passwort wird bei der Ausführung des Befehls **Löschen** nicht gelöscht.

Das Admin-Passwort kann nur durch **Urlöschen** der Bootapplikation gelöscht werden (siehe "Urlöschen" auf Seite 227).

13.5 Verfahren für die Wartung

13.5.1 Temporärer Wechsel in den unsicheren Betrieb

Das Debuggen einer laufenden Applikation (vgl. Kap. 13.3 "Diagnose von Fehlern im Betrieb" auf Seite 197) und das Aktualisieren von Bootapplikation (Kap. 13.6 "Wartung und Service" auf Seite 212) oder Firmware (Kap. 13.6 "Wartung und Service" auf Seite 212) bedingen, dass die Sicherheitssteuerung den sicheren Betrieb verlässt. Debuggen und Aktualisieren erfordern eine bestätigte Verbindung zur Sicherheitssteuerung (Fernzugriff reicht nicht) und organisatorische Sicherungsmaßnahmen.

13.5.2 Verlassen des Betriebsmodus - Debuggen der Maschine

⚠ VORSICHT **Solange sich in einer Maschine eine Sicherheitssteuerung, die eine Safety Netzwerkvariablenliste (Sender) enthält, im Debug-Betrieb befindet, muss die ganze Maschine organisatorisch abgesichert sein.**

Dies bedeutet auch, dass sich keine Personen in Gefahrenbereichen aufhalten, die durch Sicherheitssteuerungen überwacht werden, die über die sichere Querkommunikation mit der zu debuggenden Sicherheitssteuerung verbunden sind.

☞ Bitte beachten Sie: Debug-Modus und organisatorische Sicherheit, Seite 143.

13.6 Wartung und Service

13.6.1 Neue Bootapplikation aufspielen

⚠ VORSICHT **Abnahmestand der Safety-Applikation prüfen!**

Der Systemintegrierende in der Maschinenproduktion muss beim Download einer abgenommenen Bootapplikation auf die Steuerung prüfen, dass die Erzeugung einer gepinnten Bootapplikation ohne Abweichungen und mit der richtigen Pin-Kennung gemeldet wird. Wird die Erzeugung einer Bootapplikation eines "In Work"-Standes gemeldet, so muss der Vorgang abgebrochen werden.

☞ Wird vor Ort eine abgenommene Bootapplikation zum Software-Update auf die Steuerung gespielt, muss der Service-Mitarbeiter prüfen, ob eine gepinnte Bootapplikation ohne Abweichungen und mit der richtigen Pin-Kennung gemeldet wird.

☞ Der Service-Mitarbeiter muss bei Übertragung einer abgenommenen Applikation durch ein Speichermedium auf eine andere Steuerung verifizieren, dass die Bootapplikation auf der "neuen" Steuerung, die richtige Pin-Kennung hat (in der Registerkarte **Safety Online Information** der Sicherheitssteuerung).

13.6.2 Firmware-Update

☞ Bevor ein Firmware-Update durchgeführt wird, sollte sich der Service-Mitarbeiter über den Grund und die Folgen des Firmware-Updates im Klaren sein.

Siehe Kap. 12.3 "Aktualisieren der Firmware, die Ausführungsversion" auf Seite 188.

☞ Erfolgt mit dem Firmware-Update ein Release-Sprung, so muss die Bootapplikation vor dem Firmware-Update gelöscht werden. Nach erfolgtem Update kann die Bootapplikation wieder eingespielt werden. Anhand des Pins der Applikation ist der korrekte Download der Sicherheitsapplikation sicherzustellen.

Die Firmware auf der Sicherheitssteuerung kann durch einen Firmware-Update ausgetauscht werden. Hierzu dient die Schaltfläche **Firmware-Update** im Geräte-Editor **Safety Online Information**. Dabei wird die neue Firmware aus einer Datei heraus auf die Sicherheitssteuerung geladen und die bisherige Firmware der Sicherheitssteuerung ersetzt.

Der Befehl ist nur aktivierbar, wenn mindestens einmal der Befehl **Aktualisieren** ausgeführt wurde.

Zur Ausführung des Befehls ist die Autorisierung mit dem Admin-Passwort erforderlich.

☞ Nach dem Download der Firmware muss die Sicherheitssteuerung aus- und wieder eingeschaltet werden. Der anschließende Steuerungshochlauf darf unter keinen Umständen unterbrochen werden. Die Firmwareaktualisierung erfolgt während des Hochlaufs der Steuerung. In diesem Zeitraum leuchten die Power- und die Error-LED.

☞ Vor dem Update auf die Version 02V08 von einer Firmwareversion 02V06 oder kleiner muss vor dem Download der Firmware die Bootapplikation gelöscht werden. Nach erfolgreichem Firmwareupdate und anschließenden Neustart der Steuerung kann die Bootapplikation wieder aufgespielt werden.

Nach erfolgtem Firmware-Update muss geprüft werden, ob die neue Firmware auch aktiv ist. Dies erfolgt anhand der Firmware-Versionsanzeige.

13.7 Hardware-Tausch

Eine abgenommene Sicherheitsapplikation und deren Verifikationsergebnisse können auf weitere Sicherheitssteuerungen übertragen werden, ohne die Applikation neu verifizieren und abnehmen zu müssen.

☞ Die Zweiterzeugung einer im Projekt unveränderten Sicherheitsapplikation führt bei jedem kompatiblen Firmwarestand zum gleichen Verhalten der Bootapplikation.

Hardware-Tausch

Eine Bootapplikation kann auf ihrem Datenträger physikalisch in eine andere Steuerungshardware übertragen und dort weiter betrieben werden, sofern auf der neuen Steuerung keine Inkompatibilität durch einen anderen Firmwarestand entsteht (Aktualisieren der Firmware, die Ausführungsversion auf Seite 188).

Dies bedeutet:

- Eine defekte Steuerung in der Anlage kann gegen eine Neue ausgetauscht und mit der alten Applikation weiter betrieben werden.

☞ Das Admin-Passwort und die Boot-Applikation werden zusammen, bei einem Austausch des Datenträgers, auf die neue Steuerung übertragen.

Beim Tausch des Datenträgers in eine neue Steuerungshardware erfolgt ein einmaliger Logbucheintrag:

```
ID_0000230B: Baugruppentausch: HW- Seriennummer von {p0}
ID_0000230C: Baugruppentausch: auf {p0} geändert
```

Beim Tausch der SD-Karte wird der folgende Logbucheintrag generiert:

```
ID_00002309: SD-Kartentausch: SD- Seriennummer von {p0} auf {p1} geändert
```

Bei der erstmaligen Inbetriebnahme der Baugruppe werden die folgenden beiden Einträge in das Logbuch geschrieben:

```
ID_00002308: SD- Seriennummer {p0} initial angelegt
ID_0000230A: HW- Seriennummer {p0} initial angelegt
```

Serieninbetriebnahme

Die Zweitapplikation kann durch Kopieren des Datenträgers auf eine andere Steuerung übertragen werden.

Bei der Zweiterzeugung einer unveränderten Applikation auf einer weiteren SPS während der Verifikation oder nach der Abnahme muss sich der Service-Mitarbeiter vergewissern, dass die Bootapplikation dem gewünschten Ausführungsstand entspricht.

Er muss verifizieren, dass beim Download nicht die Erzeugung eines **In Work**-Standes, sondern eines gepinnten Ausführungsstandes mit der richtigen Pin-Kennung angekündigt wird.

Siehe Pinnen auf Seite 161.

Bei Zweiterzeugung einer unveränderten abgenommenen Applikation auf einer weiteren Sicherheitssteuerung durch Kopieren des Datenträgers muss nachgewiesen werden, dass die zweiterzeugte Bootapplikation dem Ausführungsstand der Abnahme entspricht.

Bei Multi-SPS-Projekten muss beim Zweiterzeugen der Bootapplikation in einem Bestätigungsdialog die Pin-Kennung geprüft werden, damit die richtige Bootapplikation auf der richtigen Steuerung erzeugt wird.

Dies bedeutet:

- Zur Applikations-Serienfertigung kann die Kopie einer Bootapplikation auf andere Datenträger übertragen und dort weiterbetrieben werden.
- Das Admin-Passwort wird auch bei Serienfertigung mit kopiert, somit befindet sich nach dem Kopiervorgang auf allen Serienmodellen das gleiche Admin-Passwort.

13.8 Änderungen in Netzwerken und Feldbussen

Bei allen Änderungen am Netzwerk auf Maschinenebene und an Feldbussen müssen die entsprechenden Systemanforderungen eingehalten werden. Siehe Feldbusse und Netzwerkvariablen, Allgemeines auf Seite 573.

14 Verfahren bei Änderungen und Wiederverwendung der abgenommenen Software

14.1 Verfahren bei Änderungen und Wiederverwendung der Software

Bei Änderungen und Wiederverwendung sind folgende Fälle zu unterscheiden:

- Unveränderte Wiederverwendung einer Sicherheitsapplikation in unverändertem Systemkontext (siehe Wiederverwendung eines abgenommenen Safety-Projekts, Seite 218). Hier wird kein Software-Entwicklungsprozess durchlaufen.
- Unveränderte Wiederverwendung einer Sicherheitsapplikation in einem neuem Systemkontext (siehe Wiederverwendung eines abgenommenen Safety-Projekts, Seite 218). Hier wird erfolgt zwar keine eigentliche Softwareentwicklung, aber die systemweite Planung von Adresseindeutigkeit und Querkommunikation zwischen Sicherheitssteuerung (siehe Planung des Gesamtsystems, Seite 19) und die Verifikation in der Maschine (siehe Verifikation in der fertigen Maschine, Seite 181) müssen wiederholt werden.
- Unveränderte Wiederverwendung einzelner Funktionsbausteine (siehe Wiederverwendung von Funktionsbausteinen, Seite 219) innerhalb des Software-Entwicklungsprozesses für eine neue Sicherheitsapplikation
- Änderung an einer Sicherheitsapplikation, die abgenommen ist (siehe Änderungen im Projekt, Seite 222). Dieses bedingt den Start eines Software-Entwicklungsprozesses für diese neue, geänderte Sicherheitsapplikation.
- Änderung an einer Sicherheitsapplikation, die in der Verifikation ist (siehe Änderungen im Projekt, Seite 222). Dieses bedingt einen Rücksprung des Software- Entwicklungsprozesses für diese Sicherheitsapplikation von der Verifikation in die Programmierung

Wiederverwendung von Software kann sich auf ein komplettes SafeLogic-Projekt oder nur auf einen oder mehrere Funktionsbausteine beziehen.

Bei Änderungen gibt es die Varianten, dass Änderungen während der Verifikation einer Sicherheitsapplikation oder an einer bereits abgenommenen Sicherheitsapplikationen vorgenommen werden.

Hinweise zu Änderungen und Wiederverwendung

Grundsätzlich gilt für alle Sicherheitsapplikationen die mit SafeLogic erstellt wurden:

☞ Es liegt in der Verantwortung des Service-Mitarbeiters sich zu vergewissern, dass die erzeugte Bootapplikation dem gewünschten Ausführungsstand der Applikation (Applikations-Pin) entspricht.

☞ Verifikation und Abnahme darf nur auf gepinnten Sicherheitsapplikationen durchgeführt werden. Kein Objekt der Applikation darf den Status **In Work** aufweisen.

Für den Nachweis von Strukturänderungen der Applikation, bzw. welche Objekte unverändert sind und für die Auswirkungsanalyse von Änderungen oder für die Differenzabnahme muss die Vergleichsansicht des Applikationsobjekts verwendet werden. Der Standard-Projektvergleich ist hierfür nicht geeignet.

Der Standard-Projektvergleich kann lediglich als Hilfsfunktion verwendet werden, um die Vergleichsansicht geänderter Objekte zu öffnen.

Es muss mittels der Pin-Kennung verifiziert werden, dass der Strukturvergleich (Vergleichseditor) die gewünschten Applikationsstände vergleicht.

Bei Erzeugung einer neuen Bootapplikation zur Verifikation nach einer Änderung muss sich der Service-Mitarbeiter vergewissern, dass die erzeugte Bootapplikation dem gewünschten Ausführungsstand (Applikations-Pin) entspricht.

Er muss verifizieren, dass beim Download nicht die Erzeugung eines **In Work**-Standes, sondern eines gepinnten Ausführungsstands mit der richtigen Pin-Kennung angekündigt wird.

Siehe Pinnen, Seite 161.

Dokumentation von Änderungen

Die für die Einflussanalyse und die Dokumentation von Änderungen qualifizierte Ansicht ist die Vergleichsansicht des Safety-Applikationsobjekts.

Ein Ausdruck der Vergleichsansicht erfolgt über den Menüpunkt **Datei ▸ Drucken...**. Es muss dabei im Auswahldialog **Projekt- und Gerätedaten drucken** der Projektbaum-Knotenpunkt **SafetyApp** ausgewählt werden.

Der Ausdruck der Querverweisliste erfolgt durch Aktivierung der Schaltfläche  im Editor der Querverweisliste.

Des Weiteren kann für die Dokumentation von Änderungen der Vergleichseditor des Safety-Applikationsobjekts ausgedruckt werden.

Der Ausdruck erfolgt in diesem Fall durch Aktivierung des Befehls **Vergleich drucken...** im Vergleichseditor.

14.2 Wiederverwendung eines abgenommenen Safety-Projekts

Die Abnahme bzw. die Verifikationsergebnisse eines IndraWorks Safety Manager-Projekts können auf weitere Sicherheitssteuerungen übertragen werden, ohne die Applikation erneut verifizieren und abnehmen zu müssen, z.B. zur Serienfertigung von Maschinen.

⚠ VORSICHT

Eine bereits abgenommene Sicherheitsapplikation kann nur auf folgende Weise auf eine andere Steuerung gebracht werden, ohne die Abnahme zu verlieren:

Duplikation der Bootapplikation aus der ursprünglichen Steuerung durch Kopieren des Datenträgers,

andernfalls ist die Sicherheitsapplikation auf der Zweitsteuerung erneut zu verifizieren und abzunehmen.

☞ Bei Serienfertigung wird das Admin-Passwort mit kopiert!

Dies bedeutet, dass alle Serienmodelle das gleiche Admin-Passwort haben.

Die Zweiterzeugung einer im Projekt unveränderten Applikation führt bei jedem kompatiblen Firmwarestand zum gleichen Verhalten der Bootapplikation.

☞ Soll eine abgenommene Sicherheitsapplikation auf einer weiteren Sicherheitssteuerung verwendet werden (Zweiterzeugung), folgen Sie den Anweisungen des Herstellers der Sicherheitssteuerung. Unter Umständen stellt er einen eigenen sicheren Mechanismus zum Kopieren der Applikation zur Verfügung.

☞ Es liegt in der Verantwortung des Service-Mitarbeiters sich zu vergewissern, dass die Bootapplikation dem gewünschten Ausführungsstand der Applikation (Applikations-Pin) entspricht.

☞ Bei Duplizieren einer unveränderten abgenommenen Applikation auf einer weiteren Sicherheitssteuerung durch Kopieren des Datenträgers muss nachgewiesen werden, dass die zweiterzeugte Bootapplikation dem Ausführungsstand (Applikations-Pin) der Abnahme entspricht.

☞ Bei Multi-SPS Projekten muss beim Zweiterzeugen der Bootapplikation in einem Bestätigungsdialog die Pin-Kennung und der Gerätename geprüft werden, damit die richtige Bootapplikation auf der richtigen Steuerung erzeugt wird.

☞ Es liegt in der Verantwortung des Anwenders sich zu vergewissern, dass die Bootapplikation dem gewünschten Ausführungsstand der Applikation entspricht. Er muss verifizieren, dass beim Download nicht die Erzeugung eines "In Work"-Standes, sondern eines gepinnten Ausführungsstandes mit der richtigen Pin-Kennung angekündigt wird (Pinnen, Seite 161).

14.3 Wiederverwendung von Funktionsbausteinen

Ein bereits validierter Funktionsbaustein einer abgenommenen Sicherheitsapplikation kann in einem anderen Projekt unter bestimmten Bedingungen wieder verwendet werden.

Dabei können die Validierung und die Testergebnisse des Funktionsbausteins auf die neue Applikation übertragen werden.

Die Identifikation eines wiederverwendeten Baustein muss nachgewiesen werden:

- Wenn der Funktionsbaustein als Teil einer Applikation validiert wurde
- Wenn der Funktionsbaustein in die Applikation kopiert und wiederverwendet wird
- Wenn der Funktionsbaustein aus einer Bibliothek heraus wiederverwendet wird

☞ Der Nachweis von wiederverwendeten Bausteinen (IEC-Bausteine und externe Bausteine) kann sowohl über die Vergleichsansicht des Applikationsobjekts als auch über seine Pin erfolgen.

☞ Wird ein Funktionsbaustein in einem anderen Projekt wiederverwendet, so muss die Prüfsumme des FBs und die Prüfsummen der FBs, die von diesem FB aufgerufen werden, den Prüfsummen der Abnahme entsprechen! Ist dies nicht der Fall, so muss der Funktionsbaustein neu verifiziert und validiert werden.

Siehe Kap. 10.1 "Software-Verifikation, Allgemeines" auf Seite 167.

☞ *Im IndraWorks Safety Manager führt ein Baustein bei jedem kompatiblem Firmwarestand immer wieder zum gleichen Verhalten auf der Steuerung wenn er*

- unverändert wieder verwendet wird und
- keine globalen Variablen enthält und
- keinen FB verwendet, dessen Datenlayout sich geändert hat.

Versionsänderung externer FBs

Bei externen FBs dient die Version dazu, die Implementierung der Sicherheitssteuerung zu identifizieren.

In der Gerätebeschreibung der Sicherheitssteuerung ist hinterlegt, welche Version der externen FBs (PLCopen, Standard) zu dieser Sicherheitssteuerung gehört. Diese Bibliotheken werden automatisch geladen.

Ändert sich die Version einer Bibliothek, so wechselt der Pin-Status der Sicherheitsapplikation auf **In Work**.

In der Vergleichsansicht des Safety-Applikationsobjekts ist deutlich erkennbar, dass sich die Applikationen (Projektstatus und gepinnter Status) nur in diesem externen FB unterscheiden. Es muss eine Auswirkungsanalyse gemacht werden. Eventuell muss nicht die gesamte Applikation neu getestet werden, sondern nur die von dem entsprechen externen FB betroffenen Objekte.

Übernahme von FBs aus anderen Projekten

Die Vorgehensweise beim Vergleich von Safelogic-Objekten ist analog zum Vergleich von Standardobjekten.

☞ Bei Safelogic-Objekten ist es generell nicht möglich Unterschiede zusammenzuführen.

Nachstehendes Beispiel zeigt, dass vom Vergleichprojekt (Datei) zum aktuellen Projekt zwei Objekte (POU_1 und GVL) gelöscht wurden (blau markiert). Dies bewirkt Änderungen im "Safety Task" (rot).

Änderungen gab es gleichfalls innerhalb des "SafeProg" (rot, unterschiedliche CRC).

Änderungen gab es ferner im Namen, der Gesamtprüfsumme und im Zeitstempel der letzten Änderung.

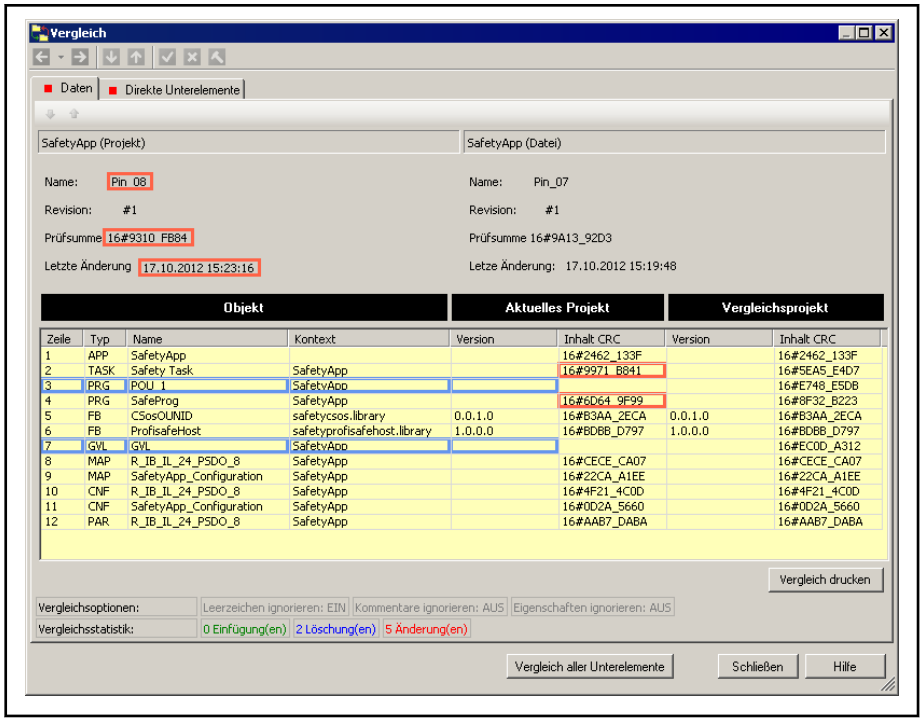


Abb. 14-1: SafetyApp im Safety-Projektvergleich

Das Registerblatt "Direkte Unterelemente" bringt die inhaltlichen Unterschiede in tabellarischer Form.

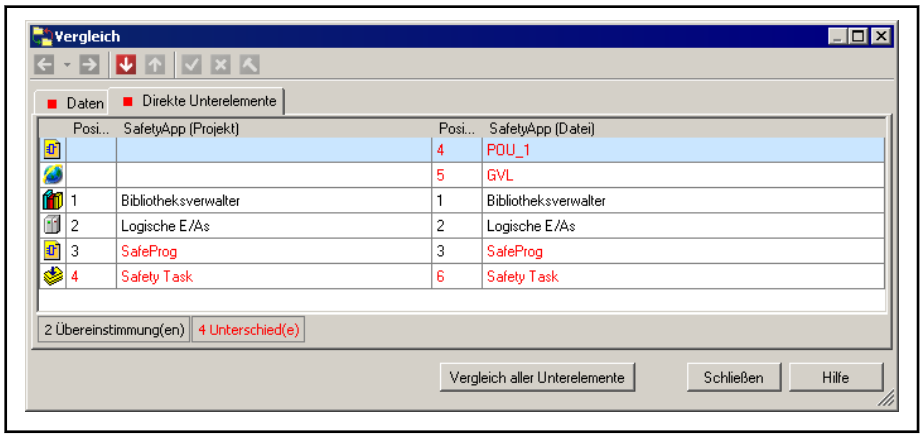


Abb. 14-2: Vergleich der Projektstruktur

Analog ist das Verhalten des Vergleiches des gesamten SPS-Projekts. Hier werden gleichartige Standard- und Safety-Objekte mit einander verglichen.

Unterschiede lassen sich erst durch den Vergleich aller Unterelemente, hier gefiltert "Alle Unterschiede", erkennen.

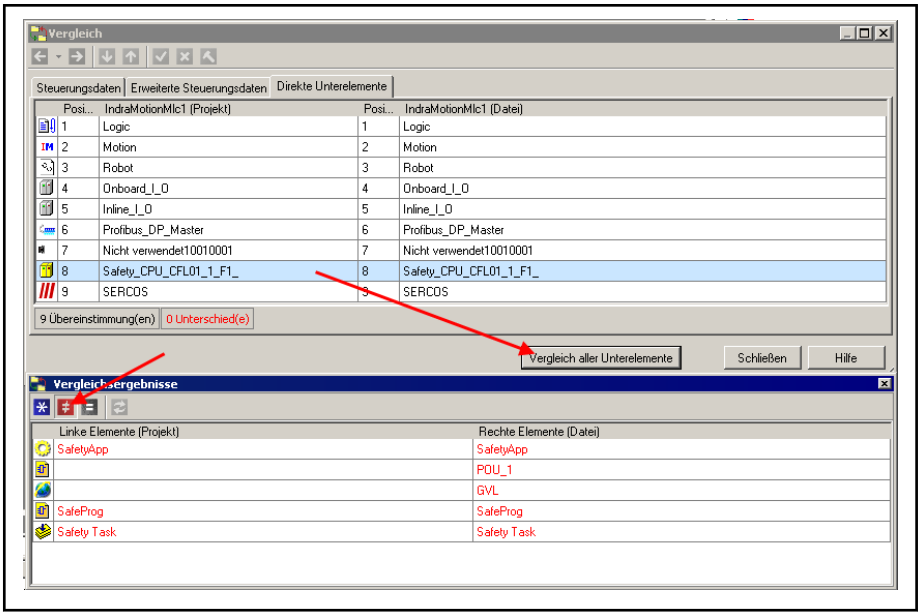


Abb. 14-3: SPS-Projektvergleich

14.4 Änderungen im Projekt

Für den Nachweis von Strukturänderungen der Applikation, bzw. welche Objekte unverändert sind und für die Auswirkungsanalyse von Änderungen oder für die Differenzabnahme muss der Vergleichseditor des Applikationsobjekts verwendet werden. Der Standard-Projektvergleich ist hierfür nicht geeignet.

Der Standard-Projektvergleich kann lediglich als Hilfsfunktion verwendet werden, um die Vergleichsansicht des gepinnten Stands der Applikation zu öffnen. Die Vergleichsansicht wird durch einen Doppelklick auf das Safety-Applikationsobjekt im Standard-Projektvergleich geöffnet.

Es muss mittels der Pin-Kennung verifiziert werden, dass der Strukturvergleich (Vergleichseditor) die gewünschten Applikationsstände vergleicht.

Bei Erzeugung einer neuen Bootapplikation zur Verifikation nach einer Änderung muss sich der Service-Mitarbeiter vergewissern, dass die erzeugte Bootapplikation dem gewünschten Ausführungsstand (Applikations-Pin) entspricht. Er muss verifizieren, dass beim Download nicht die Erzeugung eines "In Work"-Standes, sondern eines gepinnten Ausführungsstands mit der richtigen Pin-Kennung angekündigt wird (Pinnen, Seite 161).

Änderungen während der Verifikation

Werden während der Verifikation Änderungen an Objekten der Sicherheitsapplikation oder an der Projektstruktur der Sicherheitsapplikation vorgenommen, so müssen alle von den Änderungen betroffenen Safety-Objekte neu gepinnt und anschließend neu verifiziert werden.

Ein Abgleich der Pin-Stände der Applikation muss mit Hilfe des Vergleichseditors der Sicherheitsapplikation (Doppelklick auf das Safety-Applikationsobjekt im Projektbaum) durchgeführt werden.

☞ Es muss mittels der Pin-Kennung verifiziert werden, dass der Strukturvergleich (Vergleichseditor) die gewünschten Applikationsstände vergleicht.

☞ Alle im Vergleichseditor aufgelisteten geänderten Safety-Objekte (ihre aktuelle Prüfsumme entspricht nicht ihrer gepinnten Prüfsumme) müssen neu gepinnt und verifiziert werden.

Zur Einflussanalyse bei Änderungen können auch die Kontrollfluss- und Datenflussanalyse verwendet werden:

- Kontrollflussanalyse, Seite 172 und
- Datenflussanalyse, Seite 174.

Dabei kann man zu jeder Variablen in einer geänderten GVL oder einem geänderten Mapping die Liste der Verwendungsstellen erhalten. Zu einer geänderten POE kann dabei eine Liste der Verwendungsstellen (betroffenen POEs) erzeugt werden.

Im IndraWorks-Standard-Projektvergleich wird der aktuelle Stand des geöffneten Projekts mit einem früheren Projektstand aus einer Projektdatei oder in einer Quellcodeverwaltung verglichen, um die geänderten Objekte und die darin geänderten Stellen zu identifizieren. Für genauere Informationen wird auf die Online-Hilfe von IndraWorks-Standard verwiesen.

☞ Bei Multi-SPS-Projekten muss beim Fortsetzen der Verifikation nach einer Änderung beim Erzeugen der Bootapplikation in einem Bestätigungsdialog die Pin-Kennung und der Gerätename geprüft werden, damit die richtige Bootapplikation auf der richtigen Steuerung erzeugt wird.

Dokumentation der Änderungen siehe Dokumentation von Änderungen, Seite 218.

Änderungen eines bereits abgenommenen Sicherheitsprojekt

Änderungen im Projektnavigator (physikalischen Geräte) des Projektsbaums der Standardapplikation und Änderungen der Standardapplikation haben keinerlei Auswirkung auf die bereits abgenommene Sicherheitsapplikation.

☞ Wenn die Änderung im Projekt die Sicherheitsapplikation inhaltlich ändert, dass sie "In Work" ist, dann müssen folgende Software-Entwicklungsprozesse erneut durchgeführt werden:

1. Erneut pinnen, siehe Kap. 9 "Pinnen" auf Seite 161
2. Vollständig verifizieren, siehe Kap. 10 "Software-Verifikation" auf Seite 167
3. Software erneut abnehmen, siehe Kap. 11 "Software-Abnahme und Dokumentation" auf Seite 183

☞ Bei Multi-SPS-Projekten muss beim Erzeugen der Bootapplikation in einem Bestätigungsdialog die Pin-Kennung und der Gerätename geprüft werden, damit die richtige Bootapplikation auf der richtigen Steuerung erzeugt wird.

☞ Alle im Vergleichseditor des Safety-Applikationsobjekts aufgelisteten, geänderten Safety-Objekte (Prüfsumme des aktuellen Projekts entspricht nicht der Prüfsumme des gepinnten Projekts) müssen neu gepinnt und verifiziert werden.

Zur Einflussanalyse bei Änderungen können auch die Kontrollfluss- und Datenflussanalyse verwendet werden:

- Globale Kontrollflussanalyse, Seite 172 und
- Datenflussanalyse, Seite 174.

Dabei kann der Service-Mitarbeiter zu jeder Variablen in einer geänderten GVL oder einem geänderten Mapping die Liste der Verwendungsstellen erhalten. Zu einer geänderten POE kann dabei eine Liste der Verwendungsstellen (betroffenen POEs) erzeugt werden.

Bei erforderlicher, erneuter Verifikation, bzw. Teilverifikation der Sicherheitsapplikation siehe Software-Verifikation, Allgemeines, Seite 167.

Dokumentation der Änderungen siehe Dokumentation von Änderungen, Seite 218.

Dokumentation von Änderungen

☞ Die für die Einflussanalyse und die Dokumentation von Änderungen qualifizierte Ansicht ist der Vergleichseditor des Safety-Applikationsobjekts.

Die Änderung im Programm können durch Datenfluss und Kontrollfluss Auswirkungen auf unveränderte Teile des Programms haben. Um mögliche bzw. untersuchte Auswirkungen zu dokumentieren, kann die Querverweisliste zu Variablen, die nun anders beschrieben werden, oder FB-Instanzen die nun anders aufgerufen werden, ausgedruckt werden.

Der Ausdruck der Querverweisliste erfolgt durch Aktivierung der Schaltfläche  im Editor der Safety Querverweisliste.

Des Weiteren kann für Dokumentation von Änderungen der Vergleichseditor des Safety Applikationsobjekts ausgedruckt werden.

Der Ausdruck erfolgt in diesem Fall durch Aktivierung des Befehls **Vergleich drucken...** im Vergleichseditor.

14.4.1 Änderungen in Projekten mit Querkommunikation

☞ Bei Änderung des Senders, die sich auf die Variablenwerte der Netzwerkvariablenliste (Sender) auswirkt, muss der Anwender nicht nur Auswirkungen auf die Empfänger im gleichen Projekt sondern die Auswirkungen auf alle Empfänger dieser Netzwerkvariablenliste (Sender) im UDP-Netzwerk aus gegebenenfalls verschiedenen Projekten betrachten.

Wenn die betroffenen Empfänger gar nicht bekannt sind (z.B. bei Entwicklung von Teilmaschinen, welche andere zu einer vollständigen Maschine integrieren, oder von Maschinen, die in einer Maschinenhalle miteinander kommunizieren sollen), wird folgendes Vorgehen von IndraWorks SafetyManager unterstützt:

Der Anwender ändert die Objektversion der Sender-Netzwerkvariablenliste (im Eigenschaftendialog, Registerkarte "Safety"). Damit erreicht er, dass kein alter, unveränderter Empfänger im Netzwerk die neuen Werte empfangen kann.

Der Anwender erzwingt damit, dass der Integrator der Gesamtmaschine bzw. der Betreiber der Maschinenhalle alle Empfänger im Netzwerk neu erzeugen muss, wodurch er zum Analysieren der Auswirkung der Senderänderung auf den Empfänger gebracht wird. Eine Auswirkungsanalyse der semantischen Änderung der Variablen auf den Empfänger mag dann immer noch ergeben, dass keine funktionale Anpassung des Empfängers erforderlich ist und nur der Safety NVL-Empfänger aktualisiert werden muss, um die geänderte Objektversion des Senders zu übernehmen (was dann als Dokumentation der Durchführung der Analyse verstanden werden kann).

☞ Eine modulare Erweiterung einer Maschine um eine Steuerung mit eigenen für andere bestehende Steuerungen relevanten Signale (zum Beispiel ein weiteres Nothalt-Gerät), die von bestehenden Safety NVL-Empfängern mit berücksichtigt werden sollen, wird nicht unterstützt!

☞ **Änderung der Empfänger - Auswirkung auf Sender**

Wenn ein neuer Safety NVL-Empfänger hinzugefügt, oder ein bestehender Safety NVL-Empfänger beendet wird, kann dies einen zeitlichen Einfluss auf die Hauptsteuerung des Empfängers, die Hauptsteuerung des Senders, und den Sender der Variablen haben. Wenn durch das Hinzufügen von Safety NVL-Empfängern die Zykluszeit des Senders überschritten wird, könnte eine manuelle Anpassung des Senders mit anschließendem Download nötig werden.

☞ Bei Änderung von Applikation oder Signallängen im Prozess muss die Änderungsanalyse auch Auswirkungen in Form eines möglichen Undersamplings analysieren.

Projektänderung und Auswirkungsanalyse

☞ Wenn sich durch Änderungen in der Sendersteuerung oder an ihren Eingangsgeräten die Eigenschaften der gesendeten Werte ändern (andere Wertebereiche, andere Wertverläufe, andere zeitliche Eigenschaften der Werte, andere Bedeutung in der Maschine), wirkt sich das auf die empfangenen Werte in den Empfängersteuerungen aus. Das muss bei der Auswirkungsanalyse der Änderung berücksichtigt werden.

Änderungen in oder an der Empfängersteuerung oder Hinzufügen von Empfängersteuerungen wirken sich nicht funktional auf die Sendersteuerung aus; es besteht also keine funktionale Abhängigkeit des Senders von seinen Empfängern; Ausnahme: Die Zykluszeit kann überschritten werden.

Änderungen in oder an einer Empfängersteuerung oder Hinzufügen von Empfängersteuerungen können höchstens zu Kommunikationsunterbrechungen führen (z.B. bei Überschreitung verfügbarer Ressourcen oder bei doppelter Adressvergabe). Ansonsten wirken sich diese Änderungen nicht funktional auf die anderen Empfängersteuerungen aus.

15 Außerbetriebnahme der Steuerung - SPS-Ausbau

Um eine Sicherheitssteuerung außer Betrieb zu nehmen, muss Klarheit darüber bestehen, ob sich die Bootapplikation auf der Standardsteuerung (z. B. SD-Karte) oder auf der Sicherheitssteuerung (Safety-Klemme) befindet.

Urlöschen

⚠ VORSICHT

Bevor eine Sicherheitssteuerung außer Betrieb genommen und aus der Anlage ausgebaut wird, muss die sich auf der Sicherheitssteuerung befindende Bootapplikation "urgelöscht" werden.

Dadurch wird verhindert, dass es bei einem Wiederverwenden der Sicherheitssteuerung auf einer anderen Anlage zum unbeabsichtigten Starten der "alten" Bootapplikation kommt.

⚠ VORSICHT

Ein unvollständiges Ausführung des Befehls Urlöschen kann dazu führen, dass die Bootapplikation auf der Steuerung bleibt und beim nächsten Starten der Steuerung wieder aktiv wird.

Bei Ausführen des Befehls **Urlöschen** wird die laufende Applikation entladen, die Bootapplikation gelöscht und das Admin-Kennwort zurückgesetzt (bzw. das systemspezifische Default-Passwort reaktiviert). Die Logbücher werden dabei nicht gelöscht. Zur Ausführung des Befehls ist keine Autorisierung durch das Admin-Passwort erforderlich.

Ist der Service-Mitarbeiter aktuell auf der Applikation der Geräteinstanz eingeloggt, wird der Befehl nur durch den Service-Mitarbeiter nach bestätigter Rückfrage ausgeführt und die Online-Verbindung beendet.

Der Service-Mitarbeiter erhält immer eine Rückmeldung, ob der Befehl erfolgreich ausgeführt wurde oder nicht. Diese Rückmeldung muss durch den Service-Mitarbeiter bestätigt werden.

Verfahren "Urlöschen"

1. Befehl **Öffnen** des Kontextmenüs der Sicherheitssteuerung im Projektexplorer aktivieren.
2. Registerkarte **Safety Online Information** öffnen.
3. Mit Klick auf die Schaltfläche **Urlöschen** den Befehl aktivieren.

Steuerung mit eingebauter Netzwerkadresse

Wird eine Steuerung mit eingebauter Netzwerkadresse nach der Zweit-Verifikation ausgebaut und an anderer Stelle wieder eingebaut, ist eine Gefährdung möglich, wenn sich der Service-Mitarbeiter auf der organisatorisch nicht abgesicherten "neuen" Anlage oder dem Anlagenteil ohne Schutz durch Verbindungsverifikation einloggt, debuggt oder die Bootapplikation aktualisiert.

⚠ VORSICHT

Um eine Gefährdung zu vermeiden muss der Verbindungsschlüssel auf der Steuerung ungültig gemacht werden.

Hierfür muss der Service-Mitarbeiter direkt vor oder nach dem Ausbau

- die Steuerung urlöschen oder
- sich einmal mit einem neuen Projekt auf die Steuerung einloggen.

Löschen des Admin-Passworts



Bei verlorengegangenem Admin-Passwort ermöglicht der Befehl **Urlöschen** das Löschen des alten Admin-Passwort.

Anschließend muss erst die Bootapplikation neu erzeugt werden, um dann ein neues Admin-Passwort vergeben zu können.

16 Sicherheitsbibliotheken

Im IndraWorks Safety Manager stehen dem Entwickler Bibliotheken mit diversen Bausteinen zur Verfügung.

Welche Bibliotheken zur Verfügung stehen, hängt von der eingesetzten Sicherheitssteuerung ab. Sie finden die für Ihre Steuerung verfügbaren Bibliotheken im Editorfenster der Bibliotheksverwaltung der Sicherheitsapplikation.

16.1 Genauigkeit von Zeitmessungen

Die Genauigkeit von Zeitmessungen hängt von drei Faktoren ab:

1. **Safety-Zykluszeit in Relation zur einzuhaltenden Zeit**

   Die Berechnung der Ausgänge erfolgt im Zyklus der Safety-Applikation und kann maximal einen Safety-Zyklus abweichen.

   Ist z.B. eine Zeit von 15 ms eingestellt und die des Safety-Zyklus beträgt jedoch 10 ms, so wird ein zeitgesteuerter Ausgang nicht nach 15 ms, sondern erst beim nächsten durch die Safety-Zykluszeit angestoßenen Programmdurchlauf nach 20 ms schalten.

2. **Jitter der Safety-Task**

   Der Start der Safety-Task ist mit einer Ungenauigkeit von ±10 µs behaftet. Bei einer Safety-Zykluszeit von 10 ms entspricht dies einer Abweichung von 1 ‰ und ist somit vernachlässigbar klein.

3. **Jitter aufgrund der Laufzeit des Safety-Programms**

   Abhängig von Funktionsbaustein- und Programmzuständen kann die Laufzeit des Safety-Programms variieren. Die Laufzeitschwankungen werden umso größer, je weiter hinten im Programm der Funktionsbaustein aufgerufen wird. Bei größeren Programmen mit einer Laufzeit von über 10 ms kann der hierdurch hervorgerufene Jitter durchaus 1 ms betragen.

16.2 SafetyStandard.library

Diese Bibliothek enthält die meisten Bausteine der Bibliothek Standard.library in einer für Sicherheitsapplikationen angepassten Form.

Anwendungsgebiete und Funktionsbausteine von SafetyStandard

Anwendungsgebiet (Ordner in der Bibliothek)	
Funktionsbausteine	**Beschreibung**
Bistable Function Blocks (Bistabile Funktionsbausteine)	
SF_RS auf Seite 230	dominantes Rücksetzen eines bistabilen Triggers
SF_SR auf Seite 231	dominantes Setzen eines bistabilen Triggers
Counter (Zähler)	
SF_CTD auf Seite 231	Rückwärtszähler
SF_CTUD auf Seite 232	Vorwärts- und Rückwärtszähler
SF_CTU auf Seite 234	Vorwärtszähler
Timer (Zeitgeber)	
SF_TOF auf Seite 235	Timer mit Ausschaltverzögerung

SF_TON auf Seite 236	Timer mit Einschaltverzögerung
SF_TP auf Seite 237	Timer, der einen Pulsgeber realisiert
Trigger (Flankensteuerung)	
SF_F_TRIG auf Seite 238	detektiert eine fallende Flanke
SF_R_TRIG auf Seite 239	detektiert eine steigende Flanke

Tab. 16-1: SafetyStandard-Bibliothek

16.2.1 SF_RS

Kurzbeschreibung

SF_RS ist ein Baustein der Bibliothek SafetyStandard. Er ist eine Erweiterung des IEC 61131-3 Standardbausteins RS.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

SF_RS ist ein bistabiler Baustein der Standardbibliothek SafetyStandard. Seine Funktion ist das Zurücksetzen eines bistabilen Funktionsbausteins.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-2: Bibliothekszuordnung

Schnittstellenbeschreibung

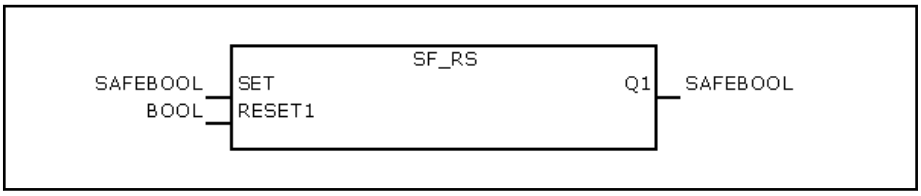


Abb. 16-1: Funktionsbaustein SF_RS

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	SET1	SAFEBOOL	Eingang, um Q1 zu setzen
	RESET	BOOL	Eingang, um Q1 zurückzusetzen, Reset dominat
VAR_OUTPUT	Q1	SAFEBOOL	Q1 = RS (SET, RESET1) bedeutet: Q1 = NOT RESET1 AND (Q1 OR SET)

Tab. 16-3: Schnittstelle SF_RS

Name	Typ	Default-Wert	Übernahme
SET1	SAFEBOOL	FALSE	Eingang, um Q1 zu setzen,
RESET	BOOL	FALSE	Eingang, um Q1 zurückzusetzen, Reset dominat

Tab. 16-4: Eingangsverhalten SF_RS

16.2.2 SF_SR

Kurzbeschreibung

SF_SR ist ein Baustein der Bibliothek SafetyStandard. Er ist eine Erweiterung des IEC 61131-3 Standardbausteins SR.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

SF_SR ist ein bistabiler Baustein der Standardbibliothek SafetyStandard. Er setzt bistabile Funktionsbausteine dominant.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-5: Bibliothekszuordnung

Schnittstellenbeschreibung

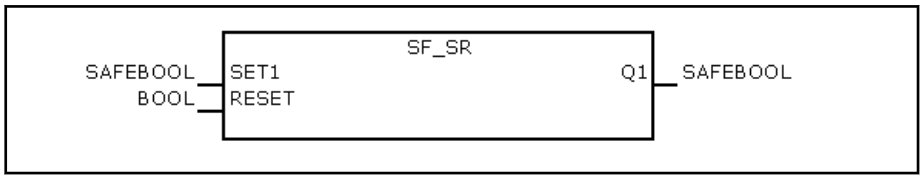


Abb. 16-2: Funktionsbaustein SF_SR

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	SET1	SAFEBOOL	Eingang, um Q1 zu setzen, Setzen dominant
	RESET	BOOL	Eingang, um Q1 zurückzusetzen
VAR_OUTPUT	Q1	SAFEBOOL	Q1 = SR (SET1, RESET) bedeutet: Q1 = (NOT RESET AND Q1) OR SET1

Tab. 16-6: Schnittstelle SF_SR

Name	Typ	Default-Wert	Übernahme
SET1	SAFEBOOL	FALSE	Eingang, um Q1 zu setzen, Setzen dominant
RESET	BOOL	FALSE	Eingang, um Q1 zurückzusetzen

Tab. 16-7: Eingangsverhalten SF_SR

16.2.3 SF_CTD

Kurzbeschreibung

SF_CTD ist ein Zähler(Counter)-Baustein der Bibliothek SafetyStandard. Dabei handelt es sich um eine Erweiterung des IEC 61131-3 Standardbausteins CTD.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

SF_CTD führt die Funktion des Rückwärtszählens aus.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-8: Bibliothekszuordnung

Schnittstellenbeschreibung

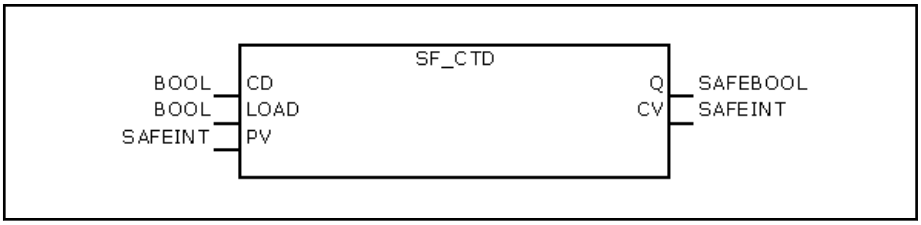


Abb. 16-3: Funktionsbaustein SF_CTD

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	CD	BOOL	Steigende Flanke an CD zählt CV um 1 zurück.
	LOAD	BOOL	TRUE: Setzen von CV auf den Startwert PV FALSE: Rückwärtszählen ist freigegeben.
	PV	SAFEINT	Startwert (Obergrenze) für das Rückwärtszählen von CV.
VAR_OUTPUT	Q	SAFEBOOL	Wird TRUE, wenn CV 0 erreicht.
	CV	SAFEINT	Momentaner Zählerwert des Zählers CV. Vom Startwert PV wird schrittweise um 1 dekrementiert, bis 0 erreicht ist.

Tab. 16-9: Schnittstelle SF_CTUD

Name	Typ	Default-Wert	Übernahme
CD	BOOL	FALSE	Flanke 0-1
LOAD	BOOL	FALSE	TRUE: Setzen von CV auf den Startwert PV FALSE: Rückwärtszählen ist freigegeben.
PV	SAFEINT	0	LOAD = TRUE

Tab. 16-10: Eingangsverhalten SF_CTUD

Baustein-Code

```
IF LOAD THEN CV := PV;
ELSIF CD AND (CV > PVmin)
    THEN CV := CV-1;
END_IF ;
Q := (CV <= 0) ;
```

Der numerische Wert von PVmin ist abhängig von der Implementierung.

16.2.4 SF_CTUD

Kurzbeschreibung

☞ SF_CTUD ist ein Zähler(Counter)-Baustein der Bibliothek SafetyStandard. Dabei handelt es sich um eine Erweiterung des IEC 61131-3 Standardbausteins CTUD.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

SF_CTUD erfüllt die Funktion eines Vorwärts- und Rückwärtszählers.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.1.1.0

Tab. 16-11: Bibliothekszuordnung

Schnittstellenbeschreibung

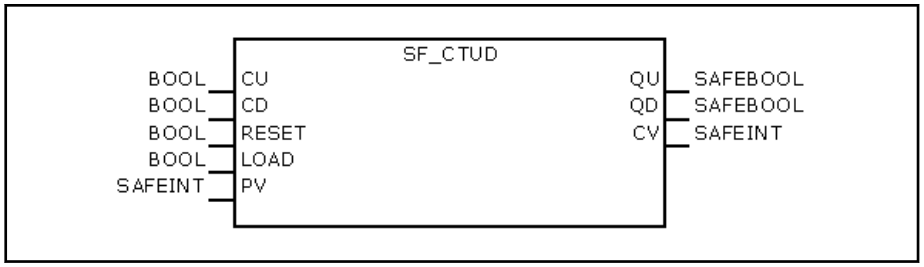


Abb. 16-4: Funktionsbaustein SF_CTUD

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	CU	BOOL	Steigende Flanke an CU zählt CV um 1 vorwärts.
	CD	BOOL	Steigende Flanke an CD zählt CV um 1 zurück.
	LOAD	BOOL	TRUE: Setzen von CV auf den Startwert PV FALSE: Rückwärtszählen ist freigegeben.
	PV	SAFEINT	Startwert (Rückwärtszählen) /oder Obergrenze für das Vorwärtszählen von CV
VAR_OUTPUT	QU	SAFEBOOL	TRUE: Zähler CV erreichte die Obergrenze
	QD	SAFEBOOL	Wird TRUE, wenn CV 0 erreicht.
	CV	SAFEINT	Momentaner Zählerwert des Zählers CV.

Tab. 16-12: Schnittstelle SF_CTUD

Name	Typ	Default-Wert	Übernahme
CU	BOOL	FALSE	Flanke 0-1
CD	BOOL	FALSE	Flanke 0-1
LOAD	BOOL	FALSE	TRUE: Setzen von CV auf den Startwert PV FALSE: Rückwärtszählen ist freigegeben.
PV	SAFEINT	0	LOAD = TRUE

Tab. 16-13: Eingangsverhalten SF_CTUD

Baustein-Code

```
IF R THEN CV := 0 ;
ESLIF LOAD THEN CV := PV ;
ELSE
        IF NOT (CU AND CD) THEN
                IF CU AND (CV < PVmax)
                THEN CV := CV+1;
                ELSIF CD AND (CV > PVmin)
                THEN CV := CV-1;
                END_IF;
        ENDIF;
ENDIF;
QU := (CV >= PV) ;
QD := (CV <= 0);
```

Die numerischen Werte von PVmin und PVmax sind abhängig von der Implementierung.

16.2.5 SF_CTU

Kurzbeschreibung

SF_CTU ist ein Zähler(Counter)-Baustein der Bibliothek SafetyStandard. Dabei handelt es sich um eine Erweiterung des IEC 61131-3 Standardbausteins CTU.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

Seine Funktion ist das Vorwärtszählen eines Zählers bis zu einer definierten Obergrenze.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-14: Bibliothekszuordnung

Schnittstellenbeschreibung

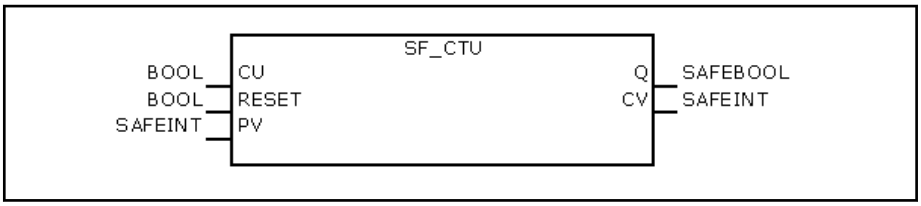


Abb. 16-5: Funktionsbaustein SF_CTU

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	CU	BOOL	Steigende Flanke an CU zählt CV um 1 vorwärts.
	RESET	BOOL	TRUE: Rücksetzen des Zählers (CV) auf 0.
	PV	SAFEINT	Obergrenze für das Hochzählen von CV.
VAR_OUTPUT	Q	SAFEBOOL	TRUE: Zähler CV erreichte die Obergrenze
	CV	SAFEINT	Momentaner Zählerwert. Wird hochgezählt, bis PV erreicht ist.

Tab. 16-15: Schnittstelle SF_CTU

Name	Typ	Default-Wert	Übernahme
CU	BOOL	FALSE	Flanke 0-1
RESET	BOOL	FALSE	TRUE: Rücksetzen des Zählers (CV) auf 0.
PV	SAFEINT	0	RESET = TRUE

Tab. 16-16: Eingangsverhalten SF_CTU

Baustein-Code

```
IF R THEN CV := 0 ;
ELSIF CU AND (CV < PVmax)
    THEN CV := CV+1;
END_IF ;
Q := (CV >= PV) ;
```

Der numerische Wert von PVmax ist abhängig von der Implementierung.

16.2.6 SF_TOF

Kurzbeschreibung

SF_TOF ist ein Timer-Baustein der Bibliothek SafetyStandard. Dabei handelt es sich um eine Erweiterung des IEC 61131-3 Standardbausteins TOF.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

SF_TOF ist ein Timer-Baustein der Standardbibliothek SafetyStandard, der eine Ausschaltverzögerung realisiert.

Wechselt der Eingang IN von TRUE auf FALSE, so wechselt der Ausgang erst nach Ablauf der Verzögerungszeit ebenfalls auf FALSE.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-17: Bibliothekszuordnung

Schnittstellenbeschreibung

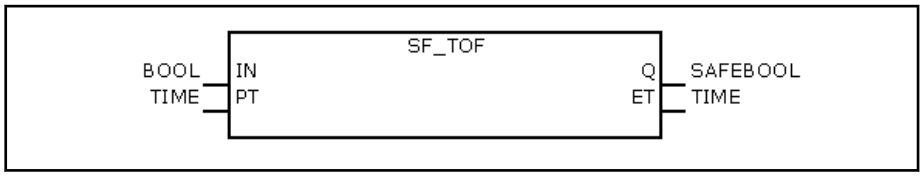


Abb. 16-6: Funktionsbaustein SF_TOF

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	BOOL	Fallende Flanke (TRUE -> FALSE): Zeitverzögerung startet. Steigende Flanke (FALSE -> TRUE): Zurücksetzen der Zeitverzögerung.
	PT	TIME	Zeitverzögerung in Millisekunden (ms)(Obergrenze für das Hochzählen von ET)
VAR_OUTPUT	Q	SAFEBOOL	Fallende Flanke, sobald ET die Obergrenze der Zeitverzögerung (PT) erreicht hat. FALSE wenn IN = FALSE und ET = PT
	ET	TIME	Aktueller Stand der Verzögerungszeit. Beginnt hochzuzählen, sobald IN = FALSE und stoppt wenn ET = PT.

Tab. 16-18: Schnittstelle SF_TOF

Name	Typ	Default-Wert	Übernahme
IN	BOOL	FALSE	Fallende Flanke (TRUE -> FALSE): Zeitverzögerung startet. Steigende Flanke (FALSE -> TRUE): Zurücksetzen der Zeitverzögerung.
PT	TIME	TIME#0ms	Flanke an IN

Tab. 16-19: Eingangsverhalten SF_TOF

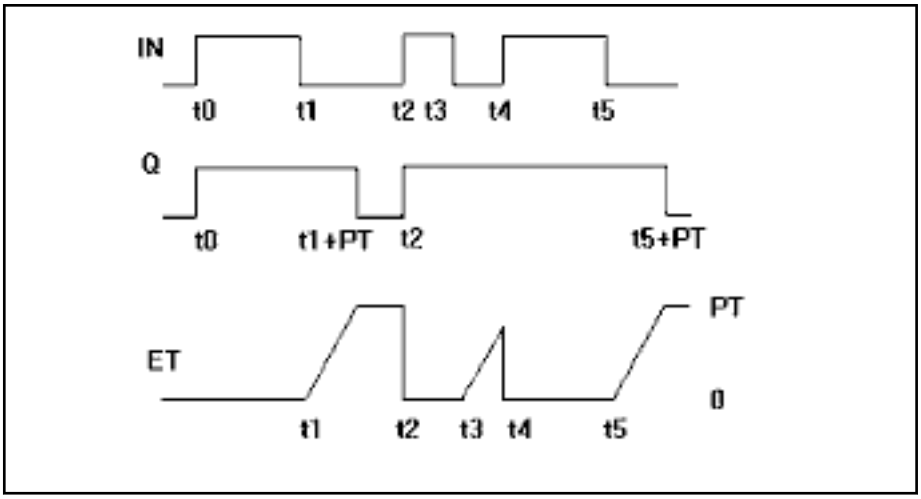


Abb. 16-7: Zeitdiagramm SF_TOF

16.2.7 SF_TON

Kurzbeschreibung

SF_TON ist ein Timer-Baustein der Bibliothek SafetyStandard. Dabei handelt es sich um eine Erweiterung des IEC 61131-3 Standardbausteins TON.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

SF_TON ist ein Timer-Baustein der Standardbibliothek SafetyStandard, der eine Einschaltverzögerung realisiert.

Wechselt der Eingang IN von FALSE auf TRUE, so wechselt der Ausgang erst nach Ablauf der Verzögerungszeit ebenfalls auf TRUE.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-20: Bibliothekszuordnung

Schnittstellenbeschreibung

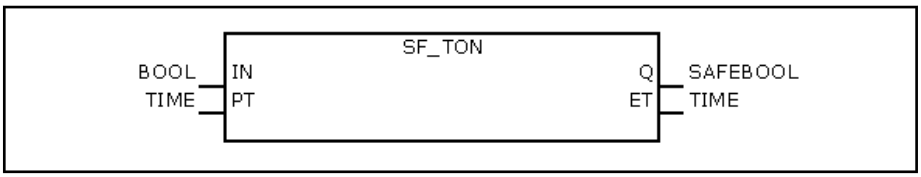


Abb. 16-8: Funktionsbaustein SF_TON

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	BOOL	Steigende Flanke (FALSE -> TRUE): Zeitverzögerung startet. Fallende Flanke (TRUE -> FALSE): Zurücksetzen der Zeitverzögerung.
	PT	TIME	Zeitverzögerung in Millisekunden (ms)(Obergrenze für das Hochzählen von ET)

E/A-Typ	Name	Datentyp	Beschreibung
VAR_OUTPUT	Q	SAFEBOOL	Steigende Flanke, sobald ET die Obergrenze der Zeitverzögerung (PT) erreicht hat. TRUE wenn IN = TRUE und ET = PT
	ET	TIME	Aktueller Stand der Verzögerungszeit. Beginnt hochzuzählen, sobald IN = TRUE und stoppt wenn ET = PT.

Tab. 16-21: Schnittstelle SF_TON

Name	Typ	Default-Wert	Übernahme
IN	BOOL	FALSE	Steigende Flanke (FALSE -> TRUE): Zeitverzögerung startet. Fallende Flanke (TRUE -> FALSE): Zurücksetzen der Zeitverzögerung.
PT	TIME	TIME#0ms	Flanke an IN

Tab. 16-22: Eingangsverhalten SF_TON

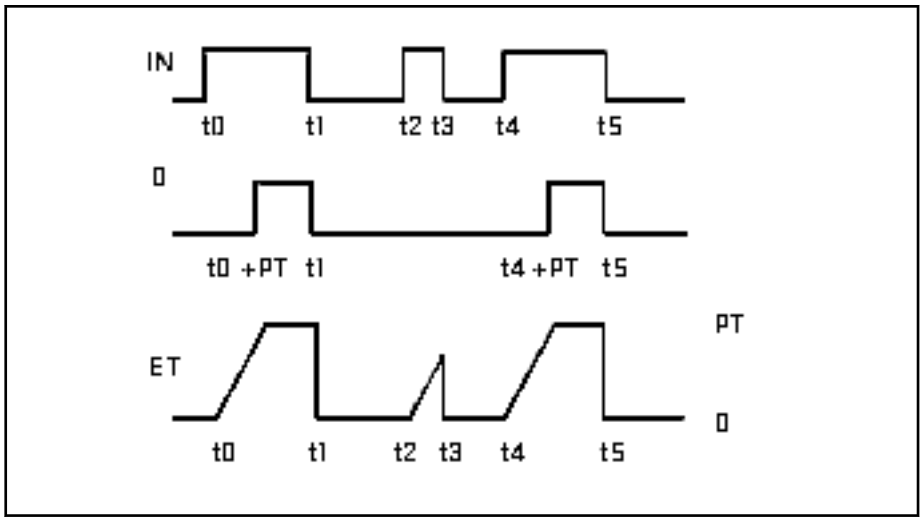


Abb. 16-9: Zeitdiagramm SF_TON

16.2.8 SF_TP

Kurzbeschreibung

SF_TP ist ein Timer-Baustein der Bibliothek SafetyStandard. Dabei handelt es sich um eine Erweiterung des IEC 61131-3 Standardbausteins TP.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

SF_TP ist ein Timer-Baustein, der als Pulsgeber arbeitet. Ein Timer(ET) wird solange hochgezählt, bis die Obergrenze(PT) erreicht ist. Während des Hochzählens ist die Puls-Variable (Q) TRUE.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-23: Bibliothekszuordnung

Schnittstellenbeschreibung

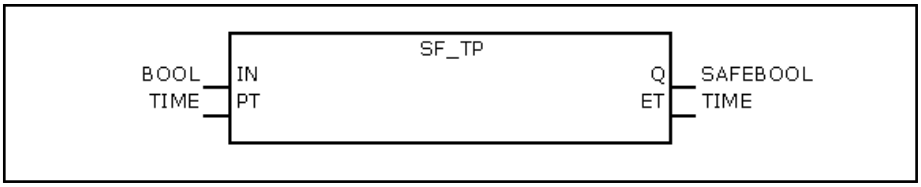


Abb. 16-10: Funktionsbaustein SF_TP

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	BOOL	Steigende Flanke an IN startet den Pulsgeber.
	PT	TIME	Zeitdauer des Pulsgebens in Millisekunden,
VAR_OUTPUT	Q	SAFEBOOL	Puls-Variable TRUE während des Hochzählens, d.h. solange ET kleiner oder gleich PT ist, ansonsten FALSE
	ET	TIME	Abgelaufene Zeit seit Starten des Pulsgebers (Q). Wenn ET = PT, bleibt der Wert konstant

Tab. 16-24: Schnittstelle SF_TP

Name	Typ	Default-Wert	Übernahme
IN	BOOL	FALSE	Flanke 0-1
PT	TIME	TIME#0ms	Flanke an IN

Tab. 16-25: Eingangsverhalten SF_TP

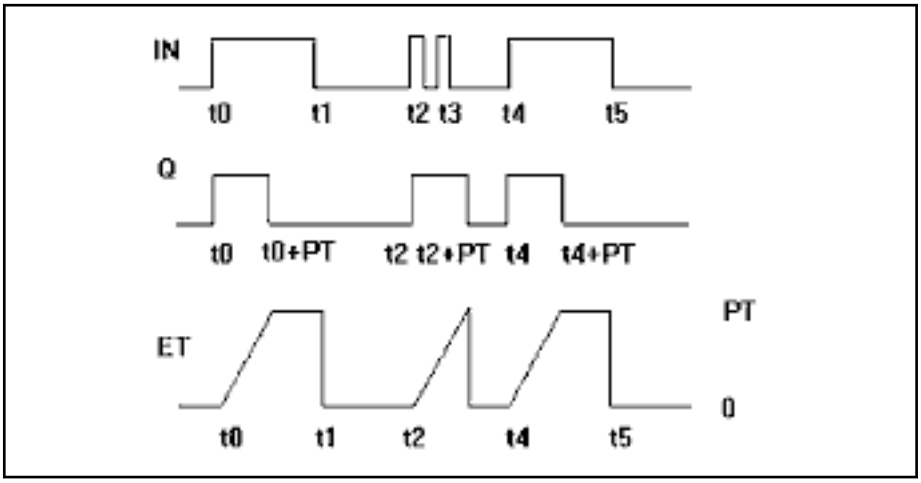


Abb. 16-11: Zeitdiagramm SF_TP

16.2.9 SF_F_TRIG

Kurzbeschreibung

☞ SF_F_TRIG ist ein Baustein der Bibliothek SafetyStandard. Dabei handelt es sich um eine Erweiterung des IEC 61131-3 Standardbausteins F_TRIG.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

Dieser Baustein detektiert eine fallende Flanke.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-26: Bibliothekszuordnung

Schnittstellenbeschreibung

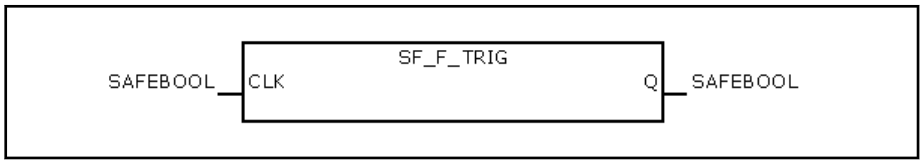


Abb. 16-12: Funktionsbaustein SF_F_TRIG

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	CLK	SAFEBOOL	Eingangssignal, das auf fallende Flanke überprüft wird.
VAR_OUTPUT	Q	SAFEBOOL	TRUE: fallende Flanke an CLK wurde erkannt.

Tab. 16-27: Schnittstelle SF_F_TRIG

Name	Typ	Default-Wert	Übernahme
CLK	SAFEBOOL	FALSE	Flanke 1-0

Tab. 16-28: Eingangsverhalten SF_F_TRIG

Der Ausgang Q ist nur dann TRUE, wenn CLK FALSE ist und beim vorherigen Aufruf des SF_ F_TRIG TRUE war (fallende Flanke). In den anderen Fällen liefert Q den Wert FALSE.

☞ **Besonderes Verhalten des Bausteins im 1. Zyklus**

Im ersten Zyklus erkennt SF_F_TRIG eine fallende Flanke bei anliegendem FALSE und setzt den Ausgang Q auf TRUE.

Dieses Verhalten ist konform zum F_TRIG von IEC 61131-3 ab edition2 .

Es weicht somit vom Verhalten des Bausteins F_TRIG von Standard-IndraLogic 2G ab, welcher konform zu IEC 61131-3 edition1 ist.

16.2.10 SF_R_TRIG

Kurzbeschreibung

☞ SF_R_TRIG ist ein Baustein der Bibliothek SafetyStandard. Dabei handelt es sich um eine Erweiterung des IEC 61131-3 Standardbausteins R_TRIG.

Der Baustein enthält sowohl Standarddatentypen, als auch sicherheitsbezogene Datentypen (Präfix: SAFE).

Dieser Baustein detektiert eine steigende Flanke.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyStandard	1.0.0.0

Tab. 16-29: Bibliothekszuordnung

Schnittstellenbeschreibung

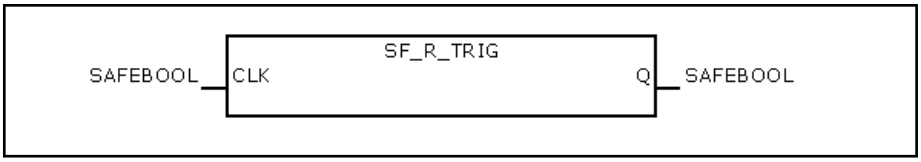


Abb. 16-13: Funktionsbaustein SF_R_TRIG

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	CLK	SAFEBOOL	Eingangssignal, das auf steigende Flanke überprüft wird.
VAR_OUTPUT	Q	SAFEBOOL	TRUE: steigende Flanke an CLK wurde erkannt.

Tab. 16-30: Schnittstelle SF_R_TRIG

Name	Typ	Default-Wert	Übernahme
CLK	SAFEBOOL	FALSE	Flanke 0-1

Tab. 16-31: Eingangsverhalten SF_R_TRIG

Der Ausgang Q ist nur dann TRUE, wenn CLK TRUE ist und beim vorherigen Aufruf des Bausteins FALSE war.

In allen anderen Fällen liefert Q FALSE.

16.3 SafetyUtilities.library

Diese Bibliothek enthält Funktionsbausteine für die Umsetzung wichtiger Safety-Hilfsfunktionen.

Anwendungsgebiete und Funktionsbausteine von SafetyUtilities

Anwendungsgebiet (Ordner in der Bibliothek)	
Funktionsbausteine	**Beschreibung**
Bit Access (Bitzugriffe)	
BoolToByte, Seite 241	fasst 8 BOOL-Eingänge zu einem BYTE zusammen
BoolToWord, Seite 242	fasst 16 BOOL-Eingänge zu einem WORD zusammen
BoolToDWord, Seite 244	fasst 32 BOOL-Eingänge zu einem DWORD zusammen
SF_BoolToByte, Seite 247	fasst 8 SAFEBOOL-Eingänge zu einem SAFEBYTE zusammen
SF_BoolToWord, Seite 249	fasst 16 SAFEBOOL-Eingänge zu einem SAFEWORD zusammen
SF_BoolToDWord, Seite 250	fasst 32 SAFEBOOL-Eingänge zu einem SAFEDWORD zusammen
ByteToBool, Seite 253	zerlegt ein BYTE zu 8 BOOL-Ausgängen
WordToBool, Seite 254	zerlegt ein WORD zu 16 BOOL-Ausgängen
DWordToBool, Seite 256	zerlegt ein DWORD zu 32 BOOL-Ausgängen
SF_ByteToBool, Seite 258	zerlegt ein SAFEBYTE zu 8 SAFEBOOL-Ausgängen
SF_WordToBool, Seite 259	zerlegt ein SAFEWORD zu 16 SAFEBOOL-Ausgängen
SF_DWordToBool, Seite 260	zerlegt ein SAFEDWORD zu 32 SAFEBOOL-Ausgängen
IOConfig	

SF_IOConfig, Seite 262	ermöglicht die Deaktivierung von sicheren Teilnehmern über einen Einlernprozess.
SF_IOConfigCntl, Seite 266	ermöglicht die Deaktivierung von sicheren Teilnehmern über einen Einlernprozess.
StartupWarning (Anlaufwarnung)	
SF_StartupWarning, Seite 268	realisiert eine Anlaufwarnung für unübersichtliche Maschinen mit mehreren Bedienstationen.
Timer (Taktgenerator)	
SF_ClockGenerator, Seite 280	stellt einen in Periodenlänge und Tastung einstellbaren Implulsgenerator bereit.
Transducer (Decoder / Encoder)	
BinaryDecoder_4_16, Seite 282	implementiert einen kaskadierbaren 16 auf 4 Bit Decoder für BOOL-Werte.
BinaryEncoder_16_4, Seite 285	implementiert einen kaskadierbaren 16 auf 4 Bit Encoder für BOOL-Werte.
SF_BinaryDecoder_4_16, Seite 287	implementiert einen kaskadierbaren 4 auf 16 Bit Binär-Dekoder für SAFEBOOL-Werte.
SF_BinaryEncoder_16_4, Seite 290	implementiert einen kaskadierbaren 16 auf 4 Bit Binär-Kodierer für SAFEBOOL-Werte.

Tab. 16-32: SafetyUtilities-Bibliothek

16.3.1 BoolToByte

Kurzbeschreibung

Der Funktionsbaustein BoolToByte fasst 8 BOOL-Eingänge zu einem BYTE zusammen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-33: Bibliothekszuordnung

Schnittstellenbeschreibung

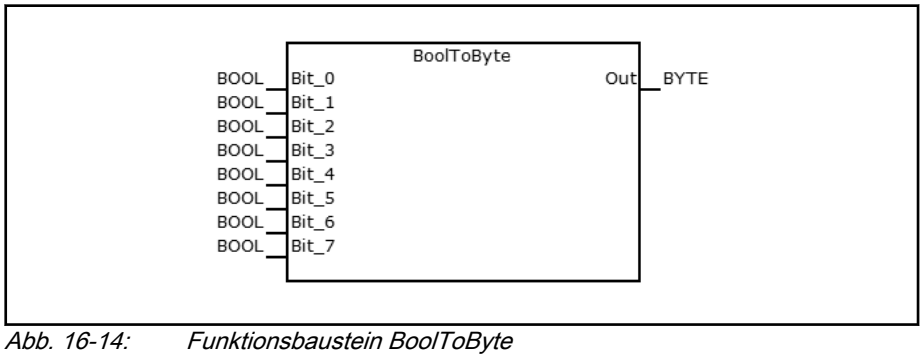


Abb. 16-14: Funktionsbaustein BoolToByte

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Bit_0	BOOL	Eingang Bit 0
	Bit_1	BOOL	Eingang Bit 1
	Bit_2	BOOL	Eingang Bit 2
	Bit_3	BOOL	Eingang Bit 3
	Bit_4	BOOL	Eingang Bit 4

E/A-Typ	Name	Datentyp	Beschreibung
	Bit_5	BOOL	Eingang Bit 5
	Bit_6	BOOL	Eingang Bit 6
	Bit_7	BOOL	Eingang Bit 7
VAR_OUTPUT	Out	BYTE	BYTE aus 8 zusammengefügten Bits

Tab. 16-34: Schnittstelle BoolToByte

Name	Typ	Default-Wert	Übernahme
Bit_0	BOOL	FALSE	kontinuierlich
Bit_1	BOOL	FALSE	kontinuierlich
Bit_2	BOOL	FALSE	kontinuierlich
Bit_3	BOOL	FALSE	kontinuierlich
Bit_4	BOOL	FALSE	kontinuierlich
Bit_5	BOOL	FALSE	kontinuierlich
Bit_6	BOOL	FALSE	kontinuierlich
Bit_7	BOOL	FALSE	kontinuierlich

Tab. 16-35: Eingangsverhalten BoolToByte

Funktionsbeschreibung

Der Funktionsbaustein BoolToByte fasst 8 BOOL-Eingänge zu einem BYTE zusammen.

Bit_0 bis Bit_7

Die Eingänge Bit_0 bis Bit_7 werden ihrer Nummer nach aufsteigend in die Bits des Bytes einsortiert. Bit_0 ist dabei das niederwertigste Bit und Bit_7 das höchstwertige Bit.

Out

Der Ausgang Out enthält das aus den Eingängen Bit_0 bis Bit_7 zusammengesetzte Byte.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.2 BoolToWord

Kurzbeschreibung

Der Funktionsbaustein BoolToWord fasst 16 BOOL-Eingänge zu einem WORD zusammen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.4.0.0

Tab. 16-36: Bibliothekszuordnung

Schnittstellenbeschreibung

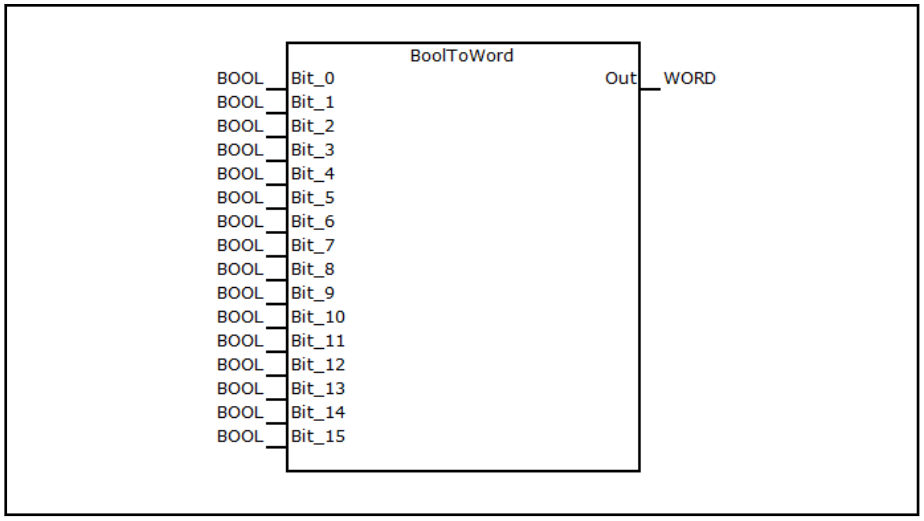


Abb. 16-15: Funktionsbaustein BoolToWord

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Bit_0	BOOL	Eingang Bit 0
	Bit_1	BOOL	Eingang Bit 1
	Bit_2	BOOL	Eingang Bit 2
	Bit_3	BOOL	Eingang Bit 3
	Bit_4	BOOL	Eingang Bit 4
	Bit_5	BOOL	Eingang Bit 5
	Bit_6	BOOL	Eingang Bit 6
	Bit_7	BOOL	Eingang Bit 7
	Bit_8	BOOL	Eingang Bit 8
	Bit_9	BOOL	Eingang Bit 9
	Bit_10	BOOL	Eingang Bit 10
	Bit_11	BOOL	Eingang Bit 11
	Bit_12	BOOL	Eingang Bit 12
	Bit_13	BOOL	Eingang Bit 13
	Bit_14	BOOL	Eingang Bit 14
	Bit_15	BOOL	Eingang Bit 15
VAR_OUTPUT	Out	WORD	WORD aus 16 zusammengefügten Bits

Tab. 16-37: Schnittstelle BoolToWord

Name	Typ	Default-Wert	Übernahme
Bit_0	BOOL	FALSE	kontinuierlich
Bit_1	BOOL	FALSE	kontinuierlich
Bit_2	BOOL	FALSE	kontinuierlich
Bit_3	BOOL	FALSE	kontinuierlich
Bit_4	BOOL	FALSE	kontinuierlich

Name	Typ	Default-Wert	Übernahme
Bit_5	BOOL	FALSE	kontinuierlich
Bit_6	BOOL	FALSE	kontinuierlich
Bit_7	BOOL	FALSE	kontinuierlich
Bit_8	BOOL	FALSE	kontinuierlich
Bit_9	BOOL	FALSE	kontinuierlich
Bit_10	BOOL	FALSE	kontinuierlich
Bit_11	BOOL	FALSE	kontinuierlich
Bit_12	BOOL	FALSE	kontinuierlich
Bit_13	BOOL	FALSE	kontinuierlich
Bit_14	BOOL	FALSE	kontinuierlich
Bit_15	BOOL	FALSE	kontinuierlich

Tab. 16-38: Eingangsverhalten BoolToWord

Funktionsbeschreibung

Der Funktionsbaustein BoolToWord fasst 16 BOOL-Eingänge zu einem WORD zusammen.

Bit_0 bis Bit_15

Die Eingänge Bit_0 bis Bit_15 werden ihrer Nummer nach aufsteigend in die Bits des Wortes einsortiert. Bit_0 ist dabei das niederwertigste Bit und Bit_15 das höchstwertige Bit.

Out

Der Ausgang Out enthält das aus den Eingängen Bit_0 bis Bit_15 zusammengesetzte Wort.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.3 BoolToDWord

Kurzbeschreibung

Der Funktionsbaustein BoolToWord fasst 32 BOOL-Eingänge zu einem DWORD zusammen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-39: Bibliothekszuordnung

Schnittstellenbeschreibung

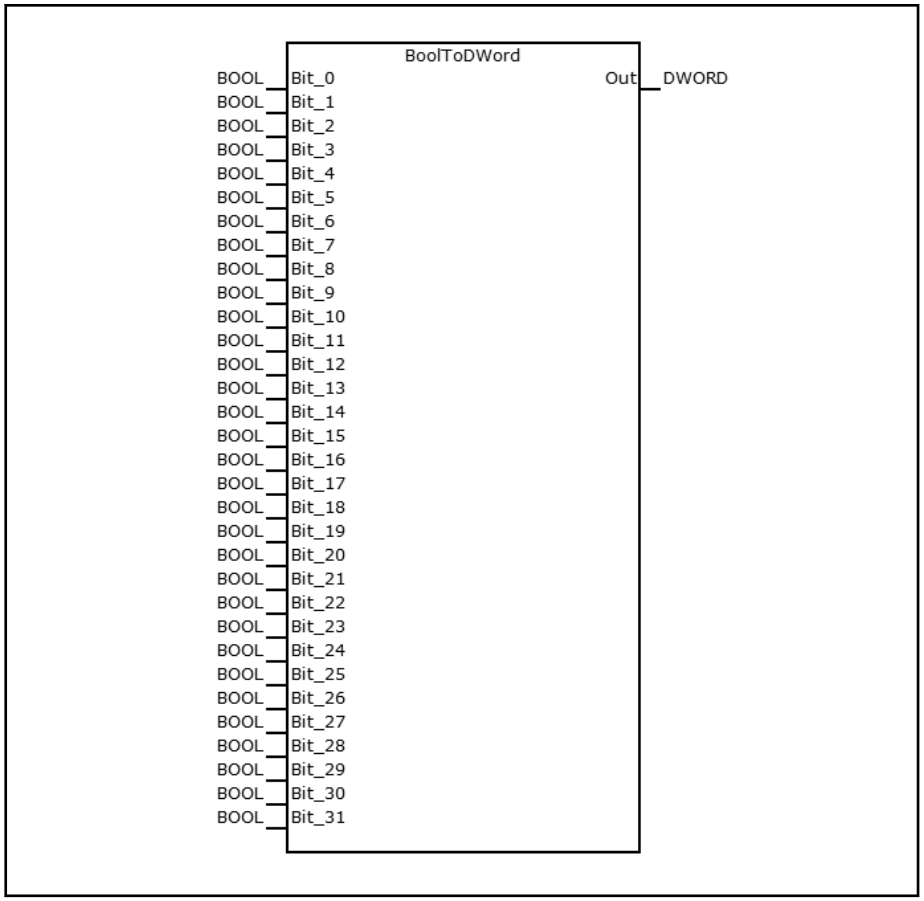


Abb. 16-16: Funktionsbaustein BoolToDWord

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Bit_0	BOOL	Eingang Bit 0
	Bit_1	BOOL	Eingang Bit 1
	Bit_2	BOOL	Eingang Bit 2
	Bit_3	BOOL	Eingang Bit 3
	Bit_4	BOOL	Eingang Bit 4
	Bit_5	BOOL	Eingang Bit 5
	Bit_6	BOOL	Eingang Bit 6
	Bit_7	BOOL	Eingang Bit 7
	Bit_8	BOOL	Eingang Bit 8
	Bit_9	BOOL	Eingang Bit 9
	Bit_10	BOOL	Eingang Bit 10
	Bit_11	BOOL	Eingang Bit 11
	Bit_12	BOOL	Eingang Bit 12
	Bit_13	BOOL	Eingang Bit 13
	Bit_14	BOOL	Eingang Bit 14
	Bit_15	BOOL	Eingang Bit 15

E/A-Typ	Name	Datentyp	Beschreibung
	Bit_16	BOOL	Eingang Bit 16
	Bit_17	BOOL	Eingang Bit 17
	Bit_18	BOOL	Eingang Bit 18
	Bit_19	BOOL	Eingang Bit 19
	Bit_20	BOOL	Eingang Bit 20
	Bit_21	BOOL	Eingang Bit 21
	Bit_22	BOOL	Eingang Bit 22
	Bit_23	BOOL	Eingang Bit 23
	Bit_24	BOOL	Eingang Bit 24
	Bit_25	BOOL	Eingang Bit 25
	Bit_26	BOOL	Eingang Bit 26
	Bit_27	BOOL	Eingang Bit 27
	Bit_28	BOOL	Eingang Bit 28
	Bit_29	BOOL	Eingang Bit 29
	Bit_30	BOOL	Eingang Bit 30
	Bit_31	BOOL	Eingang Bit 31
VAR_OUTPUT	Out	DWORD	DWORD aus 32 zusammengefügten Bits

Tab. 16-40: Schnittstelle BoolToDWord

Name	Typ	Default-Wert	Übernahme
Bit_0	BOOL	FALSE	kontinuierlich
Bit_1	BOOL	FALSE	kontinuierlich
Bit_2	BOOL	FALSE	kontinuierlich
Bit_3	BOOL	FALSE	kontinuierlich
Bit_4	BOOL	FALSE	kontinuierlich
Bit_5	BOOL	FALSE	kontinuierlich
Bit_6	BOOL	FALSE	kontinuierlich
Bit_7	BOOL	FALSE	kontinuierlich
Bit_8	BOOL	FALSE	kontinuierlich
Bit_9	BOOL	FALSE	kontinuierlich
Bit_10	BOOL	FALSE	kontinuierlich
Bit_11	BOOL	FALSE	kontinuierlich
Bit_12	BOOL	FALSE	kontinuierlich
Bit_13	BOOL	FALSE	kontinuierlich
Bit_14	BOOL	FALSE	kontinuierlich
Bit_15	BOOL	FALSE	kontinuierlich
Bit_16	BOOL	FALSE	kontinuierlich

Name	Typ	Default-Wert	Übernahme
Bit_17	BOOL	FALSE	kontinuierlich
Bit_18	BOOL	FALSE	kontinuierlich
Bit_19	BOOL	FALSE	kontinuierlich
Bit_20	BOOL	FALSE	kontinuierlich
Bit_21	BOOL	FALSE	kontinuierlich
Bit_22	BOOL	FALSE	kontinuierlich
Bit_23	BOOL	FALSE	kontinuierlich
Bit_24	BOOL	FALSE	kontinuierlich
Bit_25	BOOL	FALSE	kontinuierlich
Bit_26	BOOL	FALSE	kontinuierlich
Bit_27	BOOL	FALSE	kontinuierlich
Bit_28	BOOL	FALSE	kontinuierlich
Bit_29	BOOL	FALSE	kontinuierlich
Bit_30	BOOL	FALSE	kontinuierlich
Bit_31	BOOL	FALSE	kontinuierlich

Tab. 16-41: Eingangsverhalten BoolToDWord

Funktionsbeschreibung

Der Funktionsbaustein BoolToDWord fasst 32 BOOL-Eingänge zu einem DWORD zusammen.

Bit_0 bis Bit_31

Die Eingänge Bit_0 bis Bit_31 werden ihrer Nummer nach aufsteigend in die Bits des Doppelwortes einsortiert. Bit_0 ist dabei das niederwertigste Bit und Bit_31 das höchstwertige Bit.

Out

Der Ausgang Out enthält das aus den Eingängen Bit_0 bis Bit_31 zusammengesetzte Doppelwort.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.4 SF_BoolToByte

Kurzbeschreibung

Der Funktionsbaustein SF_BoolToByte fasst 8 SAFEBOOL-Eingänge zu einem SAFEBYTE zusammen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-42: Bibliothekszuordnung

Schnittstellenbeschreibung

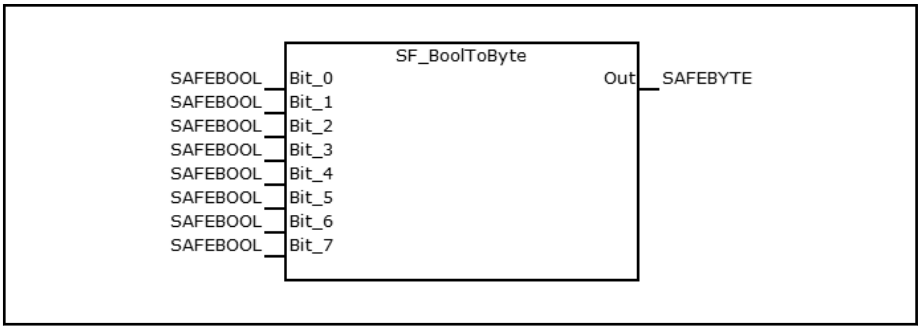


Abb. 16-17: Funktionsbaustein SF_BoolToByte

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Bit_0	SAFEBOOL	Eingang Bit 0
	Bit_1	SAFEBOOL	Eingang Bit 1
	Bit_2	SAFEBOOL	Eingang Bit 2
	Bit_3	SAFEBOOL	Eingang Bit 3
	Bit_4	SAFEBOOL	Eingang Bit 4
	Bit_5	SAFEBOOL	Eingang Bit 5
	Bit_6	SAFEBOOL	Eingang Bit 6
	Bit_7	SAFEBOOL	Eingang Bit 7
VAR_OUTPUT	Out	SAFEBYTE	SAFEBYTE aus 8 zusammengefügten Bits

Tab. 16-43: Schnittstelle SF_BoolToByte

Name	Typ	Default-Wert	Übernahme
Bit_0	SAFEBOOL	FALSE	kontinuierlich
Bit_1	SAFEBOOL	FALSE	kontinuierlich
Bit_2	SAFEBOOL	FALSE	kontinuierlich
Bit_3	SAFEBOOL	FALSE	kontinuierlich
Bit_4	SAFEBOOL	FALSE	kontinuierlich
Bit_5	SAFEBOOL	FALSE	kontinuierlich
Bit_6	SAFEBOOL	FALSE	kontinuierlich
Bit_7	SAFEBOOL	FALSE	kontinuierlich

Tab. 16-44: Eingangsverhalten SF_BoolToByte

Funktionsbeschreibung

Der Funktionsbaustein SF_BoolToByte fasst 8 SAFEBOOL-Eingänge zu einem SAFEBYTE zusammen.

Bit_0 bis Bit_7

Die Eingänge Bit_0 bis Bit_7 werden ihrer Nummer nach aufsteigend in die Bits des Bytes einsortiert. Bit_0 ist dabei das niederwertigste Bit und Bit_7 das höchstwertige Bit.

Out

Der Ausgang Out enthält das aus den Eingängen Bit_0 bis Bit_7 zusammengesetzte Byte.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.5 SF_BoolToWord

Kurzbeschreibung

Der Funktionsbaustein SF_BoolToWord fasst 16 SAFEBOOL-Eingänge zu einem SAFEWORD zusammen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.4.0.0

Tab. 16-45: Bibliothekszuordnung

Schnittstellenbeschreibung

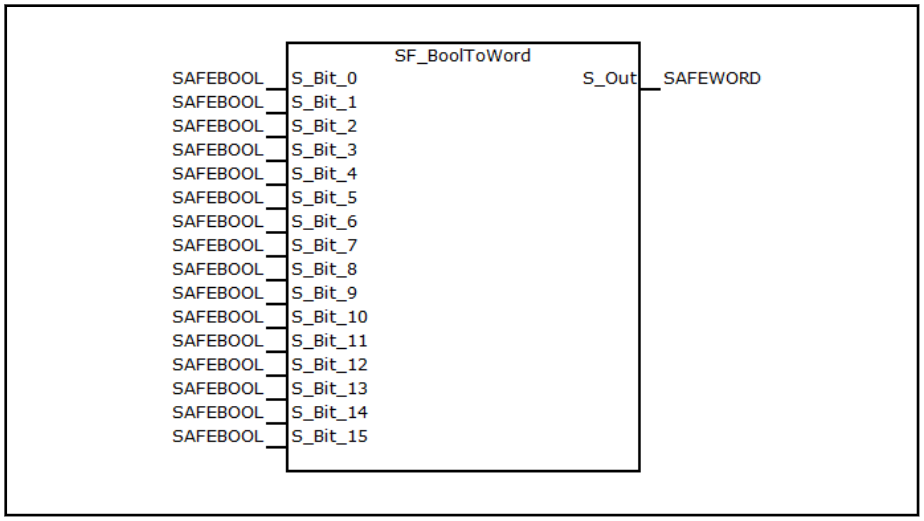


Abb. 16-18: Funktionsbaustein SF_BoolToWord

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	S_Bit_0	SAFEBOOL	Eingang Bit 0
	S_Bit_2	SAFEBOOL	Eingang Bit 1
	S_Bit_2	SAFEBOOL	Eingang Bit 2
	S_Bit_3	SAFEBOOL	Eingang Bit 3
	S_Bit_4	SAFEBOOL	Eingang Bit 4
	S_Bit_5	SAFEBOOL	Eingang Bit 5
	S_Bit_6	SAFEBOOL	Eingang Bit 6
	S_Bit_7	SAFEBOOL	Eingang Bit 7
	S_Bit_8	SAFEBOOL	Eingang Bit 8
	S_Bit_9	SAFEBOOL	Eingang Bit 9
	S_Bit_10	SAFEBOOL	Eingang Bit 10
	S_Bit_11	SAFEBOOL	Eingang Bit 11
	S_Bit_12	SAFEBOOL	Eingang Bit 12
	S_Bit_13	SAFEBOOL	Eingang Bit 13
	S_Bit_14	SAFEBOOL	Eingang Bit 14
	S_Bit_15	SAFEBOOL	Eingang Bit 15
VAR_OUTPUT	S_Out	SAFEWORD	SAFEWORD aus 16 zusammengefügten Bits

Tab. 16-46: Schnittstelle SF_BoolToWord

Name	Typ	Default-Wert	Übernahme
S_Bit_0	SAFEBOOL	FALSE	kontinuierlich
S_Bit_2	SAFEBOOL	FALSE	kontinuierlich
S_Bit_2	SAFEBOOL	FALSE	kontinuierlich
S_Bit_3	SAFEBOOL	FALSE	kontinuierlich
S_Bit_4	SAFEBOOL	FALSE	kontinuierlich
S_Bit_5	SAFEBOOL	FALSE	kontinuierlich
S_Bit_6	SAFEBOOL	FALSE	kontinuierlich
S_Bit_7	SAFEBOOL	FALSE	kontinuierlich
S_Bit_8	SAFEBOOL	FALSE	kontinuierlich
S_Bit_9	SAFEBOOL	FALSE	kontinuierlich
S_Bit_10	SAFEBOOL	FALSE	kontinuierlich
S_Bit_11	SAFEBOOL	FALSE	kontinuierlich
S_Bit_12	SAFEBOOL	FALSE	kontinuierlich
S_Bit_13	SAFEBOOL	FALSE	kontinuierlich
S_Bit_14	SAFEBOOL	FALSE	kontinuierlich
S_Bit_15	SAFEBOOL	FALSE	kontinuierlich

Tab. 16-47: Eingangsverhalten SF_BoolToWord

Funktionsbeschreibung

Der Funktionsbaustein BoolToWord fasst 16 SAFEBOOL-Eingänge zu einem SAFEWORD zusammen.

S_Bit_0 bis S_Bit_15

Die Eingänge S_Bit_0 bis S_Bit_15 werden ihrer Nummer nach aufsteigend in die Bits des Wortes einsortiert. S_Bit_0 ist dabei das niederwertigste Bit und S_Bit_15 das höchstwertige Bit.

S_Out

Der Ausgang S_Out enthält das aus den Eingängen S_Bit_0 bis S_Bit_15 zusammengesetzte Wort.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.6 SF_BoolToDWord

Kurzbeschreibung

Der Funktionsbaustein SF_BoolToDWord fasst 32 SAFEBOOL-Eingänge zu einem SAFEDWORD zusammen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-48: Bibliothekszuordnung

Schnittstellenbeschreibung

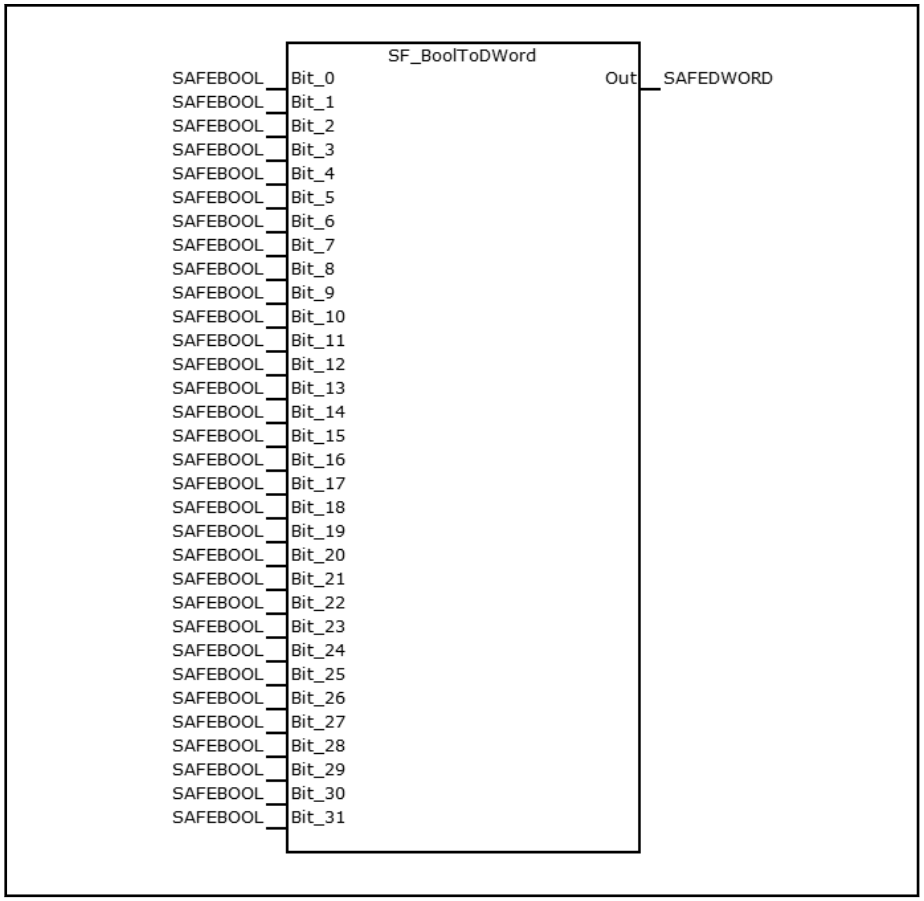


Abb. 16-19: Funktionsbaustein SF_BoolToDWord

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Bit_0	SAFEBOOL	Eingang Bit 0
	Bit_1	SAFEBOOL	Eingang Bit 1
	Bit_2	SAFEBOOL	Eingang Bit 2
	Bit_3	SAFEBOOL	Eingang Bit 3
	Bit_4	SAFEBOOL	Eingang Bit 4
	Bit_5	SAFEBOOL	Eingang Bit 5
	Bit_6	SAFEBOOL	Eingang Bit 6
	Bit_7	SAFEBOOL	Eingang Bit 7
	Bit_8	SAFEBOOL	Eingang Bit 8
	Bit_9	SAFEBOOL	Eingang Bit 9
	Bit_10	SAFEBOOL	Eingang Bit 10
	Bit_11	SAFEBOOL	Eingang Bit 11
	Bit_12	SAFEBOOL	Eingang Bit 12
	Bit_13	SAFEBOOL	Eingang Bit 13
	Bit_14	SAFEBOOL	Eingang Bit 14
	Bit_15	SAFEBOOL	Eingang Bit 15

E/A-Typ	Name	Datentyp	Beschreibung
	Bit_16	SAFEBOOL	Eingang Bit 16
	Bit_17	SAFEBOOL	Eingang Bit 17
	Bit_18	SAFEBOOL	Eingang Bit 18
	Bit_19	SAFEBOOL	Eingang Bit 19
	Bit_20	SAFEBOOL	Eingang Bit 20
	Bit_21	SAFEBOOL	Eingang Bit 21
	Bit_22	SAFEBOOL	Eingang Bit 22
	Bit_23	SAFEBOOL	Eingang Bit 23
	Bit_24	SAFEBOOL	Eingang Bit 24
	Bit_25	SAFEBOOL	Eingang Bit 25
	Bit_26	SAFEBOOL	Eingang Bit 26
	Bit_27	SAFEBOOL	Eingang Bit 27
	Bit_28	SAFEBOOL	Eingang Bit 28
	Bit_29	SAFEBOOL	Eingang Bit 29
	Bit_30	SAFEBOOL	Eingang Bit 30
	Bit_31	SAFEBOOL	Eingang Bit 31
VAR_OUTPUT	Out	SAFEDWORD	SAFEDWORD aus 32 zusammengefügten Bits

Tab. 16-49: Schnittstelle SF_BoolToDWord

Name	Typ	Default-Wert	Übernahme
Bit_0	SAFEBOOL	FALSE	kontinuierlich
Bit_1	SAFEBOOL	FALSE	kontinuierlich
Bit_2	SAFEBOOL	FALSE	kontinuierlich
Bit_3	SAFEBOOL	FALSE	kontinuierlich
Bit_4	SAFEBOOL	FALSE	kontinuierlich
Bit_5	SAFEBOOL	FALSE	kontinuierlich
Bit_6	SAFEBOOL	FALSE	kontinuierlich
Bit_7	SAFEBOOL	FALSE	kontinuierlich
Bit_8	SAFEBOOL	FALSE	kontinuierlich
Bit_9	SAFEBOOL	FALSE	kontinuierlich
Bit_10	SAFEBOOL	FALSE	kontinuierlich
Bit_11	SAFEBOOL	FALSE	kontinuierlich
Bit_12	SAFEBOOL	FALSE	kontinuierlich
Bit_13	SAFEBOOL	FALSE	kontinuierlich
Bit_14	SAFEBOOL	FALSE	kontinuierlich
Bit_15	SAFEBOOL	FALSE	kontinuierlich
Bit_16	SAFEBOOL	FALSE	kontinuierlich

Name	Typ	Default-Wert	Übernahme
Bit_17	SAFEBOOL	FALSE	kontinuierlich
Bit_18	SAFEBOOL	FALSE	kontinuierlich
Bit_19	SAFEBOOL	FALSE	kontinuierlich
Bit_20	SAFEBOOL	FALSE	kontinuierlich
Bit_21	SAFEBOOL	FALSE	kontinuierlich
Bit_22	SAFEBOOL	FALSE	kontinuierlich
Bit_23	SAFEBOOL	FALSE	kontinuierlich
Bit_24	SAFEBOOL	FALSE	kontinuierlich
Bit_25	SAFEBOOL	FALSE	kontinuierlich
Bit_26	SAFEBOOL	FALSE	kontinuierlich
Bit_27	SAFEBOOL	FALSE	kontinuierlich
Bit_28	SAFEBOOL	FALSE	kontinuierlich
Bit_29	SAFEBOOL	FALSE	kontinuierlich
Bit_30	SAFEBOOL	FALSE	kontinuierlich
Bit_31	SAFEBOOL	FALSE	kontinuierlich

Tab. 16-50: Eingangsverhalten SF_BoolToDWord

Funktionsbeschreibung

Der Funktionsbaustein SF_BoolToDWord fasst 32 SAFEBOOL-Eingänge zu einem SAFEDWORD zusammen.

Bit_0 bis Bit_31

Die Eingänge Bit_0 bis Bit_31 werden ihrer Nummer nach aufsteigend in die Bits des Doppelwortes einsortiert. Bit_0 ist dabei das niederwertigste Bit und Bit_31 das höchstwertige Bit.

Out

Der Ausgang Out enthält das aus den Eingängen Bit_0 bis Bit_31 zusammengesetzte Doppelwort.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.7 ByteToBool

Kurzbeschreibung

Der Funktionsbaustein ByteToBool zerlegt ein 8 Bit breites BYTE in 8 BOOL-Ausgänge.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-51: Bibliothekszuordnung

Schnittstellenbeschreibung

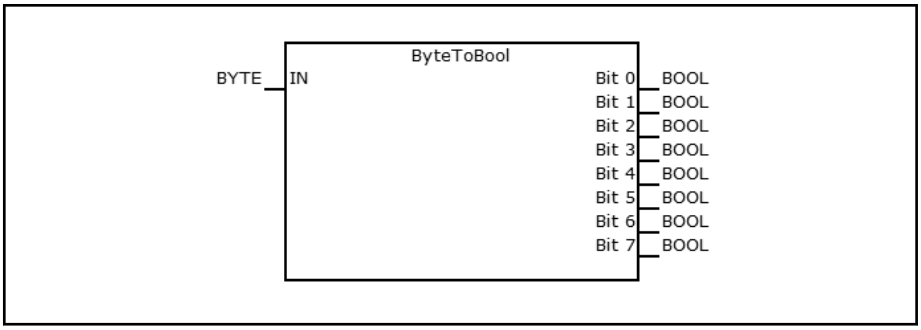


Abb. 16-20: Funktionsbaustein ByteToBool

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	BYTE	zu zerlegendes BYTE
VAR_OUTPUT	Bit_0	BOOL	Ausgang Bit 0
	Bit_1	BOOL	Ausgang Bit 1
	Bit_2	BOOL	Ausgang Bit 2
	Bit_3	BOOL	Ausgang Bit 3
	Bit_4	BOOL	Ausgang Bit 4
	Bit_5	BOOL	Ausgang Bit 5
	Bit_6	BOOL	Ausgang Bit 6
	Bit_7	BOOL	Ausgang Bit 7

Tab. 16-52: Schnittstelle ByteToBool

Name	Typ	Default-Wert	Übernahme
IN	BYTE	FALSE	kontinuierlich

Tab. 16-53: Eingangsverhalten ByteToBool

Funktionsbeschreibung

Der Funktionsbaustein ByteToBool zerlegt ein 8 Bit breites BYTE in 8 BOOL-Ausgänge.

IN

Der Eingang IN enthält das in die Ausgänge Bit_0 bis Bit_7 zu zerlegende Byte.

Bit_0 bis Bit_7

Die Ausgänge Bit_0 bis Bit_7 werden ihrer Nummer nach aufsteigend aus den Bits des Bytes gebildet. Bit_0 ist dabei das niederwertigste Bit und Bit_7 das höchstwertige Bit.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.8 WordToBool

Kurzbeschreibung

Der Funktionsbaustein WordToBool zerlegt ein 16 Bit breites WORD in 16 BOOL-Ausgänge.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-54: Bibliothekszuordnung

Schnittstellenbeschreibung

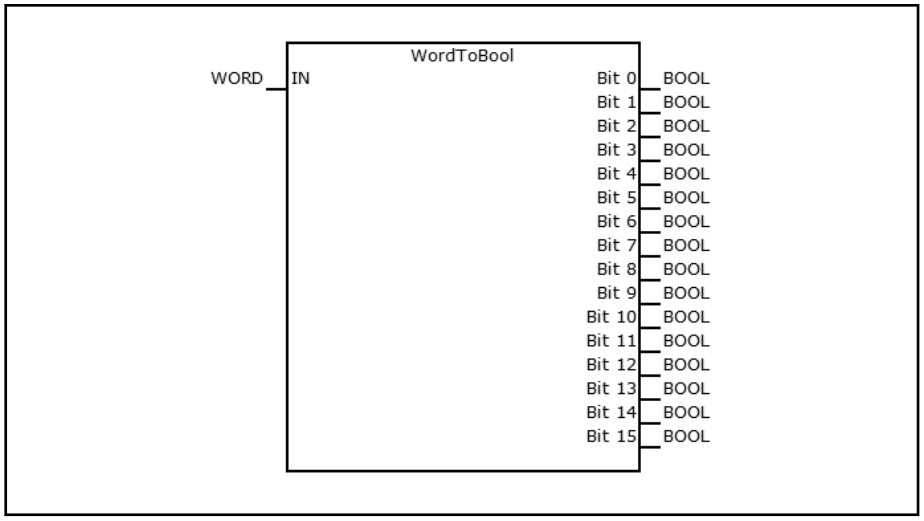


Abb. 16-21: Funktionsbaustein WordToBool

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	WORD	zu zerlegendes WORD
VAR_OUTPUT	Bit_0	BOOL	Ausgang Bit 0
	Bit_1	BOOL	Ausgang Bit 1
	Bit_2	BOOL	Ausgang Bit 2
	Bit_3	BOOL	Ausgang Bit 3
	Bit_4	BOOL	Ausgang Bit 4
	Bit_5	BOOL	Ausgang Bit 5
	Bit_6	BOOL	Ausgang Bit 6
	Bit_7	BOOL	Ausgang Bit 7
	Bit_8	BOOL	Ausgang Bit 8
	Bit_9	BOOL	Ausgang Bit 9
	Bit_10	BOOL	Ausgang Bit 10
	Bit_11	BOOL	Ausgang Bit 11
	Bit_12	BOOL	Ausgang Bit 12
	Bit_13	BOOL	Ausgang Bit 13
	Bit_14	BOOL	Ausgang Bit 14
	Bit_15	BOOL	Ausgang Bit 15

Tab. 16-55: Schnittstelle WordToBool

Name	Typ	Default-Wert	Übernahme
IN	WORD	FALSE	kontinuierlich

Tab. 16-56: Eingangsverhalten WordToBool

Funktionsbeschreibung

Der Funktionsbaustein WordToBool zerlegt ein 16 Bit breites WORD in 16 BOOL-Ausgänge.

IN

Der Eingang IN enthält das in die Ausgänge Bit_0 bis Bit_15 zu zerlegende Wort.

Bit_0 bis Bit_15

Die Ausgänge Bit_0 bis Bit_15 werden ihrer Nummer nach aufsteigend aus den Bits des Wortes gebildet. Bit_0 ist dabei das niederwertigste Bit und Bit_15 das höchstwertige Bit.

Fehlerbehandlung Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.9 DWordToBool

Kurzbeschreibung Der Funktionsbaustein DWordToBool zerlegt ein 32 Bit breites DWORD in 32 BOOL-Ausgänge.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-57: Bibliothekszuordnung

Schnittstellenbeschreibung

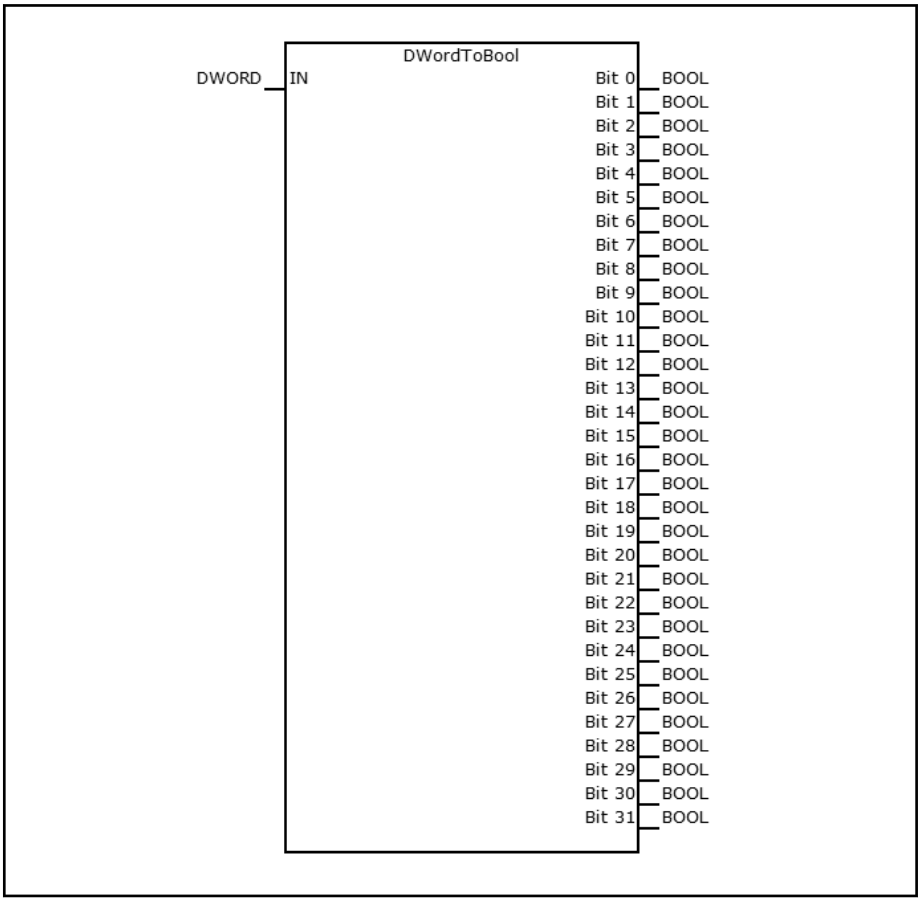


Abb. 16-22: Funktionsbaustein DWordToBool

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	DWORD	zu zerlegendes DWORD
VAR_OUTPUT	Bit_0	BOOL	Ausgang Bit 0
	Bit_1	BOOL	Ausgang Bit 1
	Bit_2	BOOL	Ausgang Bit 2

E/A-Typ	Name	Datentyp	Beschreibung
	Bit_3	BOOL	Ausgang Bit 3
	Bit_4	BOOL	Ausgang Bit 4
	Bit_5	BOOL	Ausgang Bit 5
	Bit_6	BOOL	Ausgang Bit 6
	Bit_7	BOOL	Ausgang Bit 7
	Bit_8	BOOL	Ausgang Bit 8
	Bit_9	BOOL	Ausgang Bit 9
	Bit_10	BOOL	Ausgang Bit 10
	Bit_11	BOOL	Ausgang Bit 11
	Bit_12	BOOL	Ausgang Bit 12
	Bit_13	BOOL	Ausgang Bit 13
	Bit_14	BOOL	Ausgang Bit 14
	Bit_15	BOOL	Ausgang Bit 15
	Bit_16	BOOL	Ausgang Bit 16
	Bit_17	BOOL	Ausgang Bit 17
	Bit_18	BOOL	Ausgang Bit 18
	Bit_19	BOOL	Ausgang Bit 19
	Bit_20	BOOL	Ausgang Bit 20
	Bit_21	BOOL	Ausgang Bit 21
	Bit_22	BOOL	Ausgang Bit 22
	Bit_23	BOOL	Ausgang Bit 23
	Bit_24	BOOL	Ausgang Bit 24
	Bit_25	BOOL	Ausgang Bit 25
	Bit_26	BOOL	Ausgang Bit 26
	Bit_27	BOOL	Ausgang Bit 27
	Bit_28	BOOL	Ausgang Bit 28
	Bit_29	BOOL	Ausgang Bit 29
	Bit_30	BOOL	Ausgang Bit 30
	Bit_31	BOOL	Ausgang Bit 31

Tab. 16-58: Schnittstelle DWordToBool

Name	Typ	Default-Wert	Übernahme
IN	DWORD	FALSE	kontinuierlich

Tab. 16-59: Eingangsverhalten DWordToBool

Funktionsbeschreibung

Der Funktionsbaustein DWordToBool zerlegt ein 32 Bit breites DWORD in 32 BOOL-Ausgänge.

IN

Der Eingang IN enthält das in die Ausgänge Bit_0 bis Bit_31 zu zerlegende Doppelwort.

Bit_0 bis Bit_31

Die Ausgänge Bit_0 bis Bit_31 werden ihrer Nummer nach aufsteigend aus den Bits des Doppelwortes gebildet. Bit_0 ist dabei das niederwertigste Bit und Bit_31 das höchstwertige Bit.

Fehlerbehandlung Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.10 SF_ByteToBool

Kurzbeschreibung Der Funktionsbaustein SF_ByteToBool zerlegt ein 8 Bit breites SAFEBYTE in 8 SAFEBOOL-Ausgänge.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-60: Bibliothekszuordnung

Schnittstellenbeschreibung

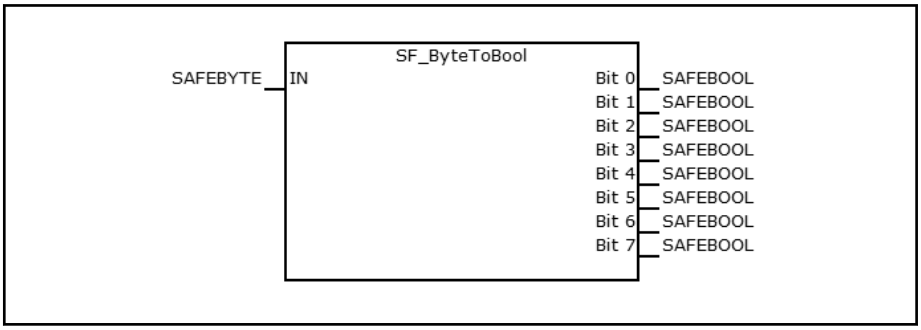


Abb. 16-23: Funktionsbaustein SF_ByteToBool

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	SAFEBYTE	zu zerlegendes SAFEBYTE
VAR_OUTPUT	Bit_0	SAFEBOOL	Ausgang Bit 0
	Bit_1	SAFEBOOL	Ausgang Bit 1
	Bit_2	SAFEBOOL	Ausgang Bit 2
	Bit_3	SAFEBOOL	Ausgang Bit 3
	Bit_4	SAFEBOOL	Ausgang Bit 4
	Bit_5	SAFEBOOL	Ausgang Bit 5
	Bit_6	SAFEBOOL	Ausgang Bit 6
	Bit_7	SAFEBOOL	Ausgang Bit 7

Tab. 16-61: Schnittstelle SF_ByteToBool

Name	Typ	Default-Wert	Übernahme
IN	SAFEBYTE	FALSE	kontinuierlich

Tab. 16-62: Eingangsverhalten SF_ByteToBool

Funktionsbeschreibung Der Funktionsbaustein SF_ByteToBool zerlegt ein 8 Bit breites SAFEBYTE in 8 SAFEBOOL-Ausgänge.

IN

Der Eingang IN enthält das in die Ausgänge Bit_0 bis Bit_7 zu zerlegende Byte.

Bit_0 bis Bit_7

Die Ausgänge Bit_0 bis Bit_7 werden ihrer Nummer nach aufsteigend aus den Bits des Bytes gebildet. Bit_0 ist dabei das niederwertigste Bit und Bit_7 das höchstwertige Bit.

Fehlerbehandlung Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.11 SF_WordToBool

Kurzbeschreibung Der Funktionsbaustein SF_WordToBool zerlegt ein 16 Bit breites SAFEWORD in 16 SAFEBOOL-Ausgänge.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-63: Bibliothekszuordnung

Schnittstellenbeschreibung

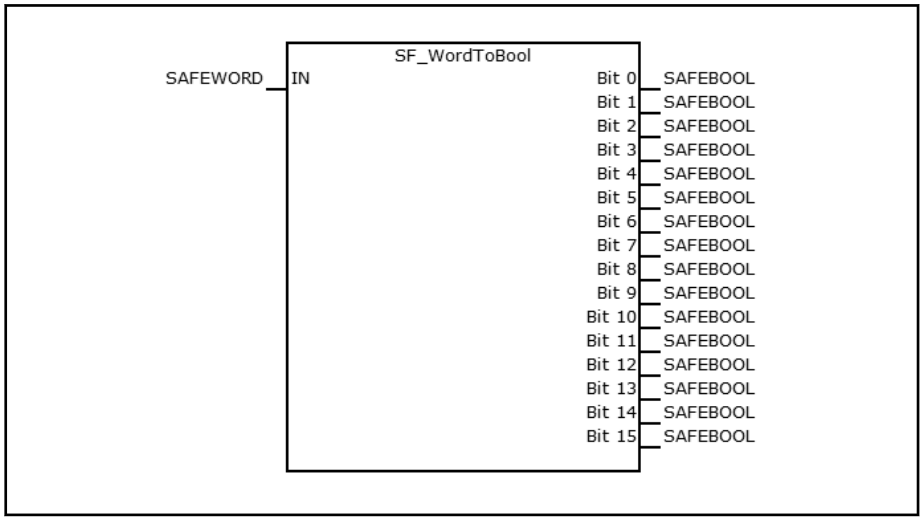


Abb. 16-24: Funktionsbaustein SF_WordToBool

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	SAFEWORD	zu zerlegendes SAFEWORD
VAR_OUTPUT	Bit_0	SAFEBOOL	Ausgang Bit 0
	Bit_1	SAFEBOOL	Ausgang Bit 1
	Bit_2	SAFEBOOL	Ausgang Bit 2
	Bit_3	SAFEBOOL	Ausgang Bit 3
	Bit_4	SAFEBOOL	Ausgang Bit 4
	Bit_5	SAFEBOOL	Ausgang Bit 5
	Bit_6	SAFEBOOL	Ausgang Bit 6
	Bit_7	SAFEBOOL	Ausgang Bit 7
	Bit_8	SAFEBOOL	Ausgang Bit 8
	Bit_9	SAFEBOOL	Ausgang Bit 9
	Bit_10	SAFEBOOL	Ausgang Bit 10

E/A-Typ	Name	Datentyp	Beschreibung
	Bit_11	SAFEBOOL	Ausgang Bit 11
	Bit_12	SAFEBOOL	Ausgang Bit 12
	Bit_13	SAFEBOOL	Ausgang Bit 13
	Bit_14	SAFEBOOL	Ausgang Bit 14
	Bit_15	SAFEBOOL	Ausgang Bit 15

Tab. 16-64: Schnittstelle SF_WordToBool

Name	Typ	Default-Wert	Übernahme
IN	SAFEWORD	FALSE	kontinuierlich

Tab. 16-65: Eingangsverhalten SF_WordToBool

Funktionsbeschreibung

Der Funktionsbaustein SF_WordToBool zerlegt ein 16 Bit breites SAFEWORD in 16 SAFEBOOL-Ausgänge.

IN

Der Eingang IN enthält das in die Ausgänge Bit_0 bis Bit_15 zu zerlegende Wort.

Bit_0 bis Bit_15

Die Ausgänge Bit_0 bis Bit_15 werden ihrer Nummer nach aufsteigend aus den Bits des Wortes gebildet. Bit_0 ist dabei das niederwertigste Bit und Bit_15 das höchstwertige Bit.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.12 SF_DWordToBool

Kurzbeschreibung

Der Funktionsbaustein SF_DWordToBool zerlegt ein 32 Bit breites SAFEDWORD in 32 SAFEBOOL-Ausgänge.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.8.0.0

Tab. 16-66: Bibliothekszuordnung

Schnittstellenbeschreibung

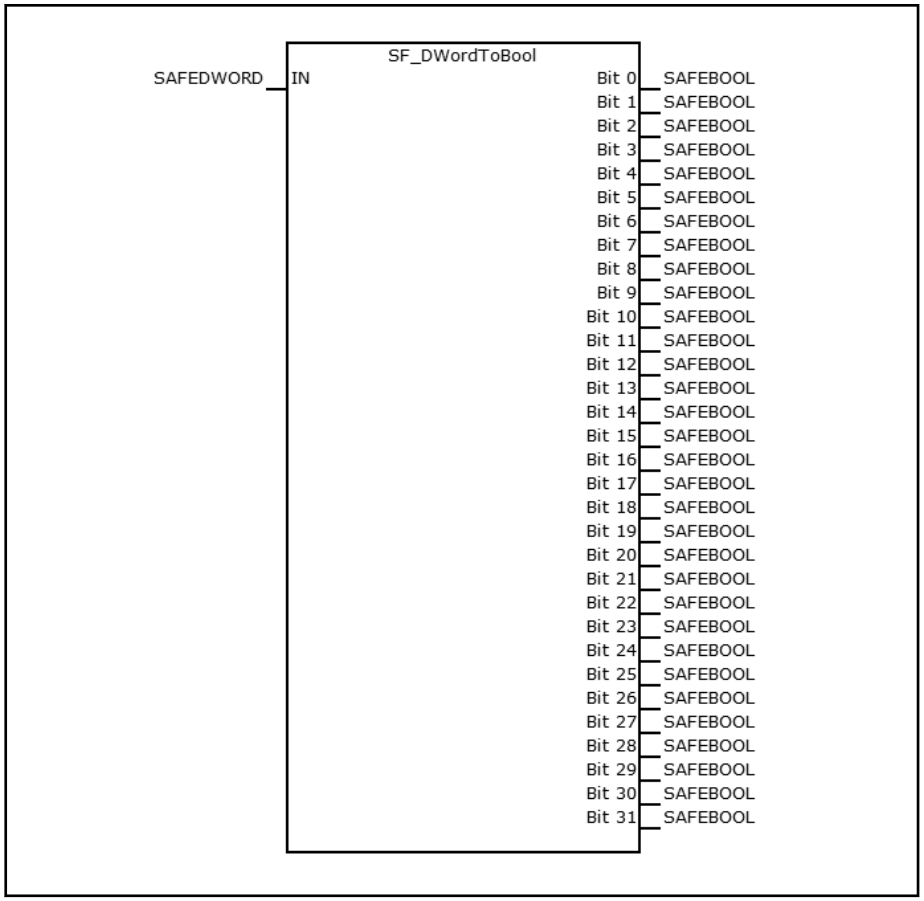


Abb. 16-25: Funktionsbaustein SF_DWordToBool

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	IN	SAFEDWORD	zu zerlegendes SAFEDWORD
VAR_OUTPUT	Bit_0	SAFEBOOL	Ausgang Bit 0
	Bit_1	SAFEBOOL	Ausgang Bit 1
	Bit_2	SAFEBOOL	Ausgang Bit 2
	Bit_3	SAFEBOOL	Ausgang Bit 3
	Bit_4	SAFEBOOL	Ausgang Bit 4
	Bit_5	SAFEBOOL	Ausgang Bit 5
	Bit_6	SAFEBOOL	Ausgang Bit 6
	Bit_7	SAFEBOOL	Ausgang Bit 7
	Bit_8	SAFEBOOL	Ausgang Bit 8
	Bit_9	SAFEBOOL	Ausgang Bit 9
	Bit_10	SAFEBOOL	Ausgang Bit 10
	Bit_11	SAFEBOOL	Ausgang Bit 11
	Bit_12	SAFEBOOL	Ausgang Bit 12
	Bit_13	SAFEBOOL	Ausgang Bit 13
	Bit_14	SAFEBOOL	Ausgang Bit 14

E/A-Typ	Name	Datentyp	Beschreibung
	Bit_15	SAFEBOOL	Ausgang Bit 15
	Bit_16	SAFEBOOL	Ausgang Bit 16
	Bit_17	SAFEBOOL	Ausgang Bit 17
	Bit_18	SAFEBOOL	Ausgang Bit 18
	Bit_19	SAFEBOOL	Ausgang Bit 19
	Bit_20	SAFEBOOL	Ausgang Bit 20
	Bit_21	SAFEBOOL	Ausgang Bit 21
	Bit_22	SAFEBOOL	Ausgang Bit 22
	Bit_23	SAFEBOOL	Ausgang Bit 23
	Bit_24	SAFEBOOL	Ausgang Bit 24
	Bit_25	SAFEBOOL	Ausgang Bit 25
	Bit_26	SAFEBOOL	Ausgang Bit 26
	Bit_27	SAFEBOOL	Ausgang Bit 27
	Bit_28	SAFEBOOL	Ausgang Bit 28
	Bit_29	SAFEBOOL	Ausgang Bit 29
	Bit_30	SAFEBOOL	Ausgang Bit 30
	Bit_31	SAFEBOOL	Ausgang Bit 31

Tab. 16-67: Schnittstelle SF_DWordToBool

Name	Typ	Default-Wert	Übernahme
IN	SAFEDWORD	FALSE	kontinuierlich

Tab. 16-68: Eingangsverhalten SF_DWordToBool

Funktionsbeschreibung

Der Funktionsbaustein SF_DWordToBool zerlegt ein 32 Bit breites SAFEDWORD in 32 SAFEBOOL-Ausgänge.

IN

Der Eingang IN enthält das in die Ausgänge Bit_0 bis Bit_31 zu zerlegende Doppelwort.

Bit_0 bis Bit_31

Die Ausgänge Bit_0 bis Bit_31 werden ihrer Nummer nach aufsteigend aus den Bits des Doppelwortes gebildet. Bit_0 ist dabei das niederwertigste Bit und Bit_31 das höchstwertige Bit.

Fehlerbehandlung

Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.13 SF_IOConfig

Kurzbeschreibung

Der Funktionsbaustein SF_IOConfig ermöglicht das Einlernen und Auswerten von deaktivierten sicheren Busteilnehmern.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.6.0.0

Tab. 16-69: Bibliothekszuordnung

Schnittstellenbeschreibung

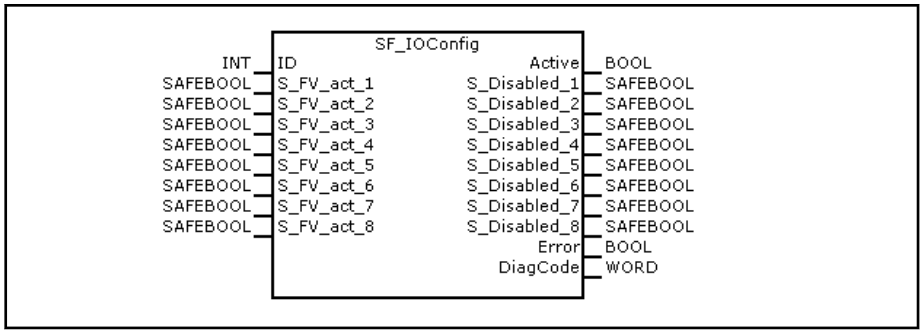


Abb. 16-26: Funktionsbaustein SF_IOConfig

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	ID	INT	Nummer des Konfigurationsblocks (0..63)
	S_FV_act_1	SAFEBOOL	"FV_Activated_S"-Signal des ProfiSafeHost bzw. CSosOriginator Funktionsbausteins für das betreffende Gerät
	S_FV_act_2	SAFEBOOL	"FV_Activated_S"-Signal des ProfiSafeHost bzw. CSosOriginator Funktionsbausteins für das betreffende Gerät
	S_FV_act_3	SAFEBOOL	"FV_Activated_S"-Signal des ProfiSafeHost bzw. CSosOriginator Funktionsbausteins für das betreffende Gerät
	S_FV_act_4	SAFEBOOL	"FV_Activated_S"-Signal des ProfiSafeHost bzw. CSosOriginator Funktionsbausteins für das betreffende Gerät
	S_FV_act_5	SAFEBOOL	"FV_Activated_S"-Signal des ProfiSafeHost bzw. CSosOriginator Funktionsbausteins für das betreffende Gerät
	S_FV_act_6	SAFEBOOL	"FV_Activated_S"-Signal des ProfiSafeHost bzw. CSosOriginator Funktionsbausteins für das betreffende Gerät
	S_FV_act_7	SAFEBOOL	"FV_Activated_S"-Signal des ProfiSafeHost bzw. CSosOriginator Funktionsbausteins für das betreffende Gerät
	S_FV_act_8	SAFEBOOL	"FV_Activated_S"-Signal des ProfiSafeHost bzw. CSosOriginator Funktionsbausteins für das betreffende Gerät
VAR_OUTPUT	Active	BOOL	Konfiguration gefunden, "S_Disabled_X"-Ausgänge sind gültig
	S_Disabled_1	SAFEBOOL	Deaktivierungssignal für das betreffende Gerät
	S_Disabled_2	SAFEBOOL	Deaktivierungssignal für das betreffende Gerät
	S_Disabled_3	SAFEBOOL	Deaktivierungssignal für das betreffende Gerät
	S_Disabled_4	SAFEBOOL	Deaktivierungssignal für das betreffende Gerät
	S_Disabled_5	SAFEBOOL	Deaktivierungssignal für das betreffende Gerät
	S_Disabled_6	SAFEBOOL	Deaktivierungssignal für das betreffende Gerät
	S_Disabled_7	SAFEBOOL	Deaktivierungssignal für das betreffende Gerät
	S_Disabled_8	SAFEBOOL	Deaktivierungssignal für das betreffende Gerät
	Error	BOOL	Es ist ein Fehler aufgetreten
	DiagCode	WORD	Diagnose-Code im Fehlerfall. s. "Fehlerbehandlung" auf Seite 265

Tab. 16-70: Schnittstelle SF_IOConfig

Name	Typ	Min	Max	Default-Wert	Übernahme
ID	INT	0	63	0	Einmalig beim Programmstart
S_FV_act_1	SAFEBOOL			FALSE	kontinuierlich
S_FV_act_2	SAFEBOOL			FALSE	kontinuierlich
S_FV_act_3	SAFEBOOL			FALSE	kontinuierlich
S_FV_act_4	SAFEBOOL			FALSE	kontinuierlich
S_FV_act_5	SAFEBOOL			FALSE	kontinuierlich
S_FV_act_6	SAFEBOOL			FALSE	kontinuierlich
S_FV_act_7	SAFEBOOL			FALSE	kontinuierlich
S_FV_act_8	SAFEBOOL			FALSE	kontinuierlich

Tab. 16-71: Eingangsverhalten SF_IOConfig

Funktionsbeschreibung

Der Funktionsbaustein SF_IOConfig wird im Zusammenhang mit zur Laufzeit rekonfigurierbaren modularen Maschinen verwendet. Er ermöglicht das Einlernen und Auswerten von deaktivierten sicheren Busteilnehmern.

(siehe auch Deaktivieren von sicheren Busteilnehmern, Seite 153)

Es lassen sich bis zu 64 Instanzen des Funktionsbausteins SF_IOConfig parallel betreiben. Dadurch sind insgesamt bis zu 512 sichere Busteilnehmer einlernbar. Diese hohe Zahl wurde vorgesehen, um nicht jede Instanz des SF_IOConfig-Funktionsbausteins voll ausnutzen zu müssen.

Die Instanzen des Funktionsbausteins SF_IOConfig übernehmen, durch den mit dem Funktionsbaustein SF_IOConfigCntl ausgelösten Lernvorgang, die Zustände von den "S_FV_act_X"-Eingängen auf die "S_Disabled_X"-Ausgänge. Eine gelernte Konfiguration wird persistent gespeichert und steht dann beim nächsten Start an den Ausgängen automatisch wieder zur Verfügung.

ID

Der Eingang "ID" wählt eine von 64 Instanzen des Funktionsbausteins SF_IOConfig aus. Typischerweise hat jedes deaktiverbare Modul der Maschine mindestens eine Instanz des Funktionsbausteins SF_IOConfig.

☞ Der Eingang ID darf sich zur Laufzeit nicht ändern und muss mit einer Konstante belegt werden.

S_FV_act_X, S_Disabled_X, Active

Die Eingänge "S_FV_act_x" werden direkt mit dem jeweiligen "FV_activated_S"-Ausgang des ProfisafeHost- bzw. CSosOriginator-Funktionsbausteins verbunden, der die Verbindung zum sicheren Busteilnehmer überwacht. Über diese Information wird ermittelt, ob der Teilnehmer vorhanden ist oder nicht. Ist während des Einlernvorgangs der "FV_activated_S"-Ausgang des Verbindungsüberwachungsbausteins TRUE, so ist der Teilnehmer entweder nicht vorhanden, oder es liegt eine Störung der Verbindung vor.

Beim Einlernen wird der an den 8 "S_FV_act_X"-Eingängen anliegende Zustand auf die "S_Disabled_X"-Ausgänge gespiegelt. Gleichzeitig wird der Zustand in einer Datei auf der SD-Card der Safety-Steuerung abgelegt. Der Ausgang "Active" wird TRUE. Beim nächsten Start des SafeLogic-Programms wird der gespeicherte Zustand an den "S_Disabled_X"-Ausgängen

wiederhergestellt. Der Ausgang "Active" zeigt an, dass eine gültige Konfiguration vorliegt und die S_Disabled_X-Ausgänge gültig sind.

Error, DiagCode

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx bzw. Fxxx, wobei xxx den Fehler-Code repräsentiert.

Im Fehlerfall werden alle "S_Disabled_X"-Ausgänge FALSE. Der "Active"-Ausgang ist ebenfalls FALSE.

Fehler sind beim Funktionsbaustein SF_IOConfig nicht rücksetzbar, weil auf jeden Fall eine Änderung am Projekt und somit ein Programmneustart notwendig sind, um die Fehlerursache zu beheben.

Fehlerbehandlung

Error:

- **TRUE:** Es ist ein Fehler aufgetreten
- **FALSE:** Es ist kein Fehler aufgetreten

DiagCode

DiagCode	Beschreibung
16#0000	Der Baustein ist nicht initialisiert. Das SafeLogic-Programm wurde noch nicht gestartet.
16#8000	Kein Fehler, keine Warnung.
16#C000	Bei der geladenen Bootapplikation stimmt die Prüfsumme über die logischen Geräte nicht mit der Prüfsumme in der auf der CF-Card abgelegten Konfiguration überein. Ursachen: • Die verwendeten logischen EAs der Bootapplikation haben sich gegenüber der ursprünglichen Bootapplikation, mit der die Konfigurationsdatei erzeugt wurde, geändert.
16#C001	Bei der geladenen Bootapplikation stimmt die Prüfsumme über die Funktionsbausteine nicht mit der Prüfsumme in der auf der CF-Card abgelegten Datei überein. Ursachen: • Es wurden SF_IOConfig-Instanzen hinzugefügt oder entfernt. • IDs an SF_IOConfig-Instanzen geändert.
16#F001	Es wurde ein Fehler durch eine andere Instanz des Funktionsbausteins IOConfig erkannt.
16#F002	Es ist keine Instanz eines SF_IOConfigCntl-Funktionsbausteins im SafeLogic-Programm vorhanden.
16#F004	Es ist mehr als eine Instanz des SF_IOConfigCntl-Funktionsbausteins im SafeLogic-Programm vorhanden.
16#F005	Es wurden mehr als 64 Instanzen des SF_IOConfig-Funktionsbausteins im SafeLogic-Programm angelegt.
16#F006	Der Eingang "ID" ist größer 63 (Gültig: 0..63) Alle anderen Instanzen des Funktionsbausteins SF_IOConfig melden am Ausgang "DiagCode" 16#F001.

DiagCode	Beschreibung
16#F007	Der Wert am Eingang "ID" wurde bereits in einer anderen Instanz des Funktionsbausteins SF_IOConfig verwendet. Alle anderen Instanzen des Funktionsbausteins SF_IOConfig melden am Ausgang "DiagCode" 16#F001.
16#F008	Im laufenden SPS-Zyklus wurde erkannt, dass sich die Anzahl oder die Reihenfolge der SF_IOConfig Instanzen geändert hat. Ursachen: • Der Wert am Eingang "ID" hat sich zur Laufzeit geändert. • Bedingtes Ausführen der Funktionsbausteins durch bedingte Sprünge im Programm • Die Aufrufreihenfolge der Funktionsbausteine wurde durch bedingtes Ausführen geändert

Tab. 16-72: Werte des Ausgangs DiagCode

16.3.14 SF_IOConfigCntl

Kurzbeschreibung

Der Funktionsbaustein SF_IOConfigCntl ermöglicht die Deaktivierung von sicheren Teilnehmern über einen Einlernprozess.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.6.0.0

Tab. 16-73: Bibliothekszuordnung

Schnittstellenbeschreibung

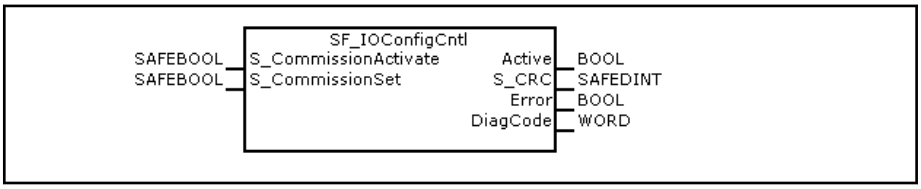


Abb. 16-27: Funktionsbaustein SF_IOConfigCntl

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	S_CommissionActivate	SAFEBOOL	Freigabe des Einlernprozesses
	S_CommissionSet	SAFEBOOL	Übernahme der aktuellen Konfiguration
VAR_OUTPUT	Active	BOOL	Konfiguration gefunden, "S_Disabled_X"-Ausgänge sind aktiv
	S_CRC	SAFEBOOL	Prüfsummer der aktuellen Konfiguration. 0: keine Konfiguration aktiv sonst: eine Konfiguration ist aktiv
	Error	BOOL	Es ist ein Fehler aufgetreten
	DiagCode	WORD	Diagnose-Code im Fehlerfall. s. "Fehlerbehandlung" auf Seite 268

Tab. 16-74: Schnittstelle SF_IOConfigCntl

Name	Typ	Min	Max	Default-Wert	Übernahme
S_CommissionActivate	SAFEBOOL			FALSE	kontinuierlich
S_CommissionSet	SAFEBOOL			FALSE	Steigende Flanke an"S_CommissionActivate"

Tab. 16-75: Eingangsverhalten SF_IOConfigCntl

Funktionsbeschreibung

Der Funktionsbaustein SF_IOConfig wird im Zusammenhang mit zur Laufzeit rekonfigurierbaren modularen Maschinen verwendet. Er ermöglicht das Einlernen und Auswerten von deaktivierten sicheren Busteilnehmern.

(s.a. Deaktivieren von sicheren Busteilnehmern, Seite 153)

S_CommissionActivate, S_CommissionSet

Die beiden Signale "S_CommissionActivate" und "S_CommissionSet" werden für die Übernahme der aktuellen Sollkonfiguration benötigt.

"S_CommissionActivate" muss mit einem Schlüsselschalter über direkt einen sicheren Eingang verbunden werden. Ist der Schaltschrank über einen Schlüssel gesichert, reicht hier ein Schalter. "S_CommisionSet" muss direkt mit einem über einen sicheren Eingang eingelesenen Taster verbunden werden.

Bevor der Einlernvorgang beginnt, sollte geprüft werden, ob alle sicheren Teilnehmer, die in der Maschine vorhanden sind auch eine fehlerfreie Verbindung haben, die Verbindungsüberwachungsbausteine (CSosOriginator oder ProfisafeHost) also keinen Fehler anzeigen.

Danach wird über den Schlüsselschalter der Eingang "S_CommissionActivate" auf TRUE gesetzt. Die Übernahme der aktuellen EA-Konfiguration erfolgt durch Drücken des Tasters am Eingang "S_CommissionSet".

Nach dem Einlernen muss der Schlüsselschalter wieder in seine Ausgangsposition gebracht werden.



Die Sollkonfiguration wird nur auf der CF-Card gespeichert, wenn die aktuelle Bootapplikation läuft. Wurde das SafeLogic-Programm nicht aus der Bootapplikation gestartet, werden die "S_FV_act_X"-Eingänge zwar auf die "S_Disabled_X"-Ausgänge übernommen und die CRC ausgegeben, die so eingelernte Sollkonfiguration wird jedoch beim nächsten Start nicht verwendet.

Active, S_CRC

Der Ausgang "Active" zeigt an, dass eine EA-Sollkonfiguration eingelernt wurde. Dann ist der Ausgang S_CRC ungleich 0. Wurde keine EA-Sollkonfiguration eingelernt, oder wurde ein SafeLogic-Programm mit einer nicht passenden EA-Konfiguration eingespielt, so ist der Ausgang "Active" FALSE.

Die in "S_CRC" ausgegebene Prüfsumme darf für die Identifizierung einer eingelernten EA-Sollkonfiguration verwendet werden. Auf diese Weise kann der Einlernvorgang an Serienmaschinen oder wiederkehrenden Sollkonfigurationen schnell überprüft werden.

Error, DiagCode

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx bzw. Fxxx, wobei xxx den Fehler-Code repräsentiert.

Im Fehlerfall wird "S_CRC" 0 und "Active" wird FALSE.

Fehler sind beim Funktionsbaustein SF_IOConfigCntl nicht rücksetzbar, weil auf jeden Fall eine Änderung am Projekt und somit ein Programmneustart notwendig sind, um die Fehlerursache zu beheben.

Fehlerbehandlung

Error:

- **TRUE:** Es ist ein Fehler aufgetreten
- **FALSE:** Es ist kein Fehler aufgetreten

DiagCode

DiagCode	Beschreibung
16#0000	Der Baustein ist nicht initialisiert. Das SafeLogic-Programm wurde noch nicht gestartet.
16#8000	Kein Fehler, keine Warnung.
16#C000	Bei der geladenen Bootapplikation stimmt die Prüfsumme über die logischen Geräte nicht mit der Prüfsumme in der auf der CF-Card abgelegten Konfiguration überein. Ursachen: • Die verwendeten logischen EAs der Bootapplikation haben sich gegenüber der ursprünglichen Bootapplikation, mit der die Konfigurationsdatei erzeugt wurde, geändert.
16#C001	Bei der geladenen Bootapplikation stimmt die Prüfsumme über die Funktionsbausteine nicht mit der Prüfsumme in der auf der CF-Card abgelegten Datei überein. Ursachen: • Es wurden SF_IOConfig-Instanzen hinzugefügt oder entfernt. • IDs an SF_IOConfig-Instanzen geändert.
16#F003	Es ist keine Instanz eines SF_IOConfig-Funktionsbausteins im SafeLogic-Programm vorhanden.
16#F004	Es ist mehr als eine Instanz des SF_IOConfigCntl-Funktionsbausteins im SafeLogic-Programm vorhanden.
16#F005	Es wurden mehr als 64 Instanzen des SF_IOConfig-Funktionsbausteins im SafeLogic-Programm angelegt.
16#F008	Im laufenden SPS-Zyklus wurde erkannt, dass sich die Anzahl oder die Reihenfolge der SF_IOConfig Instanzen geändert hat. Ursachen: • Der Wert am Eingang "ID" hat sich zur Laufzeit geändert. • Bedingtes Ausführen der Funktionsbausteins durch bedingte Sprünge im Programm • Die Aufrufreihenfolge der Funktionsbausteine wurde durch bedingtes Ausführen geändert

Tab. 16-76: Werte des Ausgangs DiagCode

16.3.15 SF_StartupWarning

Kurzbeschreibung

Der Funktionsbaustein SF_StartupWarning realisiert eine Anlaufwarnung für unübersichtliche Maschinen mit mehreren Bedienstationen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.6.0.2

Tab. 16-77: Bibliothekszuordnung

Schnittstellenbeschreibung

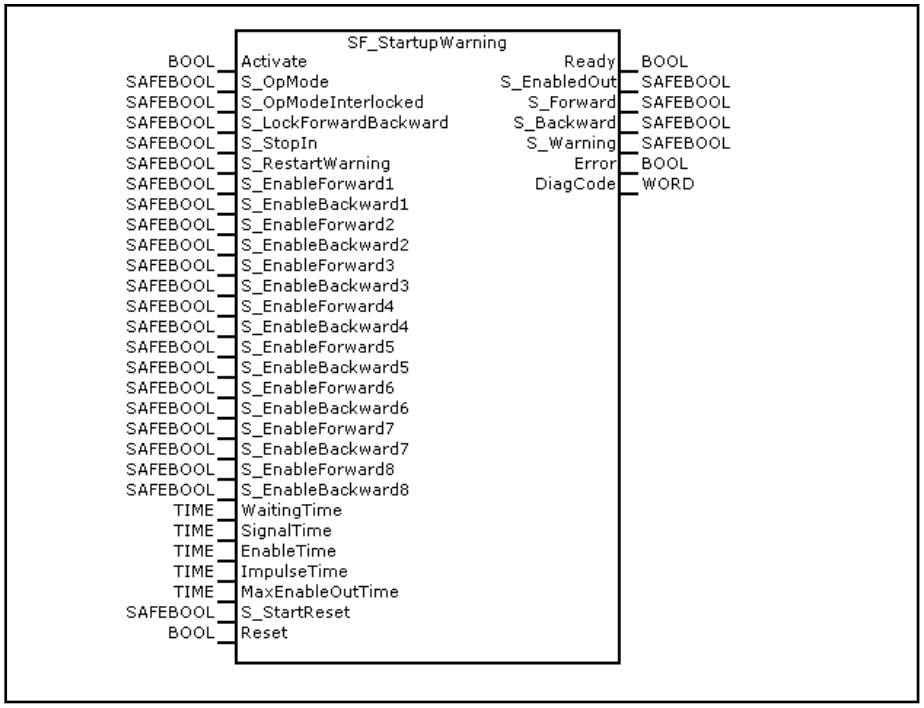


Abb. 16-28: Funktionsbaustein SF_StartupWarning

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungfreigabe des Funktionsbausteins TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv
	S_OpMode	SAFEBOOL	Freigabe der Funktion
	S_OpModeInterlocked	SAFEBOOL	Sperren der Funktion
	S_LockForward-Backward	SAFEBOOL	Sperren der Forward bzw. Backward Freischaltung während der Zeit "EnableTime"
	S_StopIn	SAFEBOOL	Halt/Sperren: mit FALSE wird die Signalfolge abgebrochen bzw. gesperrt
	S_RestartWarning	SAFEBOOL	Neustart Signalfolge: bei einer fallenden Flanke muss die Signalfolge neu gestartet werden (z.B. Schutztür wird geöffnet = neue Situation)
	S_EnableForward1	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Forward“ auf TRUE.
	S_EnableBackward1	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Backward“ auf TRUE.
	S_EnableForward2	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Forward“ auf TRUE.
	S_EnableBackward2	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Backward“ auf TRUE.
	S_EnableForward3	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Forward“ auf TRUE.

E/A-Typ	Name	Datentyp	Beschreibung
	S_EnableBackward3	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Backward“ auf TRUE.
	S_EnableForward4	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Forward“ auf TRUE.
	S_EnableBackward4	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Backward“ auf TRUE.
	S_EnableForward5	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Forward“ auf TRUE.
	S_EnableBackward5	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Backward“ auf TRUE.
	S_EnableForward6	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Forward“ auf TRUE.
	S_EnableBackward6	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Backward“ auf TRUE.
	S_EnableForward7	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Forward“ auf TRUE.
	S_EnableBackward7	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Backward“ auf TRUE.
	S_EnableForward8	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Forward“ auf TRUE.
	S_EnableBackward8	SAFEBOOL	Startet den Ablauf und schaltet während der Zeit „EnableTime“ den VAR_OUTPUT „S_Backward“ auf TRUE.
	WaitingTime	TIME	Wartezeit bevor ein „Enable“ auf einen Output durchgeschaltet wird.
	SignalTime	TIME	Zeitdauer der Warnsignalausgabe vor der Bereitschaftsphase, "SignalTime" beginnt gleichzeitig mit „WaitingTime“
	EnableTime	TIME	Zeitdauer der Bereitschaftsphase, die nach Ablauf der „WaitingTime“ gestartet wird.
	ImpulseTime	TIME	Zeitabschnitt der Warnsignalausgabe während der Bereitschaft. Wird nach Ablauf der „WaitingTime“ gleichzeitig mit "EnableTime"gestartet.
	MaxEnabledOut-Time	TIME	Maximale Zeit, in der der Tippbetrieb ununterbrochen aktiviert sein kann; ein Neustart der Signalfolge wird erzwungen. (bei t#0s ist dieser Grenzwert deaktiviert)
	S_StartReset	SAFEBOOL	„Kaltstart“: automatischer Reset, wenn der FB aktiviert wird
	Reset	BOOL	Reset wirkt auf die Zustände Init, Input Invalid Error und Reset Error 1. Siehe Zustandsdiagramm.
VAR_OUTPUT	Ready	BOOL	Zeigt die Betriebsbereitschaft des Funktionsbausteins an.
	S_EnabledOut	SAFEBOOL	Ist TRUE während die Zeit „EnableTime“ läuft und ein entsprechender Enable Eingang TRUE ist.
	S_Forward	SAFEBOOL	Ist TRUE während die Zeit „EnableTime“ läuft und „S_EnableForwardX“ TRUE ist.

E/A-Typ	Name	Datentyp	Beschreibung
	S_Backward	SAFEBOOL	Ist TRUE während die Zeit „EnableTime“ läuft und „S_EnableBackwardX“ TRUE ist.
	S_Warning	SAFEBOOL	Ist TRUE während die „Signal-“ oder „ImpulseTime“ läuft.
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe "Diagnose-Code" auf Seite 277)

Tab. 16-78: Schnittstelle SF_StartupWarning

Name	Typ	Min	Max	Default-Wert	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_OpMode	SAFEBOOL			FALSE	kontinuierlich
S_OpModeInterlocked	SAFEBOOL			FALSE	kontinuierlich
S_LockForwardBackward	SAFEBOOL			FALSE	kontinuierlich
S_StopIn	SAFEBOOL			FALSE	kontinuierlich
S_RestartWarning	SAFEBOOL			FALSE	kontinuierlich
S_EnableForward1	SAFEBOOL			FALSE	kontinuierlich
S_EnableBackward1	SAFEBOOL			FALSE	kontinuierlich
S_EnableForward2	SAFEBOOL			FALSE	kontinuierlich
S_EnableBackward2	SAFEBOOL			FALSE	kontinuierlich
S_EnableForward3	SAFEBOOL			FALSE	kontinuierlich
S_EnableBackward3	SAFEBOOL			FALSE	kontinuierlich
S_EnableForward4	SAFEBOOL			FALSE	kontinuierlich
S_EnableBackward4	SAFEBOOL			FALSE	kontinuierlich
S_EnableForward5	SAFEBOOL			FALSE	kontinuierlich
S_EnableBackward5	SAFEBOOL			FALSE	kontinuierlich
S_EnableForward6	SAFEBOOL			FALSE	kontinuierlich
S_EnableBackward6	SAFEBOOL			FALSE	kontinuierlich
S_EnableForward7	SAFEBOOL			FALSE	kontinuierlich

Name	Typ	Min	Max	Default-Wert	Übernahme
S_EnableBack-ward7	SAFEBOOL			FALSE	kontinuierlich
S_EnableFor-ward8	SAFEBOOL			FALSE	kontinuierlich
S_EnableBack-ward8	SAFEBOOL			FALSE	kontinuierlich
WaitingTime	TIME	T#1s	T#1m	T#3s	steigende Flanke an "Activate"
SignalTime	TIME	T#0s	≤ WaitingTime	T#1s	steigende Flanke an "Activate"
EnableTime	TIME	T#1s	T#10m	T#12s	steigende Flanke an "Activate"
ImpulseTime	TIME	T#0s	≤ EnableTime	T#1s	steigende Flanke an "Activate"
MaxEnabledOut-Time	TIME	T#0s		T#0s	steigende Flanke an "Activate"
S_StartReset	SAFEBOOL			FALSE	kontinuierlich
Reset	SAFEBOOL			FALSE	kontinuierlich

Tab. 16-79: Eingangsverhalten SF_StartupWarning

Normativer Verweis

Norm	Anforderung
EN1010-1:2004	5.2.7.2 Anlaufwarneinrichtung für unübersichtliche Maschinen
EN1034-1:2000+A1:2010	5.6 Anlaufwarneinrichtungen
EN 62061:2005	5.2.4.2 Sicherheits-Integritätslevel
EN ISO 13849-1:2008	4.2.2 Beitrag der Risikominimierung durch das Steuerungssystem

Funktionsbeschreibung

Der Baustein erfüllt die in der Norm EN1010-1:2004 geforderte Anlaufwarnung in Kapitel 5.2.7.2 und die in der Norm EN1034-1:2000+A1:2010 geforderte Anlaufwarneinrichtung in Kapitel 5.6 Maschinen, die unübersichtlich sind oder an denen die gegenseitige Verständigung erschwert ist, müssen eine Anlaufwarneinrichtung haben.

Bestehen Maschinen, die unübersichtlich sind oder an denen die gegenseitige Verständigung erschwert ist, aus mehreren Maschinengruppen, muss vor dem Anlauf jeder Gruppe ein Warnsignal gegeben werden können. Gerade große Maschinen sind aufgrund ihrer Länge oft unübersichtlich. Häufig ist dadurch auch die gegenseitige Verständigung erschwert. Maschinen gelten ab einer Länge von 7 m und einer Höhe von 1,6 m als unübersichtlich.

Als Warnsignal ist ein akustisches Signal erforderlich, das durch optische Signale unterstützt werden kann. Siehe EN 457. Die Steuerung der Anlaufwarneinrichtung muss den Sicherheits-Integritätslevel SIL 1 nach EN 62061:2005 oder den Performance Level PL c nach EN ISO 13849-1:2008" entsprechen, soweit für spezielle Maschinen keine andere Kategorie festgelegt ist.

Die Anlaufwarneinrichtung muss mit dem Befehlsgerät zum Ingangsetzen der Maschine so gekoppelt sein, dass zwischen Beginn des Warnsignals und der Ingangsetzung ausreichend Zeit vorhanden ist, dass sich Personen in Sicherheit bringen können. Die Startbereitschaft der Maschine beginnt erst nach Ablauf der Wartezeit (Waiting Time) und erlischt am Ende der Bereitschaftszeit (EnableTime).

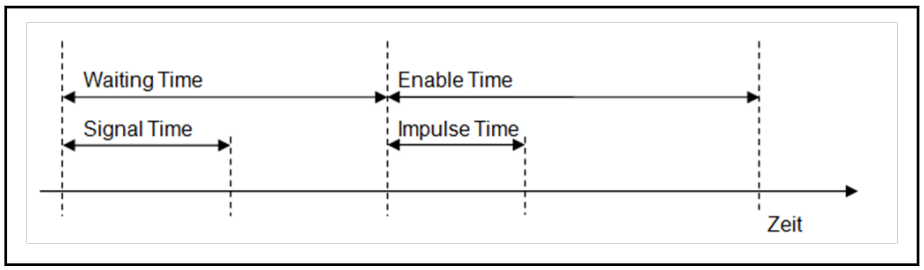


Abb. 16-29: Generelle Ablaufsequenz

Nach dem Betätigen eines "S_EnableForwardX" oder "S_EnableBackwardX" Eingangs, laufen die Wartezeit und die Signalzeit gleichzeitig los. Die Bereitschaftszeit startet automatisch nach Ablauf der Wartezeit.

Hierbei ist die folgende Tabelle zu beachten, in der Anlaufwarneinrichtungen in Kategorien A, B, und C eingeteilt werden:

Kategorie	SignalTime	WaitingTime	EnableTime
A	5s	15s	≤ 30s
B	3s	5s	≤ 30s
C	1s bis 3s	3s	≤ 15s

Tab. 16-80: Kategorien der Anlaufwarneinrichtung

Der Ablauf ist wie folgt:

1. "S_EnableForwardN" oder "S_EnableBackwardN" drücken
2. Warnton ist für "SignalTime" aktiv
3. "EnableTime" abwarten
4. Warnton ist für "ImpulseTime" aktiv
5. "S_EnableForwardN" oder "S_EnableBackwardN" drücken, um Bewegung auszulösen

N bedeutet hier, dass nur zusammengehörige Eingänge "S_EnableForwardN" oder "S_EnableBackwardN" reagieren. Alle anderen Eingänge "S_EnableForwardX" und "S_EnableBackwardX" sind während der Freigabe gesperrt.

Eine besondere Anforderung kann sein, dass während der Wartezeit keine Drehrichtungsumkehr der Maschine erfolgen darf. Dies kann über den Eingang "S_LockForwardBackward" erreicht werden.

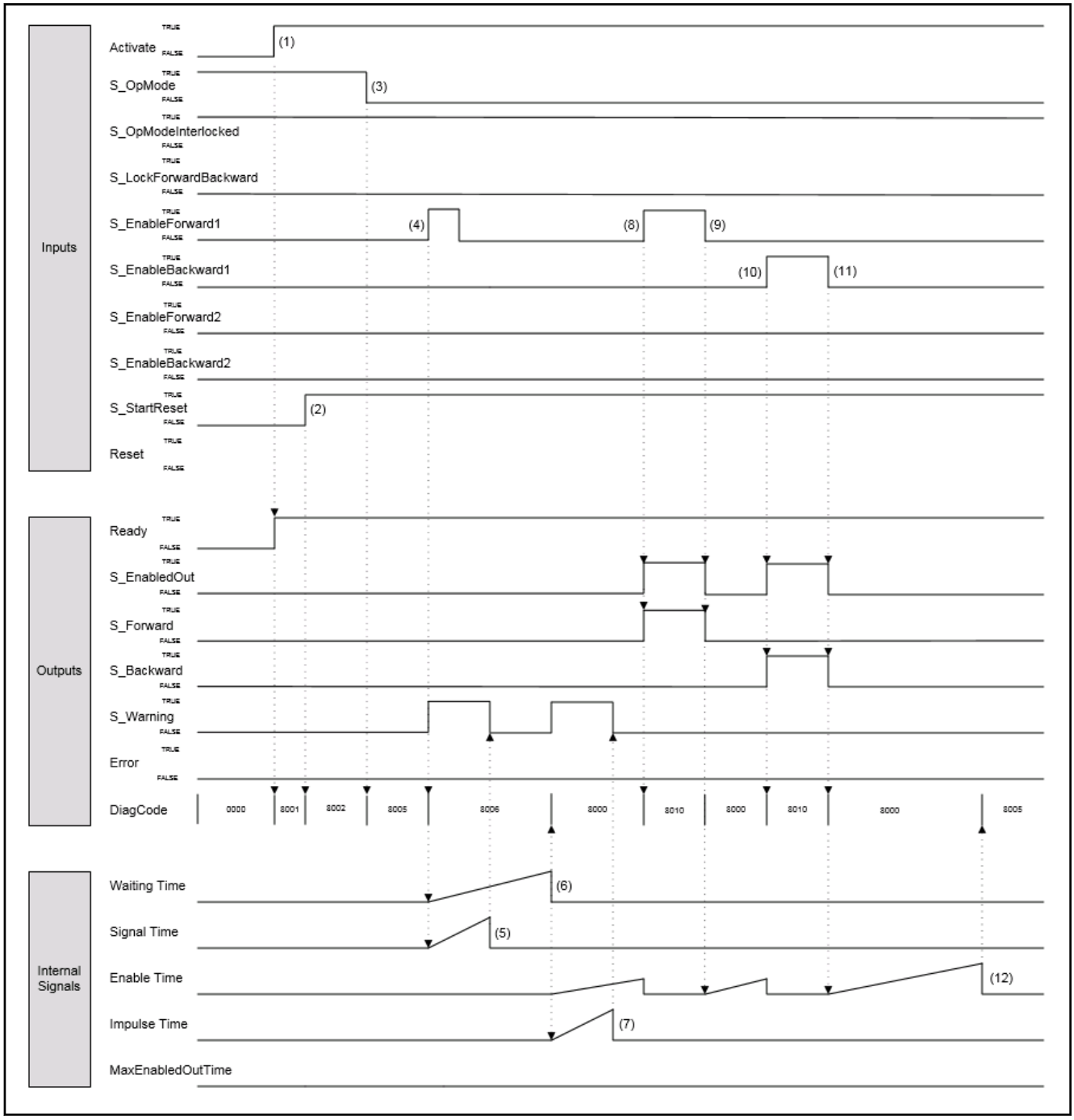


Abb. 16-30: SF_StartupWarning Timing Diagramm

Activate, Ready

Wenn "Activate" auf TRUE geht, wird der Ausgang "Ready" TRUE. Wenn "Activate" auf FALSE geht geht der Funktionsbaustein in den Zustand "Idle". Der Ausgang "Ready" ist dann FALSE. "Activate" hat Priorität vor allen anderen Eingängen.

Reset, S_StartReset

Der Eingang "Reset" quitiert aufgetretene Fehler. Nach dem Start des Funktionsbausteins wird im Zustand "Init" gewartet, bis der Anlauf einmalig über eine Steigende Flanke an "Reset" quitiert wurde. Diese Funktionsbausteinan-

laufquittierung kann über ein TRUE am Eingang "S_StartReset" übersprungen werden.

S_OpMode, S_OpModeInterlocked

Der low-aktive Eingang "S_OpMode" leitet eine Bedienhandlung ein. Dadurch werden die Eingänge "EnableForwardX" und "EnableBackwardX" aktiviert. Reichen die acht am Funktionsbaustein vorhandenen Bedienstationen nicht aus, so können weitere SF_StartupWarning-Funktionsbausteine zusammengeschaltet werden. Die Verriegelung der Funktionsbausteine untereinander erfolgt über den ebenfalls low-aktiven Eingang "S_OpModeInterlocked". Dies wird als Verblockung bezeichnet, s. auch Abb. 16-31 "Verblockung von mehreren SF_StartupWarning-Funktionsbausteinen" auf Seite 275.

Es ist sicherzustellen, dass immer nur eine Instanz des Funktionsbausteins SF_StartupWarning die Kontrolle über die Maschinen hat.

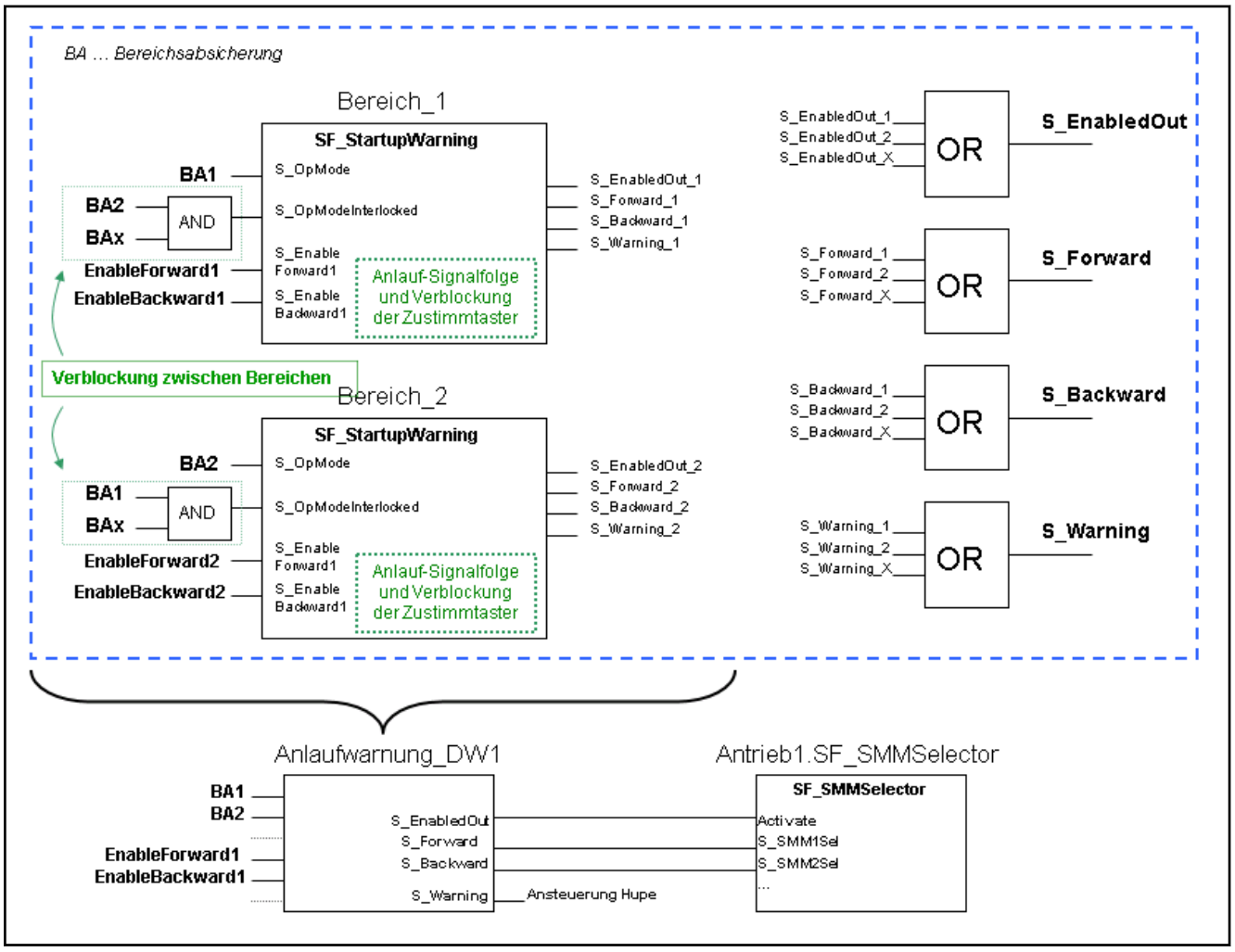


Abb. 16-31: Verblockung von mehreren SF_StartupWarning-Funktionsbausteinen

EnableForwardX, EnableBackwardX, S_Forward, S_Backward

Die Eingänge "EnableForwardX" und "EnableBackwardX" (X: 1..8) sind nur aktiv, wenn der Funktionsbaustein Kontrolle über die Maschine hat (s. Eingang "S_OpMode"). Sie dienen dem Auslösen der Bewegung. Diese Ein-

gänge sind typischerweise an Taster angeschlossen. Die Bewegungsfreigabe ist zeitüberwacht. Ist die Bewegung freigegeben, so wird der Ausgang "S_Forward" bzw. "S_Backward" gesetzt. Der Ablauf der Bewegungsfreigabe ist im Abb. 16-30 "SF_StartupWarning Timing Diagramm" auf Seite 274 zusammengefasst.

S_LockForwardBackward

Der Eingang "S_LockForwardBackward" verhindert die Umgekehr der Bewegungsrichtung nach erfolgter Freigabe. Ist **S_LockForwardBackward** True, kann die Bewegung nur in die Richtung erfolgen, mit der auch die Anlaufwarnung eingeleitet wurde.

S_StopIn

Bricht eine aktive Bewegungsfreigabe ab. "S_StopIn" ist low-aktiv. Wird "S_StopIn" nach erfolgtem Abruch wieder auf TRUE gesetzt, so erfolgt vor der Bewegungsfreigabe eine erneute Anlaufwarnung. "S_StopIn" wird wie eine NotAus-Kette verschaltet. Hierrüber kann der Maschinenanlauf effektiv verhindert werden, ohne eine NotAus-Reaktion für die gesamte Maschine auszulösen.

S_RestartWarning

Der Eingang "S_RestartWarning" erzwingt eine erneute Anlaufwarnung. Anders als bei "S_StopIn" wird die Bewegungsfreigabe jedoch nicht abgebrochen.

WaitingTime, EnableTime, MaxEnabledOutTime

Der Eingang "WaitingTime" definiert die Zeit zwischen der Anlaufwarnung und der Bewegungsfreigabe. In dieser Zeit haben müssen sich Menschen aus dem durch die Bewegung gefährdeten Bereich entfernen oder den Anlauf abbrechen. Dies kann zum Beispiel über den "S_StopIn"-Eingang erfolgen.

"EnableTime" bestimmt die Zeit, die nach erfolgter Freigabe vergehen darf ohne das eine Bewegung angefordert wird. Der entsprechende Zähler wird bei jeder Bewegung wieder zurück gesetzt.

Eine Bewegungsfreigabe bleibt maximal für die am Eingang "MaxEnabledOutTime" angegebene Zeit bestehen. Voraussetzung ist, dass ein "S_EnableForwardX" bzw. "S_EnableBackwardX"-Eingang immer wieder betätigt wird, um die in "EnableTime" definierte Zeit zurückzusetzen.

SignalTime, ImpulseTime, S_Warning

Die Eingänge "SignalTime" und "ImpulseTime" bestimmen die Länge des an "S_Warning" ausgegebenen Impulses. "SignalTime" bestimmt die Länge des Impulses für die Anlaufwarnung, "ImpulseTime" die Länge des Bestätigungsimpulses.

Folgende Randbedingungen sind zu beachten:

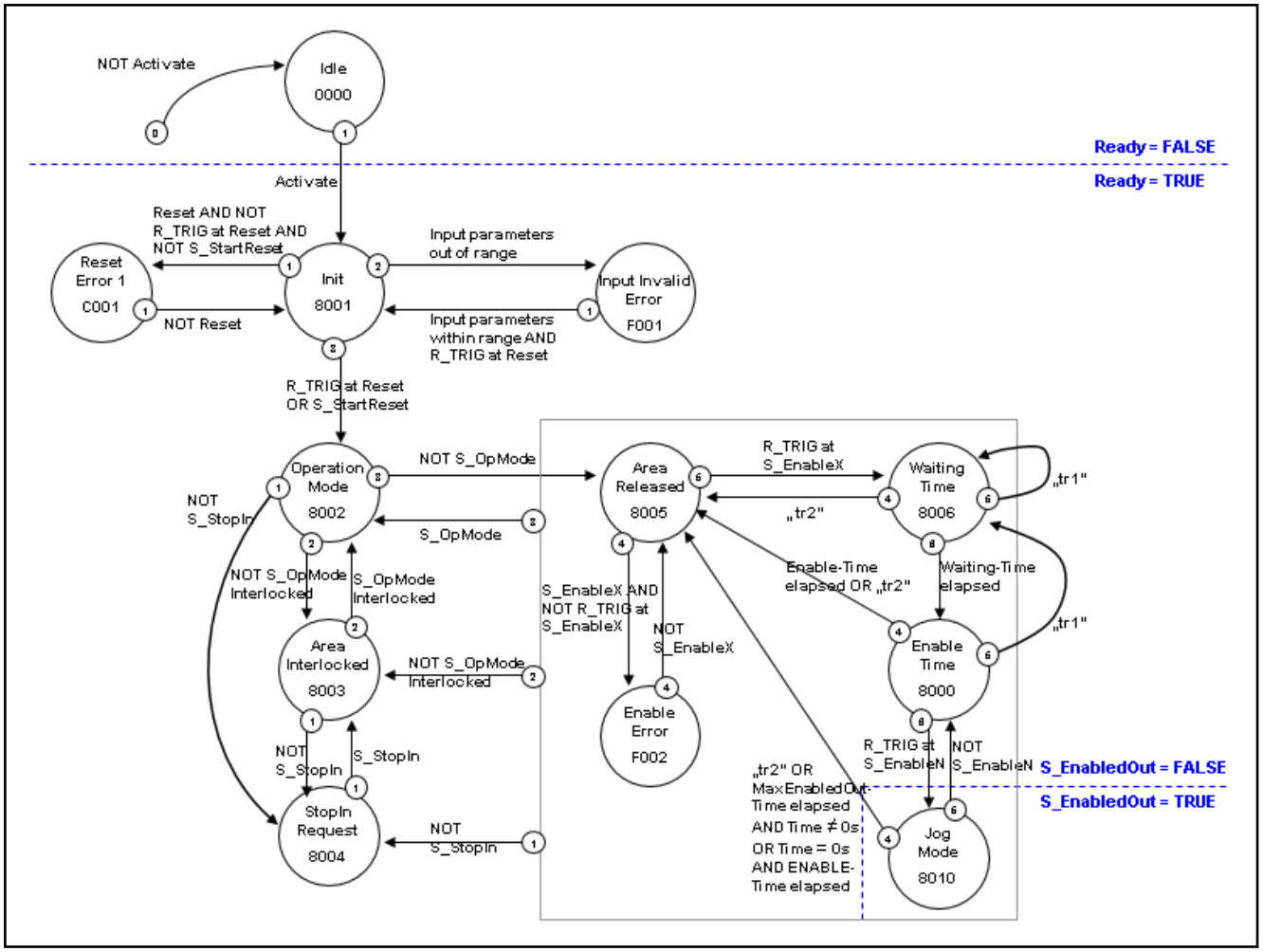
- "SignalTime" muss kleiner als "WaitingTime" sein.
- "ImpulseTime" muss kleiner als "EnableTime" sein.

Zeitgenauigkeit

Die zeitliche Genauigkeit der am Ausgang "S_Warning" anstehenden Zeit wird in Zeitmessungen erläutert.

Genauigkeit von Zeitmessungen, Seite 229

Zustandsdiagramm



tr1	S_LockForwardBackward=TRUE AND R_TRIG an komplementärem Forward/Backward Eingang zu S_EnableN
tr2	F_TRIG at S_RestartWarning
X	beliebiger Enable-Eingang
N	gleicher Enable-Eingang; keine Verblockung zw. Forward/Backward, wenn S_LockForwardBackward = FALSE

Abb. 16-32: Zustandsdiagram SF_StartupWarning

Diagnose-Code

DiagCode	Zustand	Beschreibung	Ausgänge
16#0000	Idle	FB ist nicht aktiv.	Ready := FALSE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := FALSE
16#8000	Enable Time	Anlaufwarnung ist im Bereich "Freigabezeit"	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := FALSE
16#8001	Init	Zustand, nachdem der FB aktiviert wurde.	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := FALSE
16#8002	Operation Mode	Normalbetrieb, es liegt keine Anforderung vor	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := FALSE
16#8003	Area Interlocked	Der Eingang "S_OpModeInterlocked" ist FALSE (Verblockung zwischen Bereichen)	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := FALSE
16#8004	StopIn Request	Der Eingang "S_StopIn" ist FALSE, die Signalfolge kann nicht gestartet werden.	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := FALSE

DiagCode	Zustand	Beschreibung	Ausgänge
16#8005	Area Released	Die Bereichsabsicherung wurde aktiviert, die Anlaufwarnung kann mit einem "Enable"-Eingang gestartet werden.	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := FALSE
16#8006	Waiting Time	Anlaufwarnung ist im Bereich „Wartezeit“.	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := TRUE/FALSE (→ SignalTime / ImpulseTime) Error := FALSE
16#8010	Jog Mode	„Tippbetrieb“ ist aktiv.	Ready := TRUE S_EnabledOut := TRUE S_Forward := TRUE, wenn ein S_Forward Taster gedrückt ist S_Backward := TRUE, wenn ein S_Backward Taster gedrückt ist S_Warning := FALSE Error := FALSE
16#C001	Reset Error 1	"Reset" ist TRUE, es wird aber eine steigende Flanke benötigt um aus den Zustand INIT zu kommen.	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := TRUE
16#F001	Input Invalid Error	Bei den Time-Eingängen ist ein Wert eingetragen, der außerhalb der Min- / Max-Grenzwerte liegt.	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := TRUE
16#F002	Enable Error	Der Zustand Area Released wurde erreicht (bzw. die Bereichs- absicherung wurde betätigt) und es war bereits ein "Enable"-Eingang TRUE, damit eine steigende Flanke an einem "Enable"-Eingang erkannt werden kann, müssen zunächst alle Enable-Eingänge FALSE werden.	Ready := TRUE S_EnabledOut := FALSE S_Forward := FALSE S_Backward := FALSE S_Warning := FALSE Error := TRUE

Tab. 16-81: Zustandsbeschreibung SF_StartupWarning

16.3.16 SF_ClockGenerator

Kurzbeschreibung

Der Funktionsbaustein SF_ClockGenerator stellt einen in Periodenlänge und Tastung einstellbaren Implulsgenerator bereit.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.4.0.0

Tab. 16-82: Bibliothekszuordnung

Schnittstellenbeschreibung

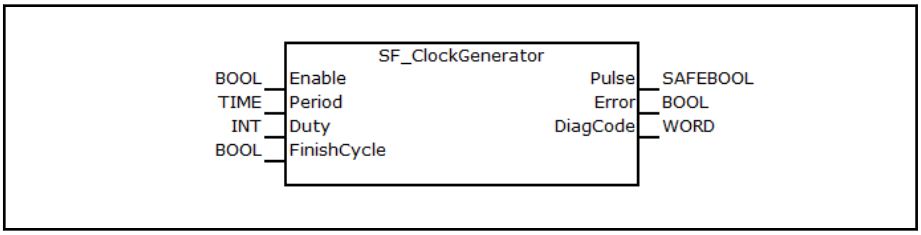


Abb. 16-33: Funktionsbaustein SF_ClockGenerator

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Enable	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv, Pulse ist FALSE (Ausnahme: FinishCycle=TRUE)
	Period	TIME	Zykluszeit
	Duty	INT	High-Signal in %
	FinishCycle	BOOL	Zyklus wird nach Rücknahme von Enable noch komplettiert. TRUE: Pulse bleibt bis zum Ende der High-Phase TRUE, obwohl Enable bereits FALSE ist FALSE: Der Pulse-Ausgang wird sofort auf FALSE gesetzt, sobald Enable auf False geht
VAR_OUTPUT	Pulse	SAFEBOOL	Generiertes Clock-Signal
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Tab. 16-85 "Werte des Ausgangs DiagCode" auf Seite 282)

Tab. 16-83: Schnittstelle SF_ClockGenerator

Name	Typ	Min	Max	Default-Wert	Übernahme
Enable	BOOL			FALSE	kontinuierlich
Period	TIME	T#100MS	T#1H	T#1S	steigende Flanke an Enable
Duty	INT	1	99	50	steigende Flanke an Enable
FinishCycle	BOOL			TRUE	steigende Flanke an Enable

Tab. 16-84: Eingangsverhalten SF_ClockGenerator

Funktionsbeschreibung

Der Funktionsbaustein SF_ClockGenerator stellt einen in Periodenlänge und Tastung einstellbaren Implulsgenerator bereit. Er ermöglicht z.B. eine einfache Blinkfunktion auf einem Hardware-Ausgang.

Enable

Wenn "Enable" auf TRUE geht, werden mit der Aufwärtsflanke die Werte an den Eingängen "Period", "Duty" und "FinishCycle" übernommen. Änderungen der Werte an diesen Eingängen werden ignoriert solange "Enable" TRUE ist. Wenn "Enable" auf FALSE geht und "FinishCycle" FALSE ist, wird der Zyklus sofort beendet.

Period

Die minimale Periodendauer muss größer gleich 100 ms und ungleich 0 ms sein. Die maximale Zykluszeit wird auf 1 Stunde festgelegt. Die Genauigkeit der Pulsbreite wird durch die Safety-Zykluszeit bestimmt. Lässt sich die Periodendauer ohne Rest durch die Safety Taskzeit teilen, so ist die Periode genau. Bleibt bei der Division ein Rest, ist die Periodendauer (Period / Safety Taskzeit) + SafetyTaskzeit.

FinishCycle

Wenn "FinishCycle" TRUE ist, werden Aufwärtsflanken an "Enable" ignoriert bis der Zyklus abgelaufen ist. Aufwärtsflanken an "Enable" werden nicht gespeichert. Pulse: Die Pulsbreite ist immer ein vielfaches der Safety-Zykluszeit und größer als der berechnete Wert aus Period / 100 * Duty.

Beispiel:

Bei einer Periode von 100ms, einem Duty-Wert von 35 % und einer Safety-Zykluszeit von 10ms ist die Pulsbreite am Ausgang "Pulse" 40 ms.

Genauigkeit Pulse-Ausgang

Die Genauigkeit des "Pulse"-Ausgangs wird durch die Genauigkeit der Safety-Task bestimmt. Diese ist mit einer Ungenauigkeit von maximal ±10 µs behaftet. Bei einer Safety-Zykluszeit von 10 ms entspricht dies einer Abweichung von 1 ‰ und ist somit vernachlässigbar klein. Es gilt, je größer die Zykluszeit desto geringer die prozentuale Ungenauigkeit. Die Fehler werden nicht aufsummiert. Die Berechnung des "Pulse"-Ausgangs erfolgt im Zyklus der Safety Applikation und kann maximal einen Safety-Zyklus abweichen. Das bedeutet, je kleiner die Safety-Zykluszeit und je größer die eingestellte Periodendauer ist, umso geringer ist die prozentuale Abweichung von Soll- zur Istzeit. Diese Betrachtung gilt für die Systemgrenzen des Safety-Funktionsmoduls. Für die Berechnung der Abweichung kann nachfolgende Formel herangezogen werden:

Anmerkung: Gerechnet wird mit Integer Variablen, d.h. ohne Nachkommastellen.

$T_{PulseRes}$	Resultierende Pulsbreite
T_{Pulse}	Parametrierte Pulsbreite
T_{Task}	Safety-Zykluszeit
T_{OnRes}	Resultierende Zeit für den High-Pegel des Pulses
Duty	High-Teil des Pulses in %

Berechnung $T_{PulseRes}$:

Wenn die Division T_{Pulse} / T_{Task} einen Rest liefert, gilt:

$T_{PulseRes} = (T_{Pulse} / T_{Task} + 1)* T_{Task}$

Wenn die Division T_{Pulse} / T_{Task} keinen Rest liefert, gilt:

$T_{PulseRes} = T_{Pulse}$

Berechnung T_{OnRes}:

Wenn die Berechnung (T_{Pulse}*Duty/100)/ T_{Task} einen Rest liefert, gilt:

T_{OnRes} = ((T_{Pulse}*Duty/100) / T_{Task} + 1)* T_{Task}

Wenn die Berechnung (T_{Pulse}*Duty/100)/ T_{Task} keinen Rest liefert, gilt:

T_{OnRes} = T_{Pulse}*Duty/100

Wird das Pulse-Signal auf einen Ausgang im Applikationsprogramm ausgegeben, wird die Ungenauigkeit zusätzlich durch unterschiedliche Zeiten für das Schalten des Ausgangs beeinflusst. Diese Zeiten sind der jeweiligen Anwenderbeschreibung der verwendeten Hardwarekomponente zu entnehmen.

Fehlerbehandlung

Error:

- **TRUE:** Es ist ein Fehler aufgetreten
- **FALSE:** Es ist kein Fehler aufgetreten

DiagCode

DiagCode	Beschreibung
16#0000	Der Baustein ist nicht initialisiert. Dieser Wert wird auch am Ausgang DiagCode ausgegeben, wenn Enable auf FALSE geht und der Zyklus abgelaufen ist
16#8000	Der Baustein ist fehlerfrei initialisiert und am Ausgang Pulse wird das parametrierte Signal ausgegeben.
16#C001	Die angegebene Zykluszeit ist 0.
16#C002	Die angegebene Zykluszeit ist kleiner 100ms.
16#C003	Die angegebene Zykluszeit ist größer 1 Stunde.
16#C004	Der angegeben Wert in der Variablen Duty ist < 1%.
16#C005	Der angegeben Wert in der Variablen Duty ist > 99%.

Tab. 16-85: Werte des Ausgangs DiagCode

16.3.17 BinaryDecoder_4_16

Kurzbeschreibung

Der Funktionsbaustein BinaryDecoder_4_16 implementiert einen kaskadierbaren 4zu16 Bit-Decoder für BOOL-Werte.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.4.0.0

Tab. 16-86: Bibliothekszuordnung

Schnittstellenbeschreibung

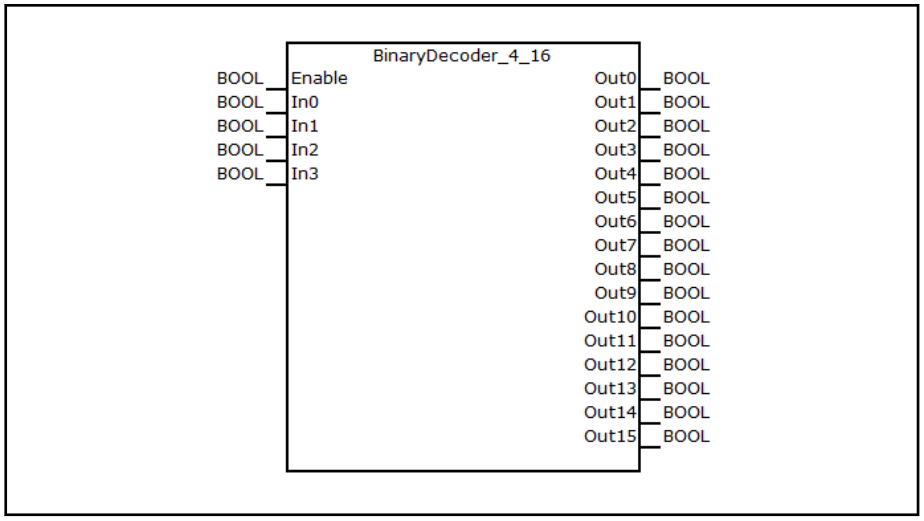


Abb. 16-34: Funktionsbaustein BinaryDecoder_4_16

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Enable	BOOL	Freigabe der Ausgänge TRUE: Out0 - Out15 sind gültig FALSE: Out0 - Out15 sind FALSE
	In0	BOOL	Eingang 0
	In1	BOOL	Eingang 1
	In2	BOOL	Eingang 2
	In3	BOOL	Eingang 3
VAR_OUTPUT	Out0	BOOL	Dekoder-Ausgang 0
	Out1	BOOL	Dekoder-Ausgang 1
	Out2	BOOL	Dekoder-Ausgang 2
	Out3	BOOL	Dekoder-Ausgang 3
	Out4	BOOL	Dekoder-Ausgang 4
	Out5	BOOL	Dekoder-Ausgang 5
	Out6	BOOL	Dekoder-Ausgang 6
	Out7	BOOL	Dekoder-Ausgang 7
	Out8	BOOL	Dekoder-Ausgang 8
	Out9	BOOL	Dekoder-Ausgang 9
	Out10	BOOL	Dekoder-Ausgang 10
	Out11	BOOL	Dekoder-Ausgang 11
	Out12	BOOL	Dekoder-Ausgang 12
	Out13	BOOL	Dekoder-Ausgang 13
	Out14	BOOL	Dekoder-Ausgang 14
	Out15	BOOL	Dekoder-Ausgang 15

Tab. 16-87: Schnittstelle BinaryDecoder_4_16

Name	Typ	Default-Wert	Übernahme
Enable	BOOL	FALSE	kontinuierlich
In0	BOOL	FALSE	kontinuierlich
In1	BOOL	FALSE	kontinuierlich
In2	BOOL	FALSE	kontinuierlich
In3	BOOL	FALSE	kontinuierlich

Tab. 16-88: Eingangsverhalten BinaryDecoder_4_16

Funktionsbeschreibung

Der Funktionsbaustein BinaryDecoder_16_4 implementiert einen kaskadierbaren 4 zu 16 Bit Dekoder für BOOL-Werte.

In3	In2	In1	In0	Enable	Out15	Out14	Out13	Out12	Out11	Out10	Out9	Out8	Out7	Out6	Out5	Out4	Out3	Out2	Out1	Out0
X	X	X	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	1	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0 FALSE
1 TRUE
X TRUE oder FALSE

Tab. 16-89: Logiktabelle des Funktionsbausteins BinaryDecoder_4_16

Enable

Wenn Enable TRUE ist, werden die Eingänge In0 bis In3 auf die Ausgänge Out0 bis Out15 dekodiert. Ist Enable FALSE, sind Out0 bis Out15 FALSE

In0 bis In3

Die Eingänge In0 bis In3 werden durch den Dekoder auf die Ausgänge Out0 bis Out15 umgesetzt.

Out0 bis Out15

Die Ausgänge Out0 bis Out15 enthalten die dekodierten Werte von In0 bis In3. Ist Enable FALSE, so sind Out0 bis Out15 immer FALSE.

Fehlerbehandlung Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.18 BinaryEncoder_16_4

Kurzbeschreibung Der Funktionsbaustein BinaryEncoder_16_4 implementiert einen kaskadierbaren 16 auf 4 Bit Encoder für BOOL-Werte.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.4.0.0

Tab. 16-90: Bibliothekszuordnung

Schnittstellenbeschreibung

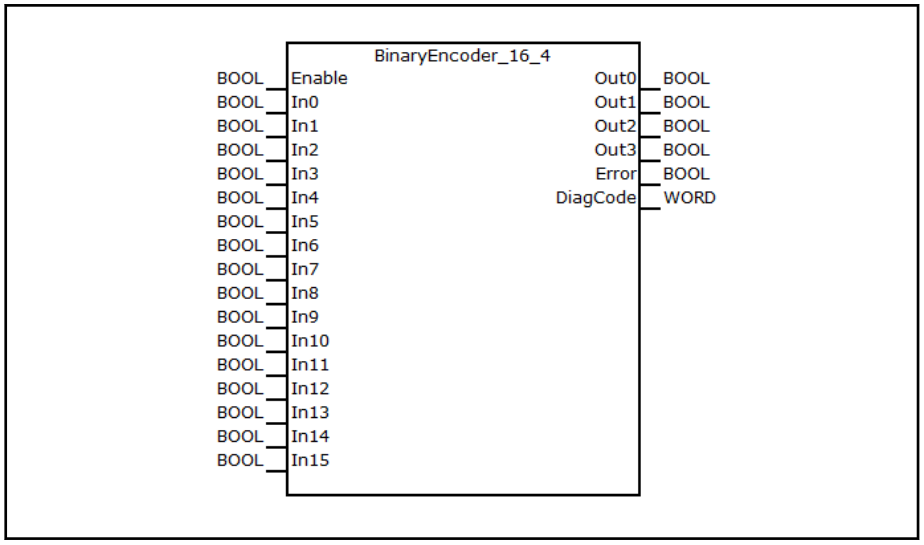


Abb. 16-35: Funktionsbaustein BinaryEncoder_16_4

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Enable	BOOL	Freigabe der Ausgänge TRUE: Out0 - Out3 sind gültig FALSE: Out0 - Out3 sind FALSE
	In0	BOOL	Eingang 0
	In1	BOOL	Eingang 1
	In2	BOOL	Eingang 2
	In3	BOOL	Eingang 3
	In4	BOOL	Eingang 4
	In5	BOOL	Eingang 5
	In6	BOOL	Eingang 6
	In7	BOOL	Eingang 7
	In8	BOOL	Eingang 8
	In9	BOOL	Eingang 9
	In10	BOOL	Eingang 10
	In11	BOOL	Eingang 11
	In12	BOOL	Eingang 12
	In13	BOOL	Eingang 13

E/A-Typ	Name	Datentyp	Beschreibung
	In14	BOOL	Eingang 14
	In15	BOOL	Eingang 15
VAR_OUTPUT	Out0	BOOL	Gemultiplexter Ausgang 0
	Out1	BOOL	Gemultiplexter Ausgang 1
	Out2	BOOL	Gemultiplexter Ausgang 2
	Out3	BOOL	Gemultiplexter Ausgang 3
	Error	BOOL	Fehlerstatus
	DiagCode	WORD	Fehlercode

Tab. 16-91: Schnittstelle BinaryEncoder_16_4

Name	Typ	Default-Wert	Übernahme
Enable	BOOL	FALSE	kontinuierlich
In0	BOOL	FALSE	kontinuierlich
In1	BOOL	FALSE	kontinuierlich
In2	BOOL	FALSE	kontinuierlich
In3	BOOL	FALSE	kontinuierlich
In4	BOOL	FALSE	kontinuierlich
In5	BOOL	FALSE	kontinuierlich
In6	BOOL	FALSE	kontinuierlich
In7	BOOL	FALSE	kontinuierlich
In8	BOOL	FALSE	kontinuierlich
In9	BOOL	FALSE	kontinuierlich
In10	BOOL	FALSE	kontinuierlich
In11	BOOL	FALSE	kontinuierlich
In12	BOOL	FALSE	kontinuierlich
In13	BOOL	FALSE	kontinuierlich
In14	BOOL	FALSE	kontinuierlich
In15	BOOL	FALSE	kontinuierlich

Tab. 16-92: Eingangsverhalten BinaryEncoder_16_4

Funktionsbeschreibung

Der Funktionsbaustein BinaryEncoder_16_4 implementiert einen kaskadierbaren 16 auf 4 Bit Multiplexer für BOOL-Werte.

In15	In14	In13	In12	In11	In10	In9	In8	In7	In6	In5	In4	In3	In2	In1	In0	Enable	Out3	Out2	Out1	Out0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	1	0	0	0	0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	0	1	0	0	0	1

In15	In14	In13	In12	In11	In10	In9	In8	In7	In6	In5	In4	In3	In2	In1	In0	Enable	Out3	Out2	Out1	Out0
X	X	X	X	X	X	X	X	X	X	X	X	X	1	0	0	1	0	0	1	0
X	X	X	X	X	X	X	X	X	X	X	X	1	0	0	0	1	0	0	1	1
X	X	X	X	X	X	X	X	X	X	X	1	0	0	0	0	1	0	1	0	0
X	X	X	X	X	X	X	X	X	X	1	0	0	0	0	0	1	0	1	0	1
X	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	1	0	1	1	0
X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	1	0	1	1	1
X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	1	1	0	0	0
X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	1	1	0	0	1
X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0
X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1
X	X	X	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
X	X	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1
X	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

0 FALSE
1 TRUE
X TRUE oder FALSE

Tab. 16-93: Logiktabelle des Funktionsbausteins BinaryEncoder_16_4

Enable

Wenn Enable TRUE ist, werden die Eingänge In0 bis In15 auf die Ausgänge Out0 bis Out3 gemultiplext. Ist Enable FALSE, sind Out0 bis Out3 FALSE

In0 bis In15

Die Eingänge In0 bis In15 werden ihrer Numer nach aufsteigend priorisiert verarbeitet. Der erste Eingang, der TRUE ist, wird verwendet.

Out0 bis Out3

Die Ausgänge Out0 bis Out3 enthalten die gemultiplexten Werte von In0 bis In15. Ist Enable FALSE, so sind Out0 bis Out3 immer FALSE.

Fehlerbehandlung Der Funktionsbaustein kann die folgenden DiagCodes ausgeben:

DiagCode	Beschreibung
16#C001	Input Enable ist TRUE aber kein Eingang InX hat den Wert TRUE

Tab. 16-94: DiagCodes des Funktionsbausteins BinaryEncoder_16_4

16.3.19 SF_BinaryDecoder_4_16

Kurzbeschreibung Der Funktionsbaustein SF_BinaryDecoder_4_16 implementiert einen kaskadierbaren 4 zu 16 Bit Binär-Dekoder für SAFEBOOL-Werte.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.4.0.0

Tab. 16-95: Bibliothekszuordnung

Schnittstellenbeschreibung

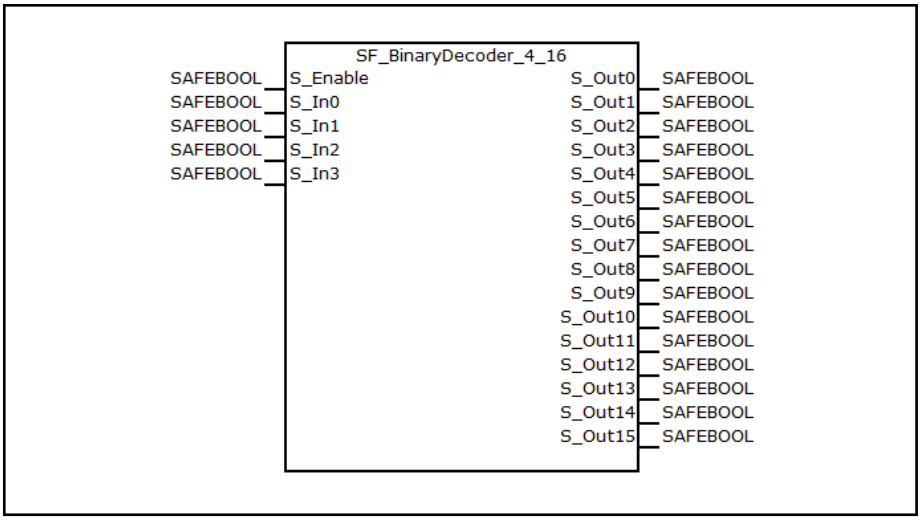


Abb. 16-36: Funktionsbaustein SF_BinaryDecoder_4_16

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	S_Enable	SAFEBOOL	Freigabe der Ausgänge TRUE: Out0 - Out15 sind gültig FALSE: Out0 - Out15 sind FALSE
	S_In0	SAFEBOOL	Eingang 0
	S_In1	SAFEBOOL	Eingang 1
	S_In2	SAFEBOOL	Eingang 2
	S_In3	SAFEBOOL	Eingang 3
VAR_OUTPUT	S_Out0	SAFEBOOL	Dekodierter Ausgang 0
	S_Out1	SAFEBOOL	Dekodierter Ausgang 1
	S_Out2	SAFEBOOL	Dekodierter Ausgang 2
	S_Out3	SAFEBOOL	Dekodierter Ausgang 3
	S_Out4	SAFEBOOL	Dekodierter Ausgang 4
	S_Out5	SAFEBOOL	Dekodierter Ausgang 5
	S_Out6	SAFEBOOL	Dekodierter Ausgang 6
	S_Out7	SAFEBOOL	Dekodierter Ausgang 7
	S_Out8	SAFEBOOL	Dekodierter Ausgang 8
	S_Out9	SAFEBOOL	Dekodierter Ausgang 9
	S_Out10	SAFEBOOL	Dekodierter Ausgang 10
	S_Out11	SAFEBOOL	Dekodierter Ausgang 11
	S_Out12	SAFEBOOL	Dekodierter Ausgang 12
	S_Out13	SAFEBOOL	Dekodierter Ausgang 13
	S_Out14	SAFEBOOL	Dekodierter Ausgang 14
	S_Out15	SAFEBOOL	Dekodierter Ausgang 15

Tab. 16-96: Schnittstelle SF_BinaryDecoder_4_16

Name	Typ	Default-Wert	Übernahme
S_Enable	SAFEBOOL	FALSE	kontinuierlich
S_In0	SAFEBOOL	FALSE	kontinuierlich
S_In1	SAFEBOOL	FALSE	kontinuierlich
S_In2	SAFEBOOL	FALSE	kontinuierlich
S_In3	SAFEBOOL	FALSE	kontinuierlich

Tab. 16-97: Eingangsverhalten SF_BinaryDecoder_4_16

Funktionsbeschreibung

Der Funktionsbaustein SF_BinaryDecoder_4_16 implementiert einen kaskadierbaren 4 zu 16 Bit Decoder für SAFEBOOL-Werte.

S_In3	S_In2	S_In1	S_In0	Enable	S_Out15	S_Out14	S_Out13	S_Out12	S_Out11	S_Out10	S_Out9	S_Out8	S_Out7	S_Out6	S_Out5	S_Out4	S_Out3	S_Out2	S_Out1	S_Out0
X	X	X	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	1	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0 FALSE
1 TRUE
X TRUE oder FALSE

Tab. 16-98: Logiktabelle des Funktionsbausteins SF_BinaryDecoder_4_16

S_Enable

Wenn Enable TRUE ist, werden die Eingänge S_In0 bis S_In3 auf die Ausgänge S_Out0 bis S_Out15 binär dekodiert. Ist Enable FALSE, sind S_Out0 bis S_Out15 FALSE

S_In0 bis S_In3

Die Eingänge S_In0 bis S_In3 werden durch den Binär-Dekoder auf die Ausgänge S_Out0 bis S_Out15 umgesetzt.

S_Out0 bis S_Out15

Die Ausgänge S_Out0 bis S_Out15 enthalten die binär dekodierten Werte von S_In0 bis S_In3. Ist Enable FALSE, so sind S_Out0 bis S_Out15 immer FALSE.

Fehlerbehandlung Dieser Funktionsbaustein verfügt über keine Fehlerbehandlung.

16.3.20 SF_BinaryEncoder_16_4

Kurzbeschreibung Der Funktionsbaustein SF_BinaryEncoder_16_44 implementiert einen kaskadierbaren 16 auf 4 Bit Binär-Kodierer für SAFEBOOL-Werte.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyUtilities	2.4.0.0

Tab. 16-99: Bibliothekszuordnung

Schnittstellenbeschreibung

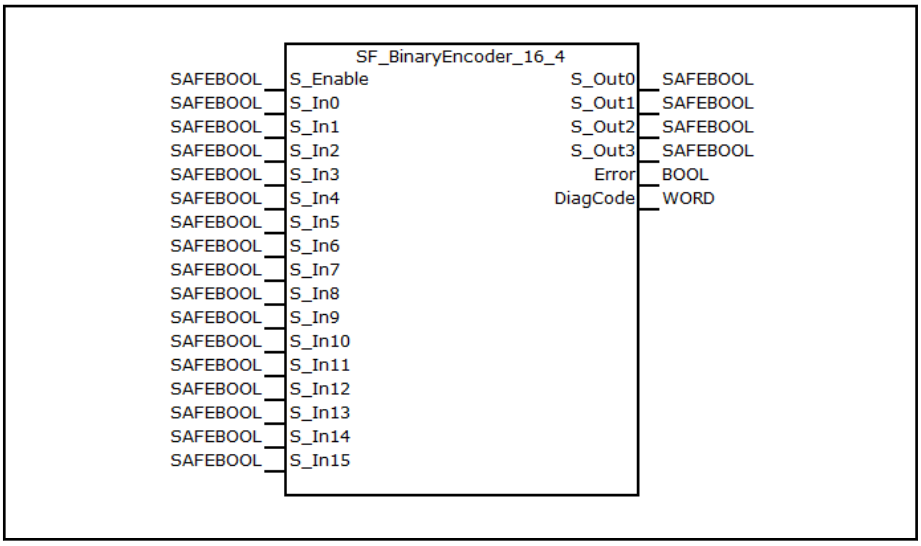


Abb. 16-37: Funktionsbaustein SF_BinaryEncoder_16_4

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	S_Enable	SAFEBOOL	Freigabe der Ausgänge TRUE: Out0 - Out3 sind gültig FALSE: Out0 - Out3 sind FALSE
	S_In0	SAFEBOOL	Eingang 0
	S_In1	SAFEBOOL	Eingang 1
	S_In2	SAFEBOOL	Eingang 2
	S_In3	SAFEBOOL	Eingang 3
	S_In4	SAFEBOOL	Eingang 4
	S_In5	SAFEBOOL	Eingang 5
	S_In6	SAFEBOOL	Eingang 6
	S_In7	SAFEBOOL	Eingang 7
	S_In8	SAFEBOOL	Eingang 8
	S_In9	SAFEBOOL	Eingang 9
	S_In10	SAFEBOOL	Eingang 10
	S_In11	SAFEBOOL	Eingang 11

E/A-Typ	Name	Datentyp	Beschreibung
	S_In12	SAFEBOOL	Eingang 12
	S_In13	SAFEBOOL	Eingang 13
	S_In14	SAFEBOOL	Eingang 14
	S_In15	SAFEBOOL	Eingang 15
VAR_OUTPUT	S_Out0	SAFEBOOL	Gemultiplexter Ausgang 0
	S_Out1	SAFEBOOL	Gemultiplexter Ausgang 1
	S_Out2	SAFEBOOL	Gemultiplexter Ausgang 2
	S_Out3	SAFEBOOL	Gemultiplexter Ausgang 3
	Error	BOOL	Fehlerstatus
	DiagCode	WORD	Fehlercode

Tab. 16-100: Schnittstelle SF_BinaryEncoder_16_4

Name	Typ	Default-Wert	Übernahme
S_Enable	SAFEBOOL	FALSE	kontinuierlich
S_In0	SAFEBOOL	FALSE	kontinuierlich
S_In1	SAFEBOOL	FALSE	kontinuierlich
S_In2	SAFEBOOL	FALSE	kontinuierlich
S_In3	SAFEBOOL	FALSE	kontinuierlich
S_In4	SAFEBOOL	FALSE	kontinuierlich
S_In5	SAFEBOOL	FALSE	kontinuierlich
S_In6	SAFEBOOL	FALSE	kontinuierlich
S_In7	SAFEBOOL	FALSE	kontinuierlich
S_In8	SAFEBOOL	FALSE	kontinuierlich
S_In9	SAFEBOOL	FALSE	kontinuierlich
S_In10	SAFEBOOL	FALSE	kontinuierlich
S_In11	SAFEBOOL	FALSE	kontinuierlich
S_In12	SAFEBOOL	FALSE	kontinuierlich
S_In13	SAFEBOOL	FALSE	kontinuierlich
S_In14	SAFEBOOL	FALSE	kontinuierlich
S_In15	SAFEBOOL	FALSE	kontinuierlich

Tab. 16-101: Eingangsverhalten SF_BinaryEncoder_16_4

Funktionsbeschreibung

Der Funktionsbaustein BinaryEncoder_16_4 implementiert einen kaskadierbaren 16 auf 4 Bit Binär-Kodierer für SAFEBOOL-Werte.

S_In 15	S_In 14	S_In 13	S_In 12	S_In 11	S_In 10	S_I n9	S_I n8	S_I n7	S_I n6	S_I n5	S_I n4	S_I n3	S_I n2	S_I n1	S_I n0	Enable	S_O ut3	S_O ut2	S_O ut1	S_O ut0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

S_In 15	S_In 14	S_In 13	S_In 12	S_In 11	S_In 10	S_In 9	S_In 8	S_In 7	S_In 6	S_In 5	S_In 4	S_In 3	S_In 2	S_In 1	S_In 0	Enable	S_Out 3	S_Out 2	S_Out 1	S_Out 0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	1	0	0	0	0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	0	1	0	0	0	1
X	X	X	X	X	X	X	X	X	X	X	X	X	1	0	0	1	0	0	1	0
X	X	X	X	X	X	X	X	X	X	X	X	1	0	0	0	1	0	0	1	1
X	X	X	X	X	X	X	X	X	X	X	1	0	0	0	0	1	0	1	0	0
X	X	X	X	X	X	X	X	X	X	1	0	0	0	0	0	1	0	1	0	1
X	X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	1	0	1	1	0
X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	1	0	1	1	1
X	X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	1	1	0	0	0
X	X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	1	1	0	0	1
X	X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0
X	X	X	X	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1
X	X	X	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
X	X	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1
X	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

0 FALSE
1 TRUE
X TRUE oder FALSE

Tab. 16-102: Logiktabelle des Funktionsbausteins BinaryEncoder_16_4

S_Enable

Wenn Enable TRUE ist, werden die Eingänge S_In0 bis S_In15 auf die Ausgänge S_Out0 bis S_Out3 binär kodiert. Ist Enable FALSE, sind S_Out0 bis S_Out3 FALSE

S_In0 bis S_In15

Die Eingänge S_In0 bis S_In15 werden ihrer Numer nach aufsteigend priorisiert verarbeitet. Der erste Eingang, der TRUE ist, wird verwendet.

S_Out0 bis S_Out3

Die Ausgänge S_Out0 bis S_Out3 enthalten die binär kodierten Werte von S_In0 bis S_In15. Ist Enable FALSE, so sind S_Out0 bis S_Out3 immer FALSE.

Fehlerbehandlung Der Funktionsbaustein kann die folgenden DiagCodes ausgeben:

DiagCode	Beschreibung
16#C001	Input Enable ist TRUE aber kein Eingang InX hat den Wert TRUE

Tab. 16-103: DiagCodes des Funktionsbausteins SF_BinaryEncoder_16_4

16.4 SafetyPLCopen.library

Die PLCopen-Bausteine (mit Präfix SF_) wurden von der Organisation PLCopen zusammen mit seinen Mitgliedern und externen Organisationen, die sich mit sicherheitsgerichteten Aspekten beschäftigen, definiert.

Diese Bausteine sind zertifiziert und reduzieren dadurch Zeit und Kosten bei der Entwicklung, Verifikation und Abnahme einer Sicherheitsapplikation. Ähnlich einer logischen Verdrahtung können die Bausteine durch logische Operationen miteinander verknüpft werden. Somit können wichtige Bestandteile einer Sicherheitsapplikation schnell und ohne großen Programmieraufwand erstellt werden.

Für Informationen, welche die einzelnen PLCopen-Bausteine noch ausführlicher beschreiben wird auf die zum Zeitpunkt der Zertifizierung aktuelle Spezifikationsversion der PLCopen - Technical Committee 5 - Safety Software verwiesen.

Anwendungsgebiete und Funktionsbausteine von SafetyPLCopen

Funktionsbausteine	Beschreibung
SF_Antivalent, Seite 300	verknüpft zwei antivalente SAFEBOOL-Eingänge (NO/NC Paar) zu einem SAFEBOOL-Ausgang unter Diskrepanzzeit-Überwachung
SF_EDM, Seite 306	überwacht, dass zwei Eingänge innerhalb einer definierten Zeit (MonitoringTime) abschalten
SF_EmergencyStop, Seite 314	überwacht NOT-HALT-Taster
SF_EnableSwitch, Seite 320	wertet die Signale eines handbetätigten, dreistufigen Zustimmungsschalters zur Unterstützung der Aufhebung einer Schutzmaßnahme aus
SF_Equivalent, Seite 327	verknüpft zwei äquivalente SAFEBOOL-Eingänge (beide NO oder NC) zu einem SAFEBOOL-Ausgang unter Diskrepanzzeit-Überwachung
SF_ESPE, Seite 333	überwacht eine berührungslos wirkenden Schutzeinrichtung
SF_GuardLocking, Seite 335	kontrolliert den Eingang zu einem Gefahrenbereich durch eine verriegelte Schutzeinrichtung mit Zuhaltung (vierstufige Verriegelung)
SF_GuardMonitoring, Seite 341	überwacht die relevante Sicherheits-Schutzeinrichtung mit einer zweistufigen Verriegelung
SF_ModeSelector, Seite 347	wählt die System-Betriebsart
SF_MutingPar, Seite 356	beabsichtigtes Unterdrücken einer Sicherheitsfunktion, paralleles Muting mit vier Mutingsensoren
SF_MutingPar_2Sensor, Seite 370	beabsichtigtes Unterdrücken einer Sicherheitsfunktion, paralleles Muting mit zwei Mutingsensoren
SF_MutingSeq, Seite 379	beabsichtigtes Unterdrücken einer Sicherheitsfunktion, sequentielles Muting mit vier Mutingsensoren
SF_OutControl, Seite 389	steuert einen Sicherheits-Ausgang mit einem Signal der Standardsteuerung und einem sicheren Signal mit optionaler Anlaufsperrre
SF_SafetyRequest, Seite 395	Schnittstelle zwischen dem Benutzerprogramm und der Systemumgebung
SF_TestableSafetySensor, Seite 401	kontrolliert die Funktion einer elektronischen/optischen Schutzeinrichtung
SF_TwoHandControlTypeII, Seite 412	unterstützt in einer Applikation die Funktion "Zweihandschaltung Typ 2"
SF_TwoHandControlTypeIII, Seite 417	unterstützt in einer Applikation die Funktion "Zweihandschaltung Typ 3"

Tab. 16-104: SafetyPLCopen-Bibliothek

16.4.1 Allgemeine Regeln für sicherheitsgerichtete Funktionsbausteine

Diese Programmierregeln entsprechen den "General Rules for Safety-Related Function Blocks" der PLCopen. Sie gelten für PLCopen-konforme Funktionsbausteine.

Funktionsbausteinspezifische Regeln

Default-Signal	Alle sicherheitsgerichteten Booleschen E/A-Signale haben den voreingestellten sicheren Wert "FALSE"
Signallevel	Der Wert SAFEBOOL ist nur wie folgt anwendbar: = 0 entspricht der Sicherheit, wie bei Systemausgängen definiert =1 bedeutet, dass die Sicherheitsaspekte des Systems korrekt arbeiten, z. B., normaler Betrieb ist möglich. Dies reflektiert die Funktionalität der IEC 61131-Umgebungen, wie beispielsweise die Regeln für den Standardwert und dass alle Ausgänge bei einem Fehlerfall auf "0" gesetzt werden.
Ausgänge	Jeder Ausgang muss in jedem Zyklus zugewiesen werden
Fehlende E/A-Parameter	Fehlende Parameter sind zulässig. Default-Werte gelten. Diese Default-Werte sollen unter keinen Umständen in einen nicht sicheren Zustand führen. Die Default-Werte einschließlich ihrer Attribute (Variable oder Constant) werden in den entsprechenden FBs spezifiziert.
Anlaufverhalten	Anfangs werden die Ausgänge auf die Default-Werte gesetzt. Nach dem ersten Aufruf des FBs, sind die Ausgänge gültig. Es gibt ein konsistentes Anlaufverhalten (Kaltstart).
Zeitdiagramme	Zeitdiagramme, wie sie bei den FBs gezeigt werden, dienen nur zur Erklärung. Sie stellen nicht das exakte Zeitverhalten dar. Das exakte Zeitverhalten hängt von der Implementierung ab.
Fehlerbehandlung und Diagnose	Alle sicherheitsgerichteten Funktionsbausteine haben zwei fehlerbezogene Ausgänge: Error und DiagCode. Sie dienen Diagnosezwecken auf Benutzeranwendungsebene, nicht zur Diagnose auf System- oder Hardeware-Ebene. Die Vorschrift für sicherheitsgerichtete Umgebungen ist, dass die Schaltung der sicherheitsgerichteten Funktion die höchste Priorität hat und dass es in der Folgeschaltung genügend Zeit zur Diagnose gibt, entweder im funktionalen Programm oder in der Bedienschnittstelle.

Allgemeine Eingangsparameter Eine einheitliche Namensgebung bei den Eingängen, sowie gleiches Verhalten dieser Standardschnittstellen erhöht das Verständnis für die Funktion der Bausteine, verkürzt die Einarbeitung und entlastet den Support.

Name	Datentyp	Beschreibung
Activate	BOOL	Variable oder Konstante. Aktivierung eines Bausteins. Initialwert ist FALSE. Dieser Parameter kann mit der Variablen verbunden werden, die den Zustand (aktiv oder nicht aktiv) des relevanten Sicherheitsgeräts darstellt. Dies gewährleistet, dass keine irrelevanten Diagnosedaten erzeugt werden, wenn das Gerät deaktiviert wird. FALSE: alle Ausgangsvariablen werden auf den Initialwert gesetzt. Wenn kein Gerät verbunden ist, muss ein statisches TRUE-Signal zugewiesen werden. Ist ein Gerät verbunden, so muss der FV_activated-Ausgang der logischen Überwachungsinstanz des Geräts negiert angeschlossen werden.
S_<sicherheitsgerichteter Eingangs-Name>	SAFExxxx	Jeder Name eines Eingangs eines SAFExxxx-Typs beginnt mit S_. Nur Variable dürfen zugewiesen werden.
S_StartReset	SAFEBOOL	Variable oder Konstante. FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt). TRUE: Automatischer Reset, wenn die S-SPS gestartet wird (warm oder kalt). Zu beachten sind "Sicherheitshinweise für S_Start-Reset" auf Seite 295.

Name	Datentyp	Beschreibung
S_AutoReset	SAFEBOOL	Variable oder Konstante. FALSE (= Initialwert): Manueller Reset wenn NOT-HALT-Knopf entriegelt ist. TRUE: Automatischer Reset, wenn NOT-HALT-Knopf entriegelt ist. Zu beachten sind "Sicherheitsheinweise für S_Auto-Reset" auf Seite 295.
Reset	BOOL	Variable. Initialwert ist FALSE *Abhängig von der Funktion, kann dieser Eingang für unterschiedliche Zwecke verwendet werden:* • Reset der Zustandsmaschine und gekoppelter Fehler- und Zustandsmeldungen wie über DiagCode angezeigt, wenn die Fehlerursache behoben wurde. Dieses Reset-Verhalten ist als Fehler-Reset konzipiert. • Manueller Reset einer Wiederanlaufsperre durch den Operator. Dieser Reset ist als funktionaler Reset konzipiert. • Zusätzliche bausteinspezifische Reset-Funktionen. Diese Funktion ist nur bei einem Signalwechsel von FALSE nach TRUE aktiv. Ein statisches TRUE-Signal erzeugt keine weiteren Aktionen, kann aber in einigen Bausteinen als Fehler entdeckt werden. Zu beachten sind "Sicherheitshinweis für Reset" auf Seite 296. Die angemessene Bedeutung ist bei jedem Baustein beschrieben.

Sicherheitshinweise für S_Start-Reset

⚠ VORSICHT **S_StartReset**

Diese Funktion soll nur aktiviert werden, wenn gewährleistet ist, dass keine Gefährdung beim Start der S-SPS entstehen kann.

Deshalb erfordert der Gebrauch des Features Automatischer-System-Reset eines Bausteins die Implementierung anderer System- oder Applikationsmaßnahmen, um sicherzustellen, dass kein unerwartetes (oder unbeabsichtigtes) Starten auftritt.

⚠ VORSICHT

Wird der Eingang mit einer Variablen (und nicht mit FALSE) verknüpft, so muss dieser Teil der Sicherheitsapplikation validiert werden.

Sicherheitsheinweise für S_Auto-Reset

⚠ VORSICHT **S_Auto-Reset**

Diese Funktion soll nur aktiviert werden, wenn gewährleistet ist, dass keine Gefährdung beim Start der S-SPS entstehen kann.

Deshalb erfordert der Gebrauch des Features Automatischer-System-Reset eines Bausteins die Implementierung anderer System- oder Applikationsmaßnahmen, um sicherzustellen, dass kein unerwartetes (oder unbeabsichtigtes) Starten auftritt.

⚠ VORSICHT **Wird der Eingang mit einer Variablen (und nicht mit FALSE) verknüpft, so muss dieser Teil der Sicherheitsapplikation validiert werden.**

Sicherheitshinweis für Reset

⚠ VORSICHT **Reset**

Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden.

Allgemeine Ausgangsparameter

Eine einheitliche Namensgebung bei den Ausgängen, sowie gleiches Verhalten dieser Standardschnittstellen erhöht das Verständnis für die Funktion der Bausteine, verkürzt die Einarbeitung und entlastet den Support.

Name	Datentyp	Beschreibung
Ready	BOOL	TRUE: Zeigt an, dass der Baustein aktiviert und die Ausgangs-Ergebnisse gültig sind (gleich wie die "POWER"-LED eines Sicherheits-Relais). FALSE: Der Baustein ist nicht aktiv und das Programm wird nicht ausgeführt. Nützlich im Debug-Modus oder um zusätzliche Bausteine zu aktivieren/deaktivieren. Ebenso für die Weiterverarbeitung im funktionalen Programm.
S_<sicherheitsgerichteter Ausgangs-Name>	SAFExxxx	Jeder Name eines SAFExxxx-Typ Ausgangs beginnt mit S_.
Error	BOOL	Fehler-Flag (gleich wie die "K1/K2"-LED eines Sicherheits-Relais.) TRUE: Zeigt an, dass ein Fehler aufgetreten ist und sich der Baustein im Fehlerzustand befindet. Der relevante Fehlerzustand wird am DiagCode-Ausgang angezeigt. FALSE: Es gibt keinen Fehler und der Baustein befindet sich in einem anderen Zustand. Dies wird wiederum durch den DiagCode angezeigt. Nützlich im Debug-Modus ebenso wie für die Weiterverarbeitung im funktionalen Programm.
DiagCode	WORD	Diagnoseregister. Alle Zustände eines Bausteins (Active, Not Active und Error) werden durch dieses Register abgebildet. Es wird immer nur ein konsistenter Code zur gleichen Zeit angezeigt. Im Falle von mehreren Fehlern zeigt der DiagCode-Ausgang den zuerst entdeckten Fehler an. Nützlich im Debug-Modus, ebenso wie für die Weiterverarbeitung im funktionalen Programm.

Diagnose-Codes

Ein transparentes und einheitliches Diagnosekonzept bildet die Basis für alle Bausteine. Dadurch wird gewährleistet, dass ohne Rücksicht auf die Implementierung des Anwenders, dem Endanwender in Form des DiagCode einheitliche Diagnose-Information zur Verfügung steht. Besteht kein Fehler, so wird der interne Zustand des Bausteins (Zustandsmaschine) dargestellt. Ein Fehler wird über einen binären Ausgang (error) angezeigt.

Detaillierte Informationen über interne oder externe Baustein-Fehler kann man über DiagCode erhalten.

Der Baustein muss über verschiedene Reset-Eingänge zurückgesetzt werden.

Allgemeine Diagnose-Code-Bereiche

DiagCode	Beschreibung
2#0000_0000_0000_0000	Der Baustein ist nicht aktiviert oder Sicherheits-CPU ist angehalten
2#10xx_xxxx_xxxx_xxxx	Zeigt an, dass sich der aktivierte Baustein im Betriebszustand ohne Fehler befindet. X = bausteinspezifischer Code
2#11xx_xxxx_xxxx_xxxx	Zeigt an, dass sich der aktivierte Baustein im Fehlerzustand befindet. X = bausteinspezifischer Code

System- oder gerätespezifische Codes

DiagCode	Beschreibung
2#0xxx_xxxx_xxxx_xxxx	X = System- oder gerätespezifische Meldung. Diese Information enthält Diagnose-Information des Systems oder des Gerätes. Anmerkung: 16#0000 ist reserviert.

Generische Diagnose-Codes

DiagCode	Beschreibung
2#0000_0000_0000_0000 16#0000	Der Baustein ist nicht aktiviert. Dieser Code stellt den Leerlauf-Zustand dar. Als allgemeines Beispiel könnte die E/A-Einstellung wie folgt sein: Activate = FALSE S_In = FALSE oder TRUE Ready = FALSE Error = FALSE S_Out = FALSE
2#1000_0000_0000_0000 16#8000	Der Baustein ist aktiviert, ohne Fehler oder eine andere Bedingung, die den Sicherheits-Ausgang auf FALSE setzt. Dies ist der Standard-Betriebszustand, in dem der Sicherheitsausgang S_Out bei Normalbetrieb TRUE ist. Als allgemeines Beispiel könnten die Ein- und Ausgänge wie folgt gesetzt sein: Activate = TRUE S_In = TRUE Ready = TRUE Error = FALSE S_Out = TRUE
2#1000_0000_0000_0001 16#8001	Eine Aktivierung wurde durch den Baustein entdeckt und der Baustein ist jetzt aktiviert, aber der S_Out-Sicherheitsausgang ist auf FALSE gesetzt. Dieser Code stellt den Init-Zustand des Betriebs-Modus an. Als allgemeines Beispiel könnten die Ein- und Ausgänge wie folgt gesetzt sein: Activate = TRUE S_In = FALSE oder TRUE Ready = TRUE Error = FALSE S_Out = FALSE

DiagCode	Beschreibung
2#1000_0000_0000_0010 16#8002	Der aktivierte Baustein entdeckt eine Sicherheitsanforderung, zum Beispiel S_In = FALSE. Der Sicherheitsausgang ist deaktiviert (S_Out = FALSE). Als allgemeines Beispiel könnten die Ein- und Ausgänge wie folgt gesetzt sein: Activate = TRUE S_IN = FALSE Ready = TRUE Error = FALSE S_Out = FALSE
2#1000_0000_0000_0011 16#8003	Der Sicherheitsausgang des aktivierten Baustein ist durch eine Sicherheitsanforderung deaktiviert worden. Die Sicherheitsanforderung ist jetzt zurückgenommen, aber der Sicherheitsausgang bleib solange FALSE, bis eine Reset-Bedingung entdeckt wird. Dies ist ein Betriebszustand, wo der Sicherheitsausgang S_Out = FALSE. Als allgemeines Beispiel könnten die Ein- und Ausgänge wie folgt gesetzt sein: Activate = TRUE S_In = FALSE => TRUE (mit statischem TRUE fortfahren) Ready = TRUE Error = FALSE S_Out = FALSE

Generisches Zustandsdiagramm

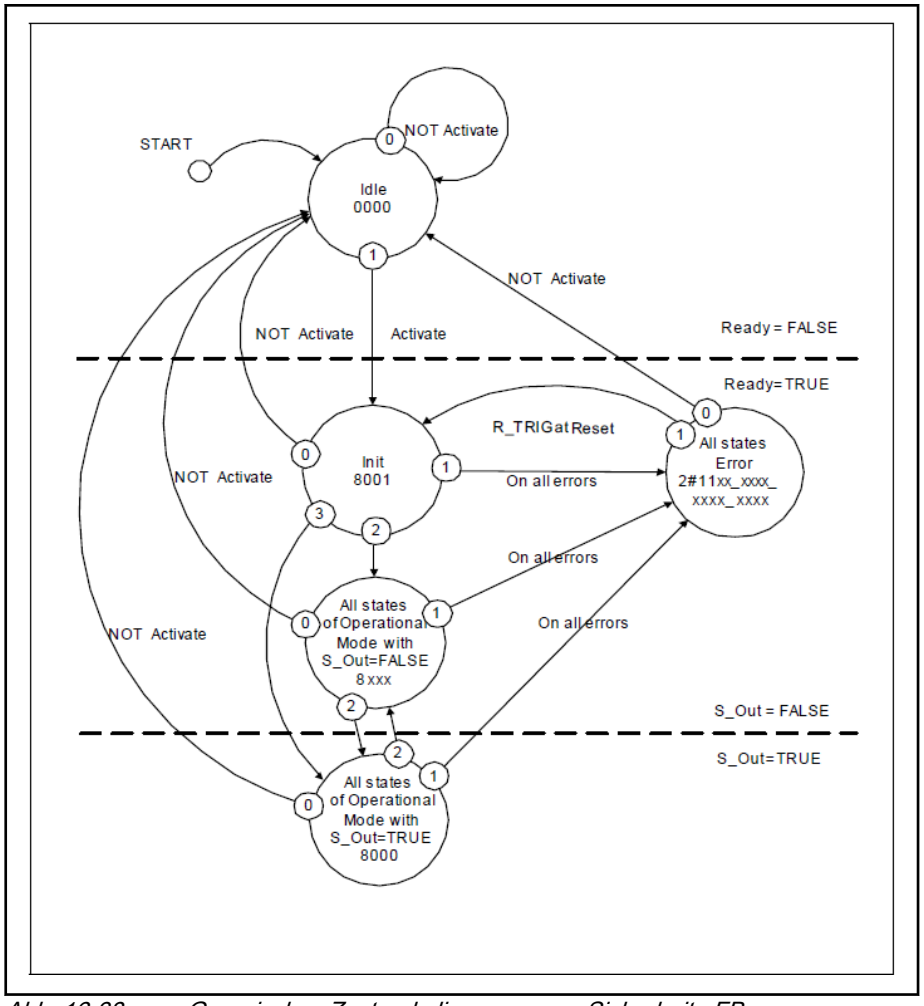


Abb. 16-38: Generisches Zustandsdiagramm von Sicherheits-FBs

Erklärung des generischen Zustandsdiagramms:

- Es wird ein genereller Überblick über die Zustände und Übergänge gezeigt. Einige Übergänge sind nicht benannt, dies bedeutet, dass sie FB-spezifisch sind und mit dem jeweiligen FB beschrieben werden müssen.
- *Das Diagramm zeigt drei Bereiche:*
  - Im oberen Bereich ist der FB nicht aktiv und ist im sicheren Zustand (sichere Ausgänge sind FALSE)
  - Im mittleren Bereich ist der FB aktiv und im sicheren Zustand (sichere Ausgänge sind FALSE)
  - Im unteren Bereich ist der FB in normalem Zustand, d. h. die sicheren Ausgänge sind TRUE
- Die erste horizontale Linie im Statusdiagramm zeigt den Übergang von einem nicht-aktiven FB zu einem aktiven FB
- Die zweite horizontale Linie zeigt den Übergang von einem nicht-sicheren zu einem sicheren Zustand
- Die Prioritäten von möglichen parallelen Übergängen werden durch Nummern angegeben (höchste Priorität ist 0)
- Die Zustände enthalten den Zustandsname und den hexadezimalen Diagnosecode

- Bedingungen "OR, AND, XOR" werden als logischen Operatoren und "NOT" als Negation verwendet.
- Innerhalb der FB-Beschreibung ist der Anfangszustand Idle, mit den Übergängen zu den einzelnen Betriebszuständen über den Zustand Init
- Activate = FALSE schaltet von jedem Zustand direkt in den Idle-Zustand (0 = höchste Priorität ist für Activate = FALSE reserviert)

  Zur besseren Übersicht werden diese Übergänge nicht in jedem Statusdiagramm eingezeichnet. Dies wird als Fußnote bei jedem Statusdiagramm erwähnt
- Aus Gründen der Übersicht, wird das Setzen der Ausgänge nicht im Zustandsdiagramm beschrieben; eine explizite Wahrheitstabelle , die die Information "FB-Zustände zu Ausgang (Ausgängen)" enthält, ist Teil jeder FB-Spezifikation mit der FB-spezifischen Fehler- und Zustands-Codes

Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#Cxxx	Error	Ready = TRUE S_Out = FALSE Error = TRUE

Bausteinspezifische Zustands-Codes (kein Fehler)

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Ready = FALSE S_Out = FALSE Error = FALSE
16#8001	Init-Zustand des Betriebsmodus	Ready = TRUE S_Out = FALSE Error = FALSE
16#8xxx	Alle Zustände des Betriebsmodus, mit S_Out = FALSE	Ready = TRUE S_Out = FALSE Error = FALSE
16#8000	Alle Zustände des Betriebsmodus, mit S_Out = TRUE	Ready = TRUE S_Out = TRUE Error = FALSE

16.4.2 SF_Antivalent

Kurzbeschreibung



SF_Antivalent ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein verknüpft zwei antivalente SAFEBOOL-Eingänge (NO/NC Paar) zu einem SAFEBOOL-Ausgang unter Diskrepanzzeit-Überwachung.

Da dieser Baustein keine Restart-Anlaufsperre besitzt, sollte er nicht standalone verwendet werden. Es ist erforderlich, den Ausgang mit anderen sicherheitsgerichteten Funktionalitäten zu verbinden.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-105: Bibliothekszuordnung

Schnittstellenbeschreibung

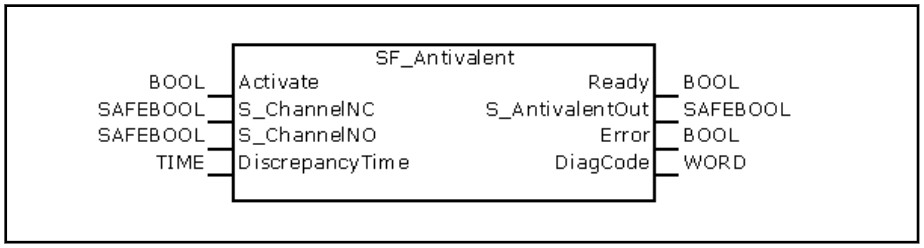


Abb. 16-39: Funktionsbaustein SF_Antivalent

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_ChannelNC	SAFEBOOL	Variable. NC bedeutet Normally Closed. Eingang für NC Verbindung FALSE: Kontakt NC offen TRUE: Kontakt NC geschlossen
	S_ChannelNO	SAFEBOOL	Variable. NO bedeutet Normally Open Eingang für NO Verbindung FALSE: Kontakt NO offen TRUE: Kontakt NO geschlossen
	DiscrepancyTime	TIME	Konstante Maximale Überwachungszeit für Diskrepanz-Zustand beider Eingänge
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_AntivalentOut	SAFEBOOL	Sicherheitsgerichteter Ausgang FALSE: Mindestens ein Eingangssignal ist "nicht aktiv", oder Statusänderung außerhalb der Überwachungszeit TRUE: Beide Eingangssignale "aktiv" und Statusänderung innerhalb der Überwachungszeit.
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-106: Schnittstelle SF_Antivalent

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_ChannelNC	SAFEBOOL	FALSE	gesteuert durch Activate
S_ChannelNO	SAFEBOOL	FALSE	gesteuert durch Activate
DiscrepancyTime	TIME	T#0ms	gesteuert durch Activate

Tab. 16-107: Eingangsverhalten SF_Antivalent

☞ "Antivalent" bedeutet, dass sich während des normalen Ablaufs die zwei Eingänge zur selben Zeit in entgegengesetzten Zuständen befinden. Dies wird manchmal auch als "komplementär" oder "nicht equivalent" bezeichnet.

Funktionsbeschreibung

Dieser Baustein verarbeitet zwei antivalente SAFEBOOL-Eingänge zu einem SAFEBOOL-Ausgang unter Diskrepanzzeit-Überwachung. Beide Eingänge sind voneinander abhängig. Der Ausgang des Bausteins zeigt das Auswertungsergebnis beider Kanäle an.

Wenn S_AntivalentOut = TRUE ist und sich einer der sicherheitsbezogenen Eingänge ändert, schaltet der Ausgang sofort auf FALSE.

Diskrepanzzeit-Überwachung: Die Diskrepanzzeit ist die maximale Zeitspanne, in der beide Eingänge den gleichen Zustand haben dürfen (diese bedeutet, dass entweder beide Eingänge auf TRUE oder beide auf FALSE sind), ohne dass der Baustein einen Fehler entdeckt. Die Diskrepanzzeit-Überwachung startet, wenn sich der Zustand eines Eingangs ändert. Der Baustein entdeckt einen Fehler, wenn beide Eingänge nach Ablauf der Diskrepanzzeit den gleichen Wert aufweisen.

Die Eingänge müssen symmetrisch geschalten werden. Dies bedeutet, dass die Überwachung sowohl für das Umschalten der Eingänge auf TRUE als auch für das Umschalten auf FALSE durchgeführt wird.

Eingänge				Ausgänge			
Activate	S_ChannelNCS	S_ChannelNO	Discrepancy-Time	Ready	S_Antivalen-tOut	Error	DiagCode
FALSE	FALSE	TRUE	nicht gestartet	FALSE	FALSE	FALSE	16#0000
TRUE	FALSE	TRUE	nicht gestartet	TRUE	FALSE	FALSE	16#8001
TRUE	FALSE -> TRUE	TRUE	gestartet	TRUE	FALSE	FALSE	16#8004
TRUE	TRUE	TRUE -> FALSE	gestartet	TRUE	TRUE	FALSE	16#8000
TRUE	TRUE	FALSE	nicht gestartet	TRUE	TRUE	FALSE	16#8000
TRUE	TRUE	FALSE -> TRUE	gestartet	TRUE	FALSE	FALSE	16#800165
TRUE	TRUE -> FALSE	TRUE	gestartet	TRUE	FALSE	FALSE	16#8001
TRUE	FALSE	TRUE	nicht gestartet	TRUE	FALSE	FALSE	16#8001
TRUE	FALSE	TRUE -> FALSE	gestartet	TRUE	FALSE	FALSE	16#8014
TRUE	FALSE -> TRUE	FALSE	gestartet	TRUE	TRUE	FALSE	16#8000
TRUE	TRUE -> FALSE	FALSE	gestartet	TRUE	FALSE	FALSE	16#8005
TRUE	FALSE	FALSE -> TRUE	gestartet	TRUE	FALSE	FALSE	16#8001

Tab. 16-108: Logiktabelle Ein-/Ausgänge

In der Logiktabelle steht der Pfeil -> für einen Übergang. TRUE -> FALSE bedeutet, dass sich der Wert der Variablen von TRUE auf FALSE ändert.

Statusdiagramm

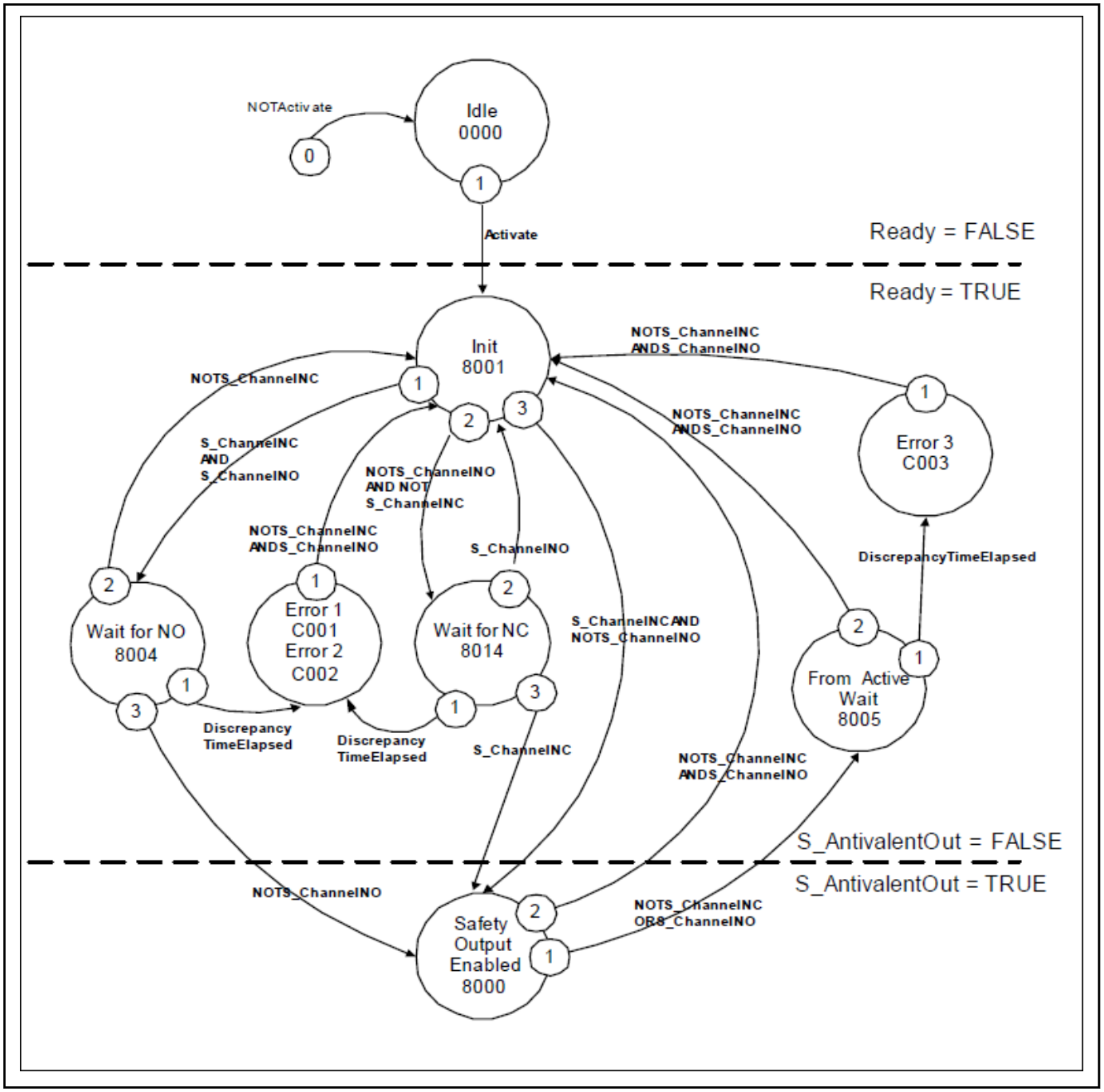


Abb. 16-40: Statusdiagramm SF_Antivalent

Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typische Zeitdiagramme

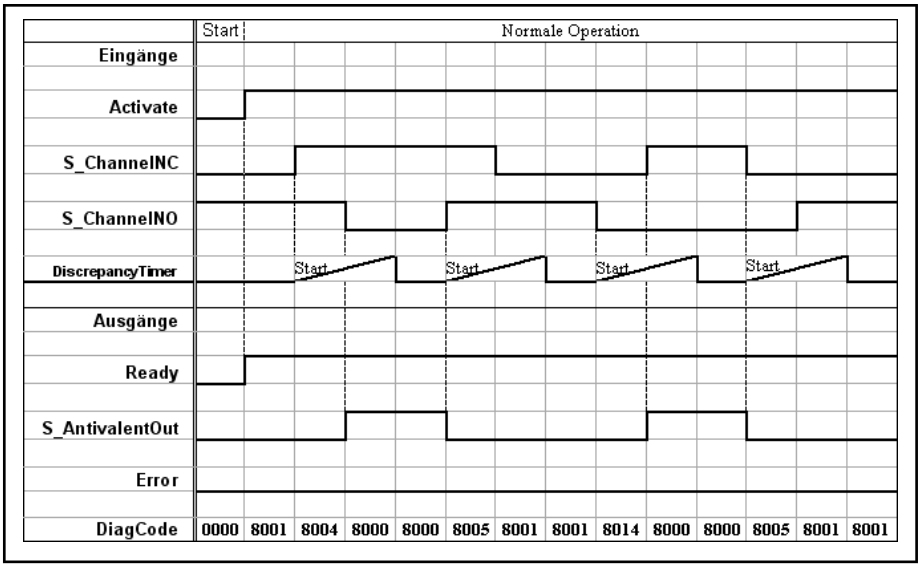


Abb. 16-41: Zeitdiagramm 1 SF_Antivalent

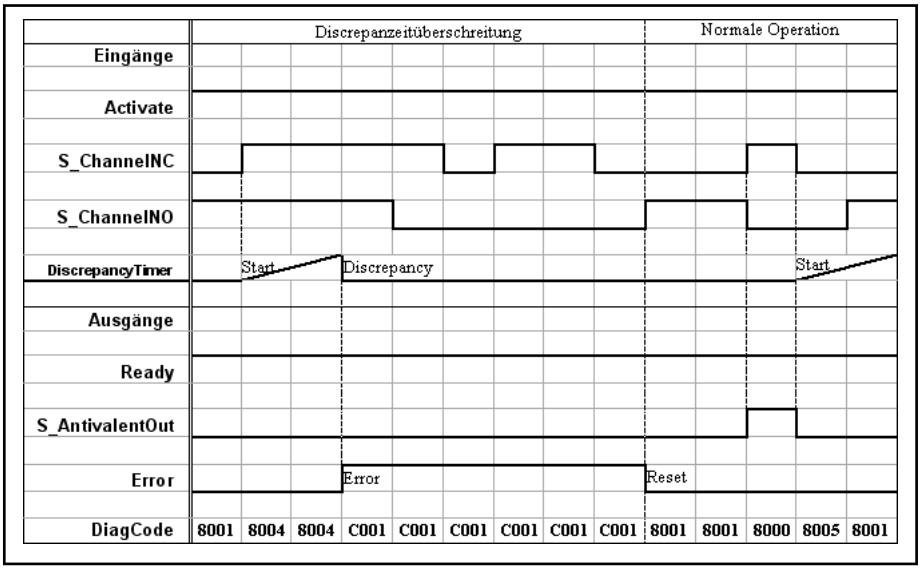


Abb. 16-42: Zeitdiagramm 2 SF_Antivalent

Fehlererkennung

Der Baustein überwacht die Diskrepanzzeit zwischen Kanal NO und Kanal NC.

Fehlerverhalten

Der Ausgang S_AntivalentOut wird auf FALSE gesetzt. Error wird auf TRUE gesetzt. DiagCode zeigt die Fehlerzustände an. Es gibt keinen als Eingang definierten Reset, der mit dem Reset eines Fehlers gekoppelt ist. Sobald ein Set von Eingangswerten mit korrektem S_AntivalentOut anliegt, ist Error wieder FALSE.

Fehler- und Zustandcodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Error 1	Diskrepanzzeit abgelaufen in Status 16#8004 Ready = TRUE S_AntivalentOut = FALSE Error = TRUE
16#C002	Error 2	Diskrepanzzeit abgelaufen in Status 16#8014 Ready = TRUE S_AntivalentOut = FALSE Error = TRUE
16#C003	Error 3	Diskrepanzzeit abgelaufen in Status 16#8005 Ready = TRUE S_AntivalentOut = FALSE Error = TRUE

Tab. 16-109: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_AntivalentOut =FALSE Error = FALSE
16#8001	Init	Eine Aktivierung wurde durch den Baustein entdeckt und der Baustein ist jetzt aktiviert. Ready = TRUE S_AntivalentOut = FALSE Error = FALSE
16#8000	Safety Output Enabled	Die Eingänge sind im Antivalent-Modus auf aktiv geschalten. Ready =TRUE S_AntivalentOut = TRUE Error = FALSE
16#8004	Wait for NO	ChannelNC ist auf TRUE umgeschaltet worden - Warten bis ChannelNO auf FALSE umgeschaltet wird; Diskrepanz-Timer gestartet. Ready = TRUE S_AntivalentOut = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8014	Wait for NC	ChannelNO ist auf FALSE umgeschaltet worden - Warten bis ChannelNC auf TRUE umgeschaltet wird; Diskrepanz-Timer gestartet. Ready = TRUE S_AntivalentOut = FALSE Error = FALSE
16#8005	From Active Wait	Ein Kanal ist auf inaktiv umgeschaltet worden; Warten, dass der zweite Kanal auch auf inaktiv geschalten wird. Ready = TRUE S_AntivalentOut = FALSE Error = FALSE

Tab. 16-110: Bausteinspezifische Status-Codes

16.4.3 SF_EDM

External Device Monitoring

Kurzbeschreibung

SF_EDM ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein legt ein Eingangssignal (S_OutControl) auf ein Ausgangssignal (S_EDM_Out).

Der Baustein überwacht, dass zwei Eingänge innerhalb einer definierten Zeit (MonitoringTime) abschalten. Schalten sie nicht innerhalb der definierten Zeit ab, wird S_EDM_OUT auf FALSE gesetzt.

Der Baustein SF_EDM (External Device Monitoring) überwacht den Grundzustand und den Schaltzustand von Aktoren, zum Beispiel Schaltverstärkern, die von sicheren Ausgangsgeräten gesteuert werden.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-111: Bibliothekszuordnung

Schnittstellenbeschreibung

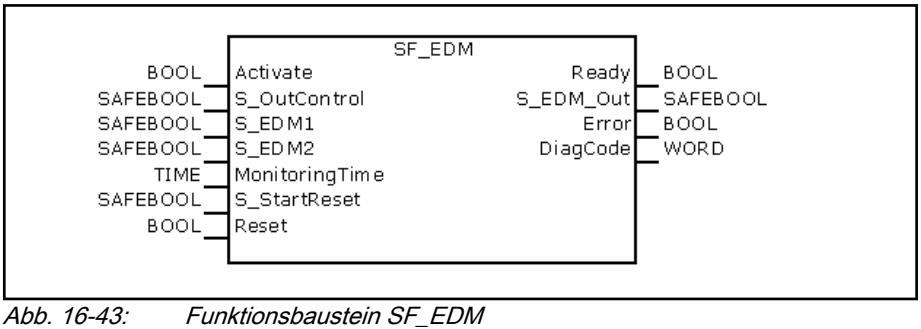


Abb. 16-43: Funktionsbaustein SF_EDM

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_OutControl	SAFEBOOL	Variable Kontrollsignal des vorangehenden Sicherheitsbausteins Typische Bausteinsignale der PLCopen-Bibliothek (z. B., SF_OutControl, SF_TwoHandControlTypeII) FALSE: Inaktiver Sicherheitsausgang (S_EDM_Out) TRUE: Aktiver Sicherheitsausgang (S_EDM_Out)
	S_EDM1	SAFEBOOL	Variable Rückmeldungssignal des ersten verbundenen Aktors (Schaltverstärkers) FALSE: Schaltzustand des ersten verbundenen Aktors TRUE: Grundzustand des ersten verbundenen Aktors
	S_EDM2	SAFEBOOL	Variable Rückmeldesignal des zweiten verbundenen Aktors (Schaltverstärkers) Wenn nur ein Signal in der Applikation verwendet wird, muss der Anwender eine grafische Verbindung verwenden, um die Parameter S_EDM1 und S_EDM2 zu überbrücken S_EDM1 und S_EDM2 werden dann durch das gleiche Signal gesteuert FALSE: Schaltzustand des zweiten verbundenen Aktors TRUE: Grundzustand des zweiten verbundenen Aktors
	MonitoringTime	TIME	Konstante Maximale Antwortzeit der verbundenen und überwachten Aktoren
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_EDM_Out	SAFEBOOL	Steuert den Aktor Das Ergebnis wird durch das Rückmeldesignal S_EDMx angezeigt FALSE: Inaktive angeschlossene Aktoren TRUE: Aktive angeschlossene Aktoren
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-112: Schnittstelle SF_EDM

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_OutControl	SAFEBOOL	FALSE	
S_EDM1	SAFEBOOL	FALSE	
S_EDM2	SAFEBOOL	FALSE	

Name	Typ	Default-Wert	Übernahme
MonitoringTime	TIME	T#0ms	Konstante
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-113: Eingangsverhalten SF_EDM

Funktionsbeschreibung

Der Baustein SF_EDM steuert einen Sicherheitsausgang und überwacht gesteuerte Aktoren.

Er überwacht den Grundzustand der Aktoren über die Rückmeldesignale (S_EDM1 und S_EDM2) bevor die Aktoren durch den Baustein aktiviert werden.

Dieser Baustein überwacht den Schaltzustand der Aktoren (MonitoringTime) nachdem die Aktoren durch den Baustein aktiviert wurden.

Zwei einzelne Rückmeldesignale müssen für die genaue Diagnose der angeschlossenen Aktoren verwendet werden. Ein gemeinsames Rückmeldesignal der beiden angeschlossenen Aktoren muss für eine begrenzte, jedoch einfache Diagnosefunktion der angeschlossenen Aktoren verwendet werden. In diesem Fall muss der Anwender dieses gemeinsame Signal mit beiden Parametern S_EDM1 und S_EDM2 verbinden. S_EDM1 und S_EDM2 werden dann durch das gleich Signal gesteuert.

Die in der Sicherheitsfunktion verwendeten Schaltgeräte sollte der Kategorie angehören, die in Risikoanalyse entsprechend der Norm festgelegt wurde.

Optionale Anlaufsperre:

Anlaufsperre bei Bausteinaktivierung

⚠ VORSICHT **S_StartReset-Eingang**

Der S_StartReset-Eingang soll nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdungssituation entstehen kann.

Statusdiagramm

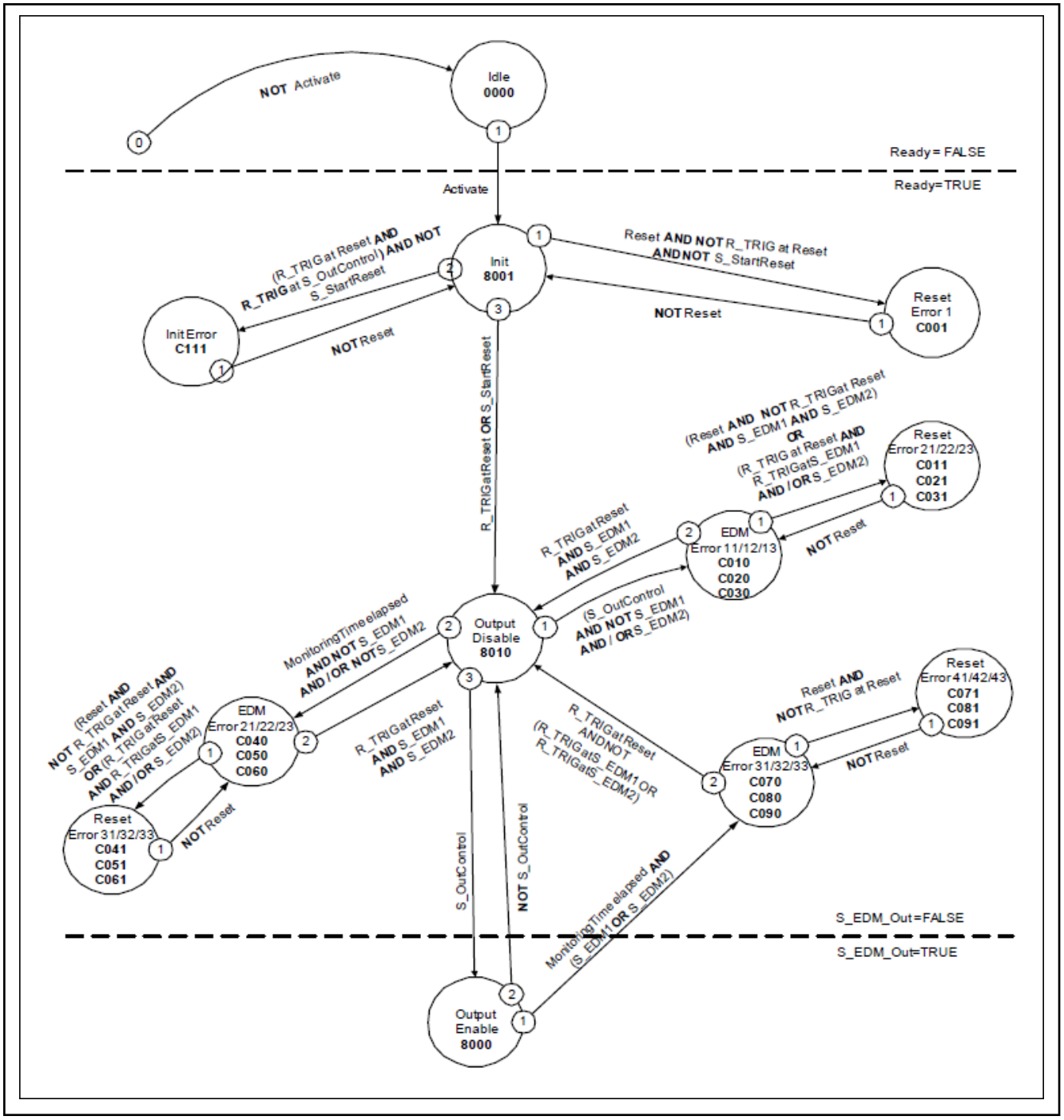


Abb. 16-44: Statusdiagramm SF_EDM

Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typische Zeitdiagramme

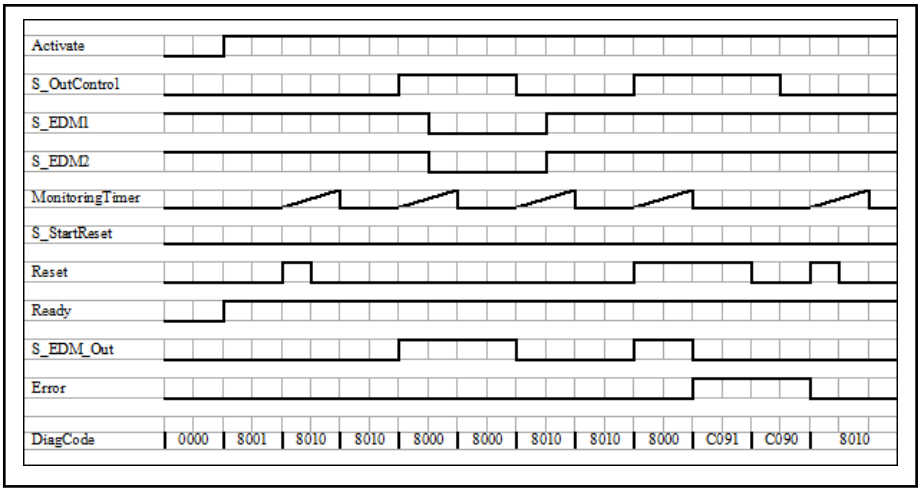


Abb. 16-45: Typisches Zeitdiagramm SF_EDM, S_StartReset = FALSE

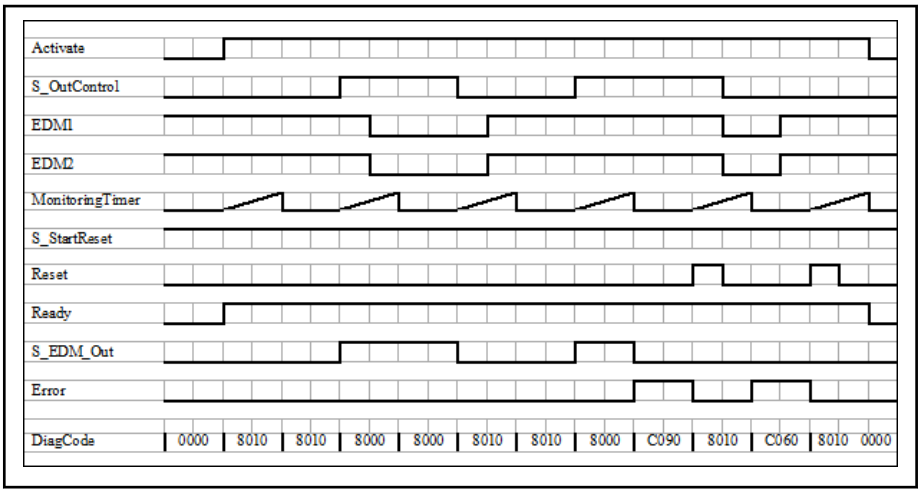


Abb. 16-46: Typisches Zeitdiagramm SF_EDM, S_StartReset = TRUE

Fehlererkennung

Die folgenden Bedingungen führen zum Fehlerzustand:

- Ungültiges, statisches Signal am Reset-Eingang.
- Ungültiges EDM-Signal im Prozess
- Die Eingänge S_OutControl und Reset sind wegen eines Progammierfehlers falsch beschaltet.

Fehlerverhalten

Im Fehlerzustand sind die Ausgänge wie folgt:

- S_EDM_Out ist auf FALSE gesetzt und bleibt im sicheren Zustand.
- Eine EDM-Fehlermeldung muss immer mit einer steigenden Triggerflanke an Reset zurückgesetzt werden.
- Eine Reset-Fehlermeldung kann durch Setzen von Reset auf FALSE zurückgesetzt werden.

Nach der Bausteinaktivierung kann die optionale Anlaufsperre durch eine steigende Flanke am Reset-Eingang zurückgesetzt werden.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error 1	Statisches Reset-Signal in Status 16#8001. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C011	Reset Error 21	Statisches Reset-Signal oder gleiche Signale an EDM1 und Reset (gleichzeitig steigende Triggerflanke an Reset und EDM1) in Status 16#C010. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C021	Reset Error 22	Statisches Reset-Signal oder gleiche Signale an EDM2 und Reset (gleichzeitig steigende Triggerflanke an Reset und EDM2) in Status16# C020. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C031	Reset Error 23	Statisches Reset-Signal oder gleiche Signale an EDM1, EDM2 und Reset (gleichzeitig steigende Triggerflanke an Reset, EDM1 und EDM2) in Status 16#C030. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C041	Reset Error 31	Statisches Reset-Signal oder gleiche Signale an EDM1 und Reset (gleichzeitig steigende Triggerflanke an Reset und EDM1) in Status16# C040. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C051	Reset Error 32	Statisches Reset-Signal oder gleiche Signale an EDM2 und Reset (gleichzeitig steigende Triggerflanke an Reset und EDM2) in Status 16#C050. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C061	Reset Error 33	Statisches Reset-Signal oder gleiche Signale An EDM1, EDM2 und Reset (gleichzeitig steigende Triggerflanke an Reset, EDM1 und EDM2) in Status16# C060. Ready = TRUE S_EDM_Out = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C071	Reset Error 41	Statisches Reset-Signal in Status 16#C070. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C081	Reset Error 42	Statisches Reset-Signal in Status 16#C080. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C091	Reset Error 43	Statisches Reset-Signal in Status 16#C090. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C010	EDM Error 11	Das Signal an EDM1 ist im Grundzustand des Aktors ungültig. In Status 16#8010 ist das EDM1-Signal auf FALSE, wenn S_OutControl aktiviert wird. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C020	EDM Error 12	Das Signal an EDM2 ist im Grundzustand des Aktors ungültig. In Status 16#8010 ist das EMD2-Signal auf FALSE, wenn S_OutControl aktiviert wird. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C030	EDM Error 13	Die Signale an EDM1 und EDM2 sind in den Grundzuständen des Aktors ungültig. In Status16# 8010 sind die Signale EDM1 und EDM2 auf FALSE, wenn S_OutControl aktiviert wird. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C040	EDM Error 21	Das Signal an EDM1 ist im Grundzustand des Aktors ungültig. In Status 16#8010 ist das EDM1-Signal auf FALSE und die Überwachungszeit ist abgelaufen. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C050	EDM Error 22	Das Signal an EDM2 ist im Grundzustand des Aktors ungültig. In Status 16#8010 ist das EDM2-Signal auf FALSE und die Überwachungszeit ist abgelaufen. Ready = TRUE S_EDM_Out = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C060	EDM Error 23	Die Signale an EDM1 und EDM2 sind in den Grundzuständen des Aktors ungültig. In Status 16#8010 sind die Signale EDM1 und EDM2 auf FALSE und die Überwachungszeit ist abgelaufen. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C070	EDM Error 31	Das Signal an EDM1 ist im Schaltzustand des Aktors ungültig. In Status 16#8000 ist das EDM1-Signal auf TRUE und die Überwachungszeit ist abgelaufen. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C080	EDM Error 32	Das Signal an EDM2 ist im Schaltzustand des Aktors ungültig. In Status 16#8000 ist das EDM2-Signal auf TRUE und die Überwachungszeit ist abgelaufen. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C090	EDM Error 33	Die Signale an EDM1 und EDM2 sind in den Schaltzuständen des Aktors ungültig. In Status 16#8000 sind die Signale EDM1 und EDM2 auf FALSE und die Überwachungszeit ist abgelaufen. Ready = TRUE S_EDM_Out = FALSE Error = TRUE
16#C111	Init Error	Gleichartige Signale an S_OutControl und Reset (R_TRIG im gleichen Zyklus) entdeckt (vielleicht ein Programmierfehler). Ready = TRUE S_EDM_Out = FALSE Error = TRUE

Tab. 16-114: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_EDM_Out = FALSE Error = FALSE
16#8001	Init	Bausteinaktivierung Anlaufsperre ist aktiv. Reset erforderlich. Ready = TRUE S_EDM_Out = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8010	Output Disable	EDM-Kontrolle ist nicht aktiv. Timer startet, wenn Status eintritt. Ready = TRUE S_EDM_Out = FALSE Error = FALSE
16#8000	Output Enable	EDM-Kontrolle ist aktiv. Timer startet, wenn Status eintritt. Ready = TRUE S_EDM_Out = TRUE Error = FALSE

Tab. 16-115: Bausteinspezifische Status-Codes

16.4.4 SF_EmergencyStop

Kurzbeschreibung

SF_EmergencyStop ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Die in den Normen aufgeführten Anforderungen müssen vom Anwender erfüllt werden.

Dieser Baustein ist ein sicherheitsgerichteter Baustein um einen NOT-HALT-Taster zu überwachen.

SF_EmergencyStop kann für die Notschalter-AUS Funktionalität (Stopp-Kategorie 0), oder - mit zusätzlicher peripherer Hilfe - als NOT-HALT (Stopp-Kategorie 1 oder 2) verwendet werden.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-116: Bibliothekszuordnung

Schnittstellenbeschreibung

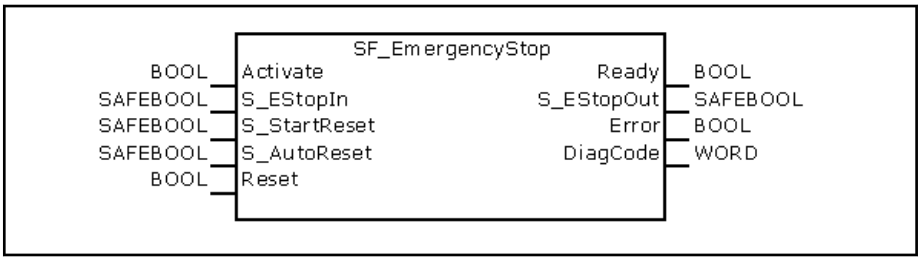


Abb. 16-47: Funktionsbaustein SF_EmergencyStop

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_EStopIn	SAFEBOOL	Eingang, der Sicherheit anfordert Variable FALSE: Anforderung einer sicherheitsgerichteten Antwort (zum Beispiel NOT-HALT-Taster ist gedrückt) TRUE: Keine Anforderung einer sicherheitsgerichteten Antwort (zum Beispiel NOT-HALT-Taster ist nicht gedrückt)
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294

E/A-TYP	Name	Datentyp	Beschreibung
	S_AutoReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_EStopOut	SAFEBOOL	Ausgang für sicherheitsgerichtete Antwort FALSE: Sicherheitsausgang deaktiviert Anforderung für sicherheitsgerichtete Antwort (zum Beispiel NOT-HALT-Taster ist gedrückt, Reset erforderlich oder interne Fehler vorhanden) TRUE: Sicherheitsausgang ist aktiviert Keine Anforderung einer sicherheitsgerichteten Antwort (zum Beispiel NOT-HALT Knopf ist nicht gedrückt, keine internen Fehler aktiv)
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-117: Schnittstelle SF_EmergencyStop

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_EStopIn	SAFEBOOL	FALSE	
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
S_AutoReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset wenn NOT-HALT-Knopf entriegelt ist TRUE: Automatischer Reset, wenn NOT-HALT-Knopf entriegelt ist
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-118: Eingangsverhalten SF_EmergencyStop

Funktionsbeschreibung

Das S_EStopOut Freigabesignal wird sobald der S_EStopIn-Eingang auf FALSE gesetzt ist, auf FALSE zurückgesetzt. Das S_EStopOut Freigabesignal wird nur auf TRUE zurückgesetzt, wenn der S_EStopIn Eingang auf TRUE gesetzt wird und ein Reset erfolgt. Das Freigabe-Reset hängt von den definierten S_StartReset, S_AutoReset und Reset Eingängen ab.

Wenn S_AutoReset = TRUE: Bestätigung ist automatisch.

Wenn S_AutoReset = FALSE: Eine steigende Triggerflanke am Reset-Eingang muss zur Freigabebestätigung verwendet werden.

Wenn S_StartReset = TRUE : Wenn die S-SPS das erste Mal gestartet wird, ist die Bestätigung automatisch.

Wenn S_StartReset = FALSE : Eine steigende Triggerflanke am Reset-Eingang muss zur Freigabebestätigung verwendet werden.

⚠ VORSICHT **S_StartReset- und S_AutoReset**

Die S_StartReset- und S_AutoReset-Eingänge sollen nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdung auftreten kann, wenn die S-SPS startet.

SF_EmergencyStop kann zum Überwachen von Single- und Zweikanal-NOT-HALT-Schaltern verwendet werden. Zum Beispiel kann für Zweikanal-Applikationen der zusätzliche Baustein SF_Equivalent verwendet werden, um zu erkennen, ob die Kontaktsynchronisierung überschritten wurde. Die Kategorie-Klassifizierung in Übereinstimmung mit der Norm wird von den endgültig verwendeten Elementen abhängen.

Der SF_EmergencyStop entdeckt automatisch eine statisches TRUE an Reset. Weitere Fehlererkennung, zum Beispiel Drahtbruch, Kurzschluss etc., hängt von der verwendeten Hardware ab.

Statusdiagramm

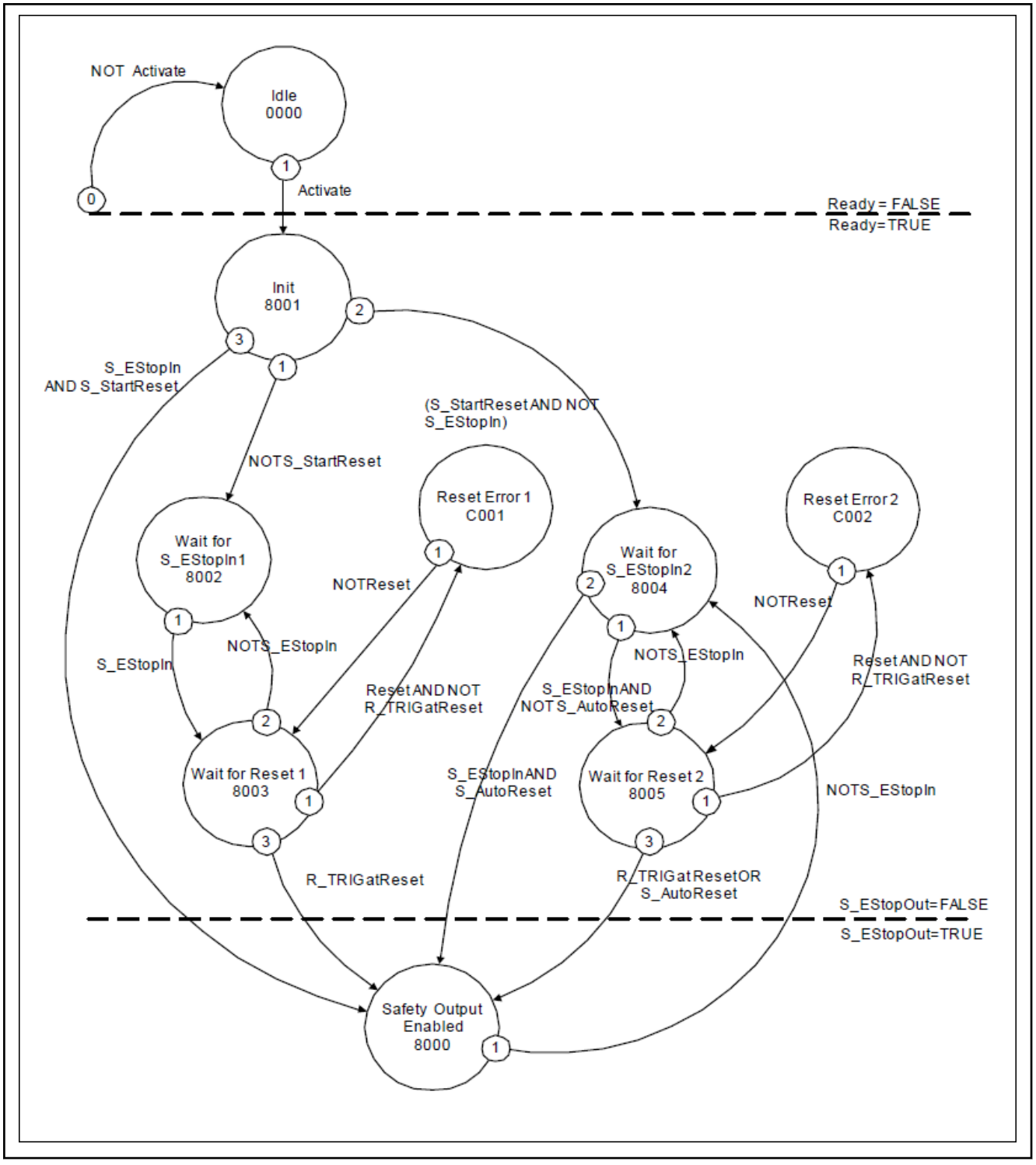


Abb. 16-48: Statusdiagramm SF_EmergencyStop

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

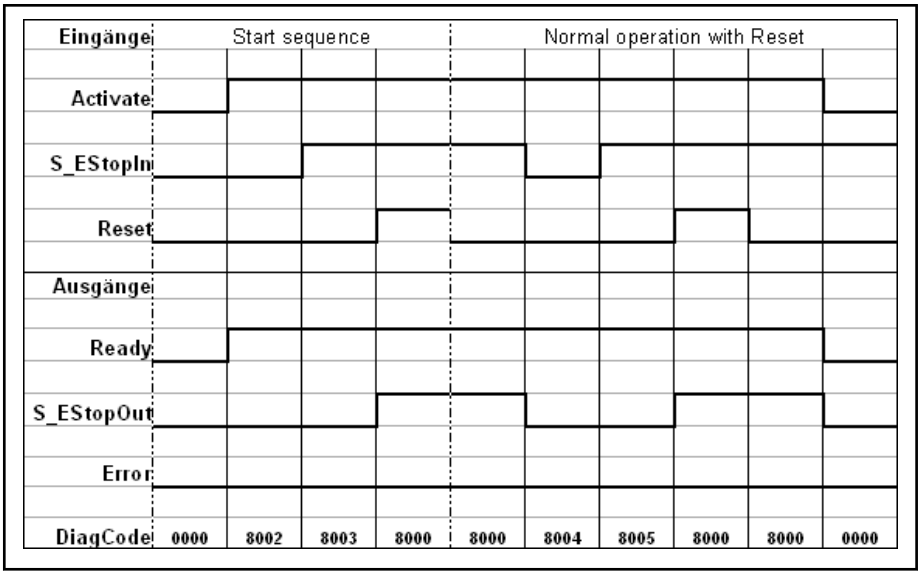


Abb. 16-49: Zeitdiagramm SF_EmergencyStop: S_StartReset = FALSE; S_AutoReset = FALSE; Start, Reset, normale Operation, Sicherheitsanforderung, Restart

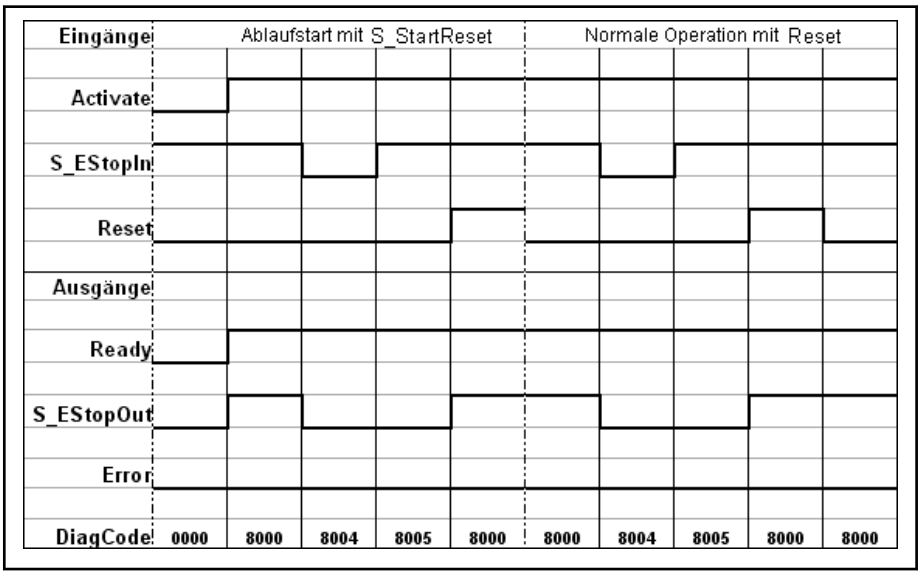


Abb. 16-50: Zeitdiagramm SF_EmergencyStop: S_StartReset = TRUE, S_AutoReset = FALSE; Start, normale Operation, Sicherheitsanforderung, Restart

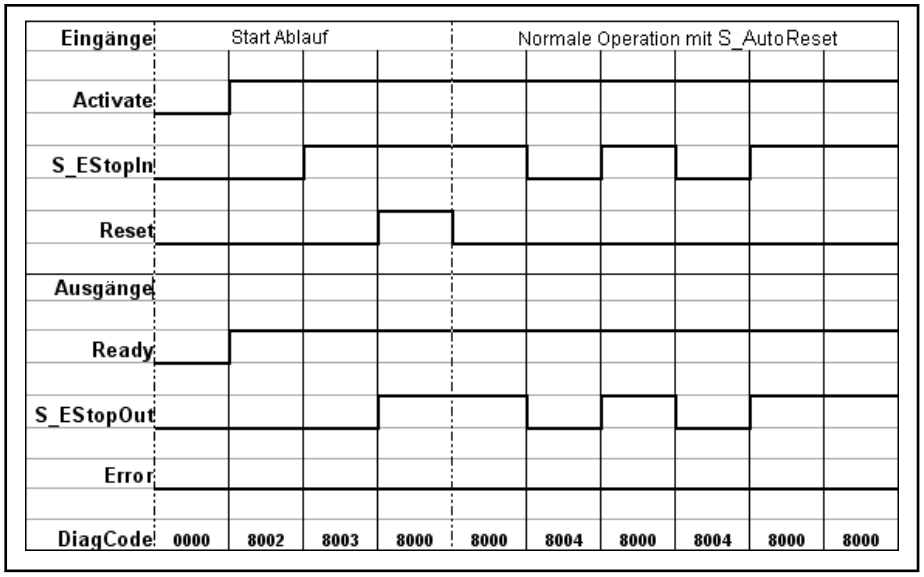


Abb. 16-51: Zeitdiagramm SF_EmergencyStop: S_StartReset = FALSE, S_AutoReset = TRUE, Start, normale Operation, Sicherheitsanforderung, Restart

Fehlererkennung

Der Baustein entdeckt eine statisches TRUE-Signal am Reset-Eingang.

Fehlerverhalten

S_EStopOut wird auf FALSE gesetzt. Im Fall eines statischen TRUE-Signals am Reset-Eingang, zeigt der DiagCode-Ausgang den relevanten Fehlercode an und der Error-Ausgang wird auf TRUE gesetzt.

Um die Fehlerzustände zu verlassen, muss Reset auf FALSE gesetzt werden.

Fehler- und Zustandscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error 1	Während des Wartens auf S_EStopIn = TRUE wird Reset TRUE. Ready = TRUE S_EStopOut = FALSE Error = TRUE
16#C002	Reset Error 2	Während des Wartens auf S_EStopIn = TRUE wird Reset TRUE. Ready = TRUE S_EStopOut = FALSE Error = TRUE

Tab. 16-119: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_EStopOut = FALSE Error = FALSE
16#8001	Init	Aktivierung ist TRUE. Der Baustein war aktiviert. Überprüfung ob S_StartReset angefordert wird. Ready = TRUE S_EStopOut = FALSE Error = FALSE
16#8002	Wait for S_EstopIn 1	Aktivierung ist TRUE. Überprüfung ob Reset FALSE ist und Warten auf S_EStopIn = TRUE. Ready = TRUE S_EStopOut = FALSE Error = FALSE
16#8003	Wait for Reset 1	Aktivierung ist TRUE. S_EStopIn = TRUE. Warten auf steigende Triggerflanke von Reset. Ready = TRUE S_EStopOut = FALSE Error = FALSE
16#8004	Wait for S_EStopIn2	Aktivierung ist TRUE. Sicherheitsanforderung entdeckt. Überprüfung ob Reset FALSE ist und Warten auf S_EStopIn = TRUE. Ready = TRUE S_EStopOut = FALSE Error = FALSE
16#8005	Wait for Reset 2	Aktivierung ist TRUE. S_EStopIn = TRUE. Überprüfung von S_Auto-Reset oder Warten auf steigende Triggerflanke von Reset. Ready = TRUE S_EStopOut = FALSE Error = FALSE
16#8000	Safety Output Enabled	Aktivierung ist TRUE. S_EStop = TRUE. Funktionsart mit S_EStopOut = TRUE. Ready = TRUE S_EStopOut = TRUE Error = FALSE

Tab. 16-120: Bausteinspezifische Status-Codes (kein Fehler)

16.4.5 SF_EnableSwitch

Kurzbeschreibung



SF_EnableSwitch ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Der Baustein SF_EnableSwitch wertet die Signale eines handbetätigten, dreistufigen Zustimmungsschalters zur Unterstützung der Aufhebung einer Schutzmaßnahme aus.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-121: Bibliothekszuordnung

Schnittstellenbeschreibung

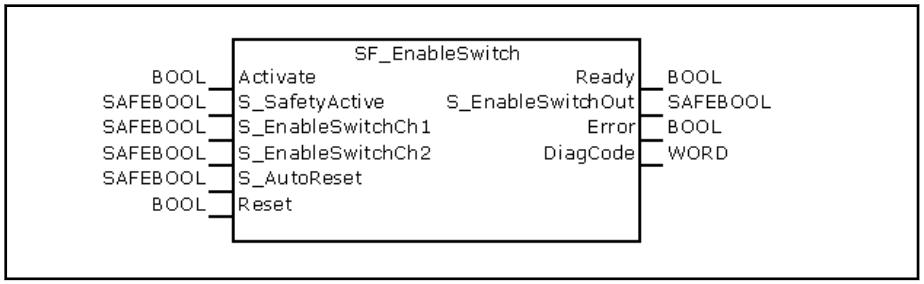


Abb. 16-52: Funktionsbaustein SF_EnableSwitch

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_SafetyActive	SAFEBOOL	Variable oder Konstante Bestätigung des sicheren Modus (Begrenzung der Geschwindigkeit, der Bewegungsenergie oder des Bewegungsbereichs) FALSE: Sicherer Modus ist nicht aktiv TRUE: Sicherer Modus ist aktiv
	S_EnableSwitchCh1	SAFEBOOL	Variable Signal der Kontakte E1 und E2 des verbundenen Zustimmungsschalters FALSE: Verbundene Kontakte sind offen TRUE: Verbundene Kontakte sind geschlossen
	S_EnableSwitchCh2	SAFEBOOL	Variable Signal der Kontakte E3 und E4 des verbundenen Zustimmungsschalters FALSE: Verbundene Kontakte sind offen TRUE: Verbundene Kontakte sind geschlossen
	S_AutoReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_EnableSwitchOut	SAFEBOOL	Sicherheitsgerichteter Ausgang: Zeigt die Aufhebung der Schutzmaßnahme an FALSE: Inaktive Aufhebung der Schutzmaßnahme TRUE: Aktive Aufhebung der Schutzmaßnahme

E/A-Typ	Name	Datentyp	Beschreibung
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-122: Schnittstelle SF_EnableSwitch

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_SafetyActive	SAFEBOOL	FALSE	
S_EnableSwitchCh1	SAFEBOOL	FALSE	
S_EnableSwitchCh2	SAFEBOOL	FALSE	
S_AutoReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset wenn NOT-HALT-Knopf entriegelt ist TRUE: Automatischer Reset, wenn NOT-HALT-Knopf entriegelt ist
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-123: Eingangsverhalten SF_EnableSwitch

Funktionsbeschreibung

Der Baustein SF_EnableSwitch unterstützt entsprechend der europäischen Norm die Aufhebung der Schutzmaßnahme unter Verwendung von Zustimmungsschalter, wenn die relevante Betriebsart selektiert und aktiv ist. Die entsprechende Betriebsart (Begrenzung der Geschwindigkeit oder der Bewegungsenergie, Begrenzung des Bewegungsbereichs) muss außerhalb dieses Bausteins ausgewählt werden.

Der Baustein SF_EnableSwitch wertet die Signale eine 3-stufigen Zustimmungsschalters aus.

Die Eingangsparameter S_EnableSwitchCh1 und S_EnableSwitchCh2 verarbeiten folgende Signalstufen der Kontakte E1 bis E4.

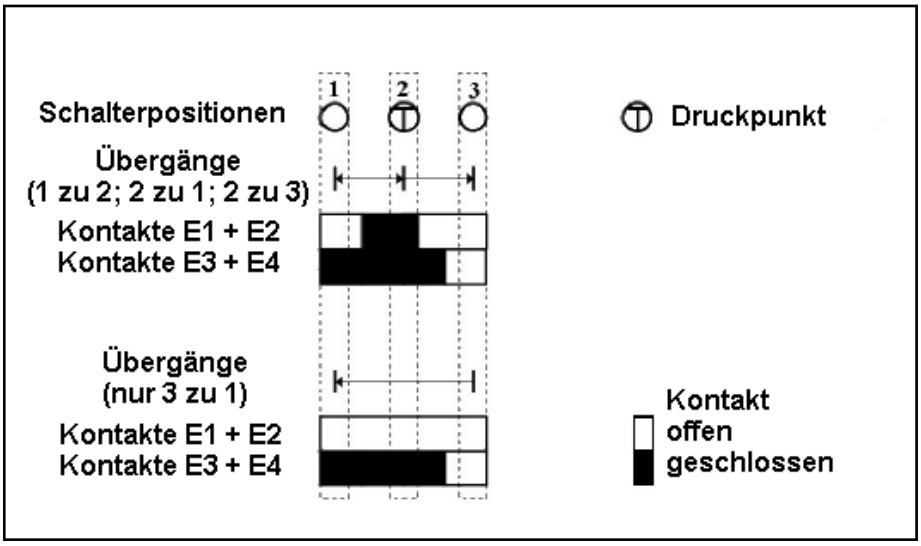


Abb. 16-53: SF_EnableSwitch Schaltpositionen

Die Signale von E1+E2 müssen mit dem Parameter S_EnableSwitchCh1 verbunden werden. Die Signale von E3+E4 müssen mit dem Parameter

S_EnableSwitchCh2 verbunden werden. Die Position des Zustimmungsschalters wird im Baustein beim durch das Verwenden der Signalsequenz erkannt. Der Übergang von Position 2 nach Position 3 kann anders sein, als hier gezeigt wird.

Die Schaltrichtung (Position 1 -> Position 2/Position 3 -> Position 2) kann im Baustein erkannt werden, wenn die definierte Signalfolge der Kontakte des Zustimmungsschalters verwendet wird. Die Aufhebung der Schutzmaßnahme kann vom Baustein nur durch eine Änderung von Position 1 nach Position 2 aktiviert werden. Andere Schaltrichtungen oder Postionen dürfen zur Aufhebung der Schutzmaßnahme nicht verwendet werden.

Der Anwender muss ein geeignetes Schaltgerät als Zustimmungsschalter verwenden. Zusätzlich muss der Anwender sicherstellen, dass der entsprechende Betriebsmodus in der Applikation ausgewählt ist (der automatische Betrieb muss bei dieser Betriebsart, durch die Anwendung geeigneter Maßnahmen, deaktiviert sein).

Die Betriebsart wird gewöhnlich durch einen Betriebsart-Auswahlschalter in Verbindung mit dem Baustein SF_ModeSelector und dem Baustein SF_SafetyRequest festgelegt.

SF_EnableSwitch verarbeitet die Bestätigung des "sicheren Zustand" Status durch den Parameter S_SafetyActive. Bei einer Implementierung in einer sicheren Applikation ohne Bestätigung, wird ein statisches TRUE-Signal mit dem Parameter S_SafetyActive verbunden.

⚠ VORSICHT **S_AutoReset-Eingang**

Der S_AutoReset-Eingang soll nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdung auftreten kann, wenn die S-SPS startet.

Statusdiagramm

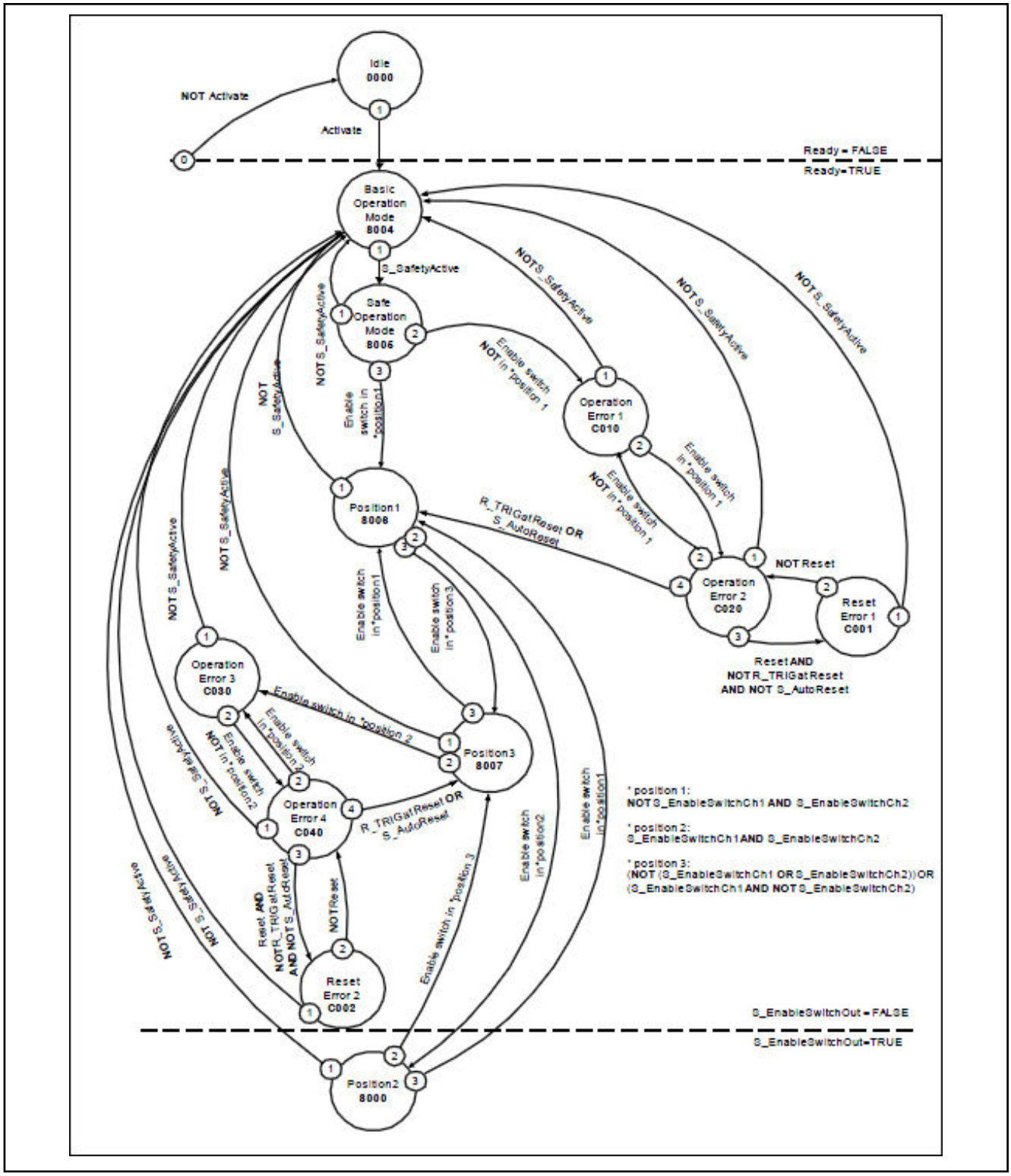


Abb. 16-54: Statusdiagramm SF_EnableSwitch

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typische Zeitdiagramme

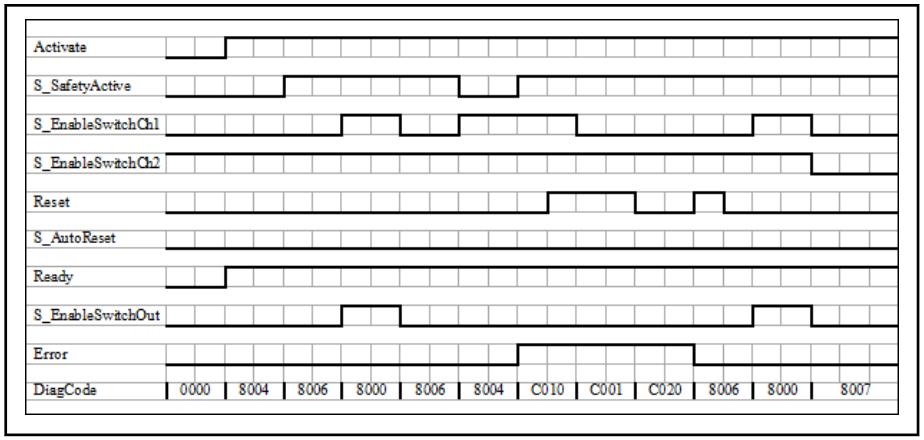


Abb. 16-55: Zeitdiagramm1 SF_EnableSwitch

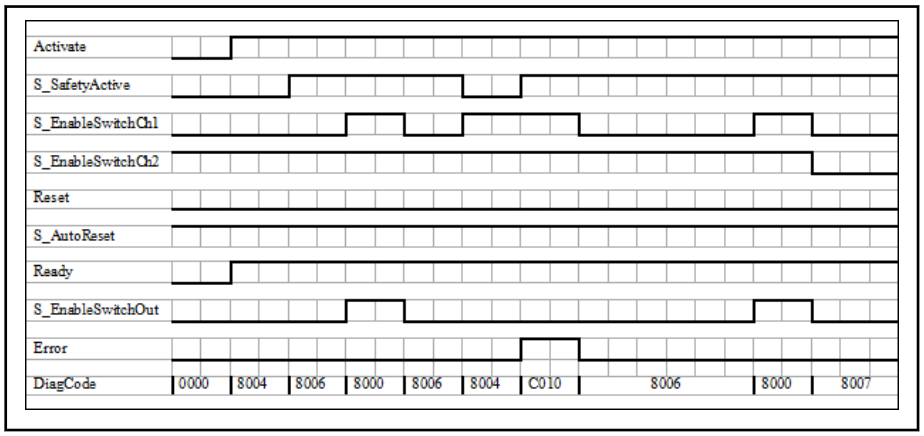


Abb. 16-56: Zeitdiagramm2 SF_EnableSwitch

Fehlererkennung

Folgende Bedingungen forcieren eine Übergang in den Fehlerstatus:

- Ungültiges Reset-Signal im Prozess
- Ungültige Schalterstellung

Fehlerverhalten

Bei einem Fehlerereignis wird der Ausgang S_EnableSwitchOut auf FALSE gesetzt und bleibt in diesem sicheren Zustand.

Im Gegensatz zu anderen Bausteinen kann ein Reset-Fehlerzustand durch die Bedingung Reset = FALSE verlassen werden, oder zusätzlich durch S_SafetyActive = FALSE.

Sobald der Fehler entfernt worden ist, muss der Zustimmungsschalter in der Grundposition befinden, die im Prozess festgelegt ist. Erst dann kann der Ausgang S_EnableSwitchOut mit Hilfe des Zustimmungsschalters auf TRUE gesetzt werden. Wenn S_AutoReset = FALSE, ist eine steigende Triggerflanke an Reset erforderlich.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error 1	Statisches Reset-Signal entdeckt in Status 16#C020 Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
16#C002	Reset Error 2	Statisches Reset-Signal entdeckt in Status 16#C040 Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
16#C010	Operation Error 1	Zustimmungsschalter ist während der Aktivierung von S_SafetyActive nicht in Position 1. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
16#C020	Operation Error 2	Zustimmungsschalter ist nach 16#C010 in Position 1 Ready =TRUE S_EnableSwitchOut = FALSE Error = TRUE
16#C030	Operation Error 3	Zustimmungsschalter ist nach Position 3 in Position 2. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE
16#C040	Operation Error 4	Zustimmungsschalter ist nach 16#C030 nicht in Position 2. Ready = TRUE S_EnableSwitchOut = FALSE Error = TRUE

Tab. 16-124: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_EnableSwtichOut = FALSE Error = FALSE
16#8004	Basic Operation Mode	Sicherer Betriebsmodus ist nicht aktiv. Ready = TRUE S_EnableSwitchOut = FALSE Error = FALSE
16#8005	Safe Operation Mode	Sicherer Betriebsmodus ist aktiv. Ready = TRUE S_EnableSwitchOut = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8006	Position 1	Sicherer Betriebsmodus ist aktiv und der Zustimmungsschalter ist in Position 1. Ready = TRUE S_EnableSwitchOut = FALSE Error = FALSE
16#8007	Position 3	Sicherer Betriebsmodus ist aktiv und der Zustimmungsschalter ist in Position 3. Ready = TRUE S_EnableSwitchOut = FALSE Error = FALSE
16#8000	Positon 2	Sicherer Betriebsmodus ist aktiv und der Zustimmungsschalter ist in Position 2. Ready = TRUE S_EnableSwitchOut = TRUE Error = FALSE

Tab. 16-125: Bausteinspezifische Status-Codes

16.4.6 SF_Equivalent

Kurzbeschreibung

SF_Equivalent ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein verknüpft zwei äquivalente SAFEBOOL-Eingänge (beide NO oder NC) zu einem SAFEBOOL-Ausgang unter Diskrepanzzeit-Überwachung.

Da dieser Baustein keine Restart-Anlaufsperre besitzt, sollte er nicht standalone verwendet werden. Es ist erforderlich, den Ausgang mit anderen sicherheitsgerichteten Funktionalitäten zu verbinden.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-126: Bibliothekszuordnung

Schnittstellenbeschreibung

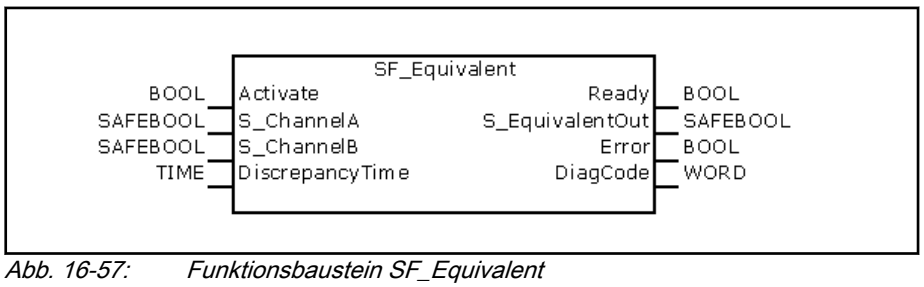


Abb. 16-57: Funktionsbaustein SF_Equivalent

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_ChannelA	SAFEBOOL	Variable Eingang A für logische Verbindung FALSE: Kontakt A offen TRUE: Kontakt A geschlossen
	S_ChannelB	SAFEBOOL	Variable Eingang B für logische Verbindung FALSE: Kontakt B offen TRUE: Kontakt B geschlossen
	DiscrepancyTime	TIME	Konstante Maximale Überwachungszeit für Diskrepanz-Zustand beider Eingänge
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_EquivalentOut	SAFEBOOL	Sicherheitsgerichteter Ausgang FALSE: Mindestens ein Eingangssignal = "FALSE" oder Statusänderung außerhalb der Überwachungszeit. TRUE: Beide Eingangssignale sind "aktiv" und Statusänderung innerhalb der Überwachungszeit
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-127: Schnittstelle SF_Equivalent

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_ChannelA	SAFEBOOL	FALSE	gesteuert durch Activate
S_ChannelB	SAFEBOOL	FALSE	gesteuert durch Activate
DiscrepancyTime	TIME	T#0ms	gesteuert durch Activate

Tab. 16-128: Eingangsverhalten SF_Equivalent

Funktionsbeschreibung

Dieser Baustein verknüpft zwei äquivalente SAFEBOOL-Eingänge (beide NO oder NC) zu einem SAFEBOOL-Ausgang unter Diskrepanzzeit-Überwachung. Die beiden Kanäle A und B sind voneinander abhängig. Der Ausgang des Bausteins zeigt das Auswertungsergebnis beider Kanäle an.

Wenn sich das Signal eines Kanals von TRUE auf FALSE ändert, schaltet der Ausgang aus Sicherheitsgründen sofort ab (FALSE).

Diskrepanzzeit-Überwachung: Die Diskrepanzzeit ist die maximale Zeitspanne, in der beide Eingänge unterschiedliche Zustände haben dürfen, ohne dass der Baustein einen Fehler entdeckt. Die Diskrepanzzeit-Überwachung startet, wenn sich der Zustand eines Eingangs ändert. Der Baustein entdeckt dann einen Fehler, wenn beide Eingänge nach Ablauf der Diskrepanzzeit unterschiedliche Zustände haben.

Beide Eingänge müssen symmetrisch geschalten werden. Das bedeutet, dass das Monitoring für beide Eingänge sowohl für das Umschalten von

TRUE auf FALSE, als auch für das Umschalten von FALSE auf TRUE durchgeführt wird.

Eingänge				Ausgänge			
Activate	S_ChannelNCS	S_ChannelNO	Discrepancy-Time	Ready	S_EquivalentOut	Error	DiagCode
FALSE	FALSE	TRUE	nicht gestartet	FALSE	FALSE	FALSE	0000
TRUE	FALSE	FALSE	nicht gestartet	TRUE	FALSE	FALSE	8001
TRUE	TRUE	FALSE	gestartet	TRUE	FALSE	FALSE	8004
TRUE	TRUE	FALSE -> TRUE	gestartet	TRUE	TRUE	FALSE	8000
TRUE	TRUE	TRUE	nicht gestartet	TRUE	TRUE	FALSE	8000
TRUE	TRUE	TRUE -> FALSE	gestartet	TRUE	FALSE	FALSE	8005
TRUE	TRUE -> FALSE	FALSE	gestartet	TRUE	FALSE	FALSE	8001
TRUE	FALSE	FALSE -> TRUE	gestartet	TRUE	FALSE	FALSE	8014
TRUE	FALSE -> TRUE	TRUE	gestartet	TRUE	TRUE	FALSE	8000

Tab. 16-129: Logiktabelle Ein-/Ausgänge

In der Logiktabelle steht der Pfeil -> für einen Übergang. TRUE -> FALSE bedeutet, dass sich der Wert der Variablen von TRUE auf FALSE ändert.

Statusdiagramm

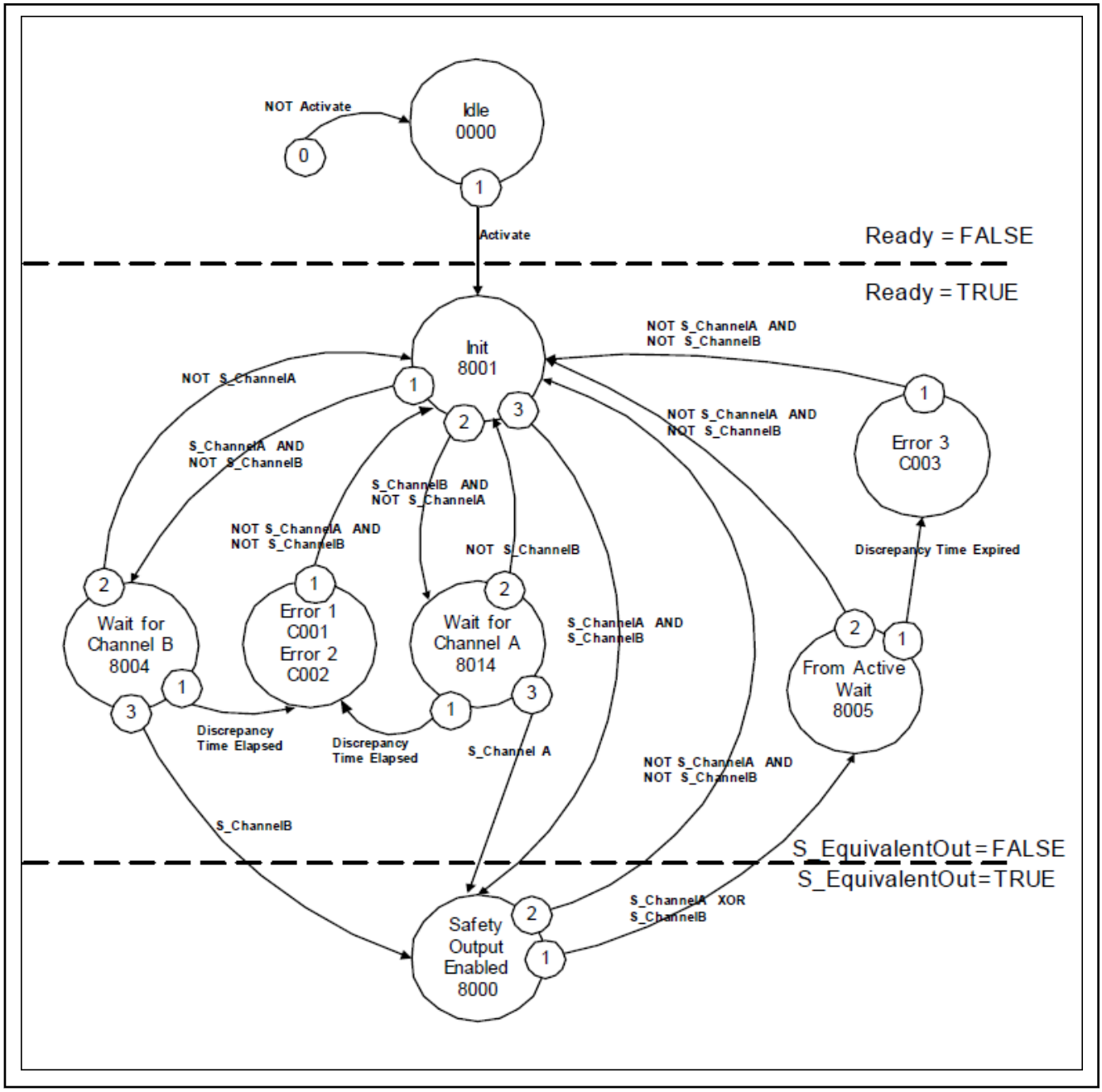


Abb. 16-58: Statusdiagramm SF_Equivalent

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

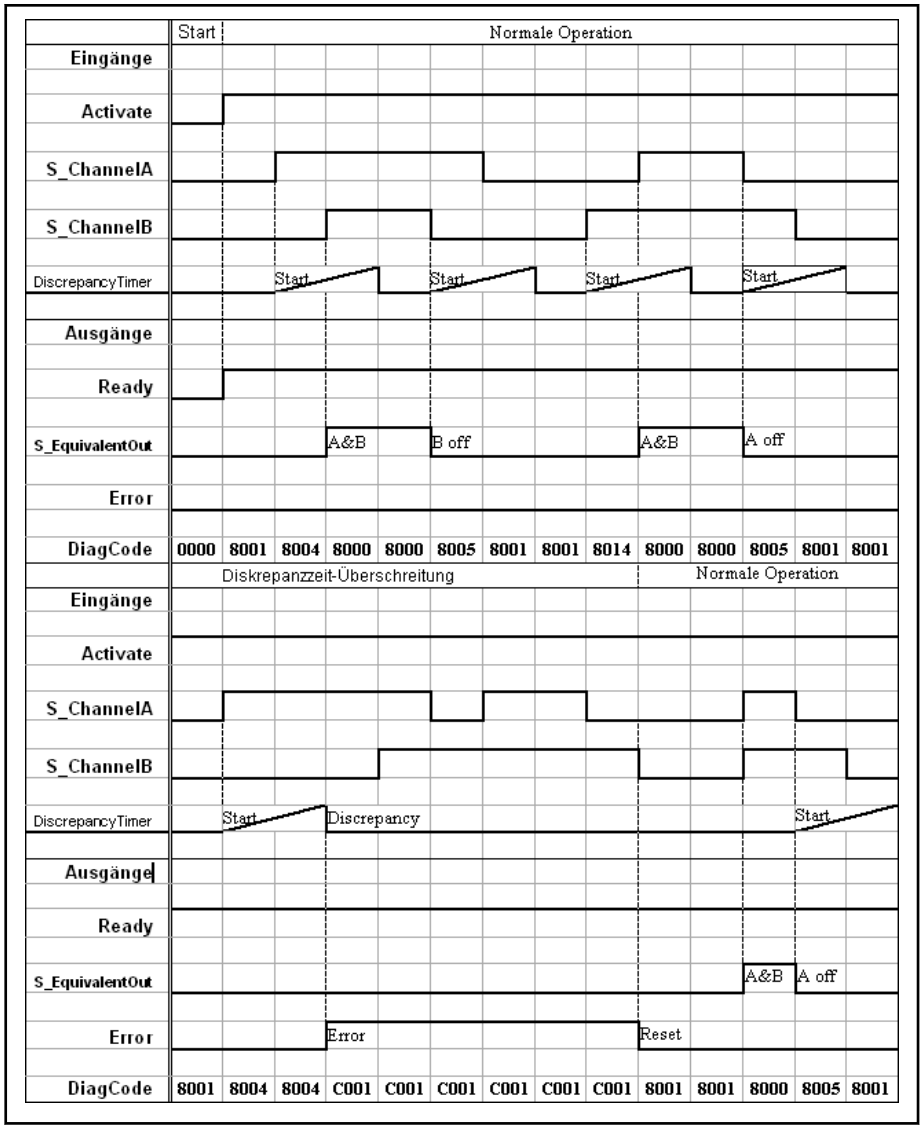


Abb. 16-59: Zeitdiagramm SF_Equivalent

Fehlererkennung

Der Baustein überwacht die Diskrepanzzeit zwischen Kanal A und Kanal B, sowohl wenn auf TRUE, als auch wenn auf FALSE umgeschaltet wird.

Fehlerverhalten

S_EquivalentOut wird auf FALSE gesetzt. Error wird auf TRUE gesetzt. DiagCode zeigt den Fehlerstatus an. Es gibt keinen als Eingang definierten Reset, der mit dem Reset eines Fehlers gekoppelt ist. Sobald ein Set von Eingangswerten mit korrektem S_EquivalentOut anliegt, ist Error wieder FALSE.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Error 1	Diskrepanzzeit abgelaufen in Status 16#8004 Ready = TRUE S_EquivalentOut = FALSE Error = TRUE
16#C002	Error 2	Diskrepanzzeit abgelaufen in Status 16#8014 Ready = TRUE S_EquivalentOut = FALSE Error = TRUE
16#C003	Error 3	Diskrepanzzeit abgelaufen in Status 16#8005 Ready = TRUE S_EquivalentOut = FALSE Error = TRUE

Tab. 16-130: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_EquivalentOut =FALSE Error = FALSE
16#8001	Init	Der Baustein wurde aktiviert. Ready = TRUE S_EquivalentOut = FALSE Error = FALSE
16#8000	Safety Output Enabled	Die Eingänge sind im Äquivalent-Modus auf TRUE geschaltet. Ready =TRUE S_EquivalentOut = TRUE Error = FALSE
16#8004	Wait for Channel B	Kanal A ist auf TRUE umgeschalten worden - Warten auf Kanal B; Diskrepanz-Timer gestartet. Ready = TRUE S_EquivalentOut = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8014	Wait for Channel A	Kanal B ist auf TRUE umgeschalten worden -Warten auf Kanal A; Diskrepanz-Timer gestartet. Ready = TRUE S_EquivalentOut = FALSE Error = FALSE
16#8005	From Active Wait	Ein Kanal ist auf FALSE umgeschalten worden; Warten, dass der andere Kanal auf FALSE geschalten wird, Diskrepanz-Timer gestartet. Ready = TRUE S_EquivalentOut = FALSE Error = FALSE

Tab. 16-131: Bausteinspezifische Status-Codes

16.4.7 SF_ESPE

Elector-Sensitve Protective Equipment

Kurzbeschreibung

SF_ESPE ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser sicherheitsgerichtete Baustein dient zur Überwachung einer berührungslos wirkenden Schutzeinrichtung.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-132: Bibliothekszuordnung

Schnittstellenbeschreibung

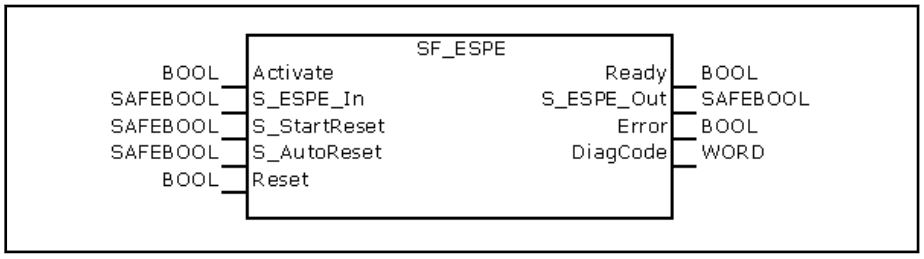


Abb. 16-60: Funktionsbaustein SF_ESPE

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_ESPE_In	SAFEBOOL	Eingang, der Sicherheit anfordert Variable FALSE: ESPE angesteuert, Anforderung für sicherheitsgerichtete Antwort TRUE: ESPE nicht angesteuert, keine Anforderung für sicherheitsgerichtete Antwort Sicherheitssteuerung muss eine sehr kurze Unterbrechung des Sensors entdecken können, wenn die ESPE in Anwendungen als Auslöser verwendet wird

E/A-TYP	Name	Datentyp	Beschreibung
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_AutoReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_ESPE_Out	SAFEBOOL	Ausgang für sicherheitsgerichtete Antwort FALSE: Sicherheits-Ausgang ist inaktiv Anforderung für sicherheitsgerichtete Antwort (zum Beispiel Reset erforderlich oder interne Fehler aktiv) TRUE: Sicherheits-Ausgang ist aktiviert. Keine Anforderung für sicherheitsgerichtete Antwort
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-133: Schnittstelle SF_ESPE

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_ESPE_In	SAFEBOOL	FALSE	
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
S_AutoReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset wenn NOT-HALT-Knopf entriegelt ist TRUE: Automatischer Reset, wenn NOT-HALT-Knopf entriegelt ist
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-134: Eingangsverhalten SF_ESPE

Funktionsbeschreibung

Dieser Baustein ist ein sicherheitsgerichteter Baustein um berührungslos wirkende Schutzeinrichtungen (ESPE) zu überwachen. Seine Funktion ist identisch mit SF_EmergencyStop. Das S_ESPE_Out-Ausgangssignal wird auf FALSE gesetzt, sobald das S_ESPE-In-Eingangssignal FALSE wird. Das S_ESPE_Out-Ausgangssignal wird nur auf TRUE gesetzt, wenn der S_ESPE_In-Eingang TRUE ist und ein Reset erfolgt. Der aktive Reset hängt von den definierten Eingängen S_StartReset, S_AutoReset und Reset ab.

Wenn S_AutoReset = TRUE : Bestätigung ist automatisch.

Wenn S_AutoReset = FALSE : Eine steigende Triggerflanke am Reset-Eingang muss für die Bestätigung der Freigabe verwendet werden.

Wenn S_StartReset = TRUE : Bestätigung ist automatisch, wenn die S-SPS das erste Mal gestartet wird.

Wenn S_StartReset = FALSE : Eine steigende Triggerflanke am Reset-Eingang muss für die Bestätigung der Freigabe verwendet werden.

⚠ VORSICHT **S_StartReset- und S_AutoReset**

Die S_StartReset- und S_AutoReset-Eingänge sollen nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdung entstehen kann, wenn die S-SPS startet.

Die Funktion des Bausteine SF_ESPE und SF_EmergencyStop ist identisch. Deshalb wird für weitere Details auf den Baustein SF_EmergencyStop ("Funktionsbeschreibung" auf Seite 315) verwiesen.

16.4.8 SF_GuardLocking

Safety Guard Interlocking with Locking

Kurzbeschreibung

SF_GuardLocking ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein kontrolliert den Eingang zu einem Gefahrenbereich durch eine verriegelte Schutzeinrichtung mit Zuhaltung (vierstufige Verriegelung).

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-135: Bibliothekszuordnung

Schnittstellenbeschreibung

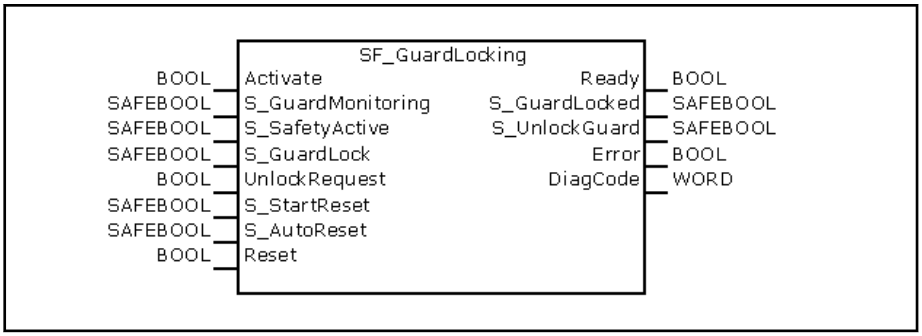


Abb. 16-61: Funktionsbaustein SF_GuardLocking

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_GuardMonitoring	SAFEBOOL	Variable Überwacht die verriegelte trennende Schutzeinrichtung FALSE: Schutzeinrichtung offen TRUE: Schutzeinrichtung geschlossen
	S_SafetyActive	SAFEBOOL	Variable Status des Gefahrenbereichs (EDM), zum Beispiel, basierend auf Geschwindigkeitsüberwachung oder sichere Rückfallverzögerungszeit FALSE: Maschine ist in nicht-sicherem Status TRUE: Maschine ist in sicherem Status

E/A-TYP	Name	Datentyp	Beschreibung
	S_GuardLock	SAFEBOOL	Variable Status der mechanischen, verriegelten trennenden Schutzeinrichtung FALSE: Schutzeinrichtung ist nicht verriegelt TRUE: Schutzeinrichtung ist verriegelt
	UnlockRequest	BOOL	Variable Eingreifen durch Operator - Anforderung zur Entriegelung der Schutzeinrichtung FALSE: Keine Anforderung TRUE: Anforderung gestellt
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_AutoReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294 Wird auch für die Anforderung verwendet, dass die Schutzeinrichtung wieder verriegelt werden soll. Die Eigenschaft des Signals muss mit einem manuellen Reset-Gerät konform sein.
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_GuardLocked	SAFEBOOL	Interface zum Gefährdungsbereich, das gestoppt werden muss FALSE: Kein sicherer Zustand TRUE: sicherer Zustand
	S_UnlockGuard	SAFEBOOL	Signal um die Schutzeinrichtung zu entriegeln FALSE: Schließen der Schutzeinrichtung TRUE: Entriegeln der Schutzeinrichtung
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-136: Schnittstelle SF_GuardLocking

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_GuardMonitoring	SAFEBOOL	FALSE	
S_SafetyActive	SAFEBOOL	FALSE	
S_GuardLock	SAFEBOOL	FALSE	
UnlockRequest	BOOL	FALSE	
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)

Name	Typ	Default-Wert	Übernahme
S_AutoReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset wenn NOT-HALT-Knopf entriegelt ist TRUE: Automatischer Reset, wenn NOT-HALT-Knopf entriegelt ist
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-137: Eingangsverhalten SF_GuardLocking

Funktionsbeschreibung

Dieser Baustein kontrolliert eine verriegelte Schutzeinrichtung und überwacht die Position der Schutzeinrichtung und der Verriegelung mit Zuhaltung. SF_GuardLocking kann mit einem mechanisch verriegelten Schalter verwendet werden.

Der Operator stellt die Anforderung, Zugang zu einem Gefährdungsbereich zu erhalten. Die Schutzeinrichtung kann nur entriegelt werden, wenn sich der Gefährdungsbereich in einem sicheren Zustand befindet. Die Schutzeinrichtung kann verriegelt werden, wenn sie geschlossen ist. Die Maschine kann gestartet werden, wenn die Schutzeinrichtung geschlossen und verriegelt ist. Eine offene oder entriegelte Schutzeinrichtung wird im Falle einer sicherheitskritischen Situation erkannt.

⚠ VORSICHT **S_StartReset- und S_AutoReset**

Die S_StartReset- und S_AutoReset-Eingänge sollen nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdung entstehen kann, wenn die S-SPS startet.

1.	External	Anforderung, den Gefährdungsbereich einen sicheren Zustand zu bringen - kein Bestandteil dieses Bausteins
2.	In	Rückmeldung des maßgeblichen Gefährdungsbereichs, dass er sich in einem sicheren Zustand befindet (über S_SafetyActive)
3.	In	Anforderung des Operators die Schutzeinrichtung zu entriegeln. (über UnlockRequest)
4.	Out	Das Öffnen der Schutzeinrichtung zulassen
5.	In	Zuhaltung der Schutzeinrichtung entriegelt (über S_GuardLock). Schutzeinrichtung kann jetzt geöffnet werden (S_GuardLocked = FALSE) Operator öffnet die trennende Schutzeinrichtung
6.	In	Überwachung des Zustands der Schutzeinrichtung über S_GuardMonitoring. Zeigt an, wenn die Schutzeinrichtung wieder geschlossen wird
7.	In	Rückmeldung des Operators, den Gefährdungsbereich wieder zu aktivieren (Reset)
8.	Out	Verriegeln der Schutzeinrichtung (S_UnlockGuard)
9.	In	Überprüfen ob Schutzeinrichtung verriegelt ist (S_GuardLock)
10.	Out	Gefährdungsbereich ist bereit zum operativen Betrieb (S_GuardLocked = TRUE)
11.	Extern	Restart des Betriebs im Gefährdungsbereichs

Tab. 16-138: Funktionsablauf

Statusdiagramm

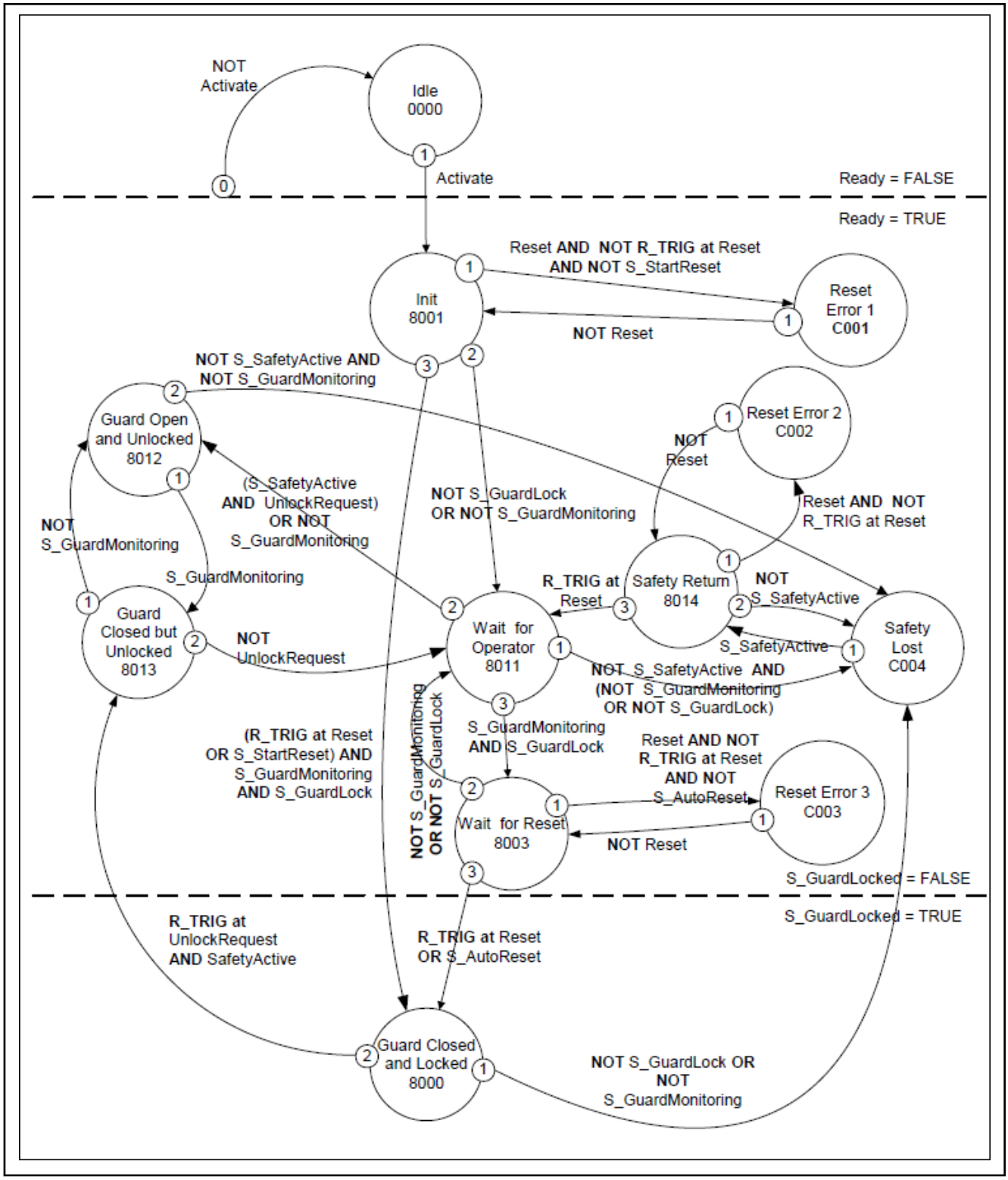


Abb. 16-62: Status SF_GuardLocking

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typische Zeitdiagramme

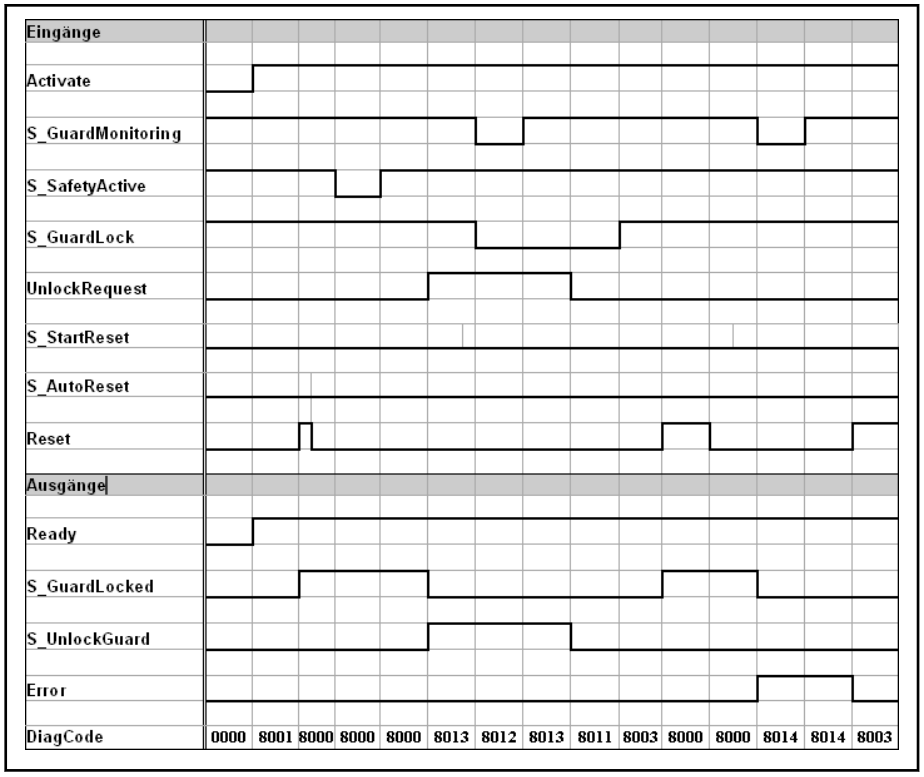


Abb. 16-63: Zeitdiagramm SF_GuardLocking

Fehlererkennung

Statische Signale an Reset werden erkannt. Fehler werden an den Schaltern der Schutzeinrichtung entdeckt.

Fehlerverhalten

Bei einem Fehlerereignis werden die Ausgänge S_GuardLocked und S_UnlockGuard auf FALSE gesetzt, der Ausgang DiagCode zeigt den relevanten Fehlercode an und der Ausgang Error wird auf TRUE gesetzt.

Ein Fehler muss durch eine steigende Triggerflanke am Reset-Eingang bestätigt werden.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error 1	Statischer Reset entdeckt in Status 16#8001 Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = TRUE
16#C002	Reset Error 2	Statischer Reset entdeckt in Status 16#C004 Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C003	Reset Error 3	Statischer Reset entdeckt in Status 16#8011 Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = TRUE
16#C004	Safety Lost	kein sicherer Zustand mehr, Schutzeinrichtung geöffnet oder Zuhaltung entriegelt Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = TRUE

Tab. 16-139: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE
16#8000	Guard Closed and Locked	Schutzeinrichtung ist verriegelt. Ready = TRUE S_GuardLocked = TRUE S_UnlockGuard = FALSE Error = FALSE
16#8001	Init	Baustein wurde aktiviert und initialisiert. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE
16#8003	Wait for Reset	Tür ist geschlossen und verriegelt, warten auf Reset durch Operator. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE
16#8011	Wait for Operator	Warten, dass Operator entweder eine Entriegelung anfordert oder Reset tätigt. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8012	Guard Open and Unlocked	Schutzeinrichtung ist entriegelt und offen. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = TRUE Error = FALSE
16#8013	Guard Closed but Unlocked	Schutzeinrichtung ist geschlossen, aber entriegelt. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = TRUE Error = FALSE
16#8014	Safety Return	Das Signal S_SafetyActive liegt an, Warten auf Bestätigung durch Operator. Ready = TRUE S_GuardLocked = FALSE S_UnlockGuard = FALSE Error = FALSE

Tab. 16-140: Bausteinspezifische Status-Codes (kein Fehler)

16.4.9 SF_GuardMonitoring

Kurzbeschreibung

SF_GuardMonitoring ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein überwacht die relevante Sicherheits-Schutzeinrichtung mit einer zweistufigen Verriegelung.

Er wird verwendet um in einer Applikation die Funktion "Schutzeinrichtung ohne Zuhaltung" (Schutztür-Überwachung) zu unterstützen.

Es gibt zwei unabhängige Eingangsparameter für zwei Schalter der Sicherheits-Schutzeinrichtung. Diese zwei unabhängigen Eingangsparameter sind mit der Zeitdifferenz (MonitoringTime) gekoppelt, um die Schutzeinrichtung zu schließen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-141: Bibliothekszuordnung

Schnittstellenbeschreibung

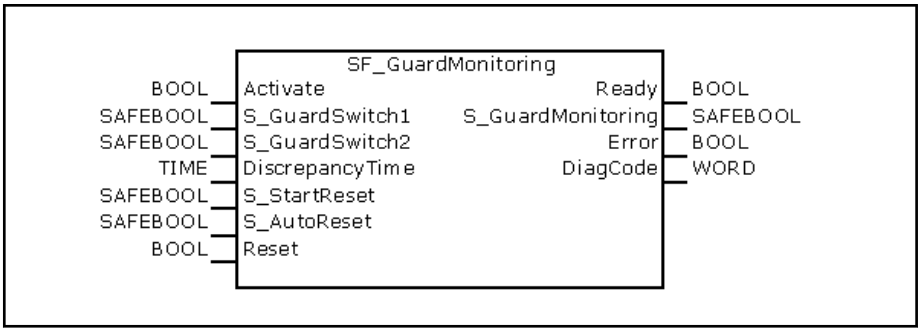


Abb. 16-64: Funktionsbaustein SF_GuardMonitoring

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_GuardSwitch1	SAFEBOOL	Variable Schutzschalter 1 Eingang FALSE: Schutzeinrichtung offen TRUE: Schutzeinrichtung geschlossen
	S_GuardSwitch2	SAFEBOOL	Variable Schutzschalter 2 Eingang FALSE: Schutzeinrichtung offen TRUE: Schutzeinrichtung geschlossen
	DiscrepancyTime	TIME	Konstante Legt die überwachte synchrone Zeit zwischen S_GuardSwitch1 und S_GuradSwitch2 fest
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_AutoReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_GuardMonitoring	SAFEBOOL	Ausgang, zur Statusanzeige der Schutzeinrichtung FALSE: Schutzeinrichtung ist nicht aktiv TRUE: S_GuardSwitch1 und S_GuardSwitch2 sind TRUE, kein Fehler und Bestätigung. Schutzeinrichtung ist aktiv
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-142: Schnittstelle SF_GuardMonitoring

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_GuardSwitch1	SAFEBOOL	FALSE	
S_GuardSwitch2	SAFEBOOL	FALSE	
DiscrepancyTime	TIME	T#0ms	Konstante

Name	Typ	Default-Wert	Übernahme
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
S_AutoReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset wenn NOT-HALT-Knopf entriegelt ist TRUE: Automatischer Reset, wenn NOT-HALT-Knopf entriegelt ist
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-143: Eingangsverhalten SF_GuardMonitoring

Funktionsbeschreibung

Dieser Baustein benötigt zwei Eingänge, welche die Position der Sicherheits-Schutzeinrichtung mit zwei Schaltern abbilden: einen Diskrepanzzeit-Eingang und einen Reset-Eingang. Hat die Sicherheits-Schutzeinrichtung nur einen Schalter, so können die Eingänge S_GuardSwitch1 und S_GuardSwitch2 überbrückt werden. Die Überwachungszeit ist die maximale Zeit, innerhalb der die beiden Schalter geschlossen werden müssen, damit die Sicherheits-Schutzeinrichtung schließt. Die Eingänge Reset, S_StartReset und S_AutoReset legen fest, wie der Baustein, nachdem die Sicherheits-Schutzeinrichtung geöffnet wurde, zurückgesetzt wird.

Wenn die Sicherheits-Schutzeinrichtung geöffnet wird, sollten die beiden Eingänge S_GuardSwitch1 und S_GuardSwitch2 auf FALSE schalten. Der Ausgang S_GuardMonitoring schaltet auf FALSE, sobald einer der beiden Eingänge S_GuardSwitch1 oder S_GuardSwitch2 auf FALSE gesetzt sind. Um die Sicherheits-Schutzeinrichtung zu schließen, müssen beide Eingänge S_GuardSwitch1 und S_GuardSwitch2 auf TRUE schalten.

SF_GuardMonitoring überwacht die Symmetrie des Schaltvorgangs. Der S_GuardMonitoring-Ausgang bleibt FALSE, wenn nur einer der Kontakte den Öffnen/Schließen-Prozess beendet hat.

Das Verhalten des S_GuardMonitoring-Ausgangs hängt von der Zeitdifferenz beider schaltender Eingänge ab. Die Diskrepanzzeit wird überwacht, sobald sich der Wert der beiden Eingänge S_GuardSwitch1/S_GuardSwitch2 unterscheidet. Wenn die Diskrepanzzeit überschritten wurde, sich beide Eingänge aber immer noch unterscheiden, bleibt der S_GuardMonitoring-Ausgang auf FALSE. Wenn der zweite, entsprechende Eingang S_GuardSwitch1/ S_GuardSwitch2 innerhalb der Diskrepanzzeit auf TRUE schaltet, wird der S_GuardMonitoring-Ausgang auf TRUE gesetzt, mit anschließender Bestätigung.

⚠ VORSICHT **S_StartReset- und S_AutoReset**

Die S_StartReset- und S_AutoReset-Eingänge sollen nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdung entstehen kann, wenn das S-SPS startet.

Statusdiagramm

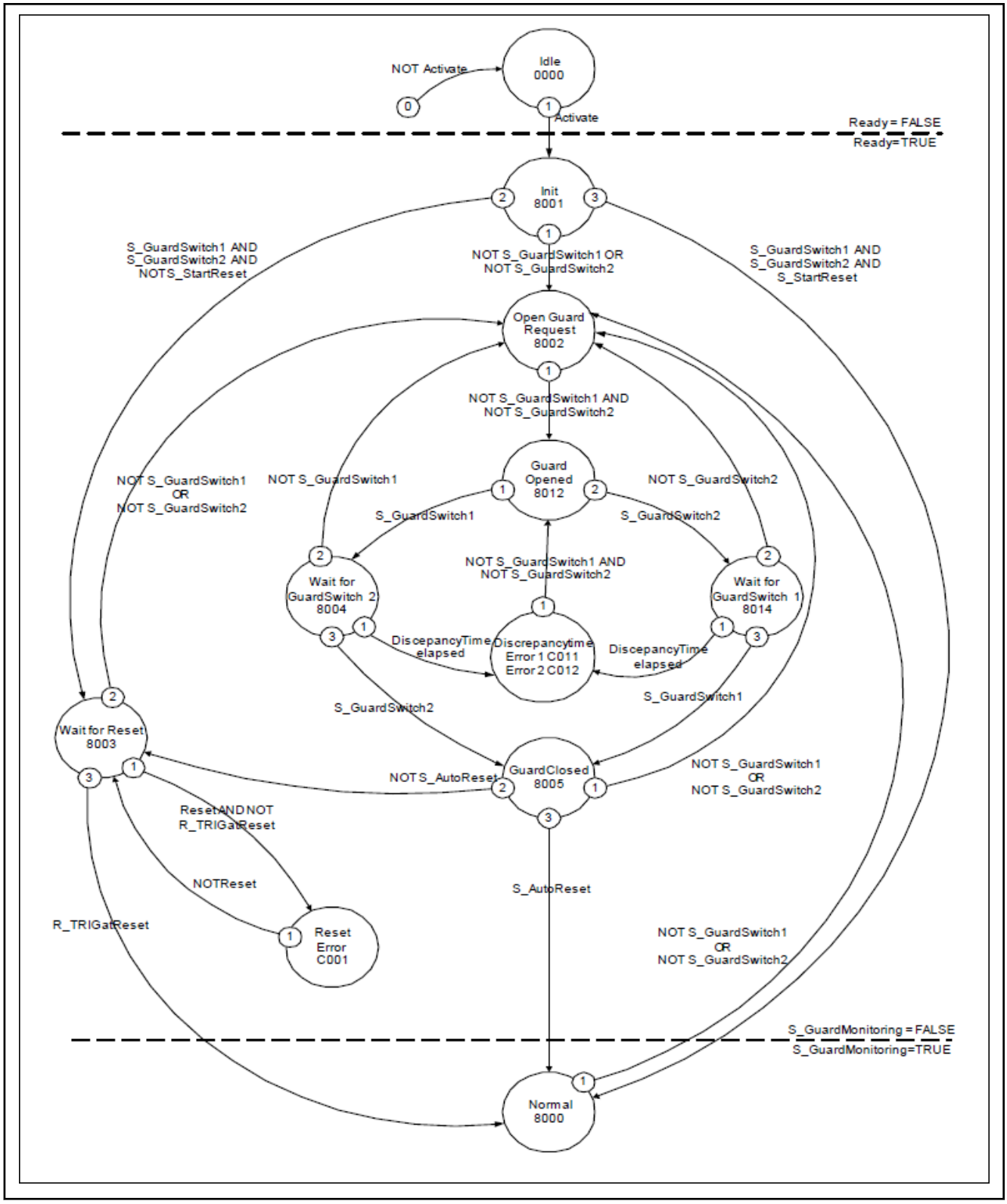


Abb. 16-65: Statusdiagramm SF_GuardMonitoring

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typische Zeitdiagramme

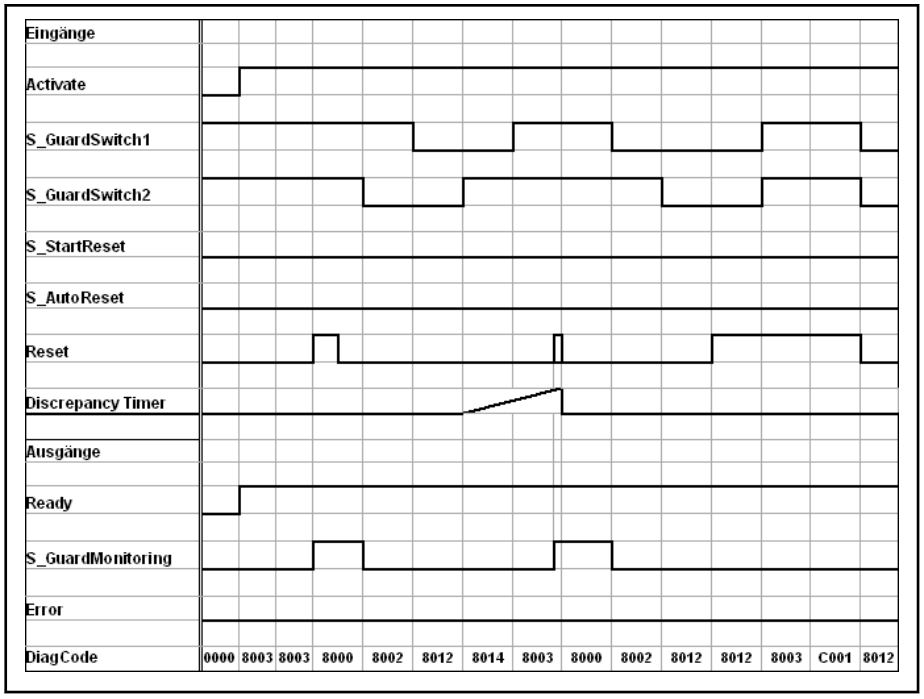


Abb. 16-66: Zeitdiagramm1 SF_GuardMonitoring

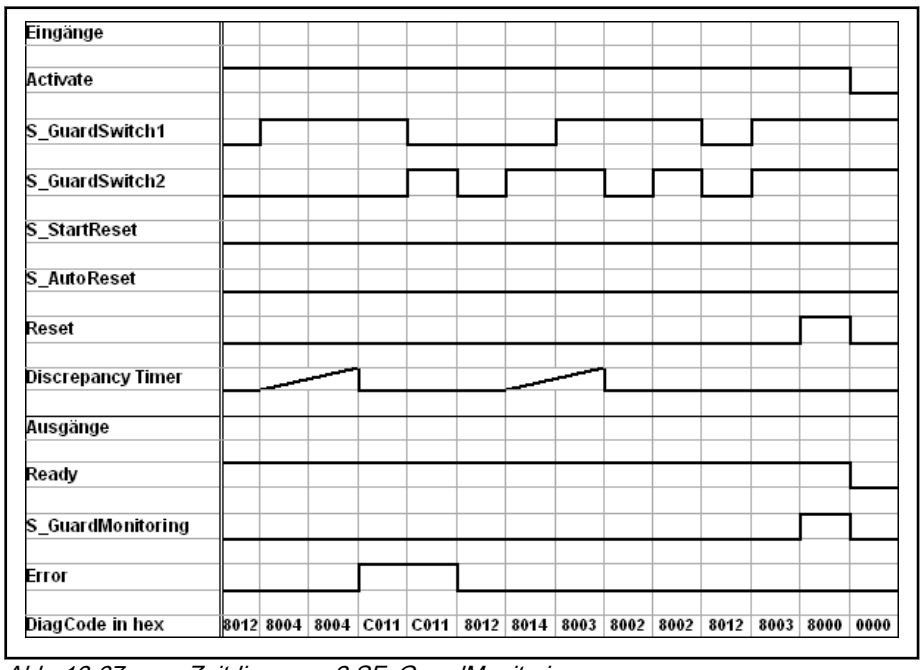


Abb. 16-67: Zeitdiagramm2 SF_GuardMonitoring

Fehlererkennung

Externe Signale: SAFEBOOL-Eingänge bieten eine eigene Fehlererkennung. Mechanischer Setup kombiniert diejenige eines offenen und geschlossenen Schalters entsprechend der Norm EN954. Diskrepanzzeitüberwachung für den Zeitverzögerung zwischen beiden mechanischen Schalterreaktionen, entsprechend der Norm EN954.

Ein Fehler wird erkannt, wenn die Zeitverzögerung zwischen dem ersten und dem zweiten S_GuardSwitch1/S_GuardSwitch2 -Eingang größer als der Wert des DiscrepancyTime-Eingangs ist. Der Fehlerausgang wird auf TRUE gesetzt.

Der Baustein erkennt ein statisches TRUE-Signal am RESET-Eingang.

Fehler- und Reset-Verhalten

Der S_GuardMonitoring-Ausgang wird auf FALSE gesetzt. Wenn die beiden Eingänge S_GuardSwitch1 und S_GuardSwitch2 gebrückt werden, wird kein Fehler erkannt. Um den Reset-Fehlerzustand zu verlassen, muss der Reset-Eingang auf FALSE gesetzt werden. Um die Diskrepanzzeit-Fehler zu verlassen, müssen beide Eingänge S_GuardSwitch1 und S_GuardSwitch2 auf FALSE gesetzt werden.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error	Statischer Reset entdeckt in Zustand 16#8003. Ready = TRUE S_GuardMonitoring = FALSE Error = TRUE
16#C011	Discrepancytime Error 1	Diskrepanzzeit abgelaufen in Zustand 16#8004 Ready = TRUE S_GuardMonitoring = FALSE Error = TRUE
16#C012	Discrepancytime Error 1	Diskrepanzzeit abgelaufen in Zustand 16#8014 Ready = TRUE S_GuardMonitoring = FALSE Error = TRUE

Tab. 16-144: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_GuardMonitoring = FALSE Error = FALSE
16#8000	Normal	Sicherheits-Schutzeinrichtung geschlossen und Sicherheitszustand bestätigt. Ready = TRUE S_GuardMonitoring = TRUE Error = FALSE
16#8001	Init	Baustein wurde aktiviert. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
16#8002	Open Guard Request	Komplette Schaltfolge erforderlich. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8003	Wait for Reset	Warten auf steigende Triggerflanke an Reset. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
16#8012	Guard opened	Schutzeinrichtung vollständig geöffnet. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
16#8004	Wait for GuardSwitch 2	S_GuardSwitch1 wurde auf TRUE geschalten - Warten auf S_GuardSwitch2; Diskrepanz-Timer gestartet. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
16#8014	Wait for GuardSwitch 1	S_GuardSwitch2 wurde auf TRUE geschalten - Warten auf S_GuardSwitch2; Diskrepanz-Timer gestartet. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE
16#8005	Guard Closed	Schutzeinrichtung geschlossen. Warten auf Reset, wenn S_AutoReset = FALSE. Ready = TRUE S_GuardMonitoring = FALSE Error = FALSE

Tab. 16-145: Bausteinspezifische Status-Codes (kein Fehler)

16.4.10 SF_ModeSelector

Kurzbeschreibung

SF_ModeSelector ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein wählt die System-Betriebsart, wie manuell, automatisch, halb-automatisch, usw.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.1.0.0

Tab. 16-146: Bibliothekszuordnung

Schnittstellenbeschreibung

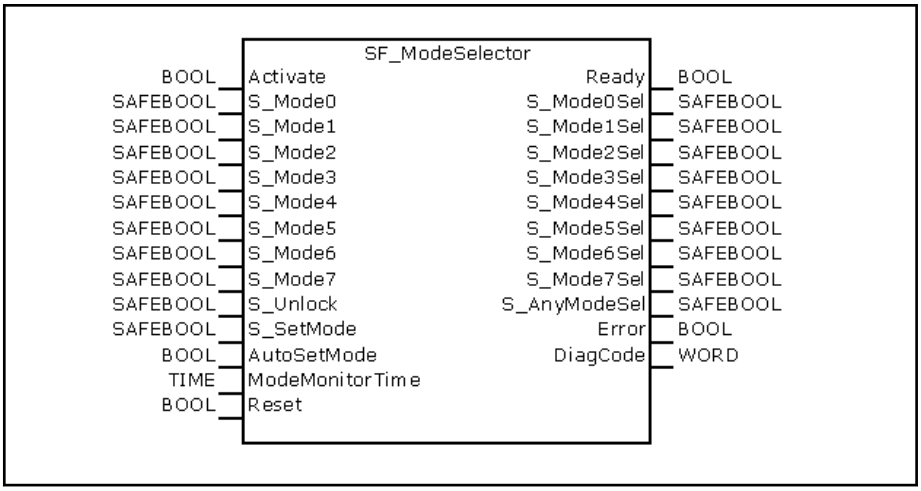


Abb. 16-68: Funktionsbaustein SF_ModeSelector

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_Mode0	SAFEBOOL	Variable oder Konstante Eingang 0 des Modus-Auswahl Schalters FALSE: Modus 0 wird vom Operator nicht angefordert TRUE: Modus 0 wird vom Operator angefordert
	S_Mode1	SAFEBOOL	Variable oder Konstante Eingang des Modus-Auswahl Schalters FALSE: Modus 1 wird vom Operator nicht angefordert TRUE: Modus 1 wird vom Operator angefordert
	S_Mode2	SAFEBOOL	Variable oder Konstante Eingang 2 des Modus-Auswahl Schalters FALSE: Modus 2 wird vom Operator nicht angefordert TRUE: Modus 2 wird vom Operator angefordert
	S_Mode3	SAFEBOOL	Variable oder Konstante Eingang 3 des Modus-Auswahl Schalters FALSE: Modus 3 wird vom Operator nicht angefordert TRUE: Modus 3 wird vom Operator angefordert
	S_Mode4	SAFEBOOL	Variable oder Konstante Eingang 4 des Mode-Selector Schalters FALSE: Modus 4 wird vom Operator nicht angefordert TRUE: Modus 4 wird vom Operator angefordert
	S_Mode5	SAFEBOOL	Variable oder Konstante Eingang 5 des Modus-Auswahl Schalters FALSE: Modus 5 wird vom Operator nicht angefordert TRUE: Modus 5 wird vom Operator angefordert

E/A-TYP	Name	Datentyp	Beschreibung
	S_Mode6	SAFEBOOL	Variable oder Konstante Eingang 6 des Modus-Auswahl Schalters FALSE: Modus 6 wird vom Operator nicht angefordert TRUE: Modus 6 wird vom Operator angefordert
	S_Mode7	SAFEBOOL	Variable oder Konstante Eingang 7 des Modus-Auswahl Schalters FALSE: Modus 7 wird vom Operator nicht angefordert TRUE: Modus 7 wird vom Operator angefordert
	S_Unlock	SAFEBOOL	Variable oder Konstante Sperrt den ausgewählten Modus FALSE: Der aktuelle S_ModeXSel Ausgang ist gesperrt, deshalb führt eine Änderung irgendeines S_ModeX-Eingangs nicht zu einer Änderung am S_ModeXSel-Ausgang, sogar beim Ereignis einer steigenden Flanke des Set-Modus TRUE: Der ausgewählte S_ModeXSel ist nicht gesperrt; eine Änderung der Modus-Auswahl ist möglich
	S_SetMode	SAFEBOOL	Variable (oder Konstante FALSE, falls AutoSetMode = TRUE) Setzt den ausgewählten Modus Operator bestätigt das Setzen eines Modus. Jede Änderung zu einem neuen S_ModeX = TRUE führt zu S_AnyModeSel/ S_ModeXSel = FALSE, nur ein ansteigende SetMode Triggerflanke führt dann zu einem neuen S_ModeXSel = TRUE
	AutoSetMode	BOOL	Konstante Parametriert den Bestätigungsmodus FALSE: Eine Änderung des Modus muss vom Operator über SetMode bestätigt werden TRUE: Eine gültige Änderung des Eingangs S_ModeX in einen anderen S_ModeX führt automatisch zu S_ModeXSel ohne Operator-Bestätigung über SetMode (so lange, wie dieser nicht durch S_Unlock gesperrt ist)
	ModeMonitorTime	TIME	Kontante Maximal zulässige Zeit für Änderung des Auswahl-Eingangs
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_Mode0Sel	SAFEBOOL	Zeigt an, dass Modus 0 ausgewählt und bestätigt ist FALSE: Modus 0 ist nicht ausgewählt oder nicht aktiv TRUE: Modus 0 ist ausgewählt und aktiv
	S_Mode1Sel	SAFEBOOL	Zeigt an, dass Modus 1 ausgewählt und bestätigt ist FALSE: Modus 1 ist nicht ausgewählt oder nicht aktiv TRUE: Modus1 ist ausgewählt und aktiv

E/A-TYP	Name	Datentyp	Beschreibung
	S_Mode2Sel	SAFEBOOL	Zeigt an, dass Modus 2 ausgewählt und bestätigt ist FALSE: Modus 2 ist nicht ausgewählt oder nicht aktiv TRUE: Modus 2 ist ausgewählt und aktiv
	S_Mode3Sel	SAFEBOOL	Zeigt an, dass Modus 3 ausgewählt und bestätigt ist FALSE: Modus 3 ist nicht ausgewählt oder nicht aktiv TRUE: Modus 3 ist ausgewählt und aktiv
	S_Mode4Sel	SAFEBOOL	Zeigt an, dass Modus 4 ausgewählt und bestätigt ist FALSE: Modus 4 ist nicht ausgewählt oder nicht aktiv TRUE: Modus 4 ist ausgewählt und aktiv
	S_Mode5Sel	SAFEBOOL	Zeigt an, dass Modus 5 ausgewählt und bestätigt ist FALSE: Modus 5 ist nicht ausgewählt oder nicht aktiv TRUE: Modus 5 ist ausgewählt und aktiv
	S_Mode6Sel	SAFEBOOL	Zeigt an, dass Modus 6 ausgewählt und bestätigt ist FALSE: Modus 6 ist nicht ausgewählt oder nicht aktiv TRUE: Modus 6 ist ausgewählt und aktiv
	S_Mode7Sel	SAFEBOOL	Zeigt an, dass Modus 7 ausgewählt und bestätigt ist FALSE: Modus 7 ist nicht ausgewählt oder nicht aktiv TRUE: Modus 7 ist ausgewählt und aktiv
	S_AnyModeSel	SAFEBOOL	Zeigt an, dass irgendeiner der 8 Modi ausgewählt und bestätigt ist FALSE: Kein S_ModeX ist ausgewählt TRUE: Einer der 8 S_ModeX ist ausgewählt und aktiv
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-147: Schnittstelle SF_ModeSelector

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_Mode0	SAFEBOOL	FALSE	S_SetMode
S_Mode1	SAFEBOOL	FALSE	S_SetMode
S_Mode2	SAFEBOOL	FALSE	S_SetMode
S_Mode3	SAFEBOOL	FALSE	S_SetMode
S_Mode4	SAFEBOOL	FALSE	S_SetMode
S_Mode5	SAFEBOOL	FALSE	S_SetMode
S_Mode6	SAFEBOOL	FALSE	S_SetMode
S_Mode7	SAFEBOOL	FALSE	S_SetMode
S_Unlock	SAFEBOOL	FALSE	kontinuierlich
S_SetMode	SAFEBOOL	FALSE	kontinuierlich
AutoSetMode	BOOL	FALSE	Konstante

Name	Typ	Default-Wert	Übernahme
ModeMonitorTime	TIME	T#0	Konstante
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-148: Eingangsverhalten SF_ModeSelector

Funktionsbeschreibung

Dieser Baustein wählt die System-Betriebsart, wie manuell, automatisch, halb-automatisch, usw. Beim Starten der Steuerung sollte vorausgesetzt sein, dass sich die Maschine in einem sicheren Modus befindet. Beim Starten der Maschine muss der Übergang zum Modus, der durch den Mode-Selector gesetzt wurde, durch einen Baustein-Eingang (z. B. START-Knopf der Maschine) initiiert werden.

Die dem Standard-Zustand folgende Aktivierung des Bausteins ist der Zustand ModeChanged. Dieser ist auch der sichere Zustand des Bausteins, wo alle S_ModeXSel und S_AnyModeSel auf FALSE sind.

Wenn sich der Baustein im Zustand ModeChanged befindet:

- Der neue S_ModeX Eingang muss mit einer steigenden S_SetMode Triggerflanke (falls AutoSetMode = FALSE) bestätigt werden, der zu einem neuen S_ModeXSel-Ausgang führt.
- Der neue S_ModeX Eingang führt automatisch zu einem S_ModeXSel Ausgang (falls AutoSetMode = TRUE).
- Solch ein Übergang von Zustand 16#8005 zu 16#8000 ist nur gültig, wenn ein S_ModeX-Eingang TRUE ist. So lange wie alle S_ModeX FALSE sind, so lange bleibt der Baustein in Zustand 16#8005, sogar wenn S_SetMode triggert.

Der Übergang vom ModeChanged zum ModeSelected-Zustand, das heißt S_SetMode wird durch den Operator gesetzt, wird nicht durch einen Timer überwacht.

Wenn sich der Baustein im ModeSelected-Zustand befindet, führt das gleichzeitige Auftreten eines neuen S_ModeX-Eingangs (höhere Priorität) und das Signal NOT S_Unlock (niedrigere Priorität) zum ModeChanged-Zustand.

Die S_ModeX-Eingangsparameter, die nicht zur Modus-Auswahl verwendet werden, sollten mit dem Standardwert FALSE belegt werden, um die Verifikation des Programms zu vereinfachen.

⚠ VORSICHT **AutoSetMode**

Der AutoSetMode-Eingang soll nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdung auftreten kann, wenn die S-SPS gestartet wird.

Statusdiagramm

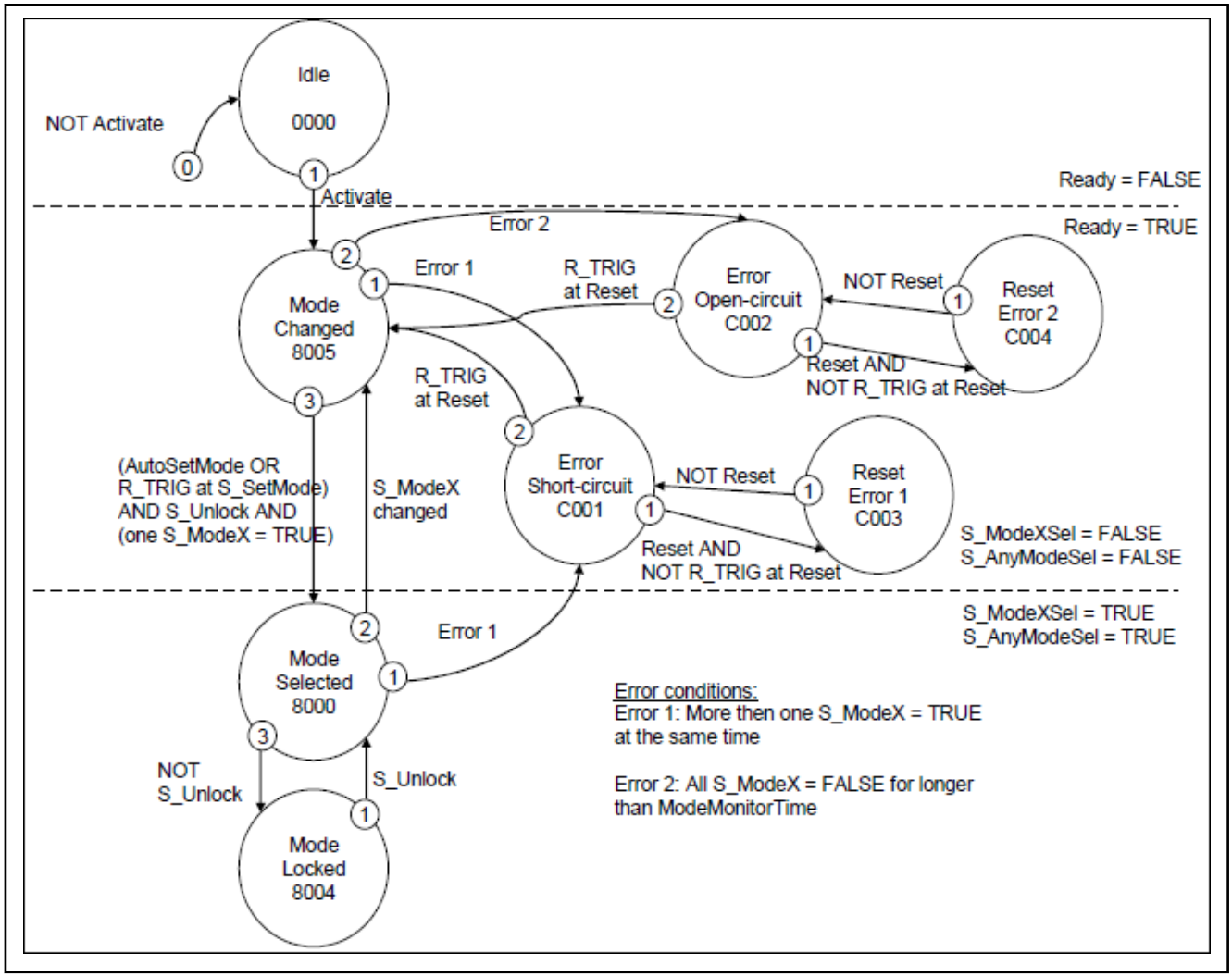


Abb. 16-69: Statusdiagramm SF_ModeSelector

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typische Zeitdiagramme

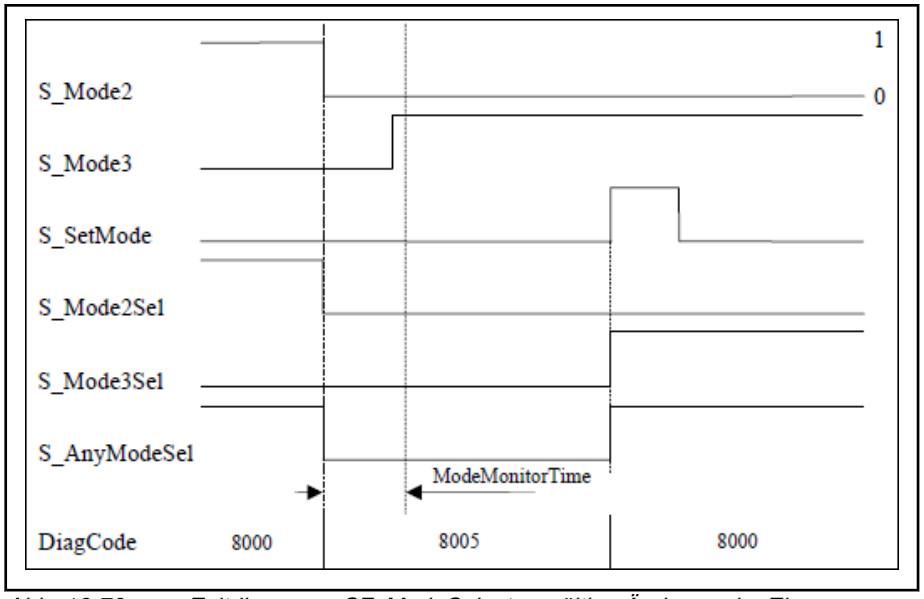


Abb. 16-70: Zeitdiagramm SF_ModeSelector, gültige Änderung im Eingangs-Modus mit Bestätigung

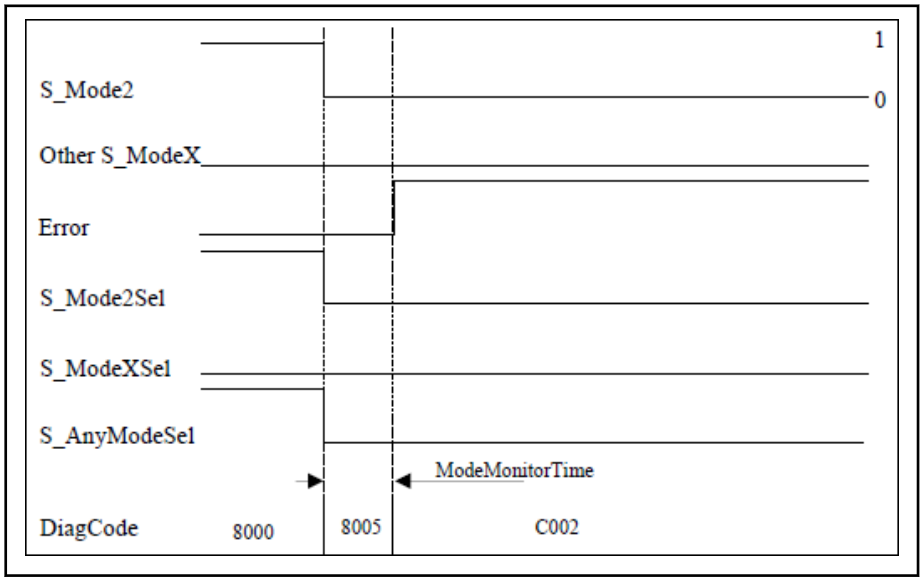


Abb. 16-71: Zeitdiagramm SF_ModeSelector,Fehlerbedingung 2 bei Modus-Eingaben

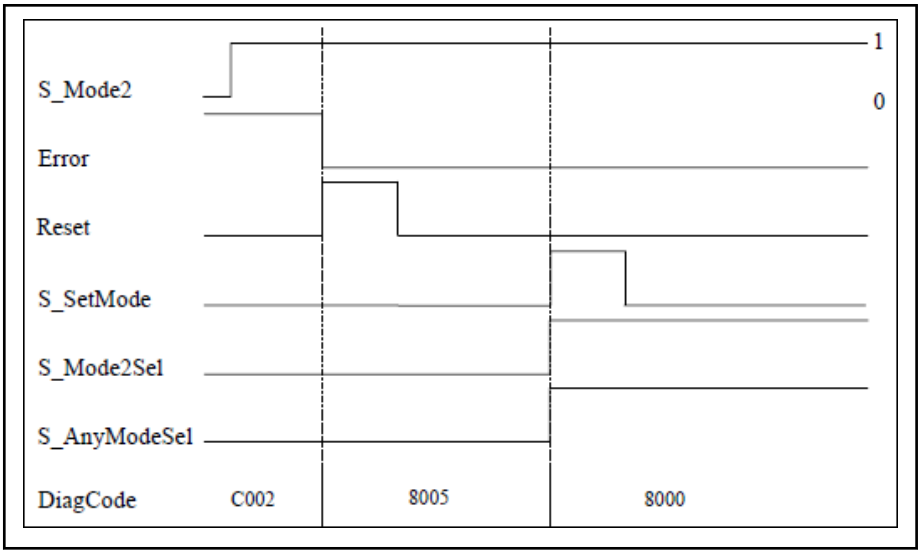


Abb. 16-72: Zeitdiagramm SF_ModeSelector, Reset der Fehler-Bedingung

Fehlererkennung

Der Baustein erkennt, wenn keiner der Modus-Eingänge ausgewählt ist. Diese ungültige Bedingung wird nach Ablauf der ModeMonitorTime entdeckt:

- Die bei jeder fallenden Triggerflanke eines S_ModeX Eingangs neu startet
- Die sich dann im Status ModeChanged befindet, welcher der Aktivierung des Bausteins folgt.

Im Gegensatz dazu entdeckt der Baustein direkt, ob mehr als ein S_ModeX Eingang zum gleichen Zeitpunkt ausgewählt ist.

Eine statische Reset-Bedingung wird entdeckt, wenn der Baustein entweder im Fehlerzustand 16#C001 oder 16#C002 ist.

Fehlerverhalten

Bei einem Fehlerereignis werden die Ausgänge S_ModeXSel und S_AnyModeSel in den sicheren Zustand = FALSE gesetzt. Der DiagCode-Eingang zeigt den relevanten Fehlercode an und der Error-Ausgang wird auf TRUE gesetzt.

Ein Fehler muss mit der steigenden Triggerflanke des Reset BOOL Eingangs erkannt werden. Der Baustein wechselt von einem Error-Zustand zum Zustand ModeChanged.

Status- und Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Error Short-circuit	Der Baustein entdeckt, dass zwei oder mehr S_ModeX TRUE sind, zum Beispiel Kurzschluss von Leitungen. Ready = TRUE Error = TRUE S_AnyModeSel = FALSE Alle S_ModeXSel = FALSE
16#C002	Error Open-circuit	Der Baustein entdeckt, dass alle S_ModeX FALSE sind: Das Zeitintervall, das einer fallenden S_ModeX Triggerflanke folgt, überschreitet die ModeMonitorTime, zum Beispiel: Kurzschluss von Leitungen. Ready = TRUE Error = TRUE S_AnyModeSel = FALSE Alle S_ModeXSel = FALSE
16#C003	Reset Error 1	Statisches Reset-Signal entdeckt in Status 16#C001. Ready = TRUE Error = TRUE S_AnyModeSel = FALSE Alle S_ModeXSel = FALSE
16#C004	Reset Error 2	Statisches Reset-Signal entdeckt in Status 16#C002. Ready = TRUE Error = TRUE S_AnyModeSel = FALSE Alle S_ModeXSel = FALSE

Tab. 16-149: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE Error = FALSE S_AnyModeSel = FALSE Alle S_ModeXSel = FALSE
16#8005	ModeChanged	Status nach Aktivierung oder wenn S_ModeX sich geändert hat (außer gesperrt) oder nach Reset eines Fehlerstatus. Ready = TRUE Error = FALSE S_AnyModeSel = FALSE Alle S_ModeXSel = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8000	ModeSelected	Gültige Modus-Auswahl, aber noch nicht gesperrt. Ready = TRUE Error = FALSE S_AnyModeSel = TRUE S_ModeXSel = Ausgewähltes X ist TRUE, andere sind FALSE
16#8004	ModeLocked	Gültige Modus-Auswahl ist gesperrt. Ready = TRUE Error = FALSE S_AnyModeSel = TRUE S_ModeXSel = Ausgewähltes X ist TRUE, andere sind FALSE.

Tab. 16-150: Bausteinspezifische Statuscodes (kein Fehler)

16.4.11 SF_MutingPar

Kurzbeschreibung

1. SF_MutingPar ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.
2. Dieser Baustein weicht von der PLCopen Spezifikation ab, siehe Statusdiagramm, Seite 360

Muting ist das bewusste Unterdrücken der Sicherheitsfunktion.

In diesem Baustein ist das parallele Muting mit vier Mutingsensoren realisiert.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.1.0.0

Tab. 16-151: Bibliothekszuordnung

Schnittstellenbeschreibung

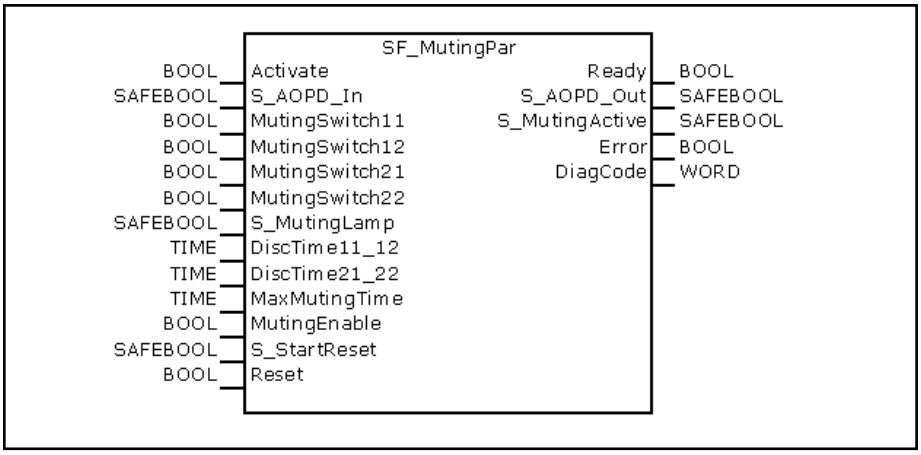


Abb. 16-73: Funktionsbaustein SF_MutingPar

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Allgemeine Eingangsparameter, Seite 294
	S_AOPD_In	SAFEBOOL	Variable Ausgangsschalt-(OSSD)-Signal des aktiven Opto-Schutz Gerätes (AOPD) FALSE: Schutzbereich unterbrochen TRUE: Schutzbereich nicht unterbrochen
	MutingSwitch11	BOOL	Variable Status des Mutingsensors MutingSwitch11 FALSE: MutingSwitch11 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch11 **Anmerkung:** Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden
	MutingSwitch12	BOOL	Variable Status des Mutingsensors MutingSwitch12 FALSE: MutingSwitch12 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch12 **Anmerkung:** Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden
	MutingSwitch21	BOOL	Variable Status des Mutingsensors MutingSwitch21 FALSE: MutingSwitch21 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch21 **Anmerkung:** Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden
	MutingSwitch22	BOOL	Variable Status des Mutingsensors MutingSwitch22 FALSE: MutingSwitch22 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch22 **Anmerkung:** Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden
	S_MutingLamp	SAFEBOOL	Variable oder Konstante Zeigt Zustand der Muting-Lampe FALSE: Muting-Lampe fehlerhaft TRUE: Muting-Lampe nicht fehlerhaft
	DiscTime11_12	TIME	Konstante 0 .. 4 s Maximale Diskrepanzzeit für MutingSwitch11 und MutingSwitch12 Der Vorsichtshinweis 'DiscTime11_12, Disc- Time21_22 und MaxMutingTime' ist zu beachten.

E/A-TYP	Name	Datentyp	Beschreibung
	DiscTime21_22	TIME	Konstante 0 .. 4 s Maximale Diskrepanzzeit für MutingSwitch21 und MutingSwitch22 Der Vorsichtshinweis 'DiscTime11_12, Disc- Time21_22 und MaxMutingTime' ist zu beachten.
	MaxMutingTime	Time	Konstante 0 .. 10 min Maximale Zeit für komplette Muting-Sequenz, Timer wird gestartet, wenn der erste Mutingsensor bedämpft wird Der Vorsichtshinweis 'DiscTime11_12, Disc- Time21_22 und MaxMutingTime' ist zu beachten.
	MutingEnable	BOOL	Variable oder Konstante Befehl des Steuerungssystems, der den Start der Muting-Funktion aktiviert, wenn vom Maschinenzyklus benötigt Nach dem Start der Mutingfunktion kann dieses Signal abgeschaltet werden FALSE: Muting nicht aktiviert TRUE: Start der Mutingfunktion aktiviert
	S_StartReset	SAFEBOOL	Allgemeine Eingangsparameter, Seite 294
	Reset	BOOL	Allgemeine Eingangsparameter, Seite 294
VAR_OUTPUT	Ready	BOOL	Allgemeine Ausgangsparameter, Seite 296
	S_AOPD_Out	SAFEBOOL	Sicherheitsgerichteter Ausgang, zeigt den Zustand der nicht trennenden Schutzeinrichtung der Muting-Anwendung an FALSE: AOPD-Schutzeinrichtungsbereich ist unterbrochen und Muting ist nicht aktiv TRUE: AOPD-Schutzeinrichtungsbereich ist nicht unterbrochen, oder Muting ist aktiv
	S_MutingActive	SAFEBOOL	Zeigt den Status des Muting-Prozesses an FALSE: Muting nicht aktiv TRUE: Muting aktiv
	Error	BOOL	Allgemeine Ausgangsparameter, Seite 296
	DiagCode	WORD	Diagnose-Codes, Seite 296

Tab. 16-152: Schnittstelle SF_MutingPar

⚠ VORSICHT **DiscTime11_12, DiscTime21_22 und MaxMutingTime**

Für Entwickler im Extended Level gilt: die Eingänge DiscTime11_12, DiscTime21_22 und Max- MutingTime müssen mit konstanten Werten beschaltet werden. D.h. diese Werte dürfen bei den Aufrufen nicht verändert werden!

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_AOPD_In	SAFEBOOL	FALSE	

Name	Typ	Default-Wert	Übernahme
MutingSwitch11	BOOL	FALSE	
MutingSwitch12	BOOL	FALSE	
MutingSwitch21	BOOL	FALSE	
MutingSwitch22	BOOL	FALSE	
S_MutingLamp	SAFEBOOL	FALSE	
DiscTime11_12	TIME	T#0s	Konstante
DiscTime21_22	TIME	T#0s	Konstante
MaxMutingTime	Time	T#0s	Konstante
MutingEnable	BOOL	FALSE	
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-153: Eingangsverhalten SF_MutingPar



Ein Kurzschluss bei den Mutingsensor-Signalen, oder ein funktionaler Applikationsfehler beim Unterstützen dieser Signale wird von diesem Baustein nicht erkannt, aber als inkorrekte Mutingsequenz interpretiert.

(Datentyp BOOL (nicht sicher), versorgt durch die funktionale Anwender-Hardware oder -Software.) Dennoch sollte es nicht zu ungewolltem Muting kommen. Der Anwender sollte dies in seiner Risikoanalyse berücksichtigen.

Funktionsbeschreibung

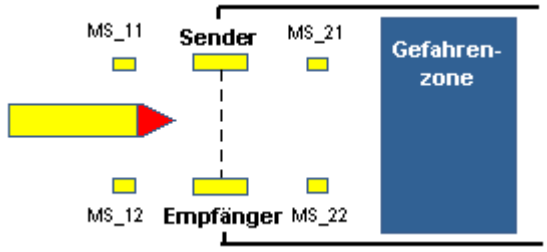
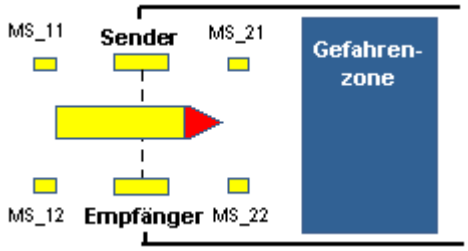
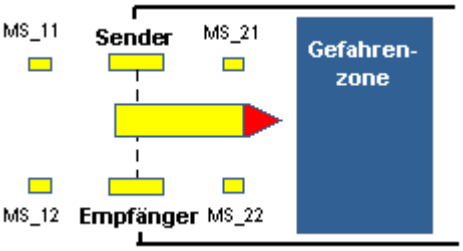
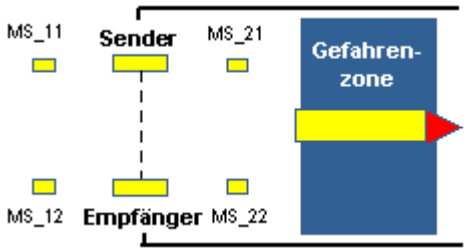
Muting ist die beabsichtigte Unterdrückung einer Sicherheitsfunktion. Dies ist erforderlich, wenn zum Beispiel Material in den Gefahrenbereich transportiert werden soll, ohne einen Stop der Maschine zu verursachen. Muting wird durch einen Mutingsensor getriggert. Die Verwendung von zwei oder vier Mutingsensoren und die richtige Integration den Fertigungsablauf muss sicherstellen, dass keine Person in den Gefahrenbereich gelangt, während das Lichtgitter gemutet ist. Mutingsensoren können Näherungsschalter, photoelektronische Barrieren, Grenztaster, etc. sein, die nicht ausfallsicher sein müssen. Aktiver Muting-Modus muss durch Anzeigelampen indiziert werden.

Es gibt sequentielle und parallele Mutingverfahren. In diesem Baustein wird paralleles Muting mit vier Mutingsensoren verwendet. Eine Ausführung der Vorwärtsrichtung des Transports wird unten aufgeführt. Der Baustein kann in beide Richtungen, vorwärts und rückwärts, verwendet werden. Das Muting sollte durch das MutingEnable-Signal vom Prozess aktiviert werden, um Manipulation zu vermeiden.

Die Eingangsparameter des Bausteins beinhalten die Signale der 4-Muting-Sensoren (MutingSwitch11, MutingSwitch12, MutingSwitch21, MutingSwitch22), das OSSD-Signal der "aktiven opto-elektronischen Schutzeinrichtung" S_AOPD_In, ebenso wie 3 parametrierbare Zeiten (DiscTime11_12, DiscTime21_22, MaxmutingTime).

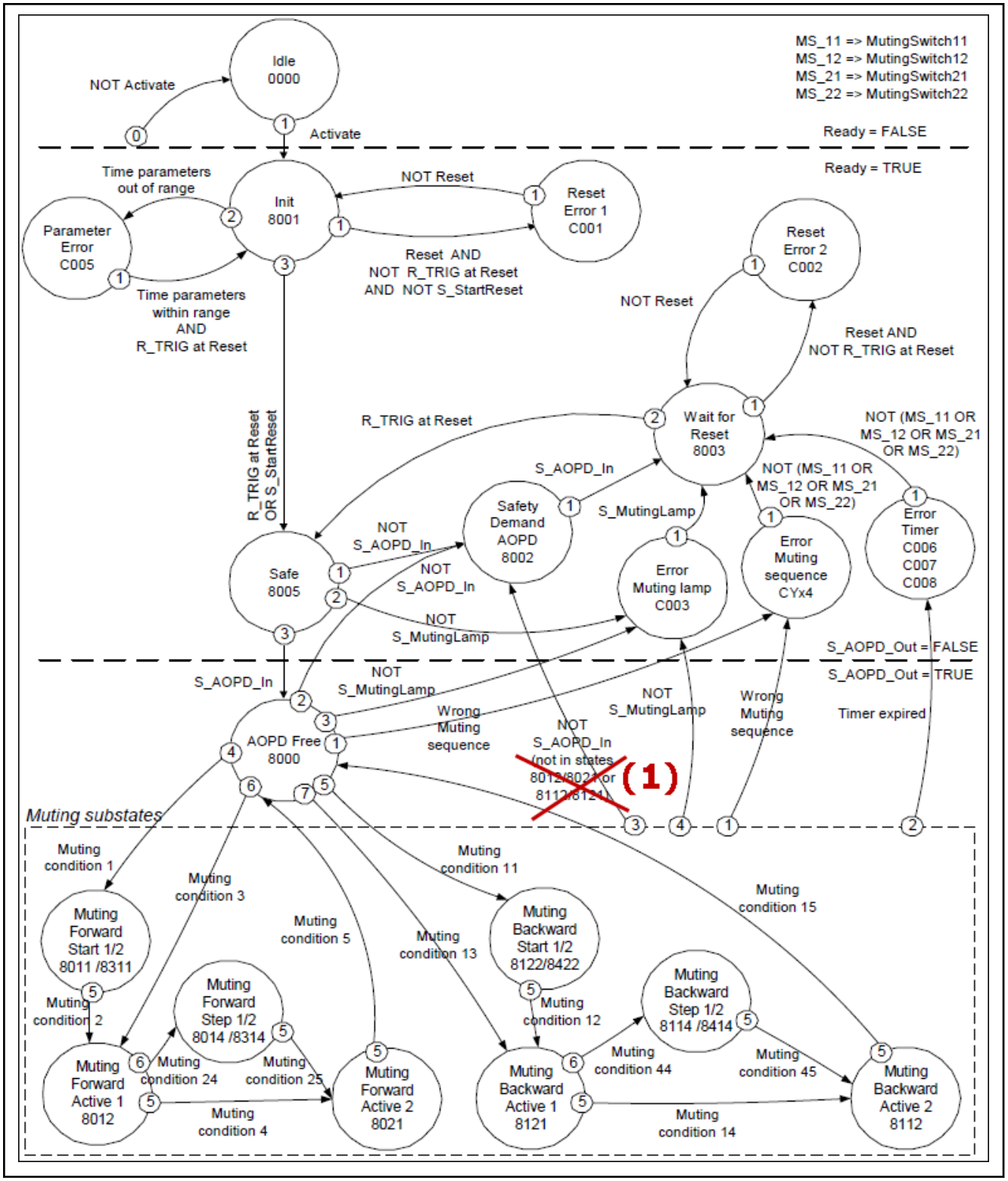
⚠ VORSICHT **S_StartReset-Eingang**

Der S_StartReset-Eingang soll nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdungssituation entstehen kann.

Nr.	Bild	Erklärung
1		Wenn die Mutingsensoren MutingSwitch11 (MS_11) und MutingSwitch12 (MS_12) von einem Werkstück innerhalb der Zeit DiscTime11_12 bedämpft werden, wir der Muting-Modus aktiviert (S_MutingActive = TRUE).
2		Muting-Modus bleibt so lange aktiv, wie MutingSwitch11 (MS_11) und MutingSwitch12 (MS_12) durch das Werkstück bedämpft werden. Das Werkstück kann das Lichtgitter passieren, ohne einen Maschinenstopp zu verursachen.
3		Bevor die Muting-Sensoren MutingSwitch11 (MS_11) und MutingSwitch12 (MS_12) deaktiviert werden, müssen MutingSwitch21 (MS_21) und MutingSwitch22 (MS_22) aktiviert werden. Dies gewährleistet, dass der Muting-Modus aktiv bleibt. Die Diskrepanzzeit zwischen dem Schalten von MutingSwitch21 und MutingSwitch22 wird von der Zeit DiscTime21_22 überwacht.
4		Muting-Modus ist beendet, wenn entweder MutingSwitch21 (MS_21) oder MutingSwitch22 (MS_22) durch das Werkstück deaktiviert wird. MaxMuting-Time ist die maximale Zeit, die der Muting-Modus aktiv bleiben darf.

Tab. 16-154: Beispiel für SF_MutingPar in Vorwärtsrichtung mit vier Sensoren

Statusdiagramm



(1) NOT S_AOPD_In only in states 8011/80311 or 8122/8422, dies stellt eine Abweichung von der PLCopen Spezifikation dar.

Abb. 16-74: Statusdiagramm SF_MutingPar

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typisches Zeitdiagramm

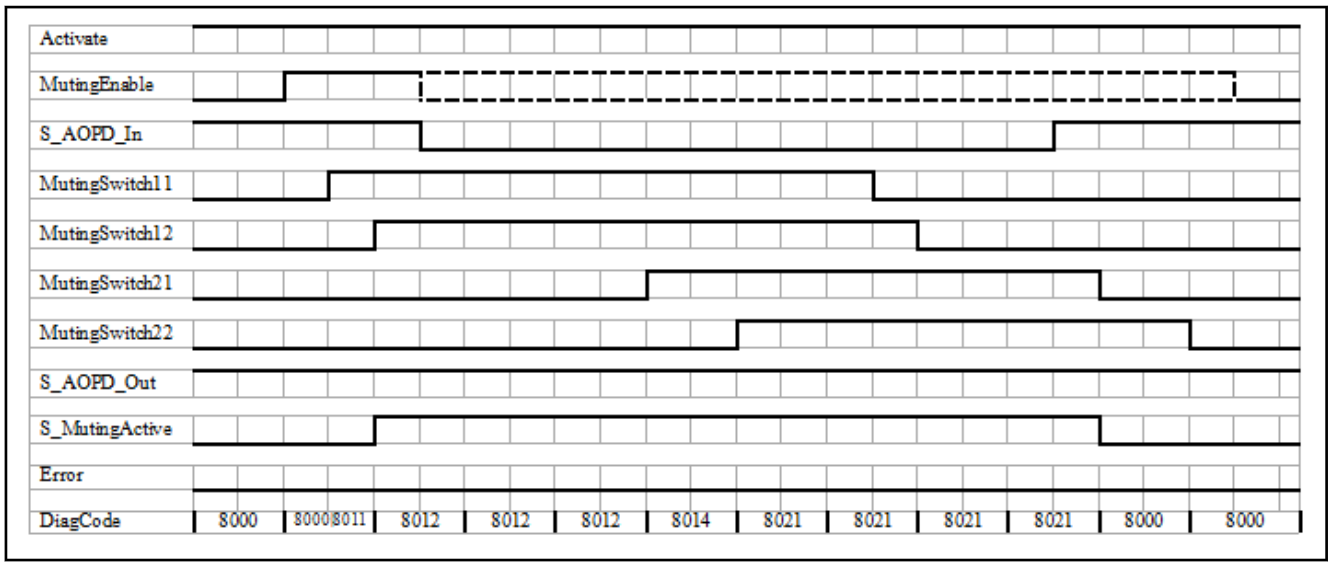


Abb. 16-75: Typisches Zeitdiagramm SF_MutingPar

Muting-Bedingungen

Vorwärtsrichtung

- **Muting-Bedingung 1 (zu Status 16#8011)** (MS_11 ist der erste bedämpfte Eingangssensor) Start der Timer MaxMutingTime und DiscTime11_12:

  MutingEnable AND (R_TRIG at MS_11 AND NOT MS_12 AND NOT MS_21 AND NOT MS_22)

- **Muting-Bedingung 1 (zu Status 16#8311)** (MS_12 ist der erste bedämpfte Eingangssensor) Start der Timer MaxMutingTime und DiscTime11_12:

  MutingEnable AND (NOT MS_11 AND R_TRIG at MS_12 AND NOT MS_22)

- **Muting-Bedingung 2 (von Status 16#8011)** (MS_12 ist der zweite bedämpfte Eingangssensor). Stopp von Timer DiscTime11_12:

  MutingEnable AND (MS_11 AND R_TRIG at MS_12 AND NOT MS_21 AND NOT MS_22)

- **Muting-Bedingung 2 (von Status 16#8311)** (MS_11 ist der zweite bedämpfte Eingangssensor). Stopp von Timer DiscTime11_12:

  MutingEnable AND(R_TRIG at MS_11 AND MS_12 AND NOT MS_21 And NOT MS_22;

- **Muting-Bedingung 3** (beide Eingangssensoren bedämpft im gleichen Zyklus). Start von Timer MaxMutingTime:

  MutingEnable AND (R_TRIG at MS_11 AND R_TRIG at MS_12 AND NOT MS_21 AND NOT MS_22)

- **Muting-Bedingung 4** (alle Sensoren bedämpft): MS_11 AND MS_12 AND MS_21 AND MS_22

- **Muting-Bedingung 24 (zu 16#8014)** (MS_21 ist der erste bedämpfte Ausgangssensor) Start Timer DiscTime21_22:

  MS_11 AND MS_12 AND R_TRIG at MS_21 AND NOT MS_22
- **Muting-Bedingung 24 (zu 16#8314)** (MS_22 ist der erste bedämpfte Ausgangssensor). Start Timer DiscTime21_22:

  MS_11 AND MS_12 AND NOT MS_21 AND R_TRIG at MS_22
- **Muting-Bedingung 25 (von 16#8014)** (MS_22 ist der zweite bedämpft Ausgangssensor). Stopp Timer DiscTime21_22:

  MS_11 AND MS_12 AND MS_21 AND R_TRIG at MS_22
- **Muting-Bedingung 25 (von 16#8314)** (MS_21 ist der zweite bedämpfte Ausgangssensor). Stopp Timer DiscTime21_22:

  MS_11 AND MS_12 AND R_TRIG at MS_21 AND MS_22
- **Muting-Bedingung 5** (einer der Ausgangs-Sensoren bedämpft). Stopp Timer MaxMutingTime:

  NOT MS_11 AND NOT MS_12 AND (F_TRIG at MS_21 OR F_TRIG at MS_22)

Rückwärtsrichtung

- **Muting-Bedingung 11 (zu16# 8122)** (MS_21 ist der erste bedämpfte Sensor). Start der Timer MaxMutingTime und DiscTime21_22:

  MutingEnable AND (NOT MS_22 AND R_TRIG at MS_21 AND NOT MS_11 AND NOT MS_12)
- **Muting-Bedingung 11 (zu 16#8422)** (MS_22 ist der erste bedämpfte Eingangssensor). Start der Timer MaxMutingTime und DiscTime21_22:

  MutingEnable AND (R_TRIG at MS_22 AND NOT MS_21 AND NOT MS_11 AND NOT MS_12)
- **Muting-Bedingung 12 (von16# 8122)** (MS_22 ist der zweite bedämpfte Eingangssensor). Stopp Timer DiscTime21_22:

  MutingEnable AND (MS_21 AND R_TRIG at MS_22 AND NOT MS_11 AND NOT MS_12)
- **Muting-Bedingung 12 (von 16#8422)** (MS_21 ist der zweite bedämpfte Eingangssensor). Stopp Timer DiscTime21_22:

  MutingEnable AND (R_TRIG at MS_21 AND MS_22 AND NOT MS_11 AND NOT MS_12)
- **Muting-Bedingung 13** (beide Eingangssensoren bedämpft im gleichen Zyklus). Start Timer MaxMutingTime:

  MutingEnable AND (R_TRIG at MS_21 AND R_TRIG at MS_22 AND NOT MS_11 AND NOT MS_12)
- **Muting-Bedingung 1**4 (alle Sensoren bedämpft):

  MS_11 AND MS_12 AND MS_21 AND MS_22
- **Muting-Bedingung 44 (zu 16#8114)** (MS_11 ist der erste bedämpfte Ausgangssensor). Start Timer DiscTime11_12:

  MS_21 AND MS_22 AND R_TRIG at MS_11 AND NOT MS_12
- **Muting-Bedingung 44 (zu 16#8414)** (MS_12 ist der erste bedämpfte Ausgangssensor). Start Timer DiscTime11_12: MS_21 AND MS_22 AND NOT MS_11 AND R_TRIG at MS_12
- **Muting-Bedingung 45 (von 16#8114)** (MS_12 ist der zweite bedämpfte Ausgangssensor). Stop Timer DiscTime11_12:

MS_21 AND MS_22 AND MS_11 AND R_TRIG at MS_12

- **Muting-Bedingung 45 (von 16#8414)** (MS_11 ist der zweite bedämpfte Ausgangssensor). Stop Timer DiscTime11_12: MS_21 AND MS_22 AND R_TRIG at MS_11 AND MS_12
- **Muting-Bedingung 15** (einer der beiden Ausgangssensoren bedämpft) Stopp Timer MaxMutingTime: NOT MS_21 AND NOT MS_22 AND (F_TRIG at MS_11 OR F_TRIG at MS_12)

Falsche Muting-Sequenzen:

- State 16#8000:

  (MutingEnable = FALSE when muting sequence starts) OR

  ((MS_11 OR MS_12) AND (MS_21 OR MS_22))OR

  (R_TRIG at MS_11 AND MS_12 AND NOT R_TRIG at MS_12) OR

  (R_TRIG at MS_12 AND MS_11 AND NOT R_TRIG at MS_11) OR

  (R_TRIG at MS_21 AND MS_22 AND NOT R_TRIG at MS_22) OR

  (R_TRIG at MS_22 AND MS_21 AND NOT R_TRIG at MS_21) OR

  ((MS_11 AND NOT R_TRIG at MS_11) AND (MS_12 AND NOT R_TRIG at MS_12)) OR

  ((MS_21 AND NOT R_TRIG at MS_21) AND (MS_22 AND NOT R_TRIG at MS_22))
- Status 16#8011: NOT MutingEnable OR NOT MS_11 OR MS_21 OR MS_22
- Status 16#8311: NOT MutingEnable OR NOT MS_12 OR MS_21 OR MS_22
- Status 16#8012: NOT MS_11 OR NOT MS_12
- Status 16#8021: R_TRIG at MS_11 OR R_TRIG at MS_12 OR R_TRIG at MS_21 OR R_TRIG at MS_22
- Status16#8014: NOT MS_11 OR NOT MS_12 OR NOT MS_21
- Status 16#8314: NOT MS_11 OR NOT MS_12 OR NOT MS_22
- Status 16#8122: NOT MutingEnable OR MS_11 OR MS_12 OR NOT MS_21
- Status 16#8422: NOT MutingEnable OR MS_11 OR MS_12 OR NOT MS_22
- Status16#8121: NOT MS_21 OR NOT MS_22
- Status 16#8112: R_TRIG at MS_11 OR R_TRIG at MS_12 OR R_TRIG at MS_21 OR R_TRIG at MS_22
- Status 16#8114: NOT MS_21 OR NOT MS_22 OR NOT MS_11
- Status 16#8414: NOT MS_21 OR NOT MS_22 OR NOT MS_12

Fehlererkennung

Der Baustein entdeckt folgende Fehlerbedingungen:

- DiscTime11_23 und DiscTime21_22 wurden auf Werte kleiner als T#0s oder größer als T#4s gesetzt
- MaxMutingTime wurde auf einen Wert kleiner als T#0s oder größer als T#10min gesetzt
- Die Diskrepanzzeit für die Sensorpaare MutingSwitch11/MutingSwitch12 oder MutingSwitch21/MutingSwitch22 wurde überschritten

- Die Mutingfunktion (S_MutingActive =TRUE) überschreitet die maximale Mutingzeit MaxMutingTime
- Die Mutingsensoren MutingSwitch11, MutingSwitch12, MutingSwitch21 und MutingSwitch22 wurden in falscher Reihenfolge aktiviert
- Die Mutingsequenz startet ohne Aktivierung durch MutingEnable
- Eine fehlerhafte Mutinglampe wird durch S_MutingLamp = FALSE angezeigt
- Eine statische Reset-Bedingung wird in Status 8001 und 8003 entdeckt

Fehlerverhalten

Bei einem Fehlerereignis werden die Ausgänge S_AOPD_Out und S_MutingActive auf FALSE gesetzt. Der Ausgang DiagCode zeigt den relevanten Fehlercode an und der Ausgang Error wird auf TRUE gesetzt.

Ein Restart wird solange unterbunden, bis die Fehlerbedingungen geklärt sind und der sichere Zustand durch den Operator mit Reset bestätigt wird.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error 1	Statische Reset-Bedingung entdeckt nach Bausteinaktivierung in Status 16#8001 Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C002	Reset Error 2	Statische Reset-Bedingung entdeckt in Status 16#8003 Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C003	Error Muting Lamp	Fehler entdeckt bei Mutinglampe Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#CYx4	Error Muting sequence	Fehler in der Mutingsequenz in Status 16#8000, 16#8011,16# 831116#,8012, 16#8021,16#8014, 16#8314,16#8122, 16#8422,16# 8121, 16#8112,16# 8114 oder 16#8414 entdeckt. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE Y = Status in der Sequenz (6 Status für Vorwärts- und 6 Status für Rückwärtsrichtung) 16#C0x4 = Fehler tritt in Status16# 8000 auf 16#C1x4 = Fehler tritt im Status Vorwärts 8011 auf 16#C2x4 = Fehler tritt im Status Vorwärts 8311 auf 16#C3x4 = Fehler tritt im Status Vorwärts 8012 auf 16#C4x4 = Fehler tritt im Status Vorwärts 8014 auf 16#C5x4 = Fehler tritt in Status Vorwärts 8314 auf 16#C6x4 = Fehler tritt in Status Vorwärts 8021 auf 16#C7x4 = Fehler tritt in Status Rückwärts 8122 auf 16#C8x4 = Fehler tritt in Status Rückwärts 8422 auf 16#C9x4 = Fehler tritt in Status Rückwärts 8121 auf 16#CAx4 = Fehler tritt in Status Rückwärts 8114 auf 16#CBx4 = Fehler tritt in Status Rückwärts 8414 auf 16#CCx4 = Fehler tritt in Status Rückwärts 8112 auf 16#CFx4 = Muting Enable fehlt x = Status des Sensors, wenn ein Fehler auftritt (4 Bits: LSB = MS_11; MS_12; MS_21; MSB = MS_22)
16#C005	Parameter Error	DiscTime11_12, DiscTime21_22 oder MaxMutingTime-Wert außerhalb des Bereichs Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C006	Error Timer MaxMuting	Timing-Fehler: Aktive Mutingzeit (wenn S_MutingActive = TRUE) überschreitet MaxMutingTime Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C007	Error Timer MS11_12	Timing-Fehler: Diskrepanzzeit für Schalten von MutingSwitch11 und MutingSwitch12 ist größer als DiscTime11_12 Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C008	Error Timer MS21_22	Timing-Fehler: Diskrepanzzeit für Schalten von MutingSwitch21 und MutingSwitch22 ist größer als DiscTime21_22 Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

Tab. 16-155: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_AOPD_Out =FALSE S_MutingActive = FALSE Error = FALSE
16#8000	AOPD Free	Muting nicht aktiv und keine Sicherheitsanforderung von AOPD. Wenn die Timer des darauffolgenden Mutings noch laufen, werden sie gestoppt. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
16#8001	Init	Baustein wurde aktiviert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8002	Safety Demand AOPD	Sicherheitsanforderung wurde von AOPD entdeckt, Muting nicht aktiv. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8003	Wait for Reset	Safety-Anforderung oder Fehler wurden entdeckt und sind jetzt geklärt. Operator-Bestätigung mit Reset angefordert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8005	Safe	Sicherheitsfunktion aktiviert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8011	Muting Forward Start 1	Muting-Vorwärts-Sequenz ist in der Startphase nach steigender Triggerflanke von MutingSwitch11. Monitoring von DiscTime11_12 ist aktiviert. Monitoring von MaxMutingTime ist aktiviert. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive =FALSE Error = FALSE
16#8311	Muting Forward Start 2	Muting-Vorwärts-Sequenz ist nach steigender Triggerflanke von MutingSwitch12 in der Startphase. Monitoring von DiscTime11_12 ist aktiviert. Monitoring von MaxMutingTime ist aktiviert. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive =FALSE Error = FALSE
16#8012	Muting Forward Active 1	Muting-Vorwärts-Sequenz ist aktiv, entweder: - nach Erkennung einer steigender Triggerflanke des zweiten Eingangs MutingSwitch12 oder 11 oder - wenn beide MutingSwitch11 und12 im gleichen Zyklus bedämpft wurden. Monitoring von DiscTime11_12 wird gestoppt. Monitoring ist aktiviert, wenn der Übergang direkt von Status 16#8000 kam. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8014	Muting Forward Step 1	Muting-Vorwärts-Sequenz ist aktiv. MutingSwitch21 ist der erste bedämpfte Ausgangssensor. Monitoring von DiscTime21_22 gestartet. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
16#8314	Muting Forward Step 2	Muting-Vorwärts-Sequenz ist aktiv. MutingSwitch22 ist der erste bedämpfte Ausgangssensor. Monitoring von DiscTime21_22 gestartet. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
16#8021	Muting Forward Active 2	Muting-Vorwärts-Sequenz ist immer noch aktiv. Beide MutingSwitch21 und 22 sind bedämpft, Monitoring von DiscTime21_22 wird gestoppt. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
16#8122	Muting Backward Start 1	Muting-Rückwärts-Sequenz ist in Startphase nach steigender Triggerflanke von MutingSwitch21. Monitoring von DiscTime21_22 ist aktiviert. Monitoring von MaxMutingTime ist aktiviert. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
16#8422	Muting Backward Start 2	Muting-Rückwärts-Sequenz ist in Startphase nach steigender Triggerflanke von MutingSwitch22. Monitoring von DiscTime21_22 ist aktiviert. Monitoring von MaxMutingTime ist aktiviert.. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8121	MutingBackward Active 1	Muting-Rückwärts-Sequenz ist aktiv, entweder: - Nach Erkennung einer steigender Triggerflanke des zweiten Eingangs MutingSwitch21 oder 22 oder - Wenn beide MutingSwitch21 und 22 im gleichen Zyklus bedämpft wurden. Monitoring von DiscTime21_22 gestoppt. Monitoring von MaxMuting-Time ist aktiviert, wenn Übergang direkt von Status 8000 kam. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
16#8114	Muting Backward Step 1	Muting-Rückwärts-Sequenz ist aktiv. MutingSwitch11 ist der erste bedämpfte Ausgangssensor. Monitoring von DiscTime11_12 gestartet. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
16#8414	Muting Backward Step 2	Muting-Rückwärts-Sequenz ist aktiv. MutingSwitch12 ist der erste bedämpfte Ausgangssensor. Monitoring von DiscTime11_12 gestartet. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
16#8112	Muting Backward Active 2	Muting-Rückwärts-Sequenz ist immer noch aktiv. Beide Ausgangssensoren MutingSwitch11 und 12 sind bedämpft, das Monitoring von DiscTime11_12 ist gestoppt. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE

Tab. 16-156: Bausteinspezifische Status-Codes

16.4.12 SF_MutingPar_2Sensor

Kurzbeschreibung

☞ SF_MutingPar_2Sensor ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Muting ist das beabsichtigte Unterdrücken einer Sicherheitsfunktion.

In diesem Baustein ist das parallele Muting mit zwei Mutingsensoren spezifiziert.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-157: Bibliothekszuordnung

Schnittstellenbeschreibung

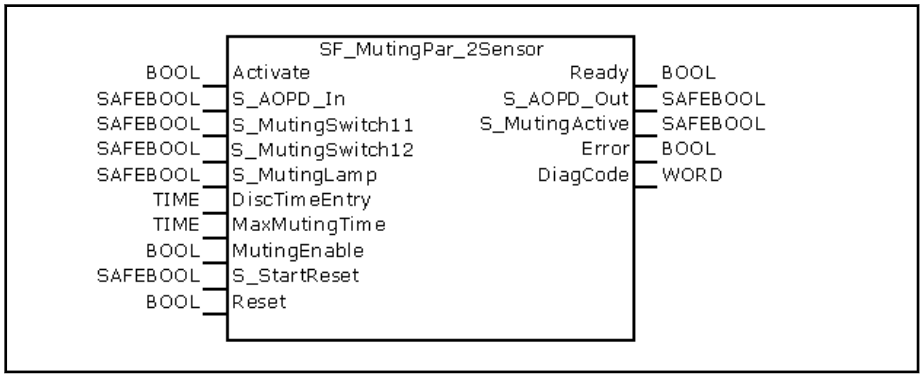


Abb. 16-76: Funktionsbaustein SF_MutingPar_2Sensor

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_AOPD_In	SAFEBOOL	Variable OSSD-Signal von AOPD FALSE: Schutzbereich unterbrochen TRUE: Schutzbereich nicht unterbrochen
	S_MutingSwitch11	SAFEBOOL	Variable Status des Mutingsensors MutingSwitch11 FALSE: MutingSwitch11 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch11
	S_MutingSwitch12	SAFEBOOL	Variable Status des Mutingsensors MutingSwitch12 FALSE: MutingSwitch12 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch12
	S_MutingLamp	SAFEBOOL	Variable oder Konstante Zeigt Zustand der Muting-Lampe FALSE: Muting-Lampe fehlerhaft TRUE: Muting-Lampe nicht fehlerhaft
	DiscTime	TIME	Konstante 0..4 s Maximale Diskrepanzzeit für S_MutingSwitch11 und S_MutingSwitch12
	MaxMutingTime	Time	Konstante 0 .. 10 min Maximale Zeit für komplette Muting-Sequenz, Timer wird gestartet, wenn der erste Mutingsensor bedämpft wird

E/A-TYP	Name	Datentyp	Beschreibung
	MutingEnable	BOOL	Variable oder Konstante Befehl des Steuerungssystems, der den Start der Muting-Funktion aktiviert, wenn vom Maschinenzyklus benötigt. Nach Start der Mutingfunktion kann dieses Signal abgeschaltet werden FALSE: Muting nicht aktiviert TRUE: Start der Mutingfunktion aktiviert
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_AOPD_Out	SAFEBOOL	Sicherheitsgerichteter Ausgang, zeigt den Zustand der nicht trennenden Schutzeinrichtung der Muting-Anwendung an FALSE: AOPD-Schutzeinrichtungsbereich ist unterbrochen und Muting ist nicht aktiv TRUE: AOPD-Schutzeinrichtungsbereich ist nicht unterbrochen oder Muting aktiv
	S_MutingActive	SAFEBOOL	Zeigt den Status des Muting-Prozesses an FALSE: Muting nicht aktiv TRUE: Muting aktiv
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-158: Schnittstelle SF_MutingPar_2Sensor

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_AOPD_In	SAFEBOOL	FALSE	
S_MutingSwitch11	SAFEBOOL	FALSE	
S_MutingSwitch12	SAFEBOOL	FALSE	
S_MutingLamp	SAFEBOOL	FALSE	
DiscTime	TIME	T#0s	Konstante
MaxMutingTime	TIME	T#0s	Konstante
MutingEnable	BOOL	FALSE	
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-159: Eingangsverhalten MutingPar_2Sensor



Leitungskontrolle der Muting-Sensorsignale muss in der Sicherheitsschleife aktiv sein.

Funktionsbeschreibung

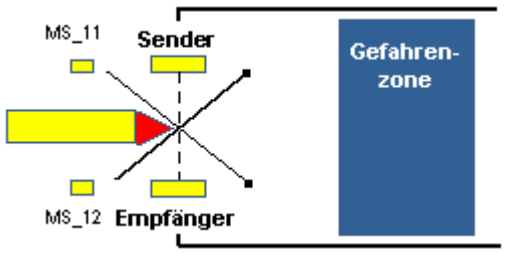
Muting ist die beabsichtigte Unterdrückung einer Sicherheitsfunktion. Dies ist erforderlich, wenn zum Beispiel Material in den Gefahrenbereich transportiert werden soll, ohne einen Stop der Maschine zu verursachen. Muting wird durch einen Mutingsensor getriggert. Die Verwendung von zwei Mutingsensoren und die korrekte Integrierung in den Fertigungsablauf muss sicherstellen, dass keine Person in den Gefahrenbereich gelangt, während das Lichtgitter gemutet ist. Mutingsensoren können Drucktaster, Näherungsschalter, photoelektronische Barrieren, Grenztaster, etc. sein, die nicht ausfallsicher sein müssen. Aktiver Muting-Modus muss durch Anzeigelampen indiziert werden.

Es gibt sequentielle und parallele Mutingverfahren. In diesem Baustein wurde paralleles Muting mit 2 Mutingsensoren verwendet. Ein Beispiel ist unten aufgeführt. Die Positionierung der Sensoren sollte entsprechend IEC Normen erfolgen. Der Baustein kann in beide Richtungen, vorwärts und rückwärts, verwendet werden. Dabei kann die aktuelle Richtung nicht ermittelt werden. Das Muting sollte mit dem MutingEnable-Signal der Prozesssteuerung aktiviert werden, um Manipulation zu vermeiden.

Die Eingangsparameter des Bausteins beinhalten die Signale der 2 Mutingsensoren (S_MutingSwitch11 und S_MutingSwitch12), das OSSD-Signal der "aktiven opto-elektronischen Schutzeinrichtung" S_AOPD, sowie die zwei parametrierbaren Zeiten DiscTimeEntry und MaxMutingTime.

⚠ VORSICHT **S_StartReset-Eingang**

Der S_StartReset-Eingang soll nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdungssituation entstehen kann.

Nr.	Bild	Erklärung
1		Wenn Reflexionslichtschranken als Mutingsensoren verwendet werden, so werden diese generell diagonal angeordnet. Generell erfordert diese Anordnung von Reflexionslichtschranken als Mutingsensoren nur zwei Lichtschranken und nur S_MutingSwitch11 (MS_11) und S_MutingSwitch12 (MS_12) werden zugeordnet.

Tab. 16-160: Beispiel für SF_MutingPar_2Sensor in Vorwärtsrichtung mit 2 Lichtschranken

Statusdiagramm

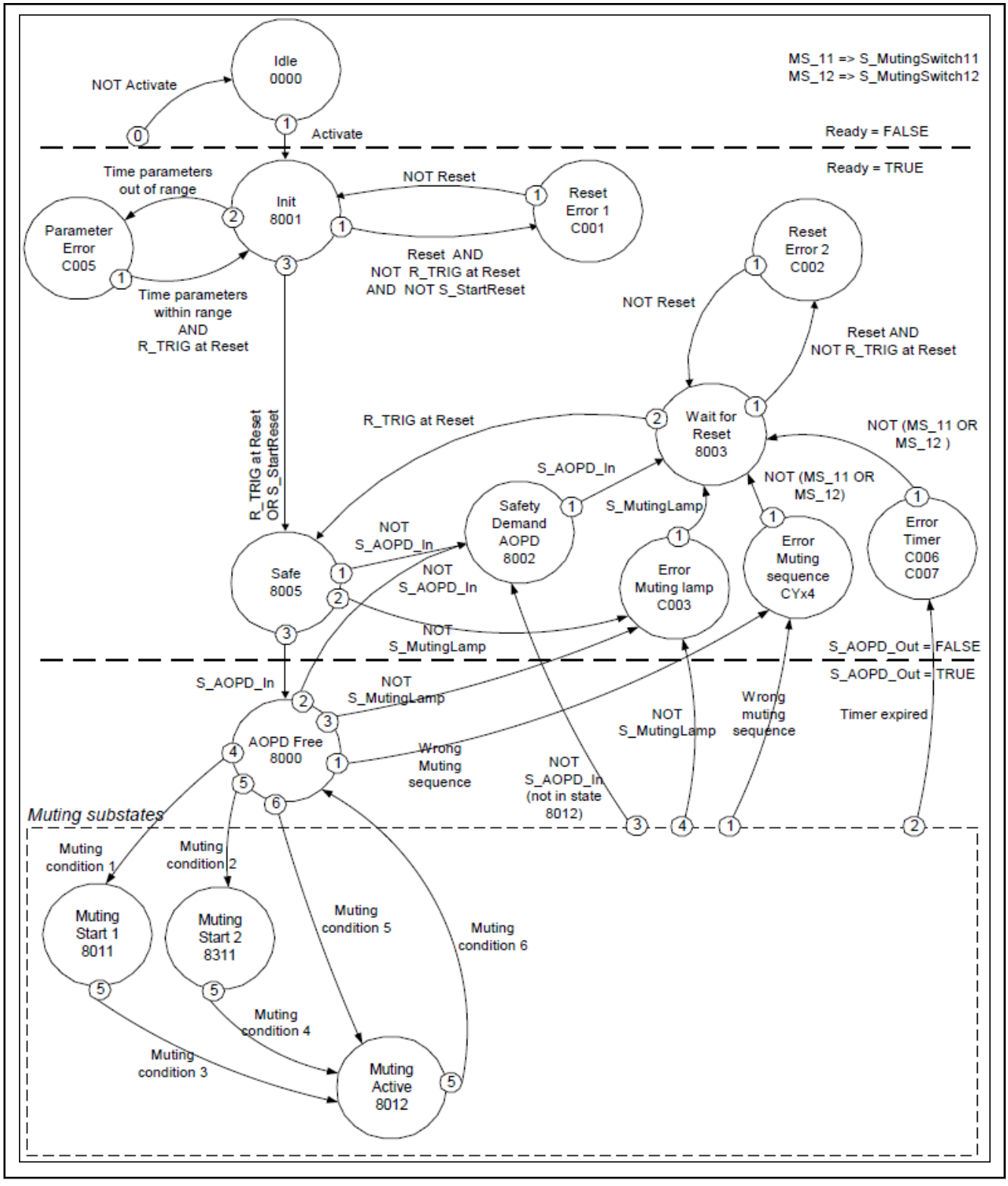


Abb. 16-77: Statusdiagramm SF_MutingPar_2Sensor

Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typisches Zeitdiagramm

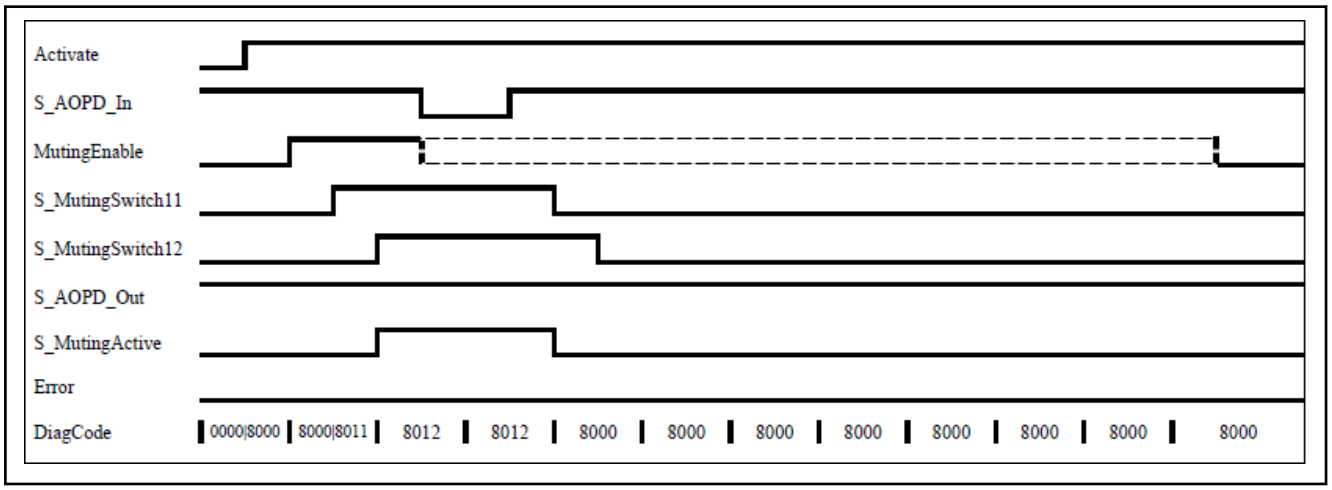


Abb. 16-78: Typisches Zeitdiagramm SF_MutingPar_2Sensor (S_StartReset = TRUE, Reset = FALSE, S_MutingLamp = TRUE)

Muting-Bedingungen

- **Muting-Bedingung 1 (zu 16#8011)** (MS_11 ist der erste bedämpfte Eingangs-Sensor) Start Timer DiscTimeEntry und MaxMutingTime:

  MutingEnable AND R_TRIG at MS_11 AND NOT MS_12
- **Muting-Bedingung 2 (zu 16#8311)**(MS_12 ist der erste bedämpfte Eingangs-Sensor): Start Timer DiscTimeEntry und MaxMutingTime:

  MutingEnable AND NOT MS_11 AND R_TRIG at MS_12
- **Muting-Bedingung 3 (von 16#8011 zu 16#8012)**(MS_12 ist der zweite bedämpfte Eingangs-Sensor): Stopp Timer DiscrimeEntry:

  MutingEnable AND MS_11 AND R_TRIG at MS_12
- **Muting-Bedingung 4 (von 16#8311 zu16#8012)**(MS_11 ist der zweite bedämpfte Eingangssensor): Stopp Timer DiscTimeEntry:

  MutingEnable AND R_TRIG at MS_11 AND MS_12
- **Muting-Bedingung 5 (von 16#8000 zu 16#8012)** (beide Sensoren bedämpft im gleichen Zyklus): Start Timer MaxMutingTime:

  MutingEnable AND R_TRIG at MS_11 AND R_TRIG at MS_12
- **Muting-Bedingung 6 (von 16#8012 zu 16#8000)**(beide Sensoren gelöst im gleichen Zyklus oder MS_11 und MS_12 nacheinander gelöst). Stopp Timer MaxMutingTime: NOT MS_11 OR NOT MS_12

Falsche Muting-Sequenzen

- **State 16#8000:**(R_TRIG at MS_11 AND MS_12 AND NOT R_TRIG at MS_12) OR

  (R_TRIG at MS_12 AND MS_11 AND NOT R_TRIG at MS_11) OR

  ((MS_11 AND NOT R_TRIG at MS_11) AND (MS_12 AND NOT R_TRIG at MS_12)) OR

  (NOT MutingEnable AND R_TRIG at MS_11) OR

  (NOT MutingEnable AND R_TRIG at MS_12)
- **State 16#8011:** NOT MutingEnable OR NOT MS_11
- **State 16#8311:** NOT MutingEnable OR NOT MS_12
- **State 16#8012:** alle möglichen Übergänge erlaubt

Fehlererkennung

Der Baustein entdeckt folgende Fehlerbedingungen:

- DiscTimeEntry wurde auf einen Wert kleiner als T#0s oder größer als T#4s gesetzt.
- MaxMutingTime wurde auf eine Wert kleiner als T#0s und größer als T#10min gesetzt.
- Die Diskrepanzzeit für das Paar S_MutingSwitch11/S_MutingSwitch12 wurde überschritten
- Die Mutingfunktion (S_MutingActive = TRUE) überschreitet die maximale Zeit MaxMutingTime
- Die Mutingsensoren S_MutingSwitch11 und S_MutingSwtich12 werden in falscher Reihenfolge bedämpft
- Mutingsequenz startet ohne Aktivierung durch MutingEnable
- Statische Signale des Mutingsensors
- Eine fehlerhafte Mutinglampe wird anzeigt durch S_MutingLamp = FALSE
- Eine statische Reset-Bedingung wird in Status 16#8001 und 16#8003 entdeckt

Fehlerverhalten

Bei einem Fehlerereignis werden die Ausgänge S_AOPD_Out und S_MutingActive auf FALSE gesetzt. Der Ausgang DiagCode zeigt den relevanten Fehlercode and und der Ausgang Error wird auf TRUE gesetzt.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error 1	Statische Reset-Bedingung entdeckt nach Bausteinaktivierung in Status 16#8001. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C002	Reset Error 2	Statische Reset-Bedingung entdeckt in Status 16#8003. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C003	Error Muting Lamp	Fehler entdeckt bei Mutinglampe. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#CYx4	Error Muting Sequence	Fehler in der Mutingsequenz in Status 16#8000, 16# 8011, 16#8311. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE Y = Status in der Sequenz 16#C0x4 = Fehler tritt im Status 16#8000 auf. 16#C1x4 = Fehler tritt im Status 16#8011 auf. 16#C2x4 = Fehler tritt im Status 16#8311 auf. 16#CFx4 = MutingEnable fehlt x = Status des Sensors, wenn ein Fehler auftritt (4 Bits: LSB = MS_11; neben LSB = MS_22).
16#C005	Parameter Error	DiscTimeEntry oder MaxMutingTime-Wert außerhalb des Bereichs. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C006	Error timer MaxMuting	Timing-Fehler: Aktive Mutingzeit (wenn S_MutingActive = TRUE) überschreitet MaxMutingTime. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C007	Error timer Entry	Timing-Fehler: Diskrepanzzeit für Schalten von MutingSwitch11 und MutingSwitch12 von FALSE nach TRUE ist größer als DiscTimeEntry. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

Tab. 16-161: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_AOPD_Out =FALSE S_MutingActive = FALSE Error = FALSE
16#8000	AOPD Free	Muting nicht aktiv und keine Sicherheitsanforderung von AOPD. Wenn die Timer des darauffolgenden Muting noch laufen, werden sie gestoppt. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE
16#8001	Init	Baustein wurde aktiviert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8002	Safety Demand AOPD	Sicherheitsanforderung wurde von AOPD entdeckt, Muting nicht aktiv. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8003	Wait for Reset	Safety-Anforderung oder Fehler wurden entdeckt und sind jetzt geklärt. Operator-Bestätigung mit Reset angefordert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8005	Safe	Sicherheitsfunktion aktiviert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8011	Muting Start 1	Mutingsequenz befindet sich nach steigender Triggerflanke von MutingSwitch11 in der Startphase Monitoring von DiscTimeEntry ist aktiviert. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive =FALSE Error = FALSE
16#8311	Muting Start 2	Mutingsequenz befindet sich nach steigender Triggerflanke von MutingSwitch12 in der Startphase. Monitoring von DiscTimeEntry ist aktiviert. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive =FALSE Error = FALSE
16#8012	Muting Active	Mutingsequenz ist aktiv, entweder: - nachdem eine steigende Triggerflanke des zweiten Eingangs MutingSwitch12 oder MutingSwitch11 entdeckt wurde oder - wenn beide Eingänge MutingSwitch11 und MutingSwitch12 im gleichen Zyklus bedämpft wurden. Monitoring von DiscTimeEntry ist gestoppt. Monitoring von MaxMutingTime ist aktiviert. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE

Tab. 16-162: Bausteinspezifische Status-Codes

16.4.13 SF_MutingSeq

Kurzbeschreibung

☞ SF_MutingSeq ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Muting ist das beabsichtigte Unterdrücken einer Sicherheitsfunktion (z. B. Lichtschranke).

In diesem Baustein ist das sequentielle Muting mit vier Mutingsensoren realisiert.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-163: Bibliothekszuordnung

Schnittstellenbeschreibung

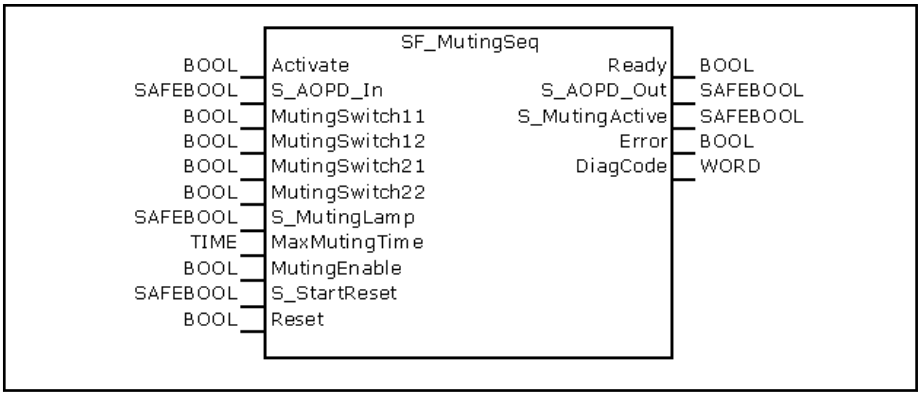


Abb. 16-79: Funktionsbaustein SF_MutingSeq

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_AOPD_In	SAFEBOOL	Variable OSSD-Signal von AOPD FALSE: Schutzbereich unterbrochen TRUE: Schutzbereich nicht unterbrochen
	MutingSwitch11	BOOL	Variable Status des Mutingsensors MutingSwitch11 FALSE: MutingSwitch11 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch11 **Anmerkung:** Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden
	MutingSwitch12	BOOL	Variable Status des Mutingsensors MutingSwitch12 FALSE: MutingSwitch12 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch12 **Anmerkung:** Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden
	MutingSwitch21	BOOL	Variable Status des Mutingsensors MutingSwitch21 FALSE: MutingSwitch21 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch21 **Anmerkung:** Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden

E/A-TYP	Name	Datentyp	Beschreibung
	MutingSwitch22	BOOL	Variable Status des Mutingsensors MutingSwitch22 FALSE: MutingSwitch22 nicht bedämpft TRUE: Werkstück bedämpft MutingSwitch22 **Anmerkung:** Abhängig von den Sicherheitsanforderungen, muss anstelle von BOOL mit SAFEBOOL verbunden werden
	S_MutingLamp	SAFEBOOL	Variable oder Konstante Zeigt Zustand der Muting-Lampe FALSE: Muting-Lampe fehlerhaft TRUE: Muting-Lampe nicht fehlerhaft
	MaxMutingTime	Time	Konstante 0 .. 10 min Maximale Zeit für komplette Muting-Sequenz, Timer wird gestartet, wenn der erste Mutingsensor bedämpft wird
	MutingEnable	BOOL	Variable oder Konstante Befehl des Steuerungssystems, der den Start der Muting-Funktion aktiviert, wenn vom Maschinenzyklus benötigt Nach Start der Mutingfunktion kann dieses Signal abgeschaltet werden FALSE: Muting nicht aktiviert TRUE: Start der Mutingfunktion aktiviert
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_AOPD_Out	SAFEBOOL	Sicherheitsgerichteter Ausgang, zeigt den Zustand der nicht trennenden Schutzeinrichtung der Muting-Anwendung an FALSE: AOPD-Schutzeinrichtungsbereich ist unterbrochen und Muting ist nicht aktiv TRUE: AOPD-Schutzeinrichtungsbereich ist nicht unterbrochen, oder Muting ist aktiv
	S_MutingActive	SAFEBOOL	Zeigt den Status des Muting-Prozesses an FALSE: Muting nicht aktiv TRUE: Muting aktiv
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-164: Schnittstelle SF_MutingSeq

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_AOPD_In	SAFEBOOL	FALSE	
MutingSwitch11	BOOL	FALSE	
MutingSwitch12	BOOL	FALSE	

Name	Typ	Default-Wert	Übernahme
MutingSwitch21	BOOL	FALSE	
MutingSwitch22	BOOL	FALSE	
S_MutingLamp	SAFEBOOL	FALSE	
MaxMutingTime	Time	T#0s	Konstante
MutingEnable	BOOL	FALSE	
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-165: Eingangsverhalten SF_MutingSeq



Ein Kurzschluss bei den Mutingsensor-Signalen, oder ein funktionaler Applikationsfehler beim Unterstützen dieser Signale wird von diesem Baustein nicht unterstützt, aber als inkorrekte Mutingsequenz interpretiert (Datentyp BOOL, versorgt durch die funktionale Anwender-Hardware oder - Software). Dennoch sollte dies nicht zu einem ungewollten Muting führen. Der Anwender sollte dies in seiner Risikoanalyse berücksichtigen.

Funktionsbeschreibung

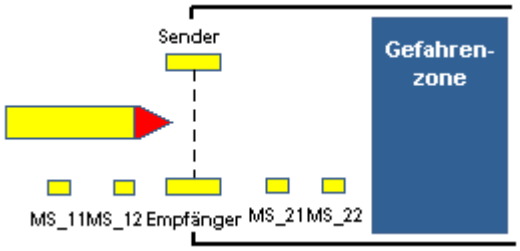
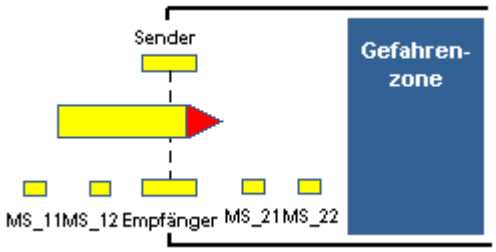
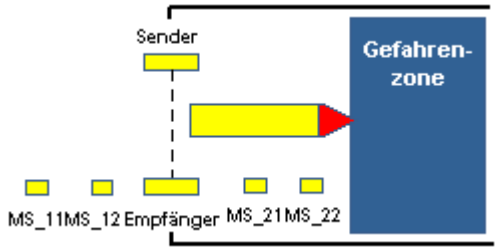
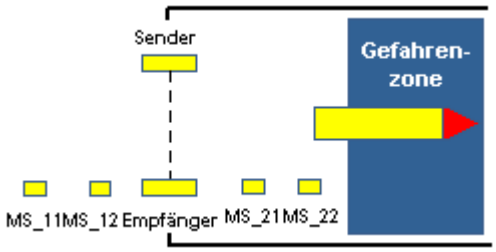
Muting ist die beabsichtigte Unterdrückung einer Sicherheitsfunktion. Dies ist erforderlich, wenn zum Beispiel Material in den Gefahrenbereich transportiert werden soll, ohne einen Stop der Maschine zu verursachen. Muting wird durch einen Mutingsensor getriggert. Die Verwendung von zwei oder vier Mutingsensoren und die korrekte Integrierung in den Fertigungsablauf muss sicherstellen, dass keine Person in den Gefahrenbereich gelangt, während das Lichtgitter gemutet ist. Mutingsensoren können Näherungsschalter, photoelektronische Barrieren, Grenztaster, etc. sein, die nicht ausfallsicher sein müssen. Aktiver Muting-Modus muss durch Anzeigelampen indiziert werden.

Es gibt sequentielle und parallele Mutingverfahren. In diesem Baustein wird sequentielles Muting mit vier Mutingsensoren verwendet. Eine Ausführung der Vorwärtsrichtung des Transports wird unten aufgeführt. Der Baustein kann in beide Richtungen, vorwärts und rückwärts, verwendet werden. Das Muting sollte durch das MutingEnable-Signal vom Prozess aktiviert werden, um Manipulation zu vermeiden. Wenn das MutingEnable-Signal nicht verfügbar ist, muss der Eingang auf TRUE gesetzt werden.

Die Eingangsparameter des Bausteins beinhalten die Signale der vier Mutingsensoren (MutingSwitch11, MutingSwitch12, MutingSwitch21, MutingSwitch22) ebenso wie das OSSD-Signal der "aktiven opto-elektronischen Schutzeinrichtung" S_AOPD_In.

⚠ VORSICHT **S_StartReset-Eingang**

Der S_StartReset-Eingang soll nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdungssituation entstehen kann.

Nr.	Bild	Erklärung
1		Wenn der Mutingsensor MutingSwitch12 (MS_12) von einem Werkstück nach MutingSwitch11(MS12) bedämpft wird, wird der Muting-Modus aktiviert.
2		Muting-Modus bleibt so lange aktiv, wie MutingSwitch11 (MS11) und MutingSwitch12 (MS_12) durch das Werkstück bedämpft werden. Das Werkstück kann das Lichtgitter passieren, ohne einen Maschinenstopp zu verursachen.
3		Bevor die Muting-Sensoren MutingSwitch11 (MS_11) und MutingSwitch12 (MS_12) deaktiviert werden, müssen MutingSwitch21 (MS_21) und MutingSwitch22 (MS_22) aktiviert werden. Dies gewährleistet, dass der Muting-Modus aktiv bleibt.
4		Muting-Modus ist beendet, wenn nur Muting-Sensor MutingSwitch22 (MS_22) vom Werkstück bedämpft wird.

Tab. 16-166: Beispiel für SF_MutingSeq in Vorwärtsrichtung mit vier Sensoren

Statusdiagramm

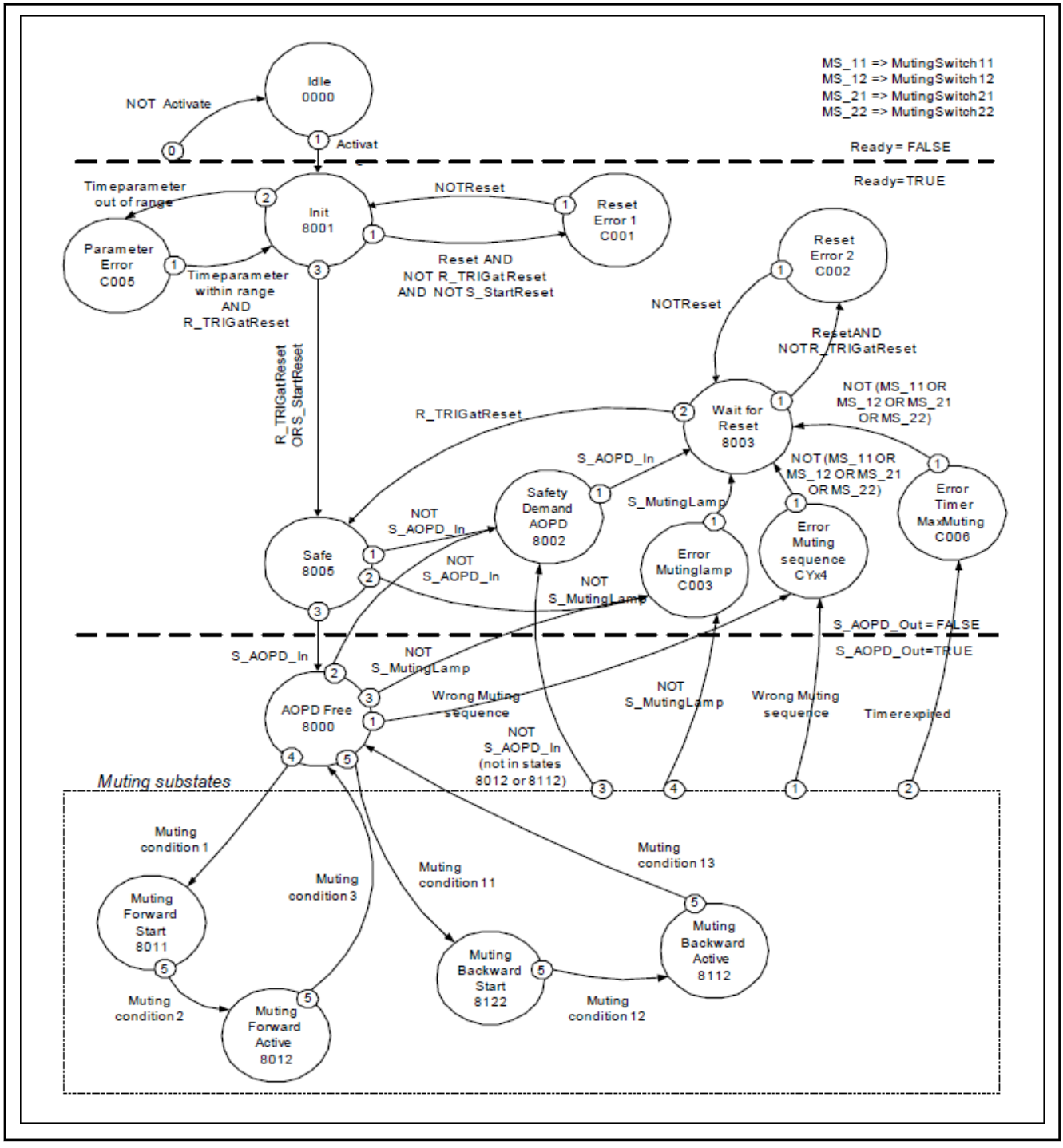


Abb. 16-80: Statusdiagramm SF_MutingSeq

Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typisches Zeitdiagramm

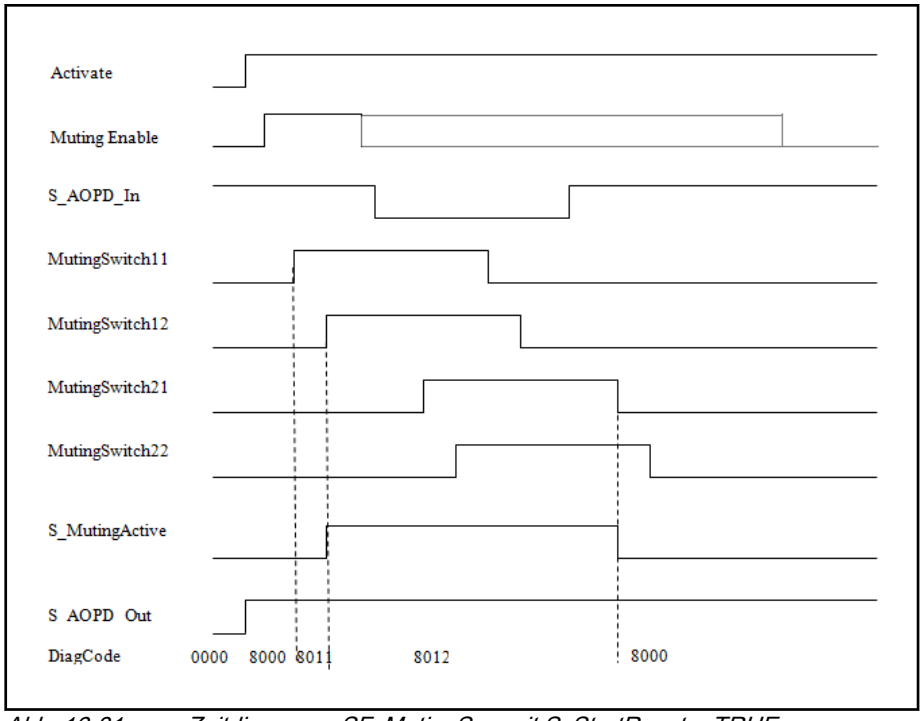


Abb. 16-81: Zeitdiagramm SF_MutingSeq mit S_StartReset = TRUE

Muting-Bedingungen

Vorwärtsrichtung

- **Muting-Bedingung 1 (zu Status 16#8011)** (MS_11 ist der erste bedämpfte Eingangssensor). Start Timer MaxMutingTime:

  MutingEnable AND (R_TRIG at MS_11 AND NOT MS_12 AND NOT MS_21 AND NOT MS_22)

- **Muting- Bedingung 2 (von Status16#8011 zu 16#8012)** (MS_12 ist der zweite bedämpfte Eingangssensor):

  MutingEnable AND (MS_11 AND R_TRIG at MS_12 AND NOT MS_21 AND NOT MS_22)

- **Muting-Bedingung 3 (von Status 16#8012 zu16# 8000)** (MS_21 erste bedämpfte Ausgangssensor). Stopp timer MaxMutingTime:

  NOT MS_11 AND NOT MS_12 AND F_TRIG at MS_21 AND MS_22

Rückwärtsrichtung

- **Muting-Bedingung 11 (zu Status 16#8122)** (MS_22 ist der erste bedämpfte Eingangssensor). Start Timer MaxMutingTime:

  MutingEnable AND (NOT MS_11 AND NOT MS_12 AND NOT MS_21 AND R_TRIG at MS_22)

- **Muting-Bedingung 12 (von Status 16#8122 zu 16#8112)** (MS_21 ist der zweite bedämpfte Eingangssensor):

  MutingEnable AND (NOT MS_11 AND NOT MS_12 AND R_TRIG at MS_21 AND MS_22)

- **Muting-Bedingung 13** (MS_12 der erste bedämpfte Ausgangssensor). Stopp Timer MaxMutingTime:

  MS_11 AND F_TRIG at MS_12 AND NOT MS_21 AND NOT MS_22

Spezifizierung falscher Muting-Sequenzen:

- In Status 16#8000: (NOT MutingEnable AND R_TRIG at MS_11) OR (NOT MutingEnable AND R_TRIG at MS_22) OR (MS_12 OR MS_21) OR (MS_11 AND MS_22)
- In Status 16#8011: NOT MutingEnable OR NOT MS_11 OR MS_21 OR MS_22
- In Status 16#8012: R_TRIG at MS_11 OR R_TRIG at MS_12 OR F_TRIG at MS_22
- In Status 16#8122: NOT MutingEnable OR MS_11 OR MS_12 OR NOT MS_22
- In Status 16#8112: F_TRIG at MS_11 OR R_TRIG at MS_21 OR R_TRIG at MS_22

Fehlererkennung

Der Baustein entdeckt folgende Fehlerbedingungen:

- Die Mutingsensoren MutingSwitch11, MutingSwitch12, MutingSwitch21 und MutingSwitch22 werden in der falschen Reihenfolge aktiviert
- Muting-Sequenz startet ohne Aktivierung durch MutingEnable
- Eine fehlerhafte Muting-Lampe wird angezeigt durch S_MutingLamp = FALSE
- Eine statische Reset-Bedingung
- MaxMutingTime wurde auf einen Wert kleiner als Null oder größer als T#10min gesetzt
- Die Muting-Funktion (S_MutingActive = TRUE) überschreitet die maximale Mutingzeit MaxMutingTime

Fehlerverhalten

Bei einem Fehlerereignis werden die Ausgänge S_AOPD und S_MutingActive auf FALSE gesetzt. Der Ausgang DiagCode zeigt den relevanten Fehlercode an und der Error-Ausgang wird auf TRUE gesetzt.

Ein Restart wird solange unterbunden, bis die Fehlerbedingungen geklärt sind und der sichere Zustand mit Reset bestätigt wird.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error 1	Statische Reset-Bedingung entdeckt nach Bausteinaktivierung Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C002	Reset Error 2	Statische Reset-Bedingung entdeckt in Status16#8003 Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C003	Error Muting Lamp	Fehler entdeckt bei Mutinglampe Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#CYx4	Error Muting Sequence	Fehler in der Mutingsequenz in Status 16#8000, 16#8011, 16#8012,16#8112 oder 16#8122 Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE Y = Status in der Sequenz (2 Status für Vorwärts- und 2 Status für Rückwärtsrichtung). 16#C0x4 = Fehler tritt in Status 16#8000 auf. 16#C1x4 = Fehler tritt im Status Vorwärts16# 8011 auf. 16#C2x4 = Fehler tritt im Status Vorwärts 16#8012 auf. 16#C3x4 = Fehler tritt im Status Rückwärts 16#8122 auf. 16#C4x4 = Fehler tritt im Status Rückwärts 16#8112 auf. 16#CFx4 = Muting Enable fehlt x = Status des Sensors, wenn ein Fehler auftritt (4 Bits: LSB = MS_11; MS_12; MS_21; MSB = MS_22).
16#C005	Parameter Error	MaxMutingTime-Wert außerhalb des gültigen Bereichs. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE
16#C006	Error Timer MaxMuting	Zeitfehler: Aktive Mutingzeit (wenn S_MutingActive = TRUE) überschreitet MaxMutingTime. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = TRUE

Tab. 16-167: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_AOPD_Out =FALSE S_MutingActive = FALSE Error = FALSE
16#8000	AOPD	Muting nicht aktiv und keine Sicherheitsanforderung von AOPD.

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8001	Init	Baustein wurde aktiviert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8002	Safety Demand AOPD	Sicherheitsanforderung wurde von AOPD entdeckt, Muting nicht aktiv. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8003	Wait for Reset	Sicherheitsanforderung oder Fehler wurden entdeckt und sind jetzt geklärt. Bestätigung Reset angefordert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8005	Safe	Sicherheitsfunktion aktiviert. Ready = TRUE S_AOPD_Out = FALSE S_MutingActive = FALSE Error = FALSE
16#8011	Muting Forward Start	Muting-Vorwärts-Sequenz ist in der Startphase, keine Sicherheitsanforderung. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive =FALSE Error = FALSE
16#8012	Muting Forward Active	Muting-Vorwärts-Sequenz ist aktiv. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8112	Muting Backward Active	Muting-Rückwärts, Sequenz ist aktiv. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = TRUE Error = FALSE
16#8122	Muting Backward Start	Muting-Rückwärts, Sequenz ist in der Startphase und keine Sicherheitsanforderung. Ready = TRUE S_AOPD_Out = TRUE S_MutingActive = FALSE Error = FALSE

Tab. 16-168: Bausteinspezifische Status-Codes

16.4.14 SF_OutControl

Kurzbeschreibung

SF_OutControl ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein steuert eine Sicherheits-Ausgang mit einem Signal der Standardsteuerung und einem sicheren Signal mit optionaler Anlaufsperre.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-169: Bibliothekszuordnung

Schnittstellenbeschreibung

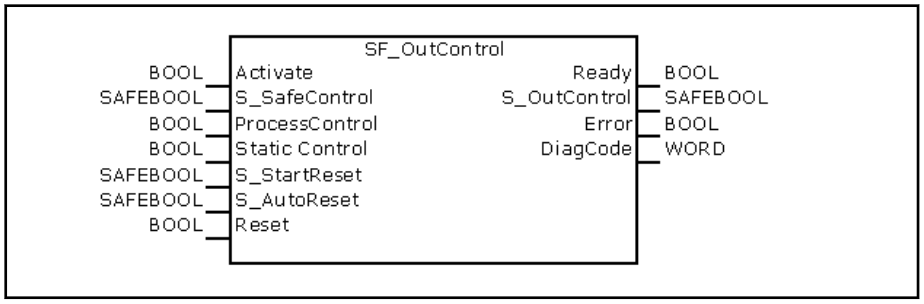


Abb. 16-82: Funktionsbaustein SF_OutControl

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_SafeControl	SAFEBOOL	Variable Steuerungssignal des vorgeschalteten Sicherheitsbausteins Typische Bausteinsignale der Bibliothek (zum Beispiel: SF_EmergencyStop, SF_GuardMonitoring, SF_TwoHand-ControlTypeII) FALSE: Die vorgeschalteten Sicherheitsbausteine sind im sicheren Zustand TRUE: Die vorgeschalteten Sicherheitsbausteine aktivieren die Sicherheitsüberwachung
	ProcessControl	BOOL	Variable oder Konstante Steuerungssignal der Standardsteuerung FALSE: Anforderung, S_OutControl auf FALSE zu setzen TRUE: Anforderung S_OutControl auf TRUE zu setzen
	StaticControl	BOOL	Kontante Optionale Bedingungen zur Prozesssteuerung FALSE: Dynamischer Wechsel von ProcessControl (FALSE -> TRUE) erforderlich nach Bausteinaktivierung oder getriggerter Sicherheitsfunktion. Zusätzlicher Funktionsstart erforderlich TRUE: Kein dynamischer Wechsel von ProcessControl (FALSE -> TRUE) erforderlich nach Bausteinaktivierung oder getriggerter Sicherheitsfunktion
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_AutoReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_OutControl	SAFEBOOL	Steuert angeschlossene, sichere Geräte oder Bausteine FALSE: Deaktivierung der sicheren Geräte oder Bausteine TRUE: Aktivierung der sicheren Geräte oder Bausteine
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-170: Schnittstelle SF_OutControl

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_SafeControl	SAFEBOOL	FALSE	
ProcessControl	SAFEBOOL	FALSE	
StaticControl	TIME	T#0ms	Konstante

Name	Typ	Default-Wert	Übernahme
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist. (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
S_AutoReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset wenn NOT-HALT-Knopf entriegelt ist TRUE: Automatischer Reset, wenn NOT-HALT-Knopf entriegelt ist
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-171: Eingangsverhalten SF_OutControl

Funktionsbeschreibung

Der Baustein SF_OutControl ist ein Ausgangstreiber für einen Sicherheitsausgang.

Der Sicherheitsausgang wird über S_OutControl gesteuert, der ein Signal der Standardsteuerung (ProcessControl (BOOL) zur Prozesssteuerung) und ein Signal der Sicherheitsanwendung (S_SafeControl (SAFEBOOL) zur Steuerung der Sicherheitsfunktion) verwendet.

Optionale Bedingungen zur Prozesssteuerung (ProcessControl):

- Ein zusätzlicher Funktionsstart (ProcessControl FALSE -> TRUE) ist erforderlich, welcher der Bausteinaktivierung oder der Rückmeldung des sicheren Signals (S_SafeControl) folgt. Ein statisches TRUE-Signal an ProcessControl setzt S_OutControl nicht auf TRUE.
- Ein zusätzlicher Funktionsstart (ProcessControl FALSE -> TRUE) ist nicht erforderlich, welcher der Bausteinaktivierung oder der Rückmeldung des sicheren Signals (S_SafeControl) folgt. Ein statisches TRUE-Signal an ProcessControl setzt S_OutControl auf TRUE, wenn die anderen Bedingungen erfüllt wurden.

Optionale Anlaufsperren

- Anlaufsperre nach Bausteinaktivierung
- Anlaufsperren nach Unterbrechung des Schutzgeräts

Statusdiagramm

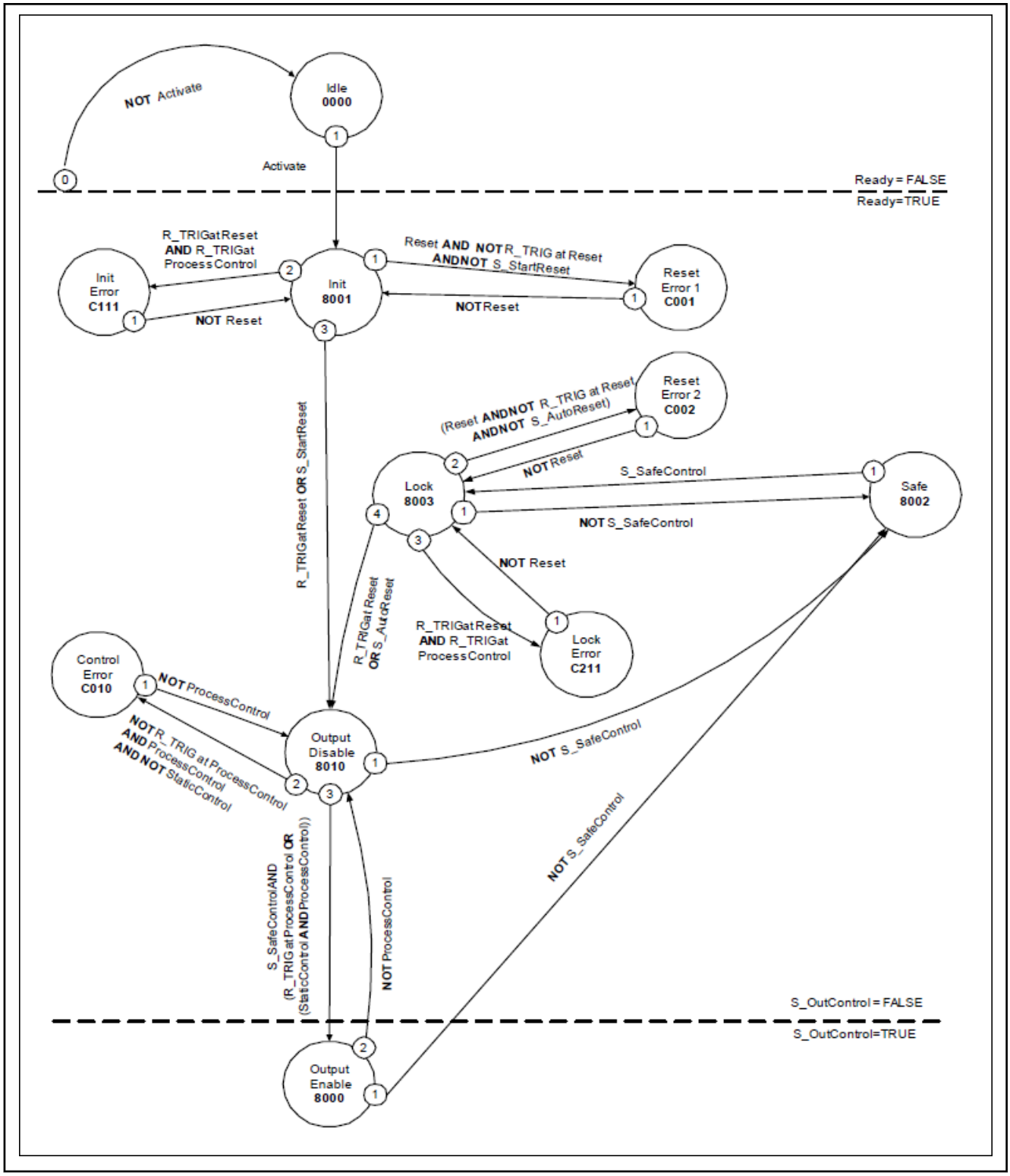


Abb. 16-83: Statusdiagramm SF_OutControl

Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typische Zeitdiagramme

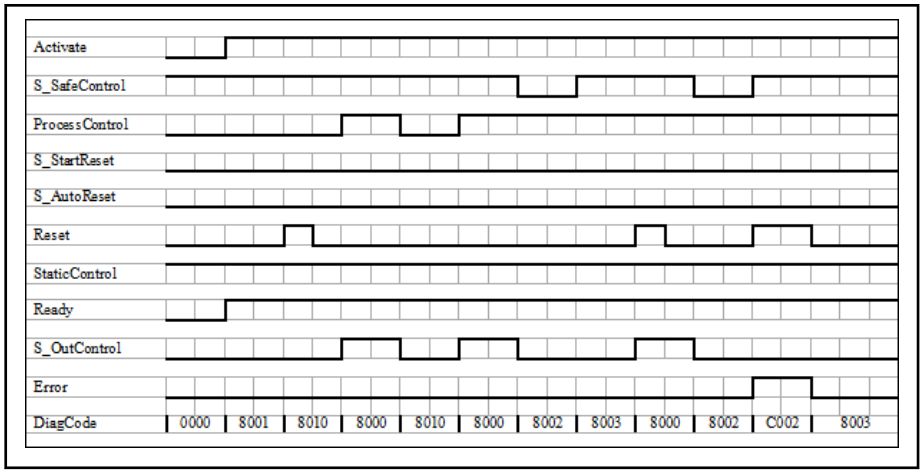


Abb. 16-84: Typisches Zeitdiagramm SF_OutControl, S_StartReset = FALSE

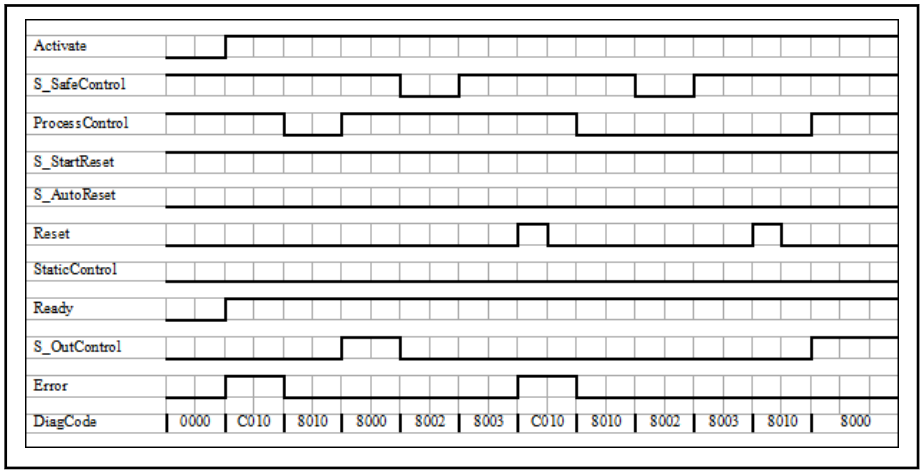


Abb. 16-85: Typisches Zeitdiagramm SF_OutControl, S_StartReset = TRUE

⚠ VORSICHT **StaticControl, S_StartReset und S_AutoReset**

Die Eingänge StaticControl, S_StartReset und S_AutoReset sollen nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdungssituation entstehen kann, wenn die S-SPS startet.

Fehlererkennung

Die folgenden Bedingungen führen in den Fehlerzustand:

- Ungültiges statisches Reset-Signal im Prozess.
- Ungültiges statisches ProcessControl-Signal.
- ProcessControl und Reset sind wegen eines Programmierfehlers falsch miteinander verbunden

Fehlerverhalten

Bei einem Fehlerereignis wird der Ausgang S_OutControl auf FALSE gesetzt und bleibt in diesem sicheren Zustand. Um die Fehlerzustände Reset Error, Init Error oder Lock Error zu verlassen, muss der Reset-Eingang auf FALSE gesetzt werden. Um den Fehlerzustand Control Error zu verlassen, muss der Eingang ProcessControl auf FALSE gesetzt werden.

Nach dem Übergang von S_SafeControl nach TRUE, kann die optionale Anlaufsperre durch eine steigende Flanke am Reset-Eingang zurückgesetzt werden.

Nach der Bausteinaktivierung kann die optionale Anlaufsperre durch eine steigende Flanke am Reset-Eingang zurückgesetzt werden.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Reset Error 1	Statisches Reset-Signal in Status 16#8001. Ready = TRUE S_OutControl = FALSE Error = TRUE
16#C002	Reset Error 2	Statisches Reset-Signal in Status 16#8003. Ready = TRUE S_OutControl = FALSE Error = TRUE
16#C010	Control Error	Statisches Signal an ProcessControl in Status 16#8010. Ready = TRUE S_OutControl = FALSE Error = TRUE
16#C111	Init Error	Gleichzeitige steigende Triggerflanke an Reset und Processcontrol in Status16#8001. Ready = TRUE S_OutControl = FALSE Error = TRUE
16#C211	Lock Error	Gleichzeitige steigende Triggerflanke an Reset und ProcessControl in Status 16#8003. Ready = TRUE S_OutControl = FALSE Error = TRUE

Tab. 16-172: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_OutControl = FALSE Error = FALSE
16#8001	Init	Bausteinaktivierung Anlaufsperre ist aktiv. Reset erforderlich. Ready = TRUE S_OutControl = FALSE Error = FALSE
16#8002	Safe	Getriggerte Sicherheitsfunktion. Ready = TRUE S_OutControl = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8003	Lock	Sicherheitsfunktion Anlaufsperre ist aktiv. Reset erforderlich. Ready = TRUE S_OutControl = FALSE Error = FALSE
16#8010	Output Disable	Prozessüberwachung ist nicht aktiv. Ready = TRUE S_OutControl = FALSE Error = FALSE
16#8000	Output Enable	Prozessüberwachung ist aktiv und Sicherheit ist aktiviert. Ready = TRUE S_OutControl = TRUE Error = FALSE

Tab. 16-173: Bausteinspezifische Status-Codes

16.4.15 SF_SafetyRequest

Kurzbeschreibung

SF_SafetyRequest ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Der Baustein SF_SafetyRequest stellt die Schnittstelle zwischen dem Benutzerprogramm und der Systemumgebung dar.

Er stellt die Schnittstelle zu einem allgemeinen Aktor, zum Beispiel einem Sicherheitsantrieb oder einem Sicherheitsventil dar, um ihn in den sicheren Zustand zu versetzen.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-174: Bibliothekszuordnung

Schnittstellenbeschreibung

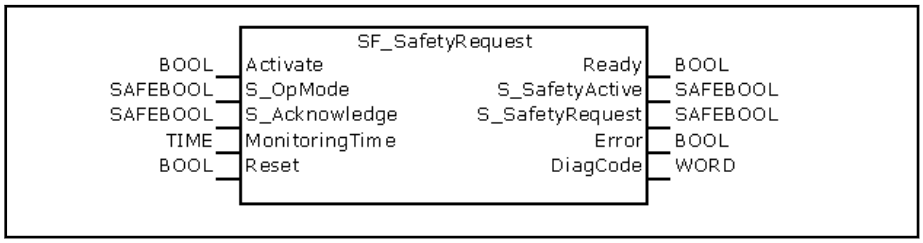


Abb. 16-86: Funktionsbaustein SF_SafetyRequest

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_OpMode	SAFEBOOL	Variable Angeforderter Modus an die angeschlossene Peripherie FALSE: Sicherer Modus ist angefordert TRUE: BetriebsModus (nicht sicher) ist angefordert

E/A-TYP	Name	Datentyp	Beschreibung
	S_Acknowledge	SAFEBOOL	Variable Bestätigung der angeschlossenen Peripherie FALSE: Betriebsmodus (nicht sicher) TRUE: Sicherer Modus
	MonitoringTime	TIME	Konstante Überwachung der Antwortzeit zwischen der Anforderung einer Sicherheitsfunktion (S_OpMode auf FALSE) und der Aktor-Bestätigung (S_Acknowledge schaltet auf TRUE)
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_SafetyActive	SAFEBOOL	Bestätigung des sicheren Zustands FALSE: Kein sicherer Zustand TRUE: Sicherer Zustand
	S_SafetyRequest	SAFEBOOL	Anforderung, die Peripherie in den sicheren Zustand zu versetzen FALSE: Sicherer Zustand ist angefordert TRUE: Nicht-sicherer Zustand ist angefordert
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-175: Schnittstelle SF_SafetyRequest

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_OpMode	SAFEBOOL	FALSE	
S_Acknowledge	SAFEBOOL	FALSE	
MonitoringTime	TIME	T#0s	Konstante
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-176: Eingangsverhalten SF_SafetyRequest

Funktionsbeschreibung Dieser Baustein stellt die Schnittstelle zwischen dem sicherheitsgerichteten System und der angeschlossenen sicheren Peripherie dar.

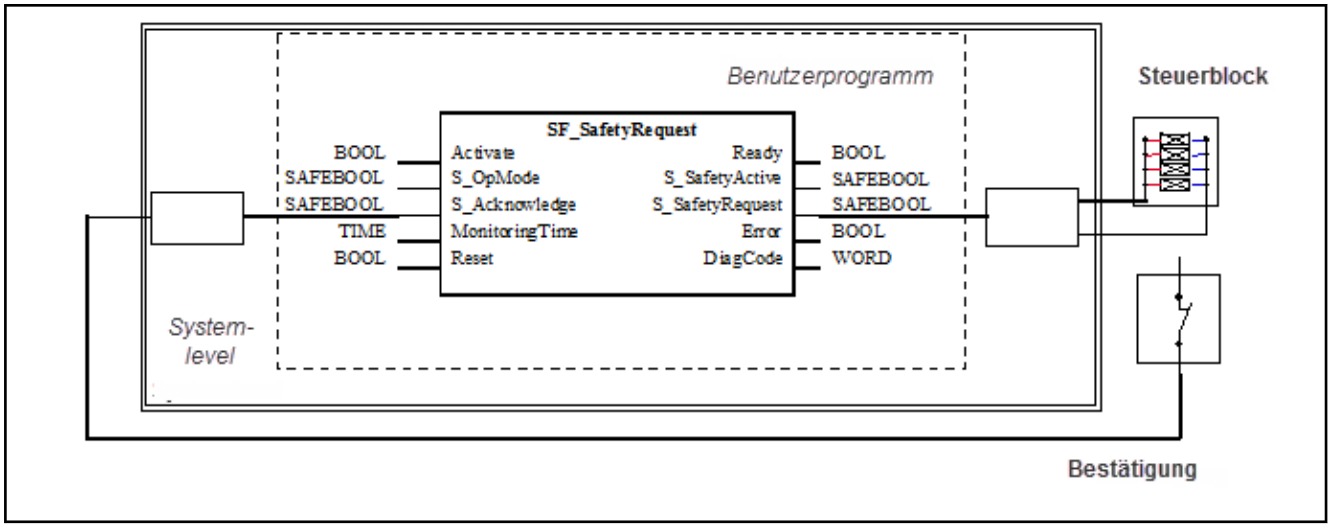


Abb. 16-87: Beispiel SF_SafetyRequest

Dies bedeutet, dass die Sicherheitsfunktionen der angeschlossenen sicheren Peripherie im Anwenderprogramm (Applikation) zur Verfügung stehen. Allerdings gibt es nur 2 binäre Signale um den sicheren Zustand der angeschlossenen Peripherie zu kontrollieren, das heißt ein Signal für die Anforderung und ein Signal für den Empfang der Bestätigung.

Die Sicherheitsfunktion wird von der angeschlossenen, sicheren Peripherie selbständig durchgeführt. Der Baustein SF_SafetyRequest initiiert hierfür nur die Anforderung, überwacht sie und setzt den Ausgang, wenn die angeschlossene, sichere Peripherie den sicheren Zustand bestätigt. Dies wird durch den Ausgang S_SafetyActive angezeigt. Der Baustein definiert nicht die Parameter der angeschlossenen, sicheren Peripherie. Diese sollten von der sicheren Peripherie selbst definiert werden. SF_SafetyRequest schaltet die angeschlossene, sichere Periphere vom Betriebsmodus in den sicheren Modus.

Statusdiagramm

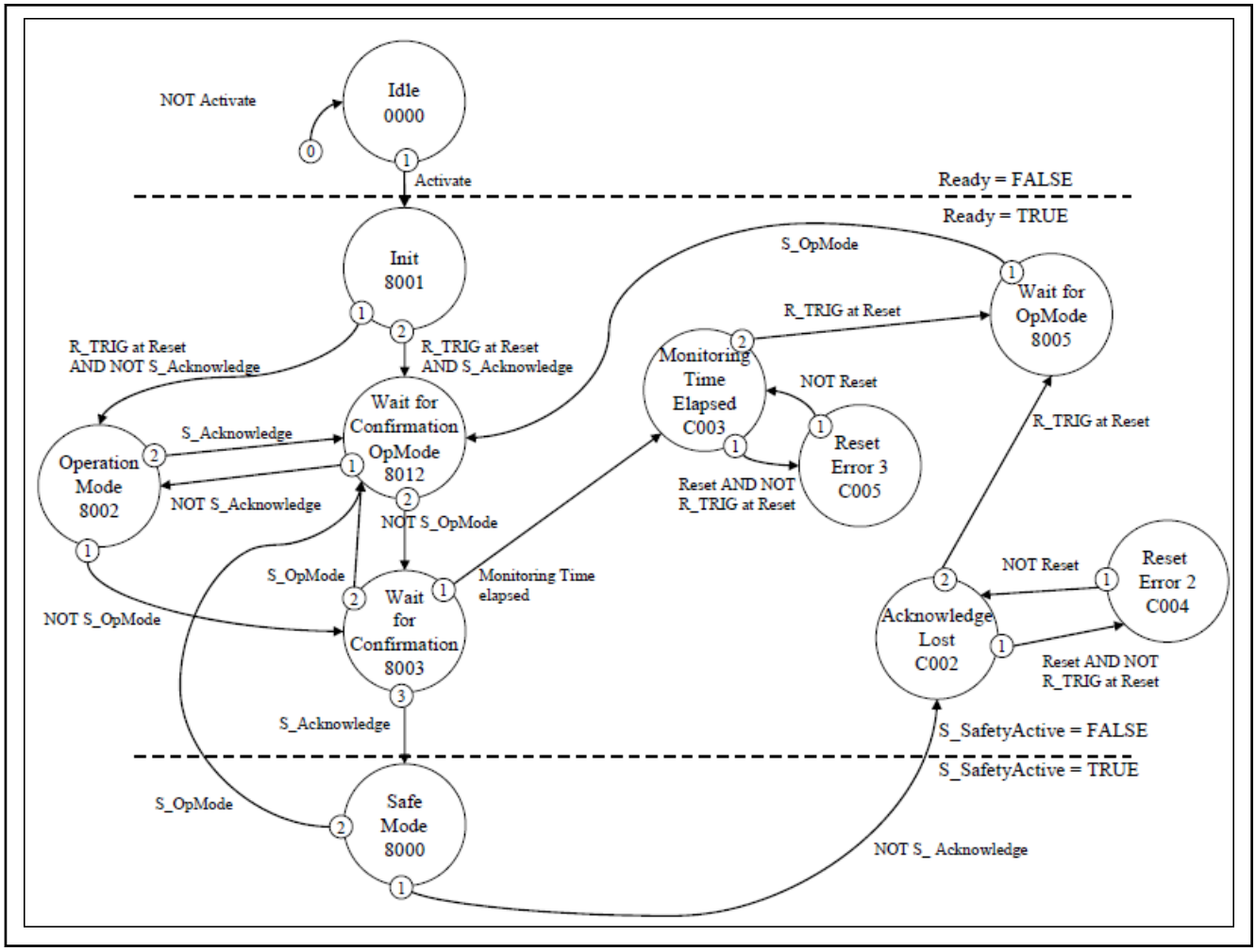


Abb. 16-88: Statusdiagramm SF_SafetyRequest



Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typische Zeitdiagramme

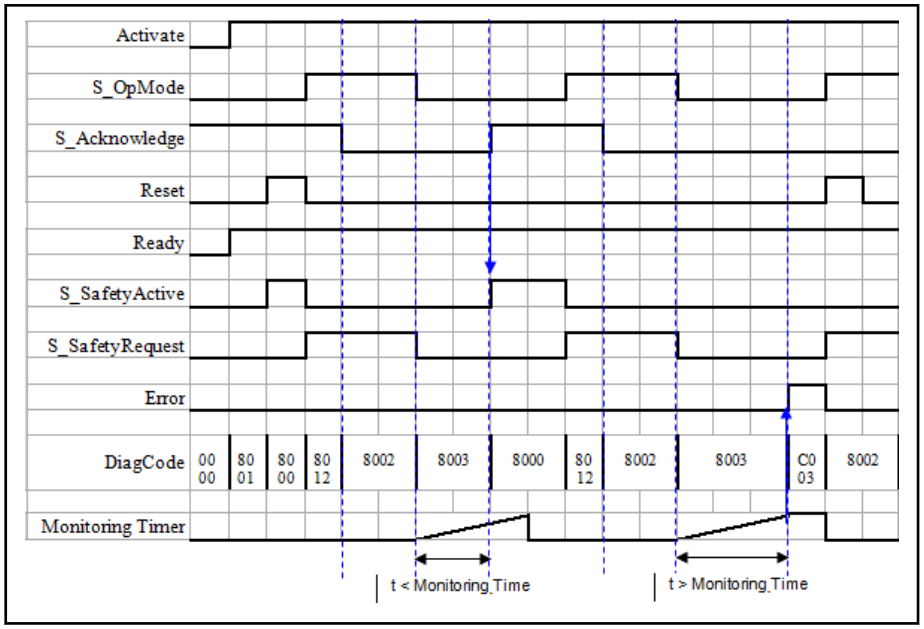


Abb. 16-89: Typisches Zeitdiagramm SF_SafetyRequest

Fehlererkennung

Der Baustein erkennt, wenn der Aktor innerhalb der Überwachungszeit nicht in den sicheren Zustand eintritt.

Der Baustein erkennt, wenn das Bestätigungssignal, obwohl die Anforderung noch aktiv ist, abhanden kommt.

Der Baustein erkennt ein statisches Reset-Signal.

Externe Baustein-Fehler: Es gibt keine externen Fehler, da es keine Fehlerbits/Fehlerinformationen gibt, die die angeschlossene, sichere Peripherie bereitstellt.

Fehlerverhalten

Bei einem Fehlerereignis wird der Ausgang S_SafetyActive auf FALSE gesetzt.

Ein Fehler muss mit einer steigenden Triggerflanke am Reset-Eingang bestätigt werden. Um mit dem Baustein nach diesem Reset fortzufahren, muss die Anforderung S_OpMode auf TRUE gesetzt werden.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C002	Acknwledge Lost	Abschalten der Bestätigung im sicheren Zustand Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = TRUE
16#C003	Monitoring Time Elapsed	Die Anforderung S_OpMode konnte während der Überwachungszeit nicht abgearbeitet werden. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C004	Reset Error 2	Statischer Reset entdeckt in Status 16#C002 (Bestätigung verloren) Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = TRUE
16#C005	Reset Error 3	Statischer Reset entdeckt in Status 16#C003 (Überwachungszeit abgelaufen) Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = TRUE

Tab. 16-177: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = FALSE
16#8000	Safe Mode	Peripherie (Aktor) ist im sicheren Modus. Ready = TRUE S_SafetyActive =TRUE S_SafetyRequest = FALSE Error = FALSE
16#8001	Init	Status nachdem Activate auf TRUE gesetzt wurde oder nach steigender Triggerflanke an Reset. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = FALSE
16#8002	Operation Mode	Betriebsmodus ohne Bestätigen des sicheren Modus. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = TRUE Error = FALSE
16#8012	Wait for Confirmation OpMode	Betriebsmodus mit Bestätigen des sicheren Modus. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = TRUE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8003	Wait for Confirmation	Warten auf Bestätigung des Antriebs(Systemschnittstelle). Ready = TRUE S_SafetyActice = FALSE S_SafetyRequest =FALSE Error = FALSE
16#8005	Wait for OpMode	Fehler wurde bestätigt. Dennoch muss S_OpMode auf TRUE gesetzt werden, bevor der Baustein initialisiert werden kann. Ready = TRUE S_SafetyActive = FALSE S_SafetyRequest = FALSE Error = FALSE

Tab. 16-178: Bausteinspezifische Status-Codes

16.4.16 SF_TestableSafetySensor

Kurzbeschreibung

SF_TestableSafetySensor ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein wird verwendet, um die Funktion einer elektronischen/optischen Schutzeinrichtung zu überprüfen.

Der Baustein kann auch für externe testbare Sicherheitssensoren (ESPE: Elektro-sensitive Schutzeinrichtung, z. B. Lichtstrahl) verwendet werden.

Er überwacht ferner den Status des Sicherheits-Sensors.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-179: Bibliothekszuordnung

Schnittstellenbeschreibung

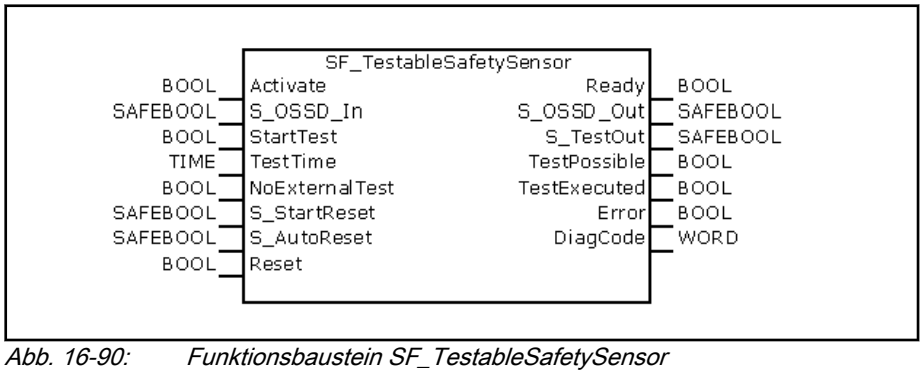


Abb. 16-90: Funktionsbaustein SF_TestableSafetySensor

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_OSSD_In	SAFEBOOL	Variable Status des Sensor-Ausgangs, z. B. Lichtgitter FALSE: Sicherheits-Sensor in Testzustand oder Anforderung für sicherheitsgerichtete Antwort TRUE: Sensor im Status für normalen Betriebzustand
	StartTest	BOOL	Variable Eingang um Sensor-Test zu starten Setzt S_TestOut und startet die interne Zeitüberwachungsfunktion des Bausteins FALSE: kein Test angefordert TRUE: Test angefordert
	TestTime	TIME	Konstante Bereich: 0 ... 150 ms Testzeit für Sicherheits-Sensor
	NoExternalTest	BOOL	Konstante Zeigt an, ob externer, manueller Sensor-Test unterstützt wird FALSE: Der externe, manuelle Sensor-Test wird unterstützt. Nach einem fehlerhaften automatischen Sensor-Test ist nur nach einer vollständigen manuellen Sensor-Schaltfolge, wieder ein automatischer Test möglich TRUE: Der externe, manuelle Sensor-Test wird nicht unterstützt. Nach einem fehlerhaften automatischen Sensor-Test ist wieder ein automatischer Test möglich - ohne manueller Sensor-Schaltfolge
	S_StartReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_AutoReset	SAFEBOOL	"Allgemeine Eingangsparameter" auf Seite 294
	Reset	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_OSSD_Out	SAFEBOOL	Sicherheitsgerichteter Ausgang, der den Status des ESPE anzeigt FALSE: Der Sensor hat eine Anforderung für sicherheitsgerichtete Aktion, oder Testfehler TRUE: Der Sensor hat keine Anforderung für sicherheitsgerichtete Aktion, UND kein Testfehler
	S_TestOut	SAFEBOOL	Gekoppelt mit dem Testeingang des Sensors Obwohl als SAFEBOOL spezifiziert, wird das Signal in der Praxis oft mit einem BOOL-Ausgang verbunden **TRUE**: Keine Testanforderung, Initialwert FALSE: Testanforderung erteilt

E/A-TYP	Name	Datentyp	Beschreibung
	TestPossible	BOOL	Rückmeldungssignal zum Prozess FALSE: Ein automatischer Sensor-Test ist nicht möglich TRUE: Ein automatischer Sensor-Test ist möglich
	TestExecuted	BOOL	Eine positive Signalflanke zeigt die erfolgreiche Ausführung des automatischen Sensor-Tests an FALSE: • Ein automatischer Sensor-Test wurde noch nicht durchgeführt • Ein automatischer Sensor-Test ist aktiv • Ein automatischer Sensor-Test war fehlerhaft TRUE: Ein Sensor-Test wurde erfolgreich durchgeführt
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-180: Schnittstelle SF_TestableSafetySensor

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_OSSD_In	SAFEBOOL	FALSE	gesteuert durch Activate
StartTest	BOOL	FALSE	
TestTime	TIME	T#10ms	Konstante
NoExternalTest	BOOL	FALSE	Konstante
S_StartReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset, wenn die S-SPS gestartet ist (warm oder kalt) TRUE: Automatischer Reset wenn die S-SPS gestartet wird (warm oder kalt)
S_AutoReset	SAFEBOOL	FALSE	FALSE (= Initialwert): Manueller Reset wenn NOT-HALT-Knopf entriegelt ist TRUE: Automatischer Reset, wenn NOT-HALT-Knopf entriegelt ist
Reset	BOOL	FALSE	Flanke 0-1

Tab. 16-181: Eingangsverhalten SF_TestableSafetySensor

☞ OSSD steht für: Output Signal Switching Device

Funktionsbeschreibung

Type 2 ESPE dient dem periodischen Testen. Gefährlichen Fehler (zum Beispiel, Ausfall des Sensors, Antwortzeit überschreitet die spezifizierte) sollen so im voraus erkannt werden.

Das Testsignal soll die Betätigung des Messgeräts simulieren, und die Dauer des periodischen Tests soll 150 ms nicht überschreiten.

Der Test soll verifizieren, dass jeder Lichtstrahl in der Art und Weise arbeitet, wie es durch den Anwender festgelegt wurde.

Wenn der Test durch ein externes, sicherheitsgerichtetes Steuerungssystem (z. B. eine Maschine) initiiert werden soll, sollte die ESPE mit geeigneten Eingabe-Einrichtungen (z. B. Terminal) ausgestattet sein.



Die ESPE muss entsprechend der Produktstandards und der geforderten Kategorien gemäß der Normen ausgewählt werden.

Es muss mit einer separaten Funktionalität überwacht werden, dass der Test innerhalb von angemessenen Intervallen initiiert wird.

⚠ VORSICHT **S_StartReset- und S_AutoReset**

Die S_StartReset- und S_AutoReset-Eingänge sollen nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdung entstehen kann, wenn die S-SPS startet.

Testmodus

1. StartTest = TRUE: STestOut = FALSE. Start der Überwachungszeit
2. S_TestOut-Signal stoppt den Transmitter (Überwachung von TestTime zum ersten Mal gestartet)
3. S_OSSD_In wechselt von TRUE nach FALSE (Überwachung von Test-Time zum zweiten Mal gestartet)
4. S_TestOut wechselt von FALSE nach TRUE
5. Start Transmitter
6. Sensor S_OSSD_In wechselt von FALSE nach TRUE
7. Stopp Überwachungszeit
8. S_OSSD_Out ist während des Testens auf TRUE

Optionale Anlaufsperren

- Anlaufsperre nach Aktivierung des Bausteins
- Anlaufsperre nach Unterbrechung der Schutzeinrichtung

Statusdiagramm

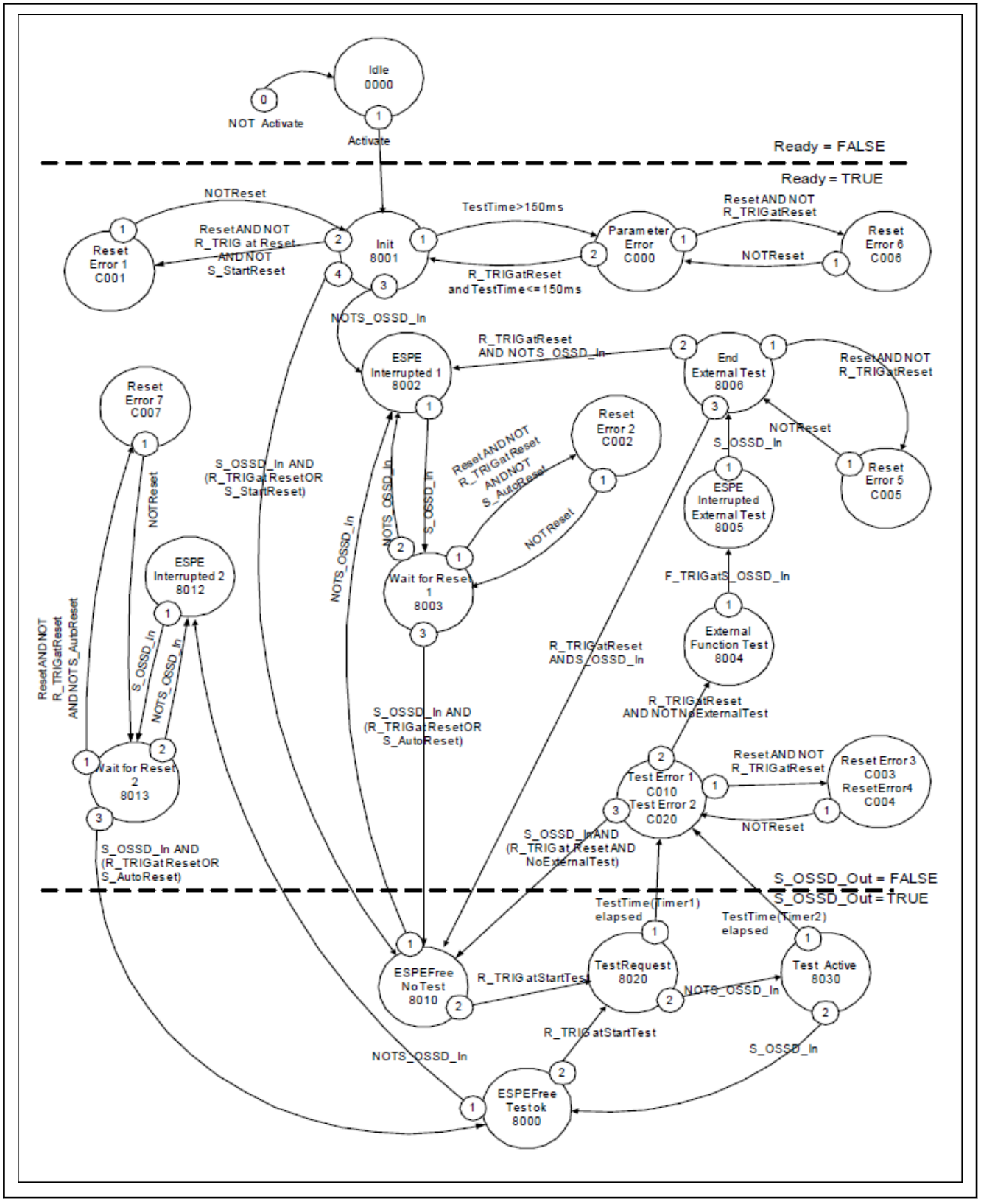


Abb. 16-91: Statusdiagramm SF_TestableSafetySensor

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Zeitdiagramm

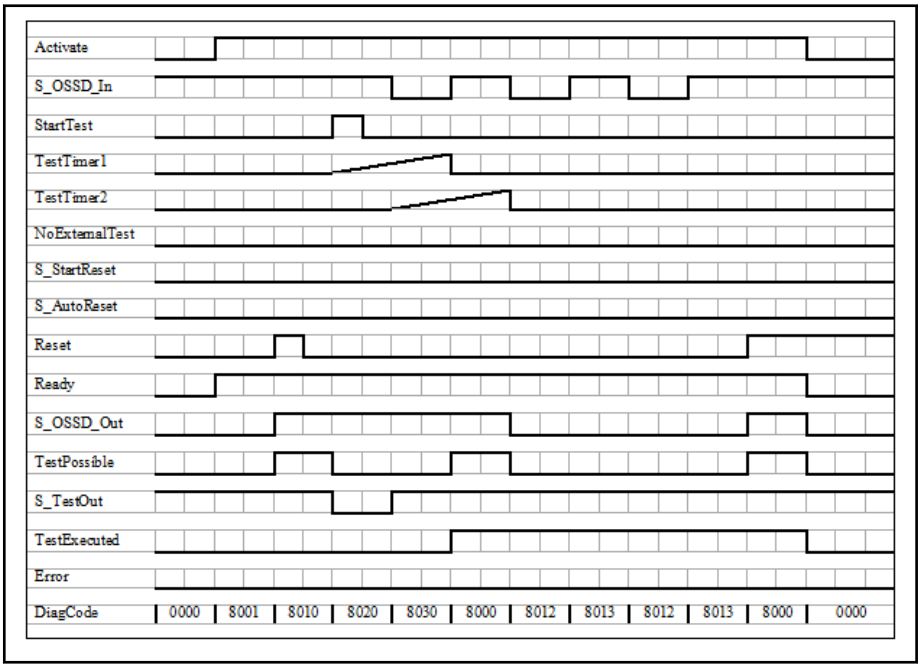


Abb. 16-92: Zeitdiagramm SF_TestableSafetySensor

Fehlererkennung

Die folgenden Bedingungen forcieren einen Übergang in den Fehlerstatus:

- Testzeitüberschreitung ohne verzögerte Sensorrückmeldung
- Test ohne Sensorsignal-Rückmeldung
- Ungültiges statisches Reset-Signal im Prozess
- Plausibilitätsprüfung der Einstellung der Überwachungszeit

Fehlerverhalten

Bei einem Fehlerereignis wird der Ausgang S_OSSD_Out auf FALSE gesetzt und bleibt in diesem Zustand. Sobald der Fehler behoben wurde und der Sensor in Betrieb ist (S_OSSD_In = TRUE), entfernt ein Reset den Fehlerstatus und setzt den Ausgang S_OSSD_Out auf TRUE.

Wenn S_AutoReset = FALSE, wird eine steigende Triggerflanke an Reset gefordert.

Nach dem Übergang von S_OSSD_In nach TRUE, kann die optionale Anlaufsperre durch eine steigende Flanke am Reset-Eingang wieder gestartet werden.

Nach Aktivierung des Bausteins kann die optionale Anlaufsperre durch eine steigende Triggerflanke am Reset-Eingang wieder gestartet werden.

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C000	Parameter Error	Ungültiger Wert des Parameters TestTime. Werte zwischen 0 ms und 150 ms sind möglich. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
16#C001	Reset Error 1	Statische Reset-Bedingung wurde nach Bausteinaktivierung entdeckt. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#C002	Reset Error 2	Statische Reset-Bedingung wurde in Status 16#8003 entdeckt. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
16#C003	Reset Error 3	Statische Reset-Bedingung entdeckt in Status 16#C010 Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
16#C004	Reset Error 4	Statische Reset-Bedingung entdeckt in Status 16#C020. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C005	Reset Error 5	Statische Reset-Bedingung entdeckt in Status 16#8006. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
16#C006	Reset Error 6	Statische Reset-Bedingung entdeckt in Status 16#C000. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = TRUE
16#C007	Reset Error 7	Statische Reset-Bedingung entdeckt in Status 16#8013. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = TRUE Error = TRUE
16#C010	Test Error 1	Testzeit abgelaufen in Status 16#8020. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted= FALSE Error = TRUE
16#C020	Test Error 2	Testzeit abgelaufen in Status 16#8030. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted= FALSE Error = TRUE

Tab. 16-182: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#8001	Init	Eine Aktivierung wurde durch den Baustein entdeckt. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#8002	ESPE Interrupted 1	Der Baustein hat eine Sicherheitsanforderung entdeckt. Der Schalter ist noch nicht automatisch getestet worden. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#8003	Wait for Reset 1	Warten auf steigende Triggerflanke nach Status 16#8002. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#8004	External Function Test	Der automatische Sensortest war fehlerhaft. Ein externer manueller Sensortest ist notwendig. Die Unterstützung für den notwendigen, externen manuellen Sensortest wurde am Baustein aktiviert (NoExternalTest = FALSE). Am Sensor ist eine negative Signalflanke gefordert. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8005	ESPE Interrupted External Test	Der automatische Sensortest war fehlerhaft. Ein externer manueller Sensortest ist notwendig. Die Unterstützung für den notwendigen, externen manuellen Sensortest wurde am Baustein aktiviert (NoExternalTest = FALSE). Am Sensor ist eine TRUE-Signal gefordert. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#8006	End External Test	Der automatische Sensortest war fehlerhaft. Ein externer manueller Sensortest ist notwendig. Die Unterstützung für den notwendigen, externen manuellen Sensortest wurde am Baustein aktiviert (NoExternalTest = FALSE). Der externe manuelle Test ist vollständig. Der Baustein entdeckte einen vollständigen Sensor-Schaltzyklus (extern kontrolliert). Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#8010	ESPE Free No Test	Der Baustein hat keine Sicherheitsanforderung entdeckt. Der Sensor wurde nicht automatisch getestet. Ready = TRUE S_OSSD_Out = TRUE S_TestOut = TRUE TestPossible = TRUE TestExecuted = FALSE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8020	Test Request	Der automatische Sensortest ist aktiv. Test Timer ist das erste Mal gestartet. Das Transmitter-Signal wurde durch den Baustein ausgeschaltet. Das Signal des Empfängers muss dem Signal des Transmitters folgen. Ready = TRUE S_OSSD_Out = TRUE S_TestOut = FALSE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#8030	Test Active	Der automatische Sensortest ist aktiv. Der Test Timer ist das zweite Mal gestartet. Das Transmitter-Signal des Sensors wurde durch den Baustein eingeschalten. Das Signal des Empfängers muss dem Signal des Transmitters folgen. Ready = TRUE S_OSSD_Out = TRUE S_TestOut = TRUE TestPossible = FALSE TestExecuted = FALSE Error = FALSE
16#8000	ESPE Free Test OK	Der Baustein hat keine Sicherheitsanforderung entdeckt. Der Sensor wurde automatisch getestet. Ready = TRUE S_OSSD_Out = TRUE S_TestOut = TRUE TestPossible = TRUE TestExecuted =TRUE Error = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8012	ESPE Interrupted 2	Der Baustein hat eine Sicherheitsanforderung entdeckt. Der Schalter wurde automatisch getestet. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted =TRUE Error = FALSE
16#8013	Wait for Reset 2	Warten auf eine steigende Triggerflanke von Reset nach Status 16#8012. Ready = TRUE S_OSSD_Out = FALSE S_TestOut = TRUE TestPossible = FALSE TestExecuted =TRUE Error = FALSE

Tab. 16-183: Bausteinspezifische Status-Codes

16.4.17 SF_TwoHandControlTypeII

Kurzbeschreibung

SF_TwoHandControlTypeII ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein unterstützt in einer Applikation die Funktion "Zweihandschaltung Typ 2" entsprechend der europäischen Norm.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-184: Bibliothekszuordnung

Schnittstellenbeschreibung

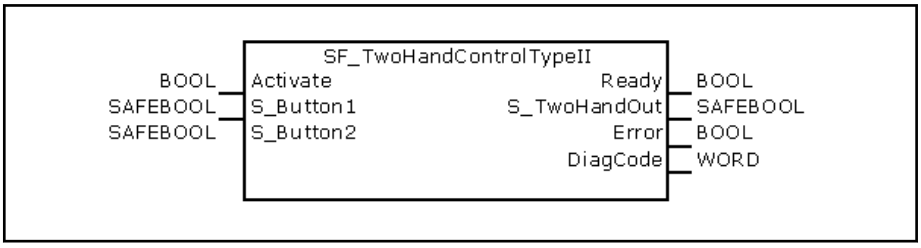


Abb. 16-93: Funktionsbaustein SF_TwoHandControlTypeII

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_Button1	SAFEBOOL	Variable Eingang für Taster 1 (für Kategorie 3 oder 4: zwei antivalente Kontakte) FALSE: Taster 1 ist gelöst TRUE: Taster 1 ist betätigt
	S_Button2	SAFEBOOL	Variable Eingang von Taster 2 (für Kategorie 3 oder 4: zwei antivalente Kontakte) FALSE: Taster 2 ist gelöst TRUE: Taster 2 ist betätigt
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_TwoHandOut	SAFEBOOL	Sicherheitsgerichtetes Ausgangssignal FALSE: Keine korrekte Zweihandbedienung TRUE: Die Eingänge S_Button1 und S_Button2 sind TRUE und kein Fehler trat auf. Korrekte Zweihandbedienung
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-185: Schnittstelle SF_TwoHandControlTypeII

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_Button1	SAFEBOOL	FALSE	
S_Button2	SAFEBOOL	FALSE	

Tab. 16-186: Eingangsverhalten SF_TwoHandControlTypeII



Es ist kein Reset-Eingang oder Fehlerausgang erforderlich, da an beiden Schaltern kein Test durchgeführt werden kann.

Funktionsbeschreibung

Dieser Baustein unterstützt die Funktion "Zweihandschaltung Typ 2" entsprechend der europäischen Norm. Wenn S_Button1 und S_Button2 in richtiger Folge auf TRUE gesteuert werden, dann wird auch der S_TwoHandOut-Ausgang auf TRUE gesetzt. Ebenso kontrolliert der Baustein das Lösen beider Taster, bevor der Ausgang S_TwoHandOut erneut auf TRUE gesetzt wird.

Statusdiagramm

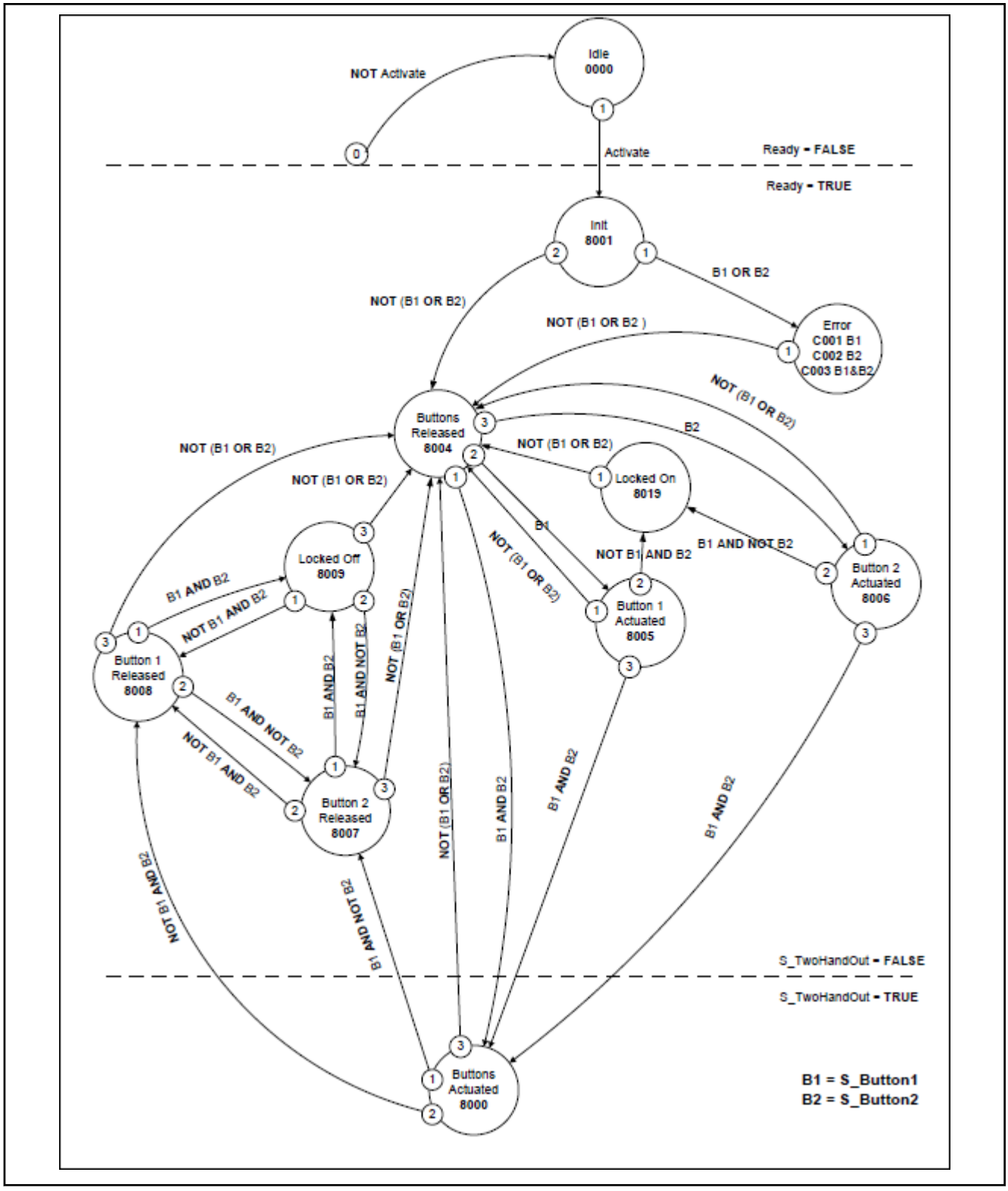


Abb. 16-94: Statusdiagramm SF_TwoHandControlTypeII

☞ Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typisches Zeitdiagramm

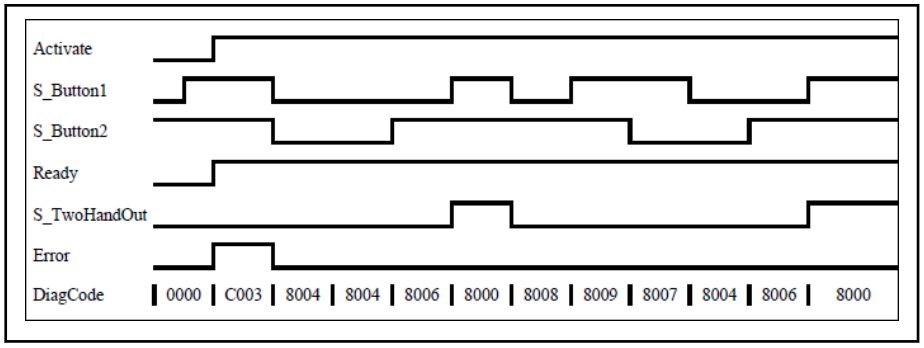


Abb. 16-95: Typisches Zeitdiagramm SF_TwoHandControlTypeII

Fehlererkennung

Nach Aktivierung des Bausteins wird das Steuern jedes Tasters auf TRUE als ungültiges Eingangssignal erkannt, das zu einem Fehler führt.

Fehlerverhalten

Bei einem Fehlerereignis wird der S_TwoHandOut-Ausgang auf FALSE gesetzt und bleibt in diesem sicheren Zustand. Der Fehlerzustand wird verlassen, wenn beide Taster gelöst werden (auf FALSE gesetzt werden).

Fehler- und Statuscodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Error B1	S_Button1 war TRUE bei Bausteinaktivierung Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
16#C002	Error B2	S_Button2 war TRUE bei Bausteinaktivierung Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
16#C003	Error B1&B2	Die Signale an S_Button1 und S_Button2 waren TRUE bei Bausteinaktivierung Ready = TRUE Error = TRUE S_TwoHandOut = FALSE

Tab. 16-187: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE Error = FALSE S_TwoHandOut = FALSE
16#8000	Buttons Actuated	Beide Taster korrekt aktiviert. Der sicherheitsgerichtete Ausgang ist aktiviert. Ready = TRUE Error = FALSE S_TwoHandOut = TRUE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8001	Init	Baustein ist aktiv, aber im Grundzustand. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8004	Buttons Released	Kein Taster ist betätigt. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8005	Button 1 Actuated	Nur Taster 1 ist betätigt. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8006	Button 2 Actuated	Nur Button2 ist betätigt. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8007	Button 2 Released	Der sicherheitsgerichtete Ausgang war aktiviert und wurde wieder deaktiviert. FALSE an beiden Eingängen S_Button1 und S_Button2 wurde nach Deaktivierung des sicherheitsgerichteten Ausgangs nicht erreicht. In diesem Zustand sind S_Button1= TRUE und S-Button2 = FALSE nach Deaktivierung des sicherheitsgerichteten Ausgangs. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8008	Button 1 Released	Der sicherheitsgerichtete Ausgang war aktiviert und wurde wieder deaktiviert. FALSE an beiden Eingängen S_Button1 und S_Button2 wurde nach Deaktivierung des sicherheitsgerichteten Ausgangs nicht erreicht. In diesem Zustand sind S_Button1 = FALSE und S_Button2 = TRUE nach Deaktivierung des sicherheitsgerichteten Ausgangs. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8009	LockedOff	Der sicherheitsgerichtete Ausgang war aktiviert und wurde wieder deaktiviert. FALSE an beiden Eingängen S_Button1 und S_Button2 wurde nach Deaktivierung des sicherheitsgerichteten Ausgangs nicht erreicht. In diesem Zustand sind S_Button1 = TRUE und S_Button2 = TRUE nach Deaktivierung des sicherheitsgerichteten Ausgangs. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8019	LockedOn	Inkorrekte Betätigung der Taster. Warten auf Lösen beider Taster. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE

Tab. 16-188: Bausteinspezifische Statuscodes (kein Fehler)

16.4.18 SF_TwoHandControlTypeIII

Kurzbeschreibung

SF_TwoHandControlTypeIII ist ein zertifizierter PLCopen-Baustein. Detaillierte Informationen zu angewendeten Normen findet der Anwender bei PLCopen - Technical Committee 5 - Safety Software.

Dieser Baustein unterstützt in einer Applikation die Funktion "Zweihandschaltung Typ 3" gemäß der europäischen Norm.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopen	1.0.0.0

Tab. 16-189: Bibliothekszuordnung

Schnittstellenbeschreibung

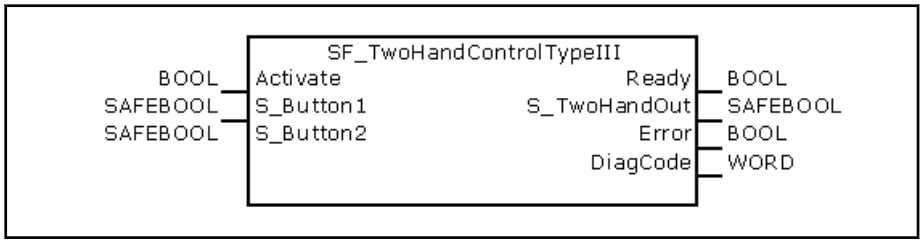


Abb. 16-96: Funktionsbaustein SF_TwoHandControlTypeIII

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	"Allgemeine Eingangsparameter" auf Seite 294
	S_Button1	SAFEBOOL	Variable Eingang für Taster 1 (für Kategorie 3 oder 4: zwei antivalente Kontakte) FALSE: Taster 1 ist gelöst TRUE: Taster 1 ist betätigt

E/A-TYP	Name	Datentyp	Beschreibung
	S_Button2	SAFEBOOL	Variable Eingang von Taster 2 (für Kategorie 3 oder 4: zwei antivalente Kontakte) FALSE: Taster 2 ist gelöst TRUE: Taster 2 ist betätigt
VAR_OUTPUT	Ready	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	S_TwoHandOut	SAFEBOOL	Sicherheitsgerichtetes Ausgangssignal FALSE: Keine korrekte zwei-Hand-Bedienung TRUE: Die Eingänge S_Button1 und S_Button2 wechselten innerhalb von 500 ms von FALSE auf TRUE und kein Fehler trat auf Zweihandbedienung wurde korrekt durchgeführt
	Error	BOOL	"Allgemeine Ausgangsparameter" auf Seite 296
	DiagCode	WORD	"Diagnose-Codes" auf Seite 296

Tab. 16-190: Schnittstelle SF_TwoHandControlTypeIII

Name	Typ	Default-Wert	Übernahme
Activate	BOOL	FALSE	TRUE
S_Button1	SAFEBOOL	FALSE	
S_Button2	SAFEBOOL	FALSE	

Tab. 16-191: Eingangsverhalten SF_TwoHandControlTypeIII

☞ Es ist kein Reset-Eingang oder Fehlerausgang erforderlich, da an beiden Schaltern kein Test durchgeführt werden kann.

Funktionsbeschreibung

Dieser Baustein unterstützt die Funktion "Zweihandschaltung Typ3" entsprechend der europäischen Norm. Wenn S_Button1 und S_Button2 innerhalb von 500 ms und richtiger Abfolge auf TRUE gesteuert werden, dann wird auch der S_TwoHandOut-Ausgang auf TRUE gesetzt. Ebenso kontrolliert der Baustein das Lösen beider Taster, bevor der Ausgang S_TwoHandOut erneut auf TRUE gesetzt wird.

Statusdiagramm

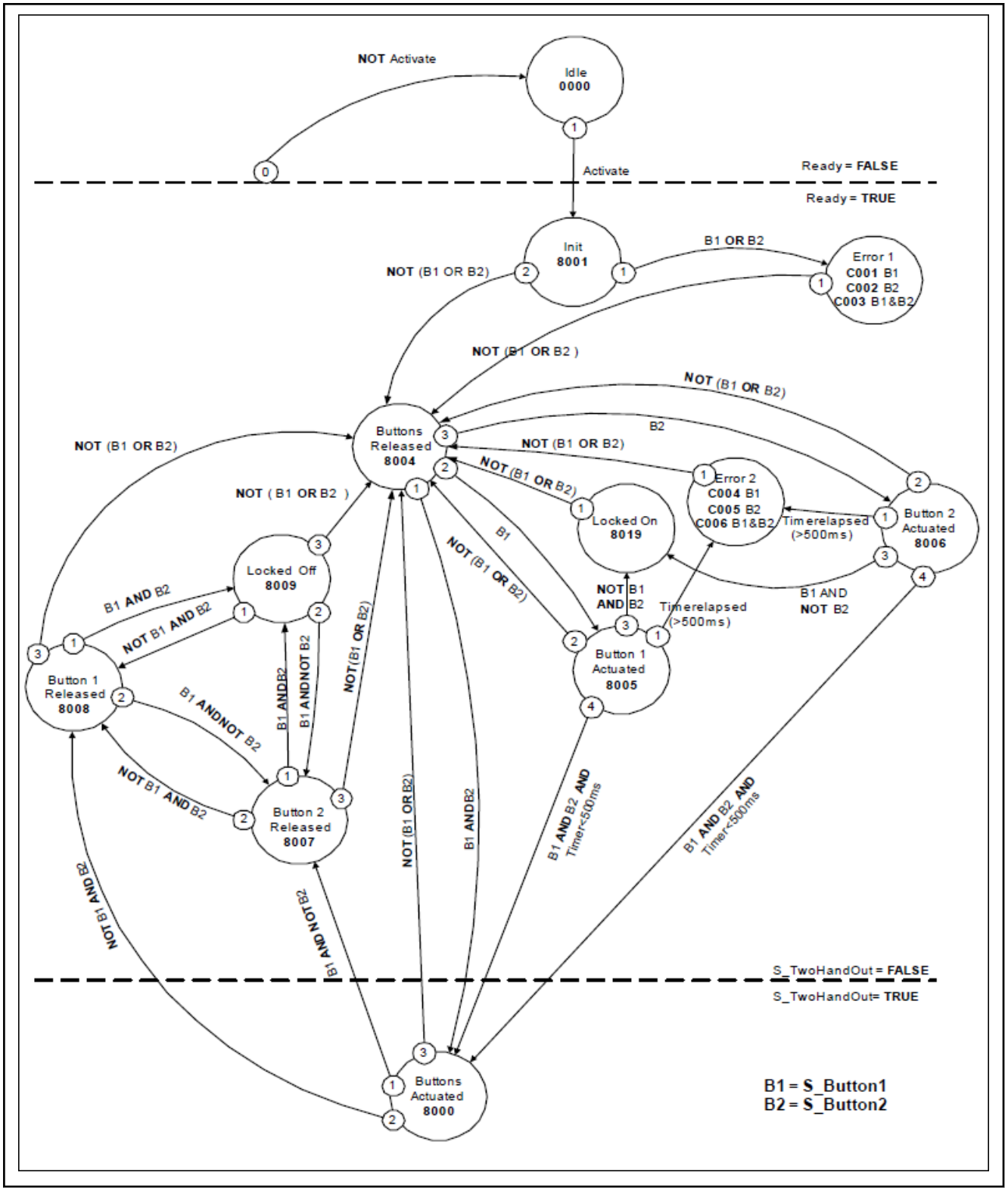


Abb. 16-97: Statusdiagramm SF_TwoHandControlTypeIII

Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt.

In jedem Fall haben diese Übergänge die höchste Priorität.

Typisches Zeitdiagramm

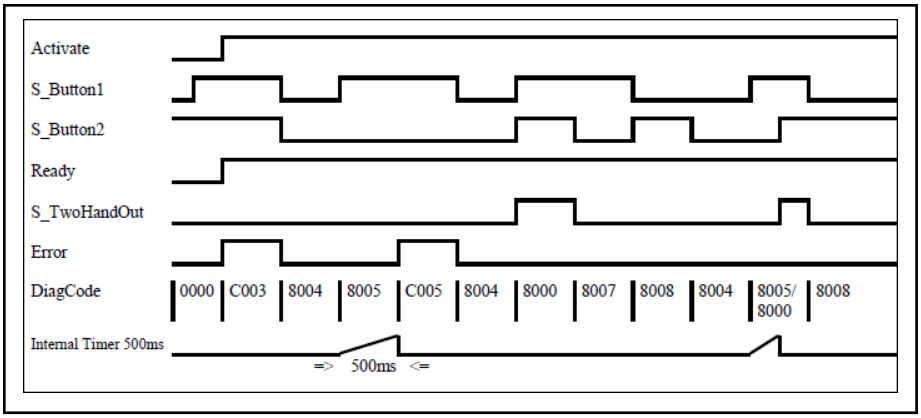


Abb. 16-98: Zeitdiagramm SF_TwoHandControlTypeIII

Fehlererkennung

Nach Aktivierung des Bausteins wird jeder Taster auf TRUE als ungültiges Eingangssignal erkannt, das zu einem Fehler führt. Der Baustein entdeckt, wenn die beiden Eingangssignale eine Divergenz von mehr als 500 ms aufweisen.

Fehlerverhalten

Bei einem Fehlerereignis wird der S_TwoHandOut-Ausgang auf FALSE gesetzt und bleibt in diesem sicheren Zustand. Der Fehlerzustand wird verlassen, wenn beide Taster gelöst werden (auf FALSE gesetzt werden).

Fehler- und Statuscodes

Der Übergang von jedem Zustand zum Zustand **Idle**, bedingt durch Activate = FALSE, wird nicht gezeigt. In jedem Fall haben diese Übergänge die höchste Priorität.

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C001	Error 1 B1	S_Button1 war TRUE bei Bausteinaktivierung Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
16#C002	Error 1 B2	S_Button2 war TRUE bei Bausteinaktivierung Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
16#C003	Error 1 B1&B2	Die Signale an S_Button1 und S_Button2 waren TRUE bei Bausteinaktivierung Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
16#C004	Error 2 B1	S_Button1 war FALSE und S_Button2 war TRUE nach 500 ms in Status 16#8005 Ready = TRUE Error = TRUE S_TwoHandOut = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#C005	Error 2 B2	S_Button1 war TRUE und S_Button2 war FALSE nach 500 ms in Status 16#8005 Ready = TRUE Error = TRUE S_TwoHandOut = FALSE
16#C006	Error 2 B1& B2	S_Button1 war TRUE und S_Button2 war TRUE nach 500 ms in Status 16#8005 oder 16#8006. Dieser Status ist nur möglich, wenn die Zustände der Eingänge (S_Button1 und S_Button2) von divergent nach konvergent (beide TRUE) simultan wechseln, wenn der Timer im gleichen Zyklus abläuft (500ms). Ready = TRUE Error = TRUE S_TwoHandOut = FALSE

Tab. 16-192: Bausteinspezifische Fehlercodes

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#0000	Idle	Der Baustein ist nicht aktiv (Grundzustand). Ready = FALSE Error = FALSE S_TwoHandOut = FALSE
16#8000	Buttons Actuated	Beide Taster korrekt aktiviert. Der sicherheitsgerichtete Ausgang ist aktiviert. Ready = TRUE Error = FALSE S_TwoHandOut = TRUE
16#8001	Init	Baustein ist aktiv, aber im Grundzustand. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8004	Buttons Released	Kein Taster ist betätigt. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8005	Button 1 Actuated	Nur Taster1 ist betätigt. Start Monitoring Timer. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8006	Button 2 Actuated	Nur Taster2 ist betätigt. Start Monitoring Timer. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE

DiagCode	Statusname	Statusbeschreibung und Setzen des Ausgangs
16#8007	Button 2 Released	Der sicherheitsgerichtete Ausgang war aktiviert und wurde wieder deaktiviert. FALSE an beiden Eingängen S_Button1 und S_Button2 wurde nach Deaktivierung des sicherheitsgerichteten Ausgangs nicht erreicht. In diesem Zustand sind S_Button1= TRUE und S-Button2 = FALSE nach Deaktivierung des sicherheitsgerichteten Ausgangs. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8008	Button 1 Released	Der sicherheitsgerichtete Ausgang war aktiviert und wurde wieder deaktiviert. FALSE an beiden Eingängen S_Button1 und S_Button2 wurde nach Deaktivierung des sicherheitsgerichteten Ausgangs nicht erreicht. In diesem Zustand sind S_Button1 = FALSE und S_Button2 = TRUE nach Deaktivierung des sicherheitsgerichteten Ausgangs. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8009	Locked Off	Der sicherheitsgerichtete Ausgang war aktiviert und wurde wieder deaktiviert. FALSE an beiden Eingängen S_Button1 und S_Button2 wurde nach Deaktivierung des sicherheitsgerichteten Ausgangs nicht erreicht. In diesem Zustand sind S_Button1 = TRUE und S_Button2 = TRUE nach Deaktivierung des sicherheitsgerichteten Ausgangs. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE
16#8019	Locked On	Inkorrekte Betätigung der Taster. Warten auf Lösen beider Taster. Ready = TRUE Error = FALSE S_TwoHandOut = FALSE

Tab. 16-193: Bausteinspezifische Statuscodes (kein Fehler)

16.5 SafetyPLCopenPress.library

Die PLCopen-Bausteine (mit Präfix SF_) wurden von der Organisation PLCopen zusammen mit seinen Mitgliedern und externen Organisationen, die sich mit sicherheitsgerichteten Aspekten beschäftigen, definiert.

Diese Bausteine sind zertifiziert und reduzieren dadurch Zeit und Kosten bei der Entwicklung, Verifikation und Abnahme einer Sicherheitsapplikation. Ähnlich einer logischen Verdrahtung können die Bausteine durch logische Operationen miteinander verknüpft werden. Somit können wichtige Bestandteile einer Sicherheitsapplikation schnell und ohne großen Programmieraufwand erstellt werden.

Diese Bibliothek enthält Funktionsbausteine für die Umsetzung von Pressenanwendungen. Sie umfasst einen Teil der in der PLCopen Safety Part 4 definierten Funktionsbausteine.

Anwendungsgebiete und Funktionsbausteine von SafetyPLCopenPress

Anwendungsgebiet (Ordner in der Bibliothek)	
Funktionsbausteine	**Beschreibung**
SF_SingleValveMonitoring, Seite 425	Überwacht das Schaltverhalten von Hydraulikventilen
SF_SingleValveCycleMonitoring, Seite 437	Überwacht den zyklischen Signalwechsel von Hydraulikventilen
SF_DoubleValveMonitoring, Seite 453	Überwacht das Schaltverhalten eines Doppelventils
SF_ValveGroupControl, Seite 470	Überwacht eine Ventilgruppe
SF_CamMonitoring, Seite 487	Überwacht die Nockenfolge für den Pressenbetrieb
SF_CamshaftMonitor, Seite 500	Überwacht eine Nockenwelle
SF_CycleControl, Seite 508	Steuert und Überwacht den Betrieb der Presse
SF_PressControl, Seite 522	Überwacht die Presse und steuert das Freigabesignal

Tab. 16-194: SafetyPLCopenPress-Bibliothek

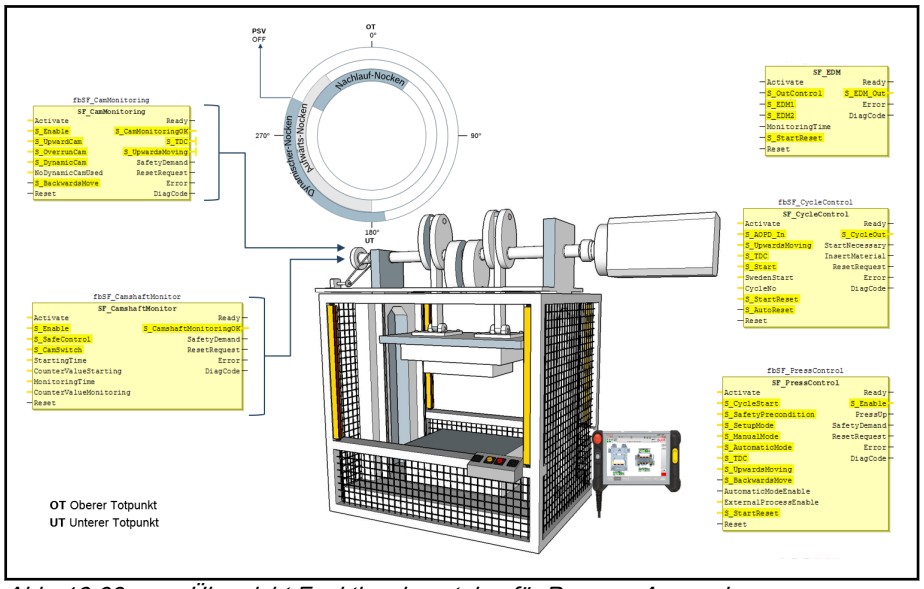


Abb. 16-99: Übersicht Funktionsbausteine für Pressen-Anwendungen

16.5.1 Standard Ein- und Ausgänge

Die nachfolgend beschriebenen Ein- und Ausgänge der PLCopen-Pressenfunktionsbausteine sind standardisiert und haben immer die gleiche Funktion.

Nicht jeder hier aufgelistete Ein- bzw. Ausgang ist an allen Pressen-Funktionsbausteinen vorhanden.

Eingänge

Name	Typ	Initialwert	Beschreibung
Activate	BOOL	FALSE	Aktivierung des Funktionsbausteins. Dieser Eingang kann mit einer Variable verbunden werden, die den Status des betreffenden Sicherheitsgeräts entspricht. Dies stellt sicher, dass keine unnötigen Diagnoseinformationen generiert werden, wenn das Gerät deaktiviert ist. Ist kein Gerät verbunden, muss der Eingang statisch auf TRUE gesetzt werden. Ist "Activate" FALSE, werden alle Ausgänge auf Ihre Initialwerte zurückgesetzt.
S_StartReset	SAFEBOOL	FALSE	Dieser Eingang dient zum einmaligen Quittieren eines Fehlers beim Start des Safety-Programms. Er sollte nur gesetzt werden, wenn beim Start des Safety-Programms keine Gefahr des unerwünschten Maschinenanlaufs besteht. FALSE: Manueller Reset über den "Reset"-Eingang beim Start der Safety-Programms. TRUE: Einmalige automatische Quittierung von Fehlern beim Start des Safety-Programms.
S_AutoReset	SAFEBOOL	FALSE	Automatische Quittierung von Fehlern. Dieser Eingang darf nur gesetzt werden, wenn die Fehlerdiagnose nicht benötigt wird und durch das Quittieren von Fehlern keine Gefahr des unerwünschten Maschinenwiederanlaufs besteht. Hierfür ist eine zusätzliche Validierung notwendig. FALSE: Manueller Reset über den "Reset"-Eingang TRUE: Automatischer Reset auftretender Fehler
Reset	BOOL	FALSE	Löscht einen am Funktionsbaustein aufgetretenen Fehler bei steigender Flanke. In Abhängigkeit von der Funktion kann dieser Eingang verschiedenen Zwecken dienen: • Reset der Zustandsmaschine des Funktionsbausteins wenn am Funktionsbaustein ein Fehler und ein Diagnose-Code ansteht. Dies ist ein Rücksetzen von Fehlern. • Manuelles Rücksetzen einer Wiederanlaufsperre durch den Bediener (siehe EN ISO 13849-1). Dies ist ein funktionaler Reset. • Weitere funktionsbausteinabhängige Rücksetzfunktionalität

Tab. 16-195: Standardisierte Eingänge

Ausgänge

Name	Typ	Initialwert	Beschreibung
Ready	BOOL	FALSE	Rückmeldung über die Aktivierung des Funktionsbausteins. TRUE: Der Funktionsbaustein hat seine Initialisierung durchlaufen und ist betriebsbereit. Alle Ausgänge sind gültig. FALSE: Der Funktionsbaustein ist inaktiv.
Error	BOOL	FALSE	Fehleranzeige des Funktionsbausteins. TRUE: Es ist ein Fehler bei der Abarbeitung aufgetreten. Der Ausgang "DiagCode" enthält weitere Diagnoseinformationen. FALSE: Fehlerfreie Bearbeitung

DiagCode	WORD	16#0000	Diagnoseinformation des Funktionsbausteins. Der Ausgang "DiagCode" enthält in der Regel den Zustand des Funktionsbausteins. Dieser entspricht der in der Dokumentation des Funktionsbausteins angegebenen Zustandsmaschine. Im Fehlerfall wird der erste aufgetretene Fehler ausgegeben. Dieser entspricht in der Regel auch einem Zustand in der Zustandsmaschine des Funktionsbausteins.
SafetyDemand	BOOL	FALSE	Sicherheitsanforderung durch den Funktionsbaustein. TRUE: Für die aktuelle Operation müssen Sicherheitsanforderungen erfüllt sein. Anhand der mit dem Funktionsbaustein verbundenen Eingänge werden die Sicherheitsanforderungen jedoch nicht erfüllt. Die Sicherheitskette ist nicht geschlossen. Es liegt aber kein Fehler vor. FALSE: Keine Sicherheitsanforderung
ResetRequest	BOOL	FALSE	Anforderung für steigende Flanke am "Reset"-Eingang TRUE: Die Fehlerursache ist behoben. Für den Wiederanlauf muss der letzte Fehler quittiert werden. FALSE: Es muss kein Fehler quittiert werden.

Tab. 16-196: Standardisierte Ausgänge

16.5.2 SF_SingleValveMonitoring

Kurzbeschreibung

Der Funktionsbaustein SF_SingleValveMonitoring stellt Funktionalität zum Überwachen des Schaltverhaltens eines Hydraulikventils mit einem statischen Überwachungssignal der sicheren Kolbenposition bereit.

Er überwacht die korrekte Signalabfolge und gibt Informationen über den aktuellen Ventilzustand aus.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopenPress	2.8.0.0

Tab. 16-197: Bibliothekszuordnung

Schnittstellenbeschreibung

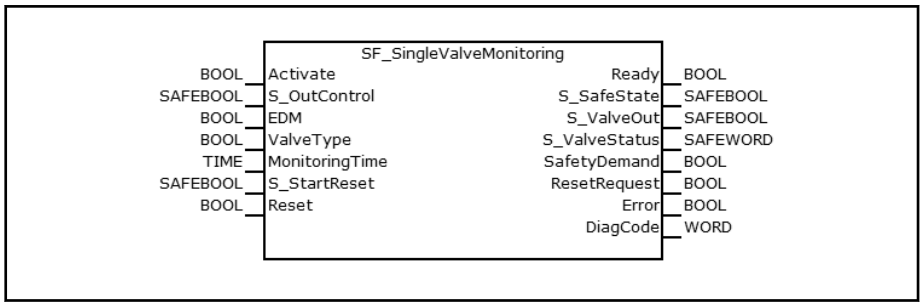


Abb. 16-100: Funktionsbaustein SF_SingleValveMonitoring

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv
	S_OutControl	SAFEBOOL	Anforderung sicherer Zustand FALSE: Sicherer Zustand nicht angefordert TRUE: Sicherer Zustand angefordert

E/A-Typ	Name	Datentyp	Beschreibung
	EDM	BOOL	Rückmeldesignal des Ventils
	ValveType	BOOL	Polarität des EDM-Eingangs FALSE: Sicherer Zustand bei Low-Signal TRUE: Sicherer Zustand bei High-Signal
	MonitoringTime	TIME	Zeit in der das Rückmeldesignal den angeforderten Schaltzustand bestätigen muss
	S_StartReset	SAFEBOOL	Automatisches Rücksetzen eines Anlauffehlers FALSE: Kein automatischer Kaltstart TRUE: Automatischer Kaltstart
	Reset	BOOL	Rücksetzen eines Fehlers bei steigender Flanke
VAR_OUTPUT	Ready	BOOL	Funktionsbaustein ist initialisiert und betriebsbereit
	S_SafeState	SAFEBOOL	Ausgang für die Überwachung des sicheren Zustands FALSE: Das Ventil befindet sich nicht im sicheren Zustand oder es ist ein Fehler aufgetreten. TRUE: Das Ventil befindet sich im sicheren Zustand und es ist kein Fehler aufgetreten.
	S_ValveOut	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveStatus	SAFEWORD	Ausgang für die Weiterverschaltung zum Funktionsbaustein Kap. 16.5.5 "SF_ValveGroupControl" auf Seite 470 um mehrere Ventile als Ventilgruppe anzusteuern, wobei alle anderen Ventile in den sicheren Zustand wechseln müssen, wenn ein Ventil der Ventilgruppe versagt.
	SafetyDemand	BOOL	Sicherheitsanforderung. Die für den sicheren Betrieb benötigten Voraussetzungen sind nicht erfüllt. Es liegt jedoch kein Fehler vor. TRUE: Sicherheitsanforderung ist aktiv FALSE: Es wird keine Absicherung benötigt
	ResetRequest	BOOL	Es wurde ein quittierbarer Fehler festgestellt. Vor dem Weiterbetrieb muss dieser mit einer steigenden Flanke am "Reset"-Eingang quittiert werden. TRUE: Reset angefordert FALSE: Reset nicht angefordert
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Werte des Ausgangs DiagCode, Seite 433)

Tab. 16-198: Schnittstelle SF_SingleValveMonitoring

Name	Typ	Min	Max	Default-Wert	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_OutControl	SAFEBOOL			FALSE	kontinuierlich

Name	Typ	Min	Max	Default-Wert	Übernahme
EDM	BOOL			FALSE	kontinuierlich
ValveType	BOOL			FALSE	bei steigender Flanke an "Activate"
MonitoringTime	TIME			FALSE	bei steigender Flanke an "Activate"
S_StartReset	SAFEBOOL			FALSE	bei steigender Flanke an "Activate"
Reset	BOOL			FALSE	kontinuierlich

Tab. 16-199: Eingangsverhalten SF_SingleValveMonitoring

Normativer Verweis

Norm	Anforderung
DIN EN ISO 13849-1:2016-6	5.2.1 Sicherheitsbezogene Stoppfunktion: sicherheitsbezogene Stoppfunktion muss die Maschine in den sicheren Zustand bringen 6.2 Spezifikation der Kategorien: Fehlererkennung (des Aktor, z.B. Drahtbruch)
DIN EN ISO 12100: 2011-03	6.2.11.4 Wiederanlauf nach Energieausfall 6.2.11.6 Verwendung von Selbstüberwachung
DIN EN ISO 4413: 2011-04	5.4.7.1 Unbeabsichtigte Bewegungen Steuerungen müssen so ausgelegt sein, dass unbeabsichtigte gefährdende Bewegungen und eine unzulässige Funktionsfolge der Aktuatoren verhindert sind. Dies gilt für alle Phasen des Betriebsablaufes. 5.4.8.1 Allgemeine Anforderungen Zur Fehlerbeseitigung und vorbeugenden Instandhaltung sollten Möglichkeiten für Fehlersuche und Zustands-überwachung vorgesehen sein. ...
DIN EN ISO 4414: 2011-04	5.4.6.1 Unbeabsichtigte Bewegung Steuerungen müssen so ausgelegt sein, dass für alle Phasen des Betriebsablaufes unbeabsichtigte gefährdende Bewegungen und eine unzulässige Funktionsfolge der Antriebe, insbesondere bei senkrechter und schräger Bewegungsrichtung, verhindert werden.

Norm	Anforderung
DIN EN 692:2009-10	5.4.1.4 ... Tritt ein Fehler in den sicherheitsrelevanten Teilen dieser Schutzeinrichtungen oder in der Steuerung auf, dann: a) darf ein unbeabsichtigter Anlauf nicht möglich sein; b) muss die sichere Funktion der Schutzeinrichtung aufrechterhalten bleiben; c) muss es möglich sein, die Maschine während der gefahrbringenden Bewegung anzuhalten; d) muss die Steuerung die Maschine während der gefährlichen Phase des Schließhubes sofort anhalten oder, in anderen Fällen, spätestens am Ende des Arbeitszyklus; e) muss die Steuerung jede Einleitung des folgenden Arbeitszyklus so lange verhindern, bis der Fehler behoben ist. 5.4.1.6 Die redundante und selbstüberwachte Pressensteuerung muss in zwei separaten Funktionssystemen arbeiten. Jedes System muss unabhängig in der Lage sein, die Gefahr bringende Bewegung anzuhalten, ohne Rücksicht auf den Zustand des anderen Systems. Das Versagen jedes der beiden Systeme muss durch Selbstüberwachung erkannt und ein weiterer Schließhub verhindert werden. Wenn das Versagen des einen Systems sich selbst anzeigt, d. h., wenn der Funktionsverlust selbst den nächsten Arbeitszyklus verhindert, ist eine weitere Überwachung dieses Systems nicht erforderlich. 5.4.2.3 ... d) sofern bei der Ventilüberwachung Sensoren zum Erkennen des Schaltzustandes der Ventile erforderlich sind, müssen diese Sensoren integrierte Bestandteile der Ventile sein. Das Doppelventil kann in sich selbst selbstüberwachend gestaltet sein, so dass das Ventilversagen selbstanzeigend ist;

Norm	Anforderung
DIN EN 693:2011-11	5.4.1.3 ... Tritt ein Fehler in den sicherheitsbezogenen Teilen dieser Schutzeinrichtungen oder in der Steuerung auf, dann a) darf ein unbeabsichtigter Anlauf nicht möglich sein; b) muss die sichere Funktion der Schutzeinrichtung aufrechterhalten bleiben; c) muss es möglich sein, die Maschine während der Gefahr bringenden Bewegung stillzusetzen; d) muss die Steuerung die Maschine während der Gefahr bringenden Phase des Schließhubes sofort stillsetzen oder, in anderen Fällen*, spätestens am Ende des Arbeitszyklus; e) muss die Steuerung jede Einleitung des folgenden Arbeitszyklus so lange verhindern, bis der Fehler behoben ist. 5.4.1.5 ... Die redundante und selbstüberwachte Pressensteuerung muss in zwei separaten Funktions-systemen arbeiten. Jedes System muss unabhängig in der Lage sein, die Gefahr bringende Bewegung anzuhalten, ohne Rücksicht auf den Zustand des anderen Systems. Das Versagen jedes der beiden Systeme muss durch Selbstüberwachung erkannt und ein weiterer Schließhub verhindert werden. Wenn das Versagen des einen Systems sich selbst anzeigt, d. h., wenn der Funktionsverlust selbst den nächsten Arbeitszyklus verhindert, ist eine weitere Überwachung dieses Systems nicht erforderlich.
DIN EN 12622:2014-02	5.2.1 ... ... Ein Ausfall von jedem einzelnen System muss durch Überwachung registriert und ein weiterer Schließhub verhindert werden. 5.2.5 ... Tabelle 2 - Sicherheit und Zuverlässigkeit des Steuerungssystems für den Einzelhubbetrieb mit manueller Zuführung und/oder Entnahme oder Einrichtbetrieb 5.2.5.2 Verhalten des Steuerungssystems im Falle eines Fehlers ... Kommt es in sicherheitsbezogenen Teilen der Schutzeinrichtungen oder dem Steuerungssystem zu einem Fehler dann: a) darf ein unerwarteter Anlauf nicht möglich sein; b) die sichere Funktionsweise der Schutzeinrichtungen muss nach dem ersten Fehler aufrecht erhalten bleiben; c) es muss weiterhin möglich sein, die Gesenkbiegepresse während einer gefährlichen Bewegung anzuhalten; d) das Steuerungssystem muss die Gesenkbiegepresse während einer gefährlichen Phase des Schließvorgangs innerhalb der vom Hersteller angegebenen Anhaltezeit stoppen oder, in anderen Fällen, bei der nächsten Aufforderung an die Sicherheitsfunktion oder aber spätestens zum Ende des Arbeitszyklus; e) das Steuerungssystem muss jede Einleitung des nächsten Produktionsbetriebszyklus verhindern bis, der Fehler behoben ist. 5.2.6.1 Anlauf ... Der unerwartete Anlauf von gefährlichen Bewegungen, wie z. B. des Pressbalkens, der hinteren Anschläge, der Werkstückauflage muss verhindert werden.

Funktionsbeschreibung Der Funktionsbaustein SF_SingleValveMonitoring überwacht das Schaltverhalten eines Hydraulikventils, das ein statisches Überwachungssignal der Schieberposition zur Verfügung stellt. Der Funktionsbaustein überwacht den Initialzustand des Ventils mittels eines Rückmeldesignals ("EDM"). Die Rückmeldung muss den sicheren Zustand signalisieren, bevor das Ventil durch

den Funktionsbaustein aktiviert werden kann. Nach der Aktivierung des Ventils wird das Rückmeldesignal weiterhin überwacht. Stimmt das Rückmeldesignal ("EDM") nach Ablauf der Überwachungszeit ("MonitoringTime") nicht mit der Erwartungshaltung überein, wird der Ausgang "S_ValveOut" FALSE. Dieser Zustand wird ebenfalls durch den "Error"-Ausgang angezeigt und eine entsprechende Diagnosemeldung erzeugt.

Activate, Ready

Ist "Activate" FALSE, so ist der Funktionsbaustein inaktiv. Es werden keine Eingänge verarbeitet. Alle Ausgänge sind FALSE. Der Ausgang "DiagCode" ist 16#0000.

Wenn "Activate" auf TRUE geht, wird der Funktionsbaustein initialisiert. Der Ausgang "Ready" wird TRUE. Eine fallende Flanke am Eingang "Activate" versetzt den Baustein wieder in den inaktiven Zustand.

S_StartReset, Reset, Error, DiagCode, ResetRequest

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx, wobei xxx den Fehler-Code repräsentiert.

Ein Fehler wird mit einer steigenden Flanke am Eingang "Reset" zurückgesetzt. Hierfür muss sich das Ventil in der sicheren Position befinden. Dies wird durch das entsprechende "EDM"-Signal an den Funktionsbaustein gemeldet. Ein rücksetzbarer Fehler wird über ein TRUE am Ausgang "ResetRequest" angezeigt. Ist der Fehler behoben, wechselt der Ausgang "Error" bei steigender Flanke am Eingang "Reset" auf FALSE, und der Funktionsbaustein wartet auf den nächsten Zyklus.

Der Anlauffehler beim ersten Start des Funktionsbausteins kann über den Eingang "S_StartReset" automatisch quittiert werden. Der Eingang "S_StartReset" ist nur im Zustand C100 aktiv.



Der Eingang "S_StartReset" darf nur aktiviert werden, wenn daraus keine gefahrbringende Situation entstehen kann.

EDM, ValveType

Da die auf dem Markt verfügbaren Ventile die sichere Schieberposition entweder mit HIGH oder LOW signalisieren, kann die Polarität des Rückmeldesignals "EDM" durch den Eingang "ValveType" festgelegt werden. Bei der Auswahl der Ventile sollte jedoch das Sicherheitsprinzip angewendet werden, nachdem der sichere Zustand durch ein HIGH-Signal bestätigt wird.

Das "EDM"-Signal muss je nach für die Applikation benötigtem Diagnosedeckungsgrad (DC) ggf. direkt über einen sicheren Eingang mit der Sicherheitssteuerung verbunden werden.

S_SafeState

Der Funktionsbaustein signalisiert den sicheren Zustand am Ausgang "S_SafeState" wenn die sichere Position des Ventilschiebers erkannt wurde und kein Fehler ansteht. Dieser Ausgang kann für eine Wiederanlaufsperre der Maschine (z.B. dem Start eines neuen Zyklus), wenn entweder das Ventil nicht im sicheren Zustand ist oder ein Fehler erkannt wurde, verwendet werden. Dies kann z.B. ein Unterschied zwischen Rückmeldesignal und erwarteten Zustand sein. Der SIL-Level des Ausgangs "S_SafeState" hängt von den charakteristischen Daten ($MTTF_d$) der Rückmeldesensoren und ihrer Integration in die Sicherheitssteuerung (z.B. DC) ab.

S_ValveStatus

Das Ausgangssignal "S_ValveStatus" kann zusammen mit dem Funktionsbaustein Kap. 16.5.5 "SF_ValveGroupControl" auf Seite 470 verwendet werden, um mehrere Ventile als Ventilgruppe anzusteuern. Hierbei werden alle Ventile einer Ventilgruppe in den sicheren Zustand versetzt, wenn ein Ventil der Ventilgruppe ausfällt. Dies ist z.B. der Fall, wenn der rückgemeldete Zustand nicht der Erwartungshaltung entspricht.

Zustandsdiagramm

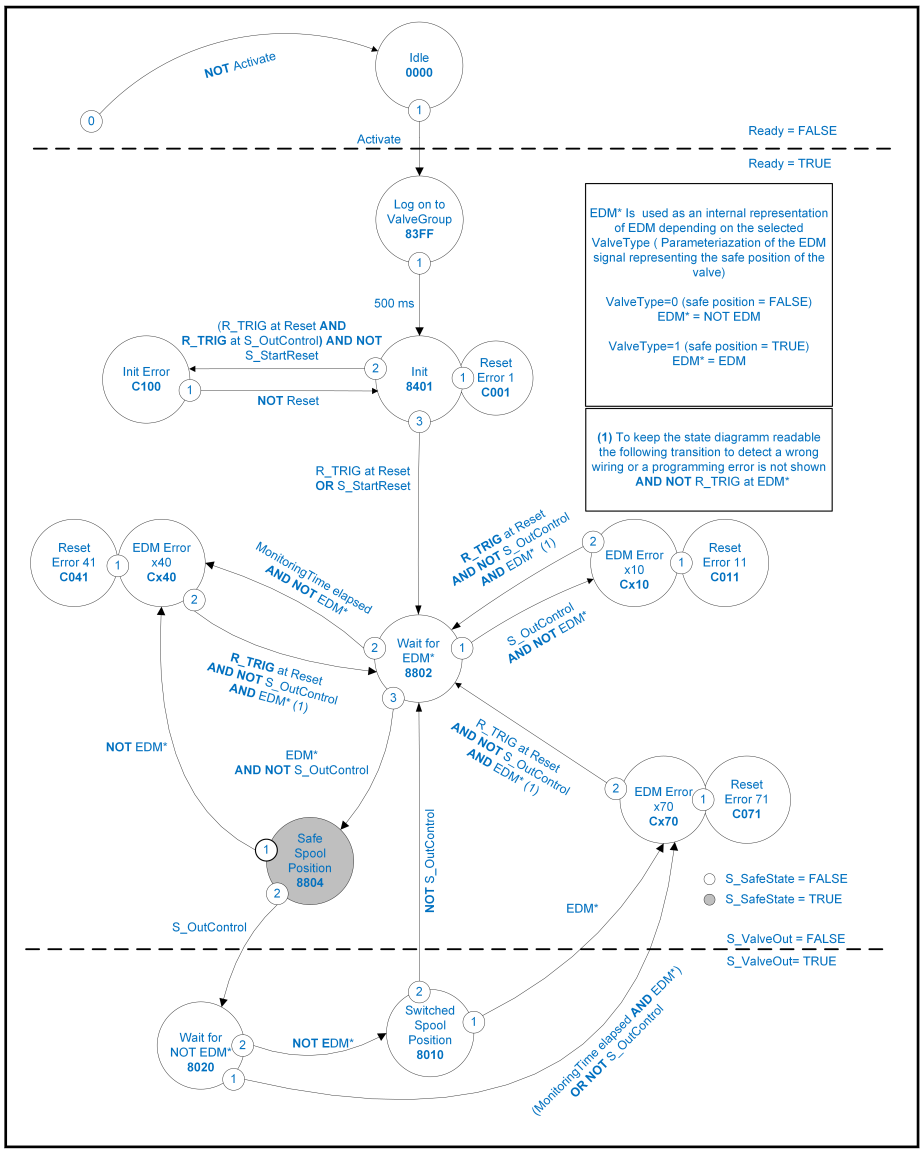
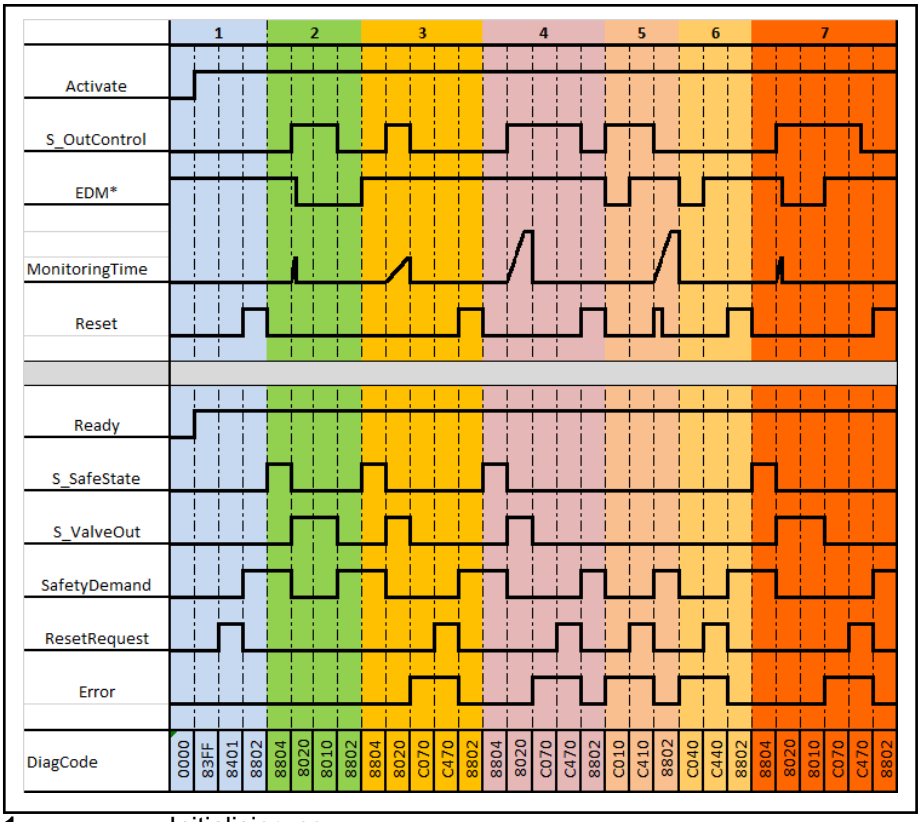


Abb. 16-101: Zustandsdiagramm SF_SingleValveMonitoring

Zeitdiagramme



1 Initialisierung
2 Normales Ein- und Ausschalten
3 Einschalten, aber "S_OutControl" schaltet aus, bevor das "EDM"-Signal FALSE ist.
4 Einschalten, aber das "EDM"-Signal ändert sich vor Ablauf der Überwachungszeit ("MonitoringTime") nicht.
5 "S_OutControl" ist TRUE, aber das" EDM"-Signal ist FALSE
6 Die Überwachungszeit ("MonitoringTime") läuft ab, bevor "EDM" TRUE wird.
7 Normales Ventilschaltspiel, aber das "EDM"-Signal ändert sich während "S_OutControl" noch TRUE ist.

Abb. 16-102: Zeitdiagramm für SF_SingleValveMonitoring

Fehlererkennung

Die folgenden Bedingungen führen zum Fehlerzustand:

- statisches "Reset"-Signal während des Prozesses
- ungültiges "EDM"-Signal während es Prozesses
- "S_OutControl" und "Reset" sind aufgrund eines Programmierfehlers falsch verbunden.
- "S_OutControl" ändert den Zustand, bevor das "EDM"-Signal die kommandierte Position bestätigt.

Fehlerverhalten

Im Fehlerzustand verhalten sich die Ausgänge wie folgt:

- Im Fehlerfall werden "S_ValveOut" und "S_SafeState" auf FALSE gesetzt und bleiben FALSE bis der Fehler behoben und der Wiederanlauf über eine steigende Flanke an "Reset" freigegeben wurde.
- Ein "EDM"-Fehler muss immer über eine steigende Flanke am "Reset"-Eingang zurückgesetzt werden.

- Ein "Reset"-Fehler kann durch rücksetzen des Eingangs "Reset" behoben werden.
- Nach der Funktionsbausteinaktivierung kann die optionale Anlaufsperre durch eine steigende Flanke an "Reset" freigegeben werden.

DiagCode

DiagCode	Zustandsname	Beschreibung
16#0000	Idle	Der Baustein ist nicht initialisiert. Dieser Wert wird auch am Ausgang DiagCode ausgegeben, wenn "Activate" auf FALSE geht und der Zyklus abgelaufen ist. Ready = FALSE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 0000 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8010	Switched Spool Position	Warte auf die Bestätigung des sicheren Ventilzustands durch das EDM-Signal. Der Zähler startet beim Eintritt in den Zustand. Ready = TRUE S_SafeState = FALSE S_ValveOut = TRUE S_ValveStatus = 8010 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8020	Wait for NOT EDM*	Das EDM-Signal bestätigt die die Ansteuerung des Ventils. Ready = TRUE S_SafeState = FALSE S_ValveOut = TRUE S_ValveStatus = 8020 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#83FF	Log on to valve group	Wartezustand um sicherzustellen, dass der Funktionsbaustein SF_ValveGroupControl nur zusammen mit den Ventilüberwachungsbausteinen genutzt werden kann. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 83FF SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#8401	Init	Blockaktivierungsstart wird verhindert. Reset benötigt. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 8401 SafetyDemand = FALSE ResetRequest = TRUE Error = FALSE
16#8802	Wait for EDM	Warte bis das "EDM"-Signal die sichere Schieberposition bestätigt. Der Timer startet bei Zustandseintritt. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 8802 SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8804	Safe spool position	Das EDM-Signal bestätigt den sicheren Zustand des Ventils (sichere Schieberposition) Ready = TRUE S_SafeState = TRUE S_ValveOut = FALSE S_ValveStatus = 8804 SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#C001	Reset Error 1	statischer Reset im Zustand 16#8001 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C001 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C011	Reset Error 11	statischer Reset im Zustand 16#Cx10 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C011 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C041	Reset Error 41	statischer Reset im Zustand 16#Cx40 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C041 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C071	Reset Error 71	statischer Reset im Zustand 16#Cx70 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C071 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C100	Init Error	Gleiche Signale an S_OutControl und Reset (Flanke im selben Zyklus) erkannt. Dies ist möglicherweise ein Programmierfehler. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C100 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx10	EDM Error x10	Das Signal EDM* ist im Startzustand des Aktors nicht korrekt. Im Zustand 16#8802 muss das EDM*-Signal FALSE sein, wenn "S_OutControl" aktiviert wird. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM* = FALSE oder S_OutControl = TRUE S_ValveStatus = C010 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C410 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx40	EDM Error x40	Das Signal EDM* ist im Zustand 16#8802 oder 16#8804 nicht korrekt. Im Zustand 16#8802 ist das EDM*-Signal FALSE und die "MonitoringTime" ist abgelaufen. Im Zustand 16#8804 wird das EDM*-Signal FALSE obwohl das Ventil nicht geschaltet wurde. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM* = FALSE oder S_OutControl = TRUE S_ValveStatus = C040 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C440 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#Cx70	EDM Error x70	Das Signal EDM* ist im Zustand 16#8010 oder 16#8020 nicht korrekt. Im Zustand 16#8020 ist das EDM*-Signal TRUE und die "MonitoringTime" ist abgelaufen oder S_OutControl wurde FALSE bevor EDM* den Schaltvorgang bestätigt hat. Im Zustand 16#8010 wird das EDM*-Signal TRUE obwohl das Ventil nicht geschaltet wurde. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM* = FALSE oder S_OutControl = TRUE S_ValveStatus = C070 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C470 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

Tab. 16-200: Werte des Ausgangs "DiagCode"

16.5.3 SF_SingleValveCycleMonitoring

Kurzbeschreibung Der Funktionsbaustein SF_SingleValveCycleMonitoring stellt Funktionalität zum Überwachen des Schaltverhaltens eines Hydraulikventils bereit. Hierbei wird der Signalwechsel der Schieberüberwachung erkannt. Findet kein Signalwechsel innerhalb eines Zyklus statt, wird der nächste Zyklus unterdrückt.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopenPress	2.8.0.0

Tab. 16-201: Bibliothekszuordnung

Schnittstellenbeschreibung

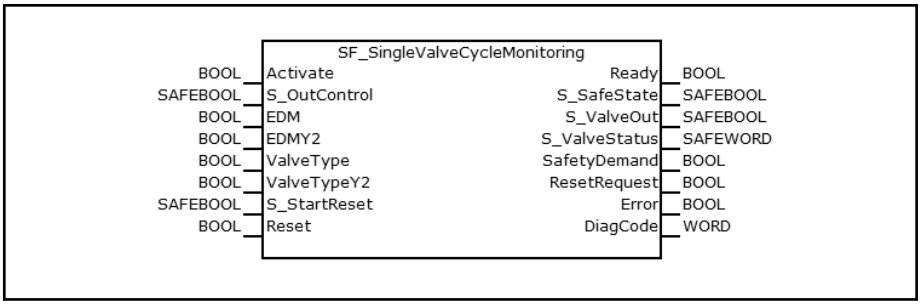


Abb. 16-103: Funktionsbaustein SF_SingleValveCycleMonitoring

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv
	S_OutControl	SAFEBOOL	Anforderung sicherer Zustand FALSE: Sicherer Zustand nicht angefordert TRUE: Sicherer Zustand angefordert
	EDM	BOOL	Rückmeldesignal 1 des Ventils
	EDMY2	BOOL	Rückmeldesignal 2 des Ventils
	ValveType	BOOL	Polarität des EDM-Eingangs FALSE: Sicherer Zustand bei LOW-Signal TRUE: Sicherer Zustand bei High-Signal
	ValveTypeY2	BOOL	Polarität des EDM-Eingangs Y2 FALSE: Sicherer Zustand bei LOW-Signal TRUE: Sicherer Zustand bei High-Signal
	S_StartReset	SAFEBOOL	Automatisches Rücksetzen eines Anlauffehlers FALSE: Kein automatischer Kaltstart TRUE: Automatischer Kaltstart
	Reset	BOOL	Rücksetzen eines Fehlers bei steigender Flanke
VAR_OUTPUT	Ready	BOOL	Funktionsbaustein ist initialisiert und betriebsbereit
	S_SafeState	SAFEBOOL	Ausgang für die Überwachung des sicheren Zustands FALSE: Das Ventil befindet sich nicht im sicheren Zustand oder es ist ein Fehler aufgetreten. TRUE: Das Ventil befindet sich im sicheren Zustand und es ist kein Fehler aufgetreten.
	S_ValveOut	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveStatus	SAFEWORD	Ausgang für die Weiterverschaltung zum Funktionsbaustein Kap. 16.5.5 "SF_ValveGroupControl" auf Seite 470 um mehrere Ventile als Ventilgruppe anzusteuern, wobei alle anderen Ventile in den sicheren Zustand wechseln müssen, wenn ein Ventil der Ventilgruppe versagt.

E/A-Typ	Name	Datentyp	Beschreibung
	SafetyDemand	BOOL	Sicherheitsanforderung. Die für den sicheren Betrieb benötigten Voraussetzungen sind nicht erfüllt. Es liegt jedoch kein Fehler vor. TRUE: Sicherheitsanforderung ist aktiv FALSE: Es wird keine Absicherung benötigt
	ResetRequest	BOOL	Es wurde ein quittierbarer Fehler festgestellt. Vor dem Weiterbetrieb muss dieser mit einer steigenden Flanke am "Reset"-Eingang quittiert werden. TRUE: Reset angefordert FALSE: Reset nicht angefordert
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Werte des Ausgangs DiagCode, Seite 448)

Tab. 16-202: Schnittstelle SF_SingleValveCycleMonitoring

Name	Typ	Min	Max	Default-Wert	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_OutControl	SAFEBOOL			FALSE	kontinuierlich
EDM	BOOL			FALSE	kontinuierlich
EDMY2	BOOL			FALSE	kontinuierlich
ValveType	BOOL			FALSE	bei steigender Flanke an "Activate"
ValveTypeY2	BOOL			FALSE	bei steigender Flanke an "Activate"
MonitoringTime	TIME			FALSE	bei steigender Flanke an "Activate"
S_StartReset	SAFEBOOL			FALSE	bei steigender Flanke an "Activate"
Reset	BOOL			FALSE	kontinuierlich

Tab. 16-203: Eingangsverhalten SF_SingleValveCycleMonitoring

Normativer Verweis

Norm	Anforderung
DIN EN ISO 13849-1:2016-6	5.2.1 Sicherheitsbezogene Stoppfunktion: sicherheitsbezogene Stoppfunktion muss die Maschine in den sicheren Zustand bringen 6.2 Spezifikation der Kategorien: Fehlererkennung (des Aktor, z.B. Drahtbruch)
DIN EN ISO 12100: 2011-03	6.2.11.4 Wiederanlauf nach Energieausfall 6.2.11.6 Verwendung von Selbstüberwachung

Norm	Anforderung
DIN EN ISO 4413: 2011-04	5.4.7.1 Unbeabsichtigte Bewegungen Steuerungen müssen so ausgelegt sein, dass unbeabsichtigte gefährdende Bewegungen und eine unzulässige Funktionsfolge der Aktuatoren verhindert sind. Dies gilt für alle Phasen des Betriebsablaufes. 5.4.8.1 Allgemeine Anforderungen Zur Fehlerbeseitigung und vorbeugenden Instandhaltung sollten Möglichkeiten für Fehlersuche und Zustands-überwachung vorgesehen sein. ...
DIN EN ISO 4414: 2011-04	5.4.6.1 Unbeabsichtigte Bewegung Steuerungen müssen so ausgelegt sein, dass für alle Phasen des Betriebsablaufes unbeabsichtigte gefährdende Bewegungen und eine unzulässige Funktionsfolge der Antriebe, insbesondere bei senkrechter und schräger Bewegungsrichtung, verhindert werden.
DIN EN 692:2009-10	5.4.1.4 ... Tritt ein Fehler in den sicherheitsrelevanten Teilen dieser Schutzeinrichtungen oder in der Steuerung auf, dann: a) darf ein unbeabsichtigter Anlauf nicht möglich sein; b) muss die sichere Funktion der Schutzeinrichtung aufrechterhalten bleiben; c) muss es möglich sein, die Maschine während der gefahrbringenden Bewegung anzuhalten; d) muss die Steuerung die Maschine während der gefährlichen Phase des Schließhubes sofort anhalten oder, in anderen Fällen, spätestens am Ende des Arbeitszyklus; e) muss die Steuerung jede Einleitung des folgenden Arbeitszyklus so lange verhindern, bis der Fehler behoben ist. 5.4.1.6 Die redundante und selbstüberwachte Pressensteuerung muss in zwei separaten Funktionssystemen arbeiten. Jedes System muss unabhängig in der Lage sein, die Gefahr bringende Bewegung anzuhalten, ohne Rücksicht auf den Zustand des anderen Systems. Das Versagen jedes der beiden Systeme muss durch Selbstüberwachung erkannt und ein weiterer Schließhub verhindert werden. Wenn das Versagen des einen Systems sich selbst anzeigt, d. h., wenn der Funktionsverlust selbst den nächsten Arbeitszyklus verhindert, ist eine weitere Überwachung dieses Systems nicht erforderlich. 5.4.2.3 ... d) sofern bei der Ventilüberwachung Sensoren zum Erkennen des Schaltzustandes der Ventile erforderlich sind, müssen diese Sensoren integrierte Bestandteile der Ventile sein. Das Doppelventil kann in sich selbst selbstüberwachend gestaltet sein, so dass das Ventilversagen selbstanzeigend ist;

Norm	Anforderung
DIN EN 693:2011-11	5.4.1.3 ... Tritt ein Fehler in den sicherheitsbezogenen Teilen dieser Schutzeinrichtungen oder in der Steuerung auf, dann a) darf ein unbeabsichtigter Anlauf nicht möglich sein; b) muss die sichere Funktion der Schutzeinrichtung aufrechterhalten bleiben; c) muss es möglich sein, die Maschine während der Gefahr bringenden Bewegung stillzusetzen; d) muss die Steuerung die Maschine während der Gefahr bringenden Phase des Schließhubes sofort stillsetzen oder, in anderen Fällen*, spätestens am Ende des Arbeitszyklus; e) muss die Steuerung jede Einleitung des folgenden Arbeitszyklus so lange verhindern, bis der Fehler behoben ist. 5.4.1.5 ... Die redundante und selbstüberwachte Pressensteuerung muss in zwei separaten Funktions-systemen arbeiten. Jedes System muss unabhängig in der Lage sein, die Gefahr bringende Bewegung anzuhalten, ohne Rücksicht auf den Zustand des anderen Systems. Das Versagen jedes der beiden Systeme muss durch Selbstüberwachung erkannt und ein weiterer Schließhub verhindert werden. Wenn das Versagen des einen Systems sich selbst anzeigt, d. h., wenn der Funktionsverlust selbst den nächsten Arbeitszyklus verhindert, ist eine weitere Überwachung dieses Systems nicht erforderlich.
DIN EN 12622:2014-02	5.2.1 ... ... Ein Ausfall von jedem einzelnen System muss durch Überwachung registriert und ein weiterer Schließhub verhindert werden. 5.2.5 ... Tabelle 2 - Sicherheit und Zuverlässigkeit des Steuerungssystems für den Einzelhubbetrieb mit manueller Zuführung und/oder Entnahme oder Einrichtbetrieb 5.2.5.2 Verhalten des Steuerungssystems im Falle eines Fehlers ... Kommt es in sicherheitsbezogenen Teilen der Schutzeinrichtungen oder dem Steuerungssystem zu einem Fehler dann: a) darf ein unerwarteter Anlauf nicht möglich sein; b) die sichere Funktionsweise der Schutzeinrichtungen muss nach dem ersten Fehler aufrecht erhalten bleiben; c) es muss weiterhin möglich sein, die Gesenkbiegepresse während einer gefährlichen Bewegung anzuhalten; d) das Steuerungssystem muss die Gesenkbiegepresse während einer gefährlichen Phase des Schließvorgangs innerhalb der vom Hersteller angegebenen Anhaltezeit stoppen oder, in anderen Fällen, bei der nächsten Aufforderung an die Sicherheitsfunktion oder aber spätestens zum Ende des Arbeitszyklus; e) das Steuerungssystem muss jede Einleitung des nächsten Produktionsbetriebszyklus verhindern bis, der Fehler behoben ist. 5.2.6.1 Anlauf ... Der unerwartete Anlauf von gefährlichen Bewegungen, wie z. B. des Pressbalkens, der hinteren Anschläge, der Werkstückauflage muss verhindert werden.

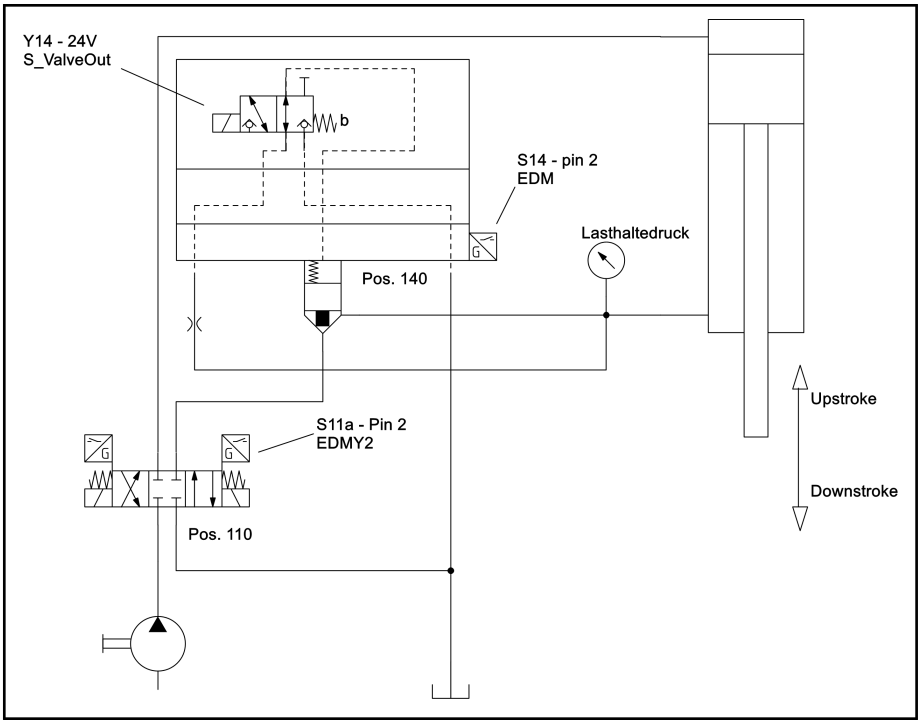


Abb. 16-104: Verschaltung des Funktionsbausteins SF_SingleValveCycleMonitoring am Beispiel des Rexroth Pressenmoduls IH04 Typ C

Funktionsbeschreibung

Der Funktionsbaustein SF_SingleValveCycleMonitoring überwacht das Schaltverhalten eines Hydraulikventils (Cartridge-Ventil). Er überwacht den Signalwechsel des Schieberüberwachungssignal ("EDM" darf nicht FALSE bleiben) während eines Maschinenzyklusses. Wird kein Signalwechsel innerhalb eines Maschinenzyklusses erkannt, so wird der nächste Zyklus verhindert.

In Pressen kann das Ventil für die Abwärtsbewegung verwendet werden. Bevor die Abwärtsbewegung kommandiert werden kann, muss das Ventil die Sichere Position anzeigen und das Aufwärtsventil geschlossen sein ("EDM2" = TRUE). Bevor die Aufwärtsbewegung erlaubt wird, muss ein Signalwechsel am Eingang "EDM" stattfinden. Der Funktionsbaustein überwacht auch den Eingang "EDMY2" auf dynamische Wechsel. Bleibt aus irgendeinem Grund "EDMY2" auf TRUE, verhindert der Funktionsbaustein die Aktivierung des Cartidge-Ventils. Deshalb können Standardeingänge für "EDMY2" verwendet werden.

Activate, Ready

Ist "Activate" FALSE, so ist der Funktionsbaustein inaktiv. Es werden keine Eingänge verarbeitet. Alle Ausgänge sind FALSE. Der Ausgang "DiagCode" ist 16#0000.

Wenn "Activate" auf TRUE geht, wird der Funktionsbaustein initialisiert. Der Ausgang "Ready" wird TRUE. Eine fallende Flanke am Eingang "Activate" versetzt den Baustein wieder in den inaktiven Zustand.

S_StartReset, Reset, Error, DiagCode, ResetRequest

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx, wobei xxx den Fehler-Code repräsentiert.

Ein Fehler wird mit einer steigenden Flanke am Eingang "Reset" zurückgesetzt. Hierfür muss sich das Ventil in der sicheren Position befinden. Dies wird durch das entsprechende "EDM"-Signal an den Funktionsbaustein ge-

meldet. Ein rücksetzbarer Fehler wird über ein TRUE am Ausgang "ResetRequest" angezeigt. Ist der Fehler behoben, wechselt der Ausgang "Error" bei steigender Flanke am Eingang "Reset" auf FALSE, und der Funktionsbaustein wartet auf den nächsten Zyklus.

Der Anlauffehler beim ersten Start des Funktionsbausteins kann über den Eingang "S_StartReset" automatisch quittiert werden. Der Eingang "S_StartReset" ist nur im Zustand C100 aktiv.

☞ Der Eingang "S_StartReset" darf nur aktiviert werden, wenn daraus keine gefahrbringende Situation entstehen kann.

EDM, EDMY2, ValveType, ValveTypeY2, ValveOut

Nach der Aktivierung des Ventils wird das Rückmeldesignal weiterhin überwacht. Stimmt das Rückmeldesignal ("EDM") nicht mit der Erwartungshaltung überein, wird der Ausgang "S_ValveOut" FALSE. Dieser Zustand wird ebenfalls durch den "Error"-Ausgang angezeigt und eine entsprechende Diagnosemeldung erzeugt.

Da die auf dem Markt verfügbaren Ventile die sichere Schieberposition entweder mit HIGH oder LOW signalisieren, kann die Polarität der Rückmeldesignale "EDM" und "EDMY2" durch den Eingänge "ValveType" bzw. "ValveTypeY2" festgelegt werden. Bei der Auswahl der Ventile sollte jedoch das Sicherheitsprinzip angewendet werden, nachdem der sichere Zustand durch ein HIGH-Signal bestätigt wird.

Das "EDM"-Signal muss je nach für die Applikation benötigtem Diagnosedeckungsgrad (DC) ggf. direkt über einen sicheren Eingang mit der Sicherheitssteuerung verbunden werden.

S_SafeState

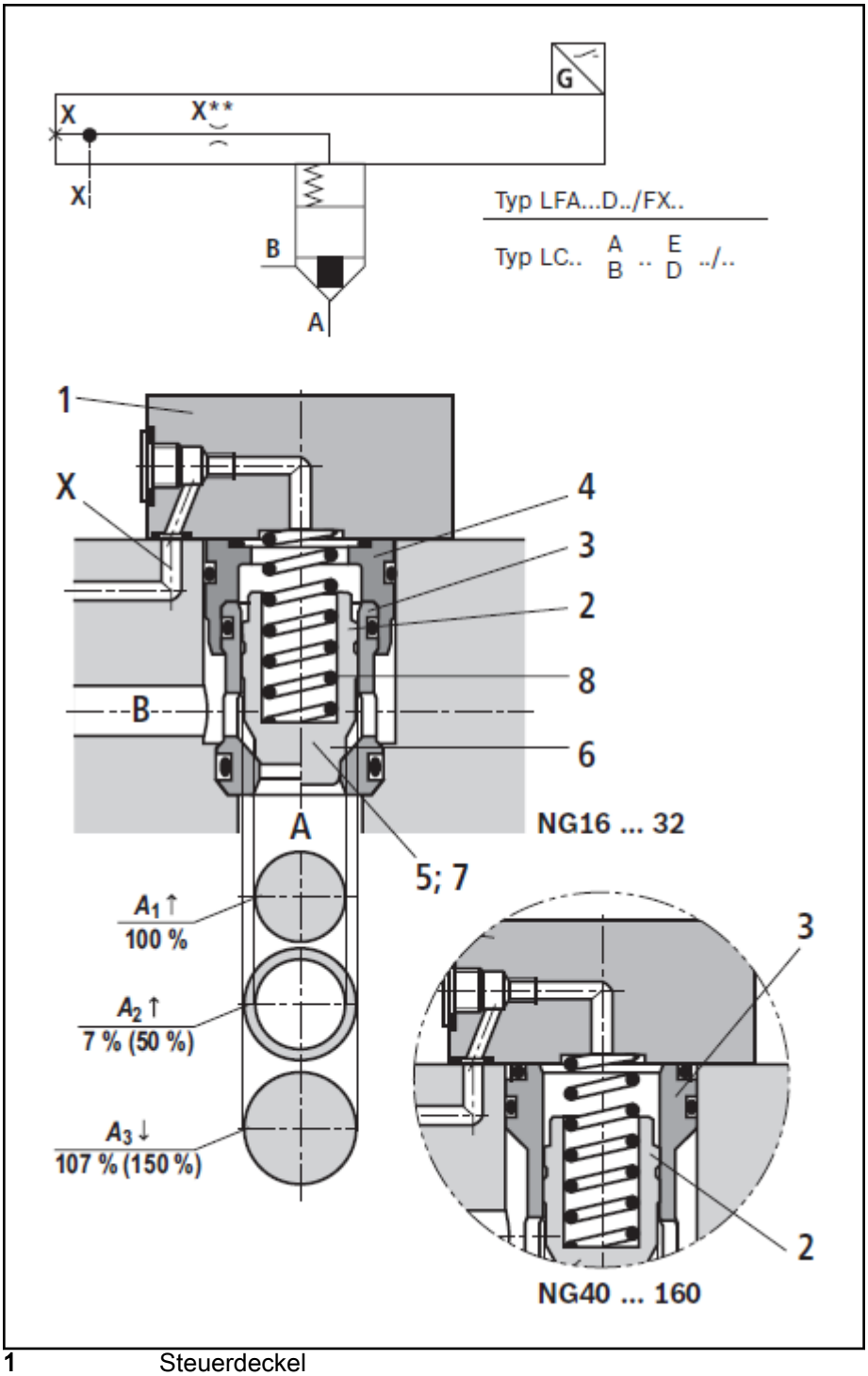
Der Funktionsbaustein überwacht den Initialzustand des Ventils mittels eines Rückmeldesignals ("EDM"). Die Rückmeldung muss den sicheren Zustand signalisieren, bevor das Ventil durch den Funktionsbaustein aktiviert werden kann. Der Funktionsbaustein bestätigt den sicheren Zustand mittels des Ausgangs "S_SafeState" (TRUE), wenn die sichere Position des Ventils erkannt wurde und kein Fehler ansteht. Dieser Ausgang kann für die Implementierung einer Wiederanlaufsperre der Maschine verwendet werden, wenn entweder das Ventil nicht im sicheren Zustand ist oder ein Fehler erkannt wurde.

☞ Der SIL-Level des "S_SafeState"-Ausgangs hängt von den charakteristischen Daten ($MTTF_d$) des Rückmeldesensors und seiner Integration in die Sicherheitssteuerung (z.B. DC) ab.

S_ValveStatus

Das Ausgangssignal "S_ValveStatus" kann zusammen mit dem Funktionsbaustein Kap. 16.5.5 "SF_ValveGroupControl" auf Seite 470 verwendet werden, um mehrere Ventile als Ventilgruppe anzusteuern. Hierbei werden alle Ventile einer Ventilgruppe in den sicheren Zustand versetzt, wenn ein Ventil der Ventilgruppe ausfällt. Dies ist z.B. der Fall, wenn der rückgemeldete Zustand nicht der Erwartungshaltung entspricht.

Aufbau Ventil



1 Steuerdeckel
2 Einbausatz
3 Buchse
5 Ventilkegel
6 Dämpfungszapfen
8 Schließfeder

Abb. 16-105: Aufbau eines 2-Wege Einbauventils am Beispiel des Rexroth RD 21015

- Die Fläche A1 und A2 wirken in Öffnungsrichtung. Die Fläche A3 und die Schließfeder wirken in Schließrichtung.

- Die Fläche A2 ist dauerhaft mit dem Lasthaltedruck beaufschlagt. Der Lasthaltedruck in dem Ringraum wirkt immer und resultiert von der bewegten/statischen Last bezogen auf die Ringfläche.
- Die Fläche A3 ist mit dem Lasthaltedruck beaufschlagt, wenn das Sitzventil Pos. 145 in Grundstellung steht. Die Fläche A3 wirkt nicht, wenn das Sitz-ventil Pos. 145 geschaltet ist.
- Die Wirkrichtung der resultierenden Kraft aus Öffnungs- und Schließkräften bestimmt das Schaltverhalten des 2-Wege-Einbauventiles.
- In Schließbewegung (downstroke) strömt das 2-Wege-Einbauventil aktiv von B nach A ("S_ValveOut" = Y14 = AUS) und in Rückzug strömt das 2-Wege-Einbauventil (upstroke) passiv von B nach A ("S_ValveOut" = Y14 = AN).

Arbeitszyklus

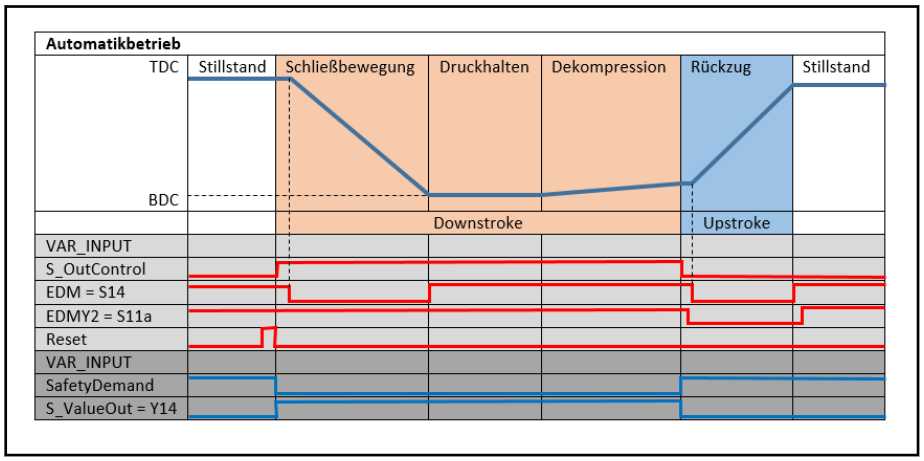


Abb. 16-106: Arbeitszyklus eines Oberkolbens (hydr. Presse) mit einem 2-Wege-Einbauvenil als sicherheitsrelevanter Steuerung

Wenn "S_ValveOut" (Y14) = FALSE und TDC erreicht

- Durch die Unterbrechung der Ölströmung zwischen Kanal A und B entsteht ein statisches Kraftverhältnis, was dazu führt, dass allein die Schließfeder nur noch wirkt und zwingt, den Ventilkegel in die sichere Grundstellung zu fahren. Daraus resultiert eine Rückmeldung des "EDM"-Werts S14 (Pin 2 high).
- Bei Druckbeaufschlagung der Fläche A3 ist der Kanal A leckagefrei gesperrt.
- Diese Konfiguration ist die Bedingung, um einen nächsten Arbeitsgang ein-zuleiten!

Wenn "S_OutControl" = FALSE und "S_ValveOut" (Y14) = TRUE - Donwstroke

- Die Fläche A3 ist von dem Lasthaltedruck getrennt und unmittelbar mit dem Tank verbunden.
- Die Schließkraft ist auf die Schließfederkraft reduziert. Die Fläche A2 ist von dem Lasthaltedruck dauerhaft beaufschlagt und wirkt in die Öffnungsrichtung.
- Ein ungleiches Kraftverhältnis entsteht, was zu der Öffnung des Ventilkegels führt, die Ölströmung geht von Kanal B nach Kanal A. Der Ventilkegel löst sich von seiner sicheren Grundstellung und daraus resultiert eine Rückmeldung des "EDM"-Werts S14 (Pin 2 = Low).
- Die tatsächliche Schließbewegung des Stößels ist eingeleitet.

Wenn "S_OutControl" = FALSE, "S_ValveOut" (Y14) = TRUE und BDC erreicht ist

- Durch die Unterbrechung der Ölströmung von Kanal B nach Kanal A entsteht ein statisches Kraftverhältnis, was dazu führt, dass allein die Schließ-feder nur noch wirkt und zwingt, den Ventilkegel in die sichere Grundstellung zu fahren. Daraus resultiert eine Rückmeldung des "EDM"-Wert (Pin 2 = high).
- Ein Signalwechsel von "EDM" S14 ist nach den EN693-Anforderungen entstanden.

Wenn "S_OutControl" = TRUE, "S_ValveOut" (Y14) = FALSE ⇒ Rückzug eingeleitet (Upstroke)

- Der auf der Fläche A1 herrschende Druck resultiert von dem Lasthaltedruck aus Fläche A2, die Schließkraft aus Fläche A3 und die Druckverluste am Ventilkegel. Durch die höhere wirkende Öffnungskraft im Kanal A (Fläche A1) öffnet der Ventilkegel, was dazu führt, dass das Öl von der Steuerkammer (Fläche A3) in den Ringraum des Zylinders gelangt.
- Dabei bleibt das Sitzventil Pos. 145 in Grundstellung.
- Hinweis: die Energietrennung beendet den Rückzug unabhängig von dem Richtungsventil Pos. 110.

Zustandsdiagramm

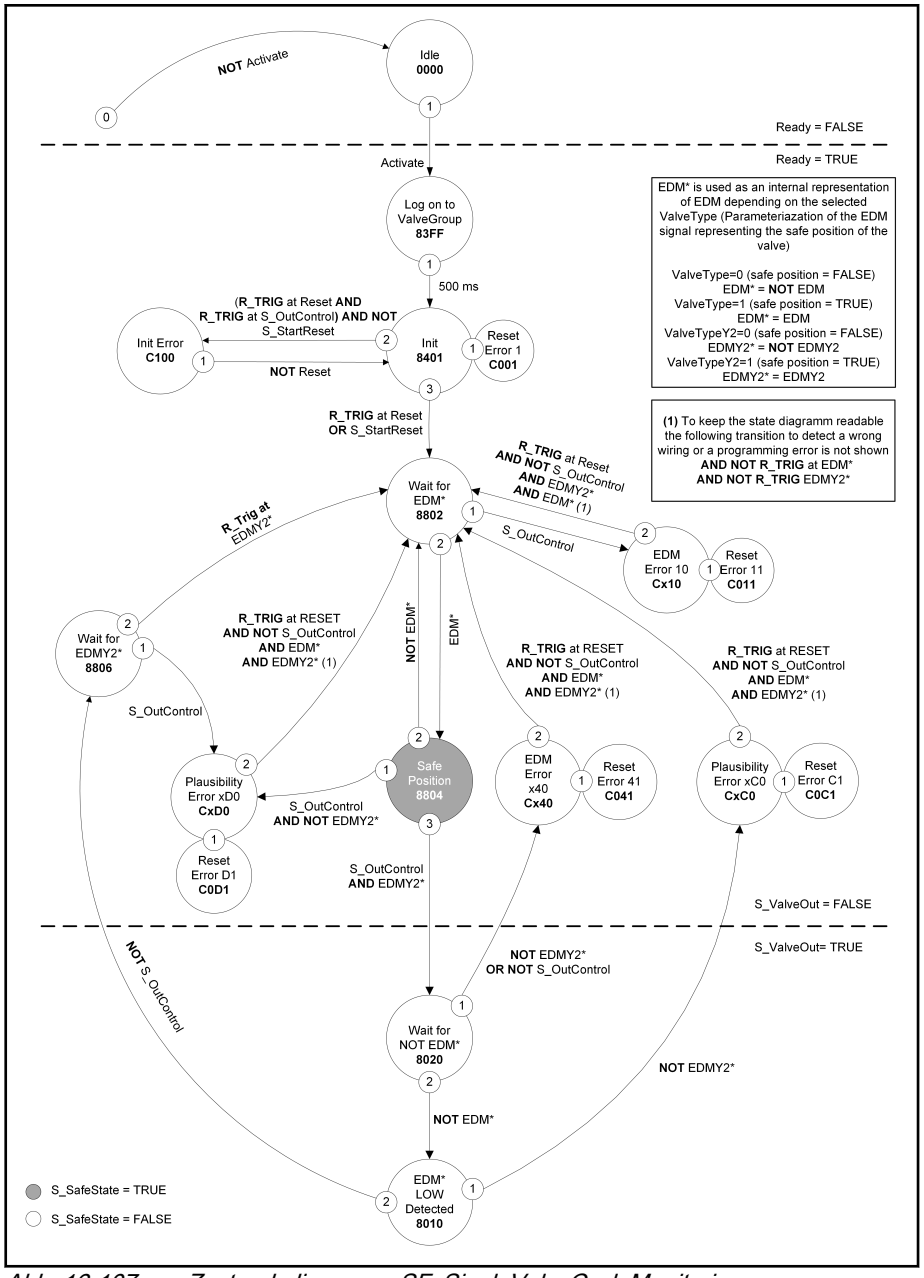


Abb. 16-107: Zustandsdiagramm SF_SingleValveCycleMonitoring

Zeitdiagramme

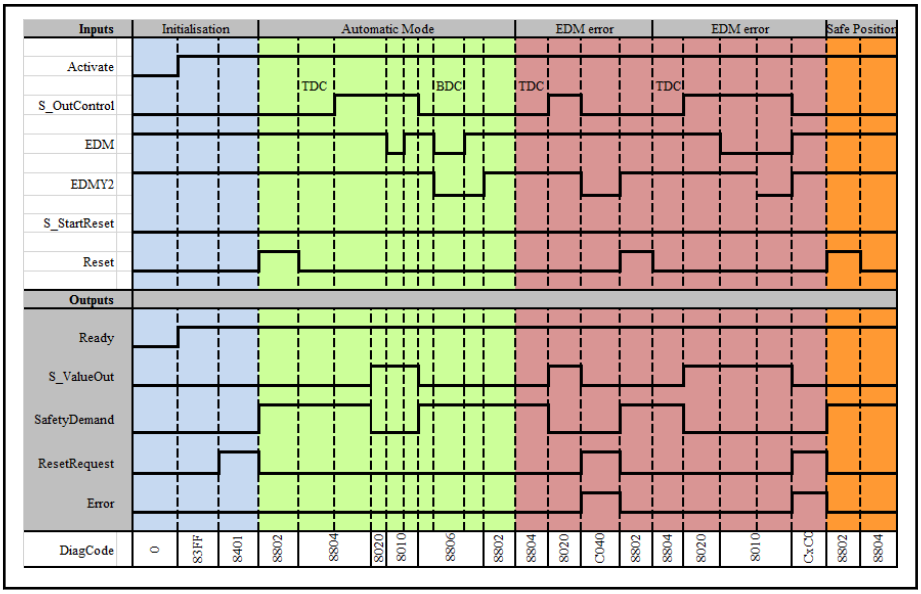


Abb. 16-108: Zeitdiagramm für SF_SingleValveCycleMonitoring

Fehlererkennung

Die folgenden Bedingungen führen zum Fehlerzustand:

- statisches "Reset"-Signal während des Prozesses
- Das "EDM"-Signal ändert sich innerhalb eines Maschinenzyklus nicht.
- Das "EDMY2"-Signal ändert sich innerhalb eines Maschinenzyklus nicht.
- "EDMY2" wird FALSE wenn "S_OutControl" TRUE ist
- "S_OutControl" und "Reset" sind aufgrund eines Programmierfehlers falsch verbunden.
- "S_OutControl" wird TRUE, bevor das Ventil seine sichere (Initiale) Position bestätigt hat.

Fehlerverhalten

Im Fehlerzustand sind die Ausgänge wie folgt gesetzt:

- Die Ausgänge "S_ValveOut" und "S_SafeState" sind FALSE und bleiben FALSE, bis der Fehler behoben und die Wiederanlaufsperre durch eine steigende Flanke am Eingang "Reset" aufgehoben wurde.
- Ein Fehler an den Eingängen "EDM" oder "EDMY2" muss durch eine steigende Flanke am Eingang "Reset" quittiert werden.
- Ein Reset-Fehler kann durch Rücksetzen des Eingangs "Reset" auf FALSE behoben werden.
- Nach der Bausteinaktivierung kann die optionale Anlaufsperre durch eine steigende Flanke am Eingang "Reset" aufgehoben werden.

DiagCode

DiagCode	Zustandsname	Beschreibung
16#0000	Idle	Der Baustein ist nicht initialisiert. Dieser Wert wird auch am Ausgang DiagCode ausgegeben, wenn "Activate" auf FALSE geht und der Zyklus abgelaufen ist. Ready = FALSE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 0000 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8010	EDM LOW Detected	Warte auf die Bestätigung des sicheren Ventilzustands durch das EDM-Signal. Der Zähler startet beim Eintritt in den Zustand. Ready = TRUE S_SafeState = FALSE S_ValveOut = TRUE S_ValveStatus = 8010 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8020	Wait for NOT EDM	Das EDM-Signal bestätigt die die Ansteuerung des Ventils. Ready = TRUE S_SafeState = FALSE S_ValveOut = TRUE S_ValveStatus = 8020 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#83FF	Log on to valve group	Wartezustand um sicherzustellen, dass der Funktionsbaustein SF_ValveGroupControl nur zusammen mit den Ventilüberwachungsbausteinen genutzt werden kann. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 83FF SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#8401	Init	Blockaktivierungsstart wird verhindert. Reset benötigt. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 8401 SafetyDemand = FALSE ResetRequest = TRUE Error = FALSE
16#8802	Wait for EDM	Der Ausgang "S_ValveOut" ist FALSE Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 8802 SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8804	Safe spool position	Das EDM-Signal bestätigt den sicheren Zustand des Ventils (sichere Schieberposition) Ready = TRUE S_SafeState = TRUE S_ValveOut = FALSE S_ValveStatus = 8804 SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8806	Wait for EDM Y2	Das Signal "EDMY2" muss sich ändern. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 8806 SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#C001	Reset Error 1	statischer Reset im Zustand 16#8401 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C001 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C011	Reset Error 11	statischer Reset im Zustand 16#Cx10 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C011 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C041	Reset Error 41	statischer Reset im Zustand 16#Cx40 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C041 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C0C1	Reset Error C1	statischer Reset im Zustand 16#CxC0 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C0C1 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C0D1	Reset Error D1	statischer Reset im Zustand 16#CxD0 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C0D1 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C100	Init Error	Gleiche Signale an S_OutControl und Reset (Flanke im selben Zyklus) erkannt. Dies ist möglicherweise ein Programmierfehler. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C100 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#Cx10	EDM Error x10	Das Signal EDM* ist im Startzustand des Aktors nicht korrekt. Im Zustand 16#8802 muss das EDM*-Signal FALSE sein, wenn "S_OutControl" aktiviert wird. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM* = FALSE oder EDMY2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C010 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C410 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#Cx40	EDM Error x40	Das Signal EDM* ist im Startzustand des Aktors nicht korrekt. Im Zustand 16#8020 muss das EDM*-Signal TRUE sein, wenn EDMY2* FALSE wird, was einen neuen Zyklus markiert. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM* = FALSE oder EDMY2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C040 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C440 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#CxC0	Plausibility Error xC0	Im Zustand 16#8010 wird das EDMY2*-Signal FALSE, was einen neuen Zyklus markiert, bevor das Ventil deaktiviert wird. ("S_OutControl" = FALSE) Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM* = FALSE oder EDMY2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C0C0 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C4C0 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#CxD0	Plausibility Error xD0	S_OutControl wird im Zustand 16#8806 TRUE. EDMY2* ist noch FALSE. Der Zyklus von Y2 wurde nicht beendet. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM* = FALSE oder EDMY2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C0D0 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C4D0 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

Tab. 16-204: Werte des Ausgangs "DiagCode"

16.5.4 SF_DoubleValveMonitoring

Kurzbeschreibung

Der Funktionsbaustein SF_DoubleValveMonitoring stellt Funktionalität zum Überwachen des Schaltverhaltens eines doppelten Hydraulikventils, das den sicheren Zustand innerhalb einer parametrierbaren Überwachungszeit durch ein statisches Rückmeldesignal für jedes Ventil bestätigt. Beide Ventile sollten vom selben Typ sein.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopenPress	2.8.0.0

Tab. 16-205: Bibliothekszuordnung

Schnittstellenbeschreibung

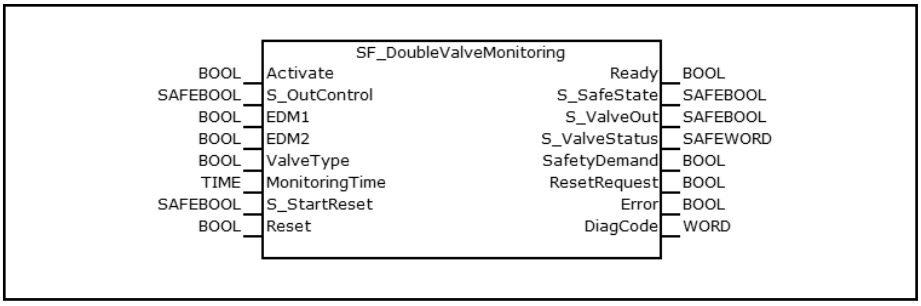


Abb. 16-109: Funktionsbaustein SF_DoubleValveMonitoring

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv
	S_OutControl	SAFEBOOL	Anforderung sicherer Zustand FALSE: Sicherer Zustand nicht angefordert TRUE: Sicherer Zustand angefordert
	EDM1	BOOL	Rückmeldesignal des 1. Ventils
	EDM2	BOOL	Rückmeldesignal des 2. Ventils
	ValveType	BOOL	Polarität der EDMx-Eingänge FALSE: Sicherer Zustand bei LOW-Signal TRUE: Sicherer Zustand bei High-Signal
	MonitoringTime	TIME	Zeit in der das Rückmeldesignal den angeforderten Schaltzustand bestätigen muss
	S_StartReset	SAFEBOOL	Automatisches Rücksetzen eines Anlauffehlers FALSE: Kein automatischer Kaltstart TRUE: Automatischer Kaltstart
	Reset	BOOL	Rücksetzen eines Fehlers bei steigender Flanke
VAR_OUTPUT	Ready	BOOL	Funktionsbaustein ist initialisiert und betriebsbereit
	S_SafeState	SAFEBOOL	Ausgang für die Überwachung des sicheren Zustands FALSE: Das Ventil befindet sich nicht im sicheren Zustand oder es ist ein Fehler aufgetreten. TRUE: Das Ventil befindet sich im sicheren Zustand und es ist kein Fehler aufgetreten.
	S_ValveOut	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveStatus	SAFEWORD	Ausgang für die Weiterverschaltung zum Funktionsbaustein Kap. 16.5.5 "SF_ValveGroupControl" auf Seite 470 um mehrere Ventile als Ventilgruppe anzusteuern, wobei alle anderen Ventile in den sicheren Zustand wechseln müssen, wenn ein Ventil der Ventilgruppe versagt.

E/A-Typ	Name	Datentyp	Beschreibung
	SafetyDemand	BOOL	Sicherheitsanforderung. Die für den sicheren Betrieb benötigten Voraussetzungen sind nicht erfüllt. Es liegt jedoch kein Fehler vor. TRUE: Sicherheitsanforderung ist aktiv FALSE: Es wird keine Absicherung benötigt
	ResetRequest	BOOL	Es wurde ein quittierbarer Fehler festgestellt. Vor dem Weiterbetrieb muss dieser mit einer steigenden Flanke am "Reset"-Eingang quittiert werden. TRUE: Reset angefordert FALSE: Reset nicht angefordert
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Werte des Ausgangs DiagCode, Seite 461)

Tab. 16-206: Schnittstelle SF_DoubleValveMonitoring

Name	Typ	Min	Max	Default-Wert	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_OutControl	SAFEBOOL			FALSE	kontinuierlich
EDM1	BOOL			FALSE	kontinuierlich
EDM2	BOOL			FALSE	kontinuierlich
ValveType	BOOL			FALSE	bei steigender Flanke an "Activate"
MonitoringTime	TIME			FALSE	bei steigender Flanke an "Activate"
S_StartReset	SAFEBOOL			FALSE	bei steigender Flanke an "Activate"
Reset	BOOL			FALSE	kontinuierlich

Tab. 16-207: Eingangsverhalten SF_DoubleValveMonitoring

Normativer Verweis

Norm	Anforderung
DIN EN ISO 13849-1:2016-6	5.2.1 Sicherheitsbezogene Stoppfunktion: sicherheitsbezogene Stoppfunktion muss die Maschine in den sicheren Zustand bringen 6.2 Spezifikation der Kategorien: Fehlererkennung (des Aktor, z.B. Drahtbruch)
DIN EN ISO 12100: 2011-03	6.2.11.4 Wiederanlauf nach Energieausfall 6.2.11.6 Verwendung von Selbstüberwachung

Norm	Anforderung
DIN EN ISO 4413: 2011-04	5.4.7.1 Unbeabsichtigte Bewegungen Steuerungen müssen so ausgelegt sein, dass unbeabsichtigte gefährdende Bewegungen und eine unzulässige Funktionsfolge der Aktuatoren verhindert sind. Dies gilt für alle Phasen des Betriebsablaufes. 5.4.8.1 Allgemeine Anforderungen Zur Fehlerbeseitigung und vorbeugenden Instandhaltung sollten Möglichkeiten für Fehlersuche und Zustands-überwachung vorgesehen sein. ...
DIN EN ISO 4414: 2011-04	5.4.6.1 Unbeabsichtigte Bewegung Steuerungen müssen so ausgelegt sein, dass für alle Phasen des Betriebsablaufes unbeabsichtigte gefährdende Bewegungen und eine unzulässige Funktionsfolge der Antriebe, insbesondere bei senkrechter und schräger Bewegungsrichtung, verhindert werden.
DIN EN 692:2009-10	5.4.1.4 ... Tritt ein Fehler in den sicherheitsrelevanten Teilen dieser Schutzeinrichtungen oder in der Steuerung auf, dann: a) darf ein unbeabsichtigter Anlauf nicht möglich sein; b) muss die sichere Funktion der Schutzeinrichtung aufrechterhalten bleiben; c) muss es möglich sein, die Maschine während der gefahrbringenden Bewegung anzuhalten; d) muss die Steuerung die Maschine während der gefährlichen Phase des Schließhubes sofort anhalten oder, in anderen Fällen, spätestens am Ende des Arbeitszyklus; e) muss die Steuerung jede Einleitung des folgenden Arbeitszyklus so lange verhindern, bis der Fehler behoben ist. 5.4.1.6 Die redundante und selbstüberwachte Pressensteuerung muss in zwei separaten Funktionssystemen arbeiten. Jedes System muss unabhängig in der Lage sein, die Gefahr bringende Bewegung anzuhalten, ohne Rücksicht auf den Zustand des anderen Systems. Das Versagen jedes der beiden Systeme muss durch Selbstüberwachung erkannt und ein weiterer Schließhub verhindert werden. Wenn das Versagen des einen Systems sich selbst anzeigt, d. h., wenn der Funktionsverlust selbst den nächsten Arbeitszyklus verhindert, ist eine weitere Überwachung dieses Systems nicht erforderlich. 5.4.2.3 ... d) sofern bei der Ventilüberwachung Sensoren zum Erkennen des Schaltzustandes der Ventile erforderlich sind, müssen diese Sensoren integrierte Bestandteile der Ventile sein. Das Doppelventil kann in sich selbst selbstüberwachend gestaltet sein, so dass das Ventilversagen selbstanzeigend ist;

Norm	Anforderung
DIN EN 693:2011-11	5.4.1.3 ... Tritt ein Fehler in den sicherheitsbezogenen Teilen dieser Schutzeinrichtungen oder in der Steuerung auf, dann a) darf ein unbeabsichtigter Anlauf nicht möglich sein; b) muss die sichere Funktion der Schutzeinrichtung aufrechterhalten bleiben; c) muss es möglich sein, die Maschine während der Gefahr bringenden Bewegung stillzusetzen; d) muss die Steuerung die Maschine während der Gefahr bringenden Phase des Schließhubes sofort stillsetzen oder, in anderen Fällen*, spätestens am Ende des Arbeitszyklus; e) muss die Steuerung jede Einleitung des folgenden Arbeitszyklus so lange verhindern, bis der Fehler behoben ist. 5.4.1.5 ... Die redundante und selbstüberwachte Pressensteuerung muss in zwei separaten Funktions-systemen arbeiten. Jedes System muss unabhängig in der Lage sein, die Gefahr bringende Bewegung anzuhalten, ohne Rücksicht auf den Zustand des anderen Systems. Das Versagen jedes der beiden Systeme muss durch Selbstüberwachung erkannt und ein weiterer Schließhub verhindert werden. Wenn das Versagen des einen Systems sich selbst anzeigt, d. h., wenn der Funktionsverlust selbst den nächsten Arbeitszyklus verhindert, ist eine weitere Überwachung dieses Systems nicht erforderlich.
DIN EN 12622:2014-02	5.2.1 ... ... Ein Ausfall von jedem einzelnen System muss durch Überwachung registriert und ein weiterer Schließhub verhindert werden. 5.2.5 ... Tabelle 2 - Sicherheit und Zuverlässigkeit des Steuerungssystems für den Einzelhubbetrieb mit manueller Zuführung und/oder Entnahme oder Einrichtbetrieb 5.2.5.2 Verhalten des Steuerungssystems im Falle eines Fehlers ... Kommt es in sicherheitsbezogenen Teilen der Schutzeinrichtungen oder dem Steuerungssystem zu einem Fehler dann: a) darf ein unerwarteter Anlauf nicht möglich sein; b) die sichere Funktionsweise der Schutzeinrichtungen muss nach dem ersten Fehler aufrecht erhalten bleiben; c) es muss weiterhin möglich sein, die Gesenkbiegepresse während einer gefährlichen Bewegung anzuhalten; d) das Steuerungssystem muss die Gesenkbiegepresse während einer gefährlichen Phase des Schließvorgangs innerhalb der vom Hersteller angegebenen Anhaltezeit stoppen oder, in anderen Fällen, bei der nächsten Aufforderung an die Sicherheitsfunktion oder aber spätestens zum Ende des Arbeitszyklus; e) das Steuerungssystem muss jede Einleitung des nächsten Produktionsbetriebszyklus verhindern bis, der Fehler behoben ist. 5.2.6.1 Anlauf ... Der unerwartete Anlauf von gefährlichen Bewegungen, wie z. B. des Pressbalkens, der hinteren Anschläge, der Werkstückauflage muss verhindert werden.

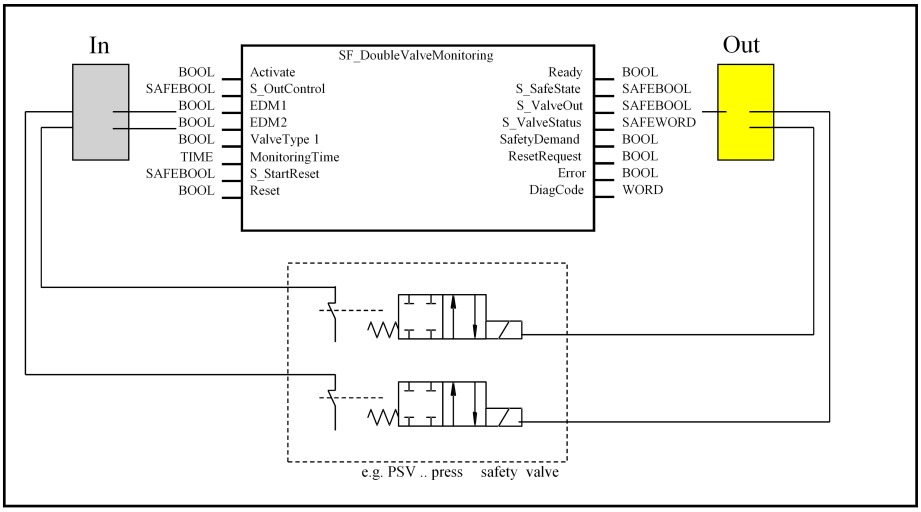


Abb. 16-110: Verschaltung des Funktionsbausteins SF_DoubleValveMonitoring

Funktionsbeschreibung

Der Funktionsbaustein SF_DoubleValveMonitoring stellt Funktionalität zum Überwachen des Schaltverhaltens eines doppelten Hydraulikventils mit zwei statischen Überwachungssignalen der sicheren Schieberposition jedes Ventils. Der Funktionsbaustein überwacht den Startzustand der Ventile mittels der Rückmeldesignale "EDM1" und "EDM2". Beide EDM-Signale müssen den sicheren Zustand signalisieren, bevor das Doppelventil durch den Funktionsbaustein aktiviert werden kann. Die Rückmeldesignale der Ventile werden auch nach der Aktivierung vom Funktionsbaustein überwacht. Sollte sein nicht innerhalb der eingestellten Überwachungszeit reagieren, wird der Ausgang "S_ValveOut" auf FALSE gesetzt. Dieser Zustand wird ebenfalls durch den "Error"-Ausgang angezeigt und eine entsprechende Diagnosemeldung ausgegeben.

Activate, Ready

Ist "Activate" FALSE, so ist der Funktionsbaustein inaktiv. Es werden keine Eingänge verarbeitet. Alle Ausgänge sind FALSE. Der Ausgang "DiagCode" ist 16#0000.

Wenn "Activate" auf TRUE geht, wird der Funktionsbaustein initialisiert. Der Ausgang "Ready" wird TRUE. Eine fallende Flanke am Eingang "Activate" versetzt den Baustein wieder in den inaktiven Zustand.

S_StartReset, Reset, Error, DiagCode, ResetRequest

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx, wobei xxx den Fehler-Code repräsentiert.

Ein Fehler wird mit einer steigenden Flanke am Eingang "Reset" zurückgesetzt. Hierfür muss sich die Ventile in der sicheren Position befinden. Dies wird durch die entsprechenden "EDMx"-Signale an den Funktionsbaustein gemeldet. Ein rücksetzbarer Fehler wird über ein TRUE am Ausgang "ResetRequest" angezeigt. Ist der Fehler behoben, wechselt der Ausgang "Error" bei steigender Flanke am Eingang "Reset" auf FALSE, und der Funktionsbaustein wartet auf den nächsten Zyklus.

Der Anlauffehler beim ersten Start des Funktionsbausteins kann über den Eingang "S_StartReset" automatisch quittiert werden. Der Eingang "S_StartReset" ist nur im Zustand C100 aktiv.



Der Eingang "S_StartReset" darf nur aktiviert werden, wenn daraus keine gefahrbringende Situation entstehen kann.

EDM1, EDM2, ValveType

Da die auf dem Markt verfügbaren Ventile die sichere Schieberposition entweder mit HIGH oder LOW signalisieren, kann die Polarität der Rückmeldesignal "EDM1" und "EDM2" über den Eingang "ValveType" eingestellt werden. Bei der Auswahl der Ventile sollte jedoch das Sicherheitsprinzip angewendet werden, nachdem der sichere Zustand durch ein HIGH-Signal bestätigt wird.

Das "EDM"-Signal muss je nach für die Applikation benötigtem Diagnosedeckungsgrad (DC) ggf. direkt über einen sicheren Eingang mit der Sicherheitssteuerung verbunden werden.

S_SafeState

Der Funktionsbaustein signalisiert den sicheren Zustand am Ausgang "S_SafeState" wenn die sichere Position der Ventilschieber erkannt wurde und kein Fehler ansteht. Dieser Ausgang kann für eine Wiederanlaufsperre der Maschine (z.B. dem Start eines neuen Zyklus), wenn entweder das Ventil nicht im sicheren Zustand ist oder ein Fehler erkannt wurde, verwendet werden. Dies kann z.B. ein Unterschied zwischen Rückmeldesignal und erwarteten Zustand sein.



Der SIL-Level des Ausgangs "S_SafeState" hängt von den charakteristischen Daten ($MTTF_d$) der Rückmeldesensoren und ihrer Integration in die Sicherheitssteuerung (z.B. DC) ab.

S_ValveStatus

Das Ausgangssignal "S_ValveStatus" kann zusammen mit dem Funktionsbaustein Kap. 16.5.5 "SF_ValveGroupControl" auf Seite 470 verwendet werden, um mehrere Ventile als Ventilgruppe anzusteuern. Hierbei werden alle Ventile einer Ventilgruppe in den sicheren Zustand versetzt, wenn ein Ventil der Ventilgruppe ausfällt. Dies ist z.B. der Fall, wenn der rückgemeldete Zustand nicht der Erwartungshaltung entspricht.

Zustandsdiagramm

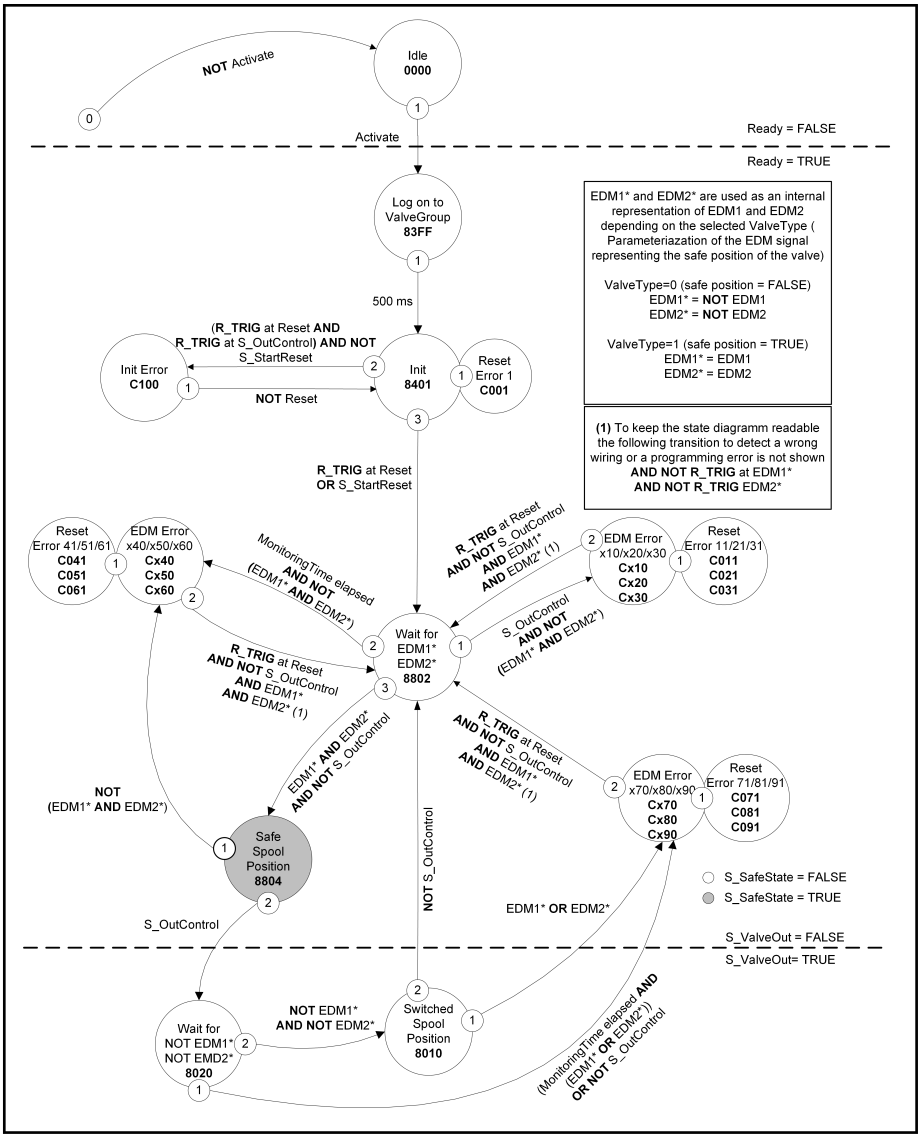
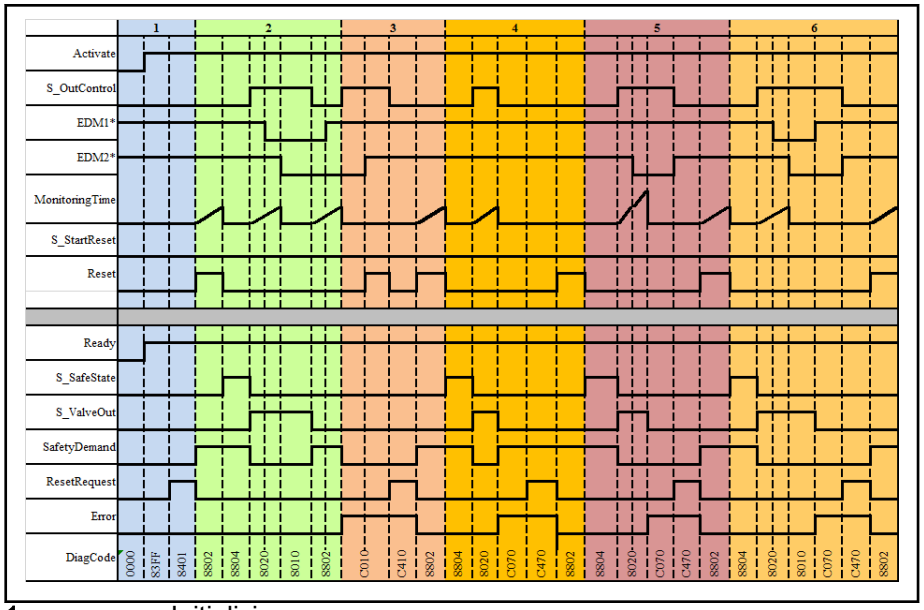


Abb. 16-111: Zustandsdiagramm SF_DoubleValveMonitoring

Zeitdiagramme



1 Initialisierung
2 Normale Operation mit Ein- und Ausschalten
3 Fehler - "S_OutControl" ist TRUE aber "EDM2" wurde nach der letzen Operation nicht TRUE.
4 Fehler - "S_OutControl" wird FALSE, bevor die Signale "EDM1" und "EDM2" FALSE werden.
5 Fehler - Die Überwachungszeit ("MonitoringTime") ist abgelaufen, "EDM1" wurde jedoch nicht FALSE.
6 Fehler - Das Signal "EDM1" wurde TRUE, "S_OutControl" ist jedoch noch TRUE.

Abb. 16-112: Zeitdiagramm für SF_DoubleValveMonitoring

Fehlererkennung

Die folgenden Bedingungen führen zum Fehlerzustand:

- Ungültiges statisches "Reset"-Signal im Prozess
- Ungültige "EDMx"-Signale im Prozess
- "S_OutControl" und "Reset" sind aufgrund eines Programmierfehlers falsch verbunden
- "S_OutControl" ändert seinen Zustand, bevor die "EDMx"-Signale die kommandierte Position bestätigt haben

Fehlerverhalten

Im Fehlerzustand sind die Ausgänge wie folgt gesetzt:

- Die Ausgänge "S_ValveOut" und "S_SafeState" sind FALSE und bleiben FALSE, bis der Fehler behoben und die Wiederanlaufsperre durch eine steigende Flanke am Eingang "Reset" aufgehoben wurde.
- Ein Fehler an den Eingängen "EDM1" oder "EDM2" muss durch eine steigende Flanke am Eingang "Reset" quittiert werden.
- Ein Reset-Fehler kann durch Rücksetzen des Eingangs "Reset" auf FALSE behoben werden.
- Nach der Bausteinaktivierung kann die optionale Anlaufsperre durch eine steigende Flanke am Eingang "Reset" aufgehoben werden.

DiagCode

DiagCode	Zustandsname	Beschreibung
16#0000	Idle	Der Baustein ist nicht initialisiert. Dieser Wert wird auch am Ausgang DiagCode ausgegeben, wenn "Activate" auf FALSE geht und der Zyklus abgelaufen ist. Ready = FALSE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 0000 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8010	Switched Spool Position	Warte auf die Bestätigung des sicheren Ventilzustands durch die "EDMx"-Signale. Ready = TRUE S_SafeState = FALSE S_ValveOut = TRUE S_ValveStatus = 8010 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8020	Wait for NOT EDM*	Die EDMx-Signale bestätigen die die Ansteuerung der Ventile. Die Überwachungszeit startet bei Eintritt in den Zustand. Ready = TRUE S_SafeState = FALSE S_ValveOut = TRUE S_ValveStatus = 8020 SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#83FF	Log on to valve group	Wartezustand um sicherzustellen, dass der Funktionsbaustein SF_ValveGroupControl nur zusammen mit den Ventilüberwachungsbausteinen genutzt werden kann. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 83FF SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#8401	Init	Blockaktivierungsstart wird verhindert. Reset benötigt. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 8401 SafetyDemand = FALSE ResetRequest = TRUE Error = FALSE
16#8802	Wait for EDM1* /EDM2*	Warte bis die "EDMx"-Signale die sichere Schieberposition bestätigt. Der Timer startet bei Zustandseintritt. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = 8802 SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8804	Safe spool position	Die "EDMx"-Signale bestätigt den sicheren Zustand des Ventils (sichere Schieberposition) Ready = TRUE S_SafeState = TRUE S_ValveOut = FALSE S_ValveStatus = 8804 SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#C001	Reset Error 1	statischer Reset im Zustand 16#8401 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C001 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C011	Reset Error 11	statischer Reset im Zustand 16#Cx10 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C011 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C021	Reset Error 21	statischer Reset im Zustand 16#Cx20 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C021 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C031	Reset Error 31	statischer Reset im Zustand 16#Cx30 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C031 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C041	Reset Error 41	statischer Reset im Zustand 16#Cx40 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C041 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C051	Reset Error 51	statischer Reset im Zustand 16#Cx50 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C051 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C061	Reset Error 61	statischer Reset im Zustand 16#Cx60 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C061 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C071	Reset Error 71	statischer Reset im Zustand 16#Cx70 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C071 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C081	Reset Error 81	statischer Reset im Zustand 16#Cx80 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C081 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C091	Reset Error 91	statischer Reset im Zustand 16#Cx90 erkannt Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C091 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C100	Init Error	Gleiche Signale an S_OutControl und Reset (Flanke im selben Zyklus) erkannt. Dies ist möglicherweise ein Programmierfehler. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE S_ValveStatus = C100 SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#Cx10	EDM Error x10	Das Signal EDM1* ist im Startzustand des Aktors nicht korrekt. Im Zustand 16#8802 muss das EDM1*-Signal FALSE sein, wenn "S_OutControl" aktiviert wird. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EDM2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C010 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C410 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#Cx20	EDM Error x20	Das Signal EDM2* ist im Startzustand des Aktors nicht korrekt. Im Zustand 16#8802 muss das EDM2*-Signal FALSE sein, wenn "S_OutControl" aktiviert wird. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EDM2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C020 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C420 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx30	EDM Error x30	Die Signale EDM1* und EDM2* sind im Startzustand des Aktors nicht korrekt. Im Zustand 16#8802 müssen EDM1* und EDM2* FALSE sein, wenn "S_OutControl" aktiviert wird. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EDM2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C030 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C430 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#Cx40	EDM Error x40	Das Signal EDM1* ist im Zustand 16#8802 oder 16#8804 nicht korrekt. Im Zustand 16#8802 ist das EDM1*-Signal FALSE und die "MonitoringTime" ist abgelaufen. Im Zustand 16#8804 wird das EDM1*-Signal FALSE obwohl das Ventil nicht geschaltet wurde. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EDM2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C040 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C440 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx50	EDM Error x50	Das Signal EDM2* ist im Zustand 16#8802 oder 16#8804 nicht korrekt. Im Zustand 16#8802 ist das EDM2*-Signal FALSE und die "MonitoringTime" ist abgelaufen. Im Zustand 16#8804 wird das EDM2*-Signal FALSE obwohl das Ventil nicht geschaltet wurde. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EDM2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C050 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C450 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#Cx60	EDM Error x60	Die Signale EDM1* und EDM2* sind im Zustand 16#8802 oder 16#8804 nicht korrekt. Im Zustand 16#8802 sind EDM1* und EDM2* FALSE und die "MonitoringTime" ist abgelaufen. Im Zustand 16#8804 werden EDM1* und EDM2* FALSE obwohl das Ventil nicht geschaltet wurde. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EDM2* = FALSE oder S_OutControl = TRUE S_ValveStatus = C060 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C460 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx70	EDM Error x70	Das Signal EDM1* ist im Zustand 16#8010 oder 16#8020 nicht korrekt. Im Zustand 16#8020 ist das EDM1*-Signal TRUE und die "MonitoringTime" ist abgelaufen oder "S_OutControl" wurde FALSE bevor EDM1* den Schaltvorgang bestätigt hat. Im Zustand 16#8010 wird das EDM1*-Signal TRUE obwohl das Ventil nicht geschaltet wurde. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EMD2* =FALSE oder S_OutControl = TRUE S_ValveStatus = C070 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C470 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx80	EDM Error x80	Das Signal EDM2* ist im Zustand 16#8010 oder 16#8020 nicht korrekt. Im Zustand 16#8020 ist das EDM2*-Signal TRUE und die "MonitoringTime" ist abgelaufen oder "S_OutControl" wurde FALSE bevor EDM2* den Schaltvorgang bestätigt hat. Im Zustand 16#8010 wird das EDM2*-Signal TRUE obwohl das Ventil nicht geschaltet wurde. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EMD2* =FALSE oder S_OutControl = TRUE S_ValveStatus = C080 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C480 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#Cx90	EDM Error x90	Die Signal EDM1* und EDM2* sind im Zustand 16#8010 oder 16#8020 nicht korrekt. Im Zustand 16#8020 sind EDM1* und EDM2* TRUE und die "MonitoringTime" ist abgelaufen oder "S_OutControl" wurde FALSE bevor EDM1* und EDM2* den Schaltvorgang bestätigt haben. Im Zustand 16#8010 wird EDM1* und EDM2* TRUE obwohl das Ventil nicht geschaltet wurde. Ready = TRUE S_SafeState = FALSE S_ValveOut = FALSE Wenn EDM1* = FALSE oder EMD2* =FALSE oder S_OutControl = TRUE S_ValveStatus = C090 SafetyDemand = FALSE ResetRequest = FALSE sonst S_ValveStatus = C490 SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

Tab. 16-208: Werte des Ausgangs "DiagCode"

16.5.5 SF_ValveGroupControl

Kurzbeschreibung Der Funktionsbaustein SF_ValveGroupControl fasst alle verbundenen Ventile zu einer Gruppe zusammen. Hat ein Ventil einen Fehler, werden alle Ventile abgeschaltet.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopenPress	2.8.0.1

Tab. 16-209: Bibliothekszuordnung

Schnittstellenbeschreibung

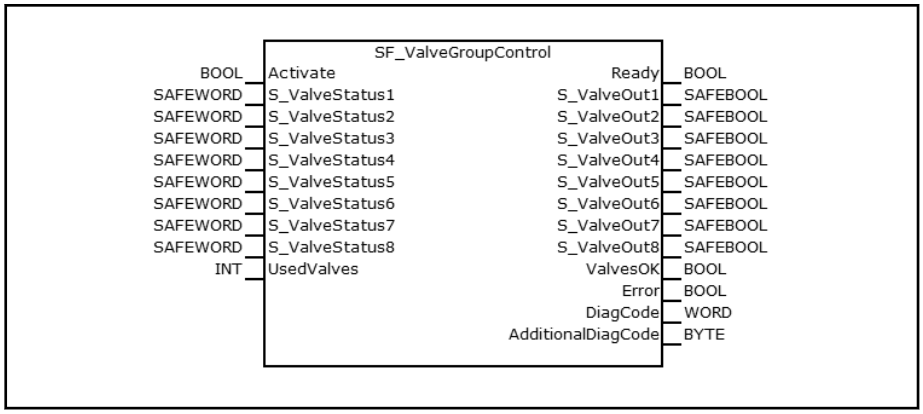


Abb. 16-113: Funktionsbaustein SF_ValveGroupControl

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv
	S_ValveStatus1	SAFEWORD	"ValveStatus"-Signal eines vorgeschalteten Ventil-Bausteins (z.B. Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425)
	S_ValveStatus2	SAFEWORD	"ValveStatus"-Signal eines vorgeschalteten Ventil-Bausteins (z.B. Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425)
	S_ValveStatus3	SAFEWORD	"ValveStatus"-Signal eines vorgeschalteten Ventil-Bausteins (z.B. Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425)
	S_ValveStatus4	SAFEWORD	"ValveStatus"-Signal eines vorgeschalteten Ventil-Bausteins (z.B. Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425)
	S_ValveStatus5	SAFEWORD	"ValveStatus"-Signal eines vorgeschalteten Ventil-Bausteins (z.B. Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425)
	S_ValveStatus6	SAFEWORD	"ValveStatus"-Signal eines vorgeschalteten Ventil-Bausteins (z.B. Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425)
	S_ValveStatus7	SAFEWORD	"ValveStatus"-Signal eines vorgeschalteten Ventil-Bausteins (z.B. Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425)
	S_ValveStatus8	SAFEWORD	"ValveStatus"-Signal eines vorgeschalteten Ventil-Bausteins (z.B. Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425)
	UsedValves	INT	Anzahl der verwendeten "S_ValveStatusx"-Eingänge für die Gruppenbildung
VAR_OUTPUT	Ready	BOOL	Funktionsbaustein ist initialisiert und betriebsbereit
	S_ValveOut1	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule

E/A-Typ	Name	Datentyp	Beschreibung
	S_ValveOut2	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveOut3	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveOut4	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveOut5	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveOut6	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveOut7	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	S_ValveOut8	SAFEBOOL	Ansteuerung der Ventilspule FALSE: Keine Ansteuerung der Ventilspule TRUE: Ansteuerung der Ventilspule
	ValvesOk	BOOL	Status aller gruppierten Ventile. Dieser Ausgang kann verwendet werden, um die funktionale Anwendung über den Status der Ventile zu informieren.
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Werte des Ausgangs DiagCode, Seite 476)
	AdditionalDiagCode	BYTE	Zusätzliche Diagnoseinformation, um anzuzeigen, welches Ventil einen Fehler hat. Die Ventile sind als einzelne Bits im Byte aufgelegt.

Tab. 16-210: Schnittstelle SF_ValveGroupControl

Name	Typ	Min	Max	Default-Wert	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_ValveStatus1	SAFEWORD	16#0000	16#FFFF	16#0000	kontinuierlich
S_ValveStatus2	SAFEWORD	16#0000	16#FFFF	16#0000	kontinuierlich
S_ValveStatus3	SAFEWORD	16#0000	16#FFFF	16#0000	kontinuierlich
S_ValveStatus4	SAFEWORD	16#0000	16#FFFF	16#0000	kontinuierlich
S_ValveStatus5	SAFEWORD	16#0000	16#FFFF	16#0000	kontinuierlich
S_ValveStatus6	SAFEWORD	16#0000	16#FFFF	16#0000	kontinuierlich

Name	Typ	Min	Max	Default-Wert	Übernahme
S_ValveStatus7	SAFEWORD	16#0000	16#FFFF	16#0000	kontinuierlich
S_ValveStatus8	SAFEWORD	16#0000	16#FFFF	16#0000	kontinuierlich
UsedValves	INT	0	8	0	bei steigender Flanke an "Activate"

Tab. 16-211: Eingangsverhalten SF_ValveGroupControl

Funktionsbeschreibung

Der Funktionsbaustein SF_ValveGroupControl wird zum Sperren aller Ventile der verbundenen Ventilgruppe im Fehlerfall verwendet. Er wird immer zusammen mit den drei Ventil-Funktionsbausteinen (Kap. 16.5.2 "SF_SingleValveMonitoring" auf Seite 425, Kap. 16.5.3 "SF_SingleValveCycleMonitoring" auf Seite 437 und Kap. 16.5.4 "SF_DoubleValveMonitoring" auf Seite 453) verwendet. Der Funktionsbaustein prüft, ob ein verbundener Ventil-Funktionsbaustein einen Fehler erkannt hat, und schaltet ggf. alle Ventile der Ventilgruppe ab. Ist ein "S_ValveStatusX"-Eingang nicht verbunden, kann der entsprechende "S_ValveOutX"-Ausgang nicht TRUE werden. Der Funktionsbaustein enthält keine Funktionalität für gegenseitiges Sperren von Ventilen.

Ein Ausgang kann nur gesetzt werden, wenn sich der entsprechende Ventil-Funktionsbaustein in einem 16#8xx0-Zustand (wie z.B. 0x8010 oder 0x8020) befindet. Der Ausgang "AdditionalDiagCode" zeigt an, welcher Ausgang gesetzt ist.

Der Funktionsbaustein enthält Mechanismen um systematische Fehler zu verhindern:

- In den Ventil-Funktionsblöcken ist ein Login-Algorithmus implementiert, der vom Funktionsbaustein SF_ValveGroupControl ausgewertet wird (16#83FF⇒16#8x01).
- Die Anzahl der verbundenen Ventile wird mit der angegebenen Ventilanzahl am Eingang "UsedValves" verglichen. Die Ventil-Funktionsbausteine müssen lückenlos aufsteigend an den "S_ValveStatusX"-Eingängen angeschlossen werden.

Zustandsdiagramm

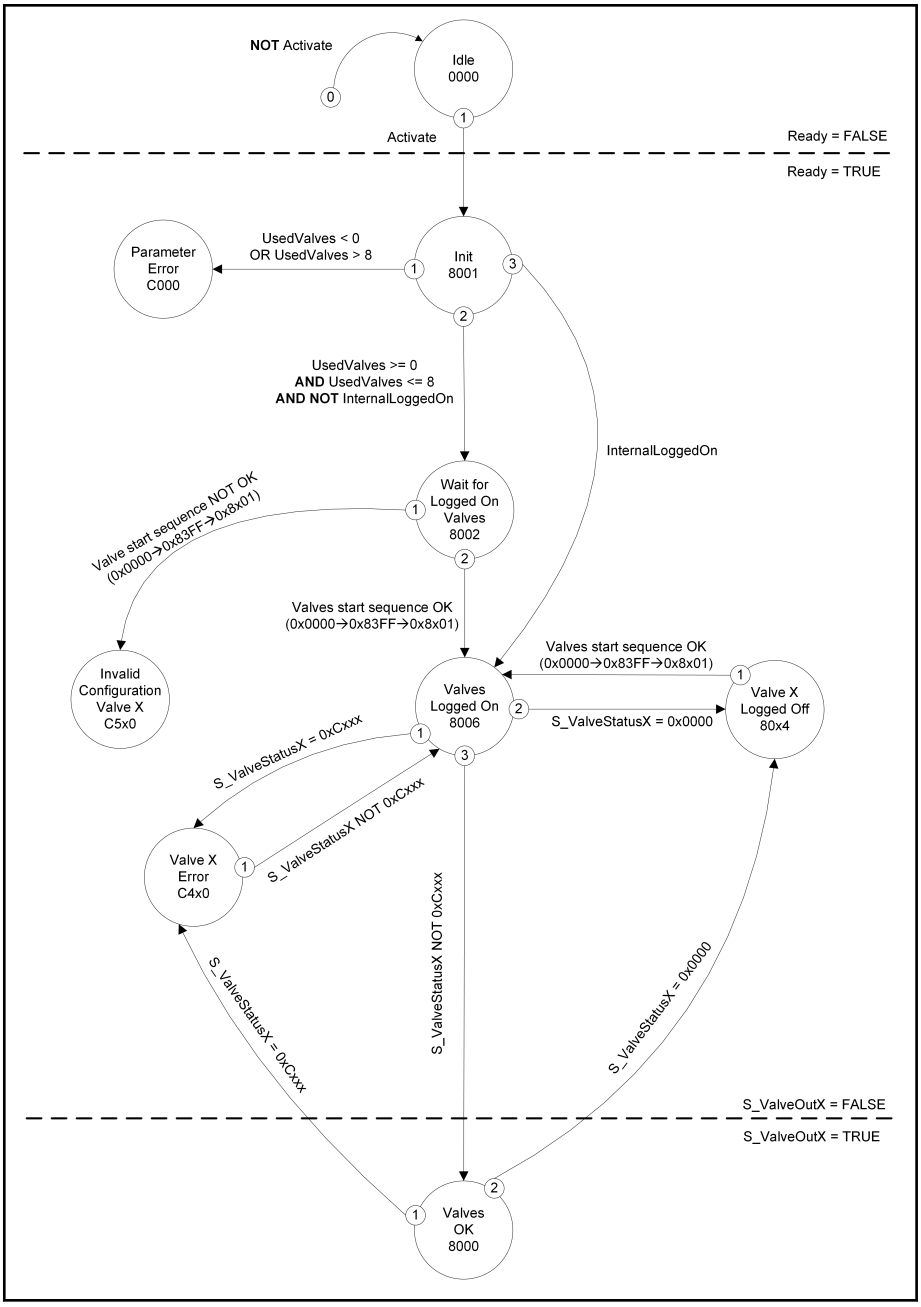


Abb. 16-114: Zustandsdiagramm SF_ValveGroupControl

Zeitdiagramme

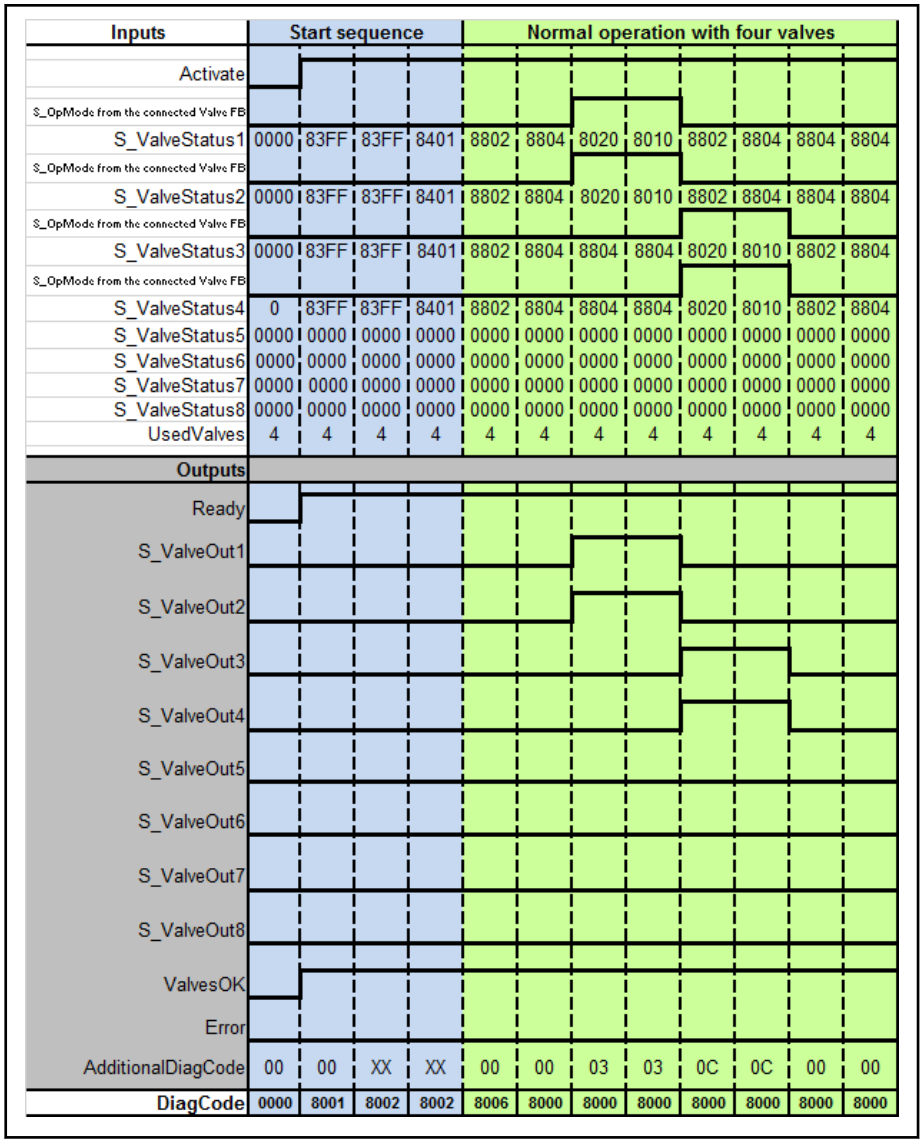


Abb. 16-115: Zeitdiagramm für SF_ValveGroupControl

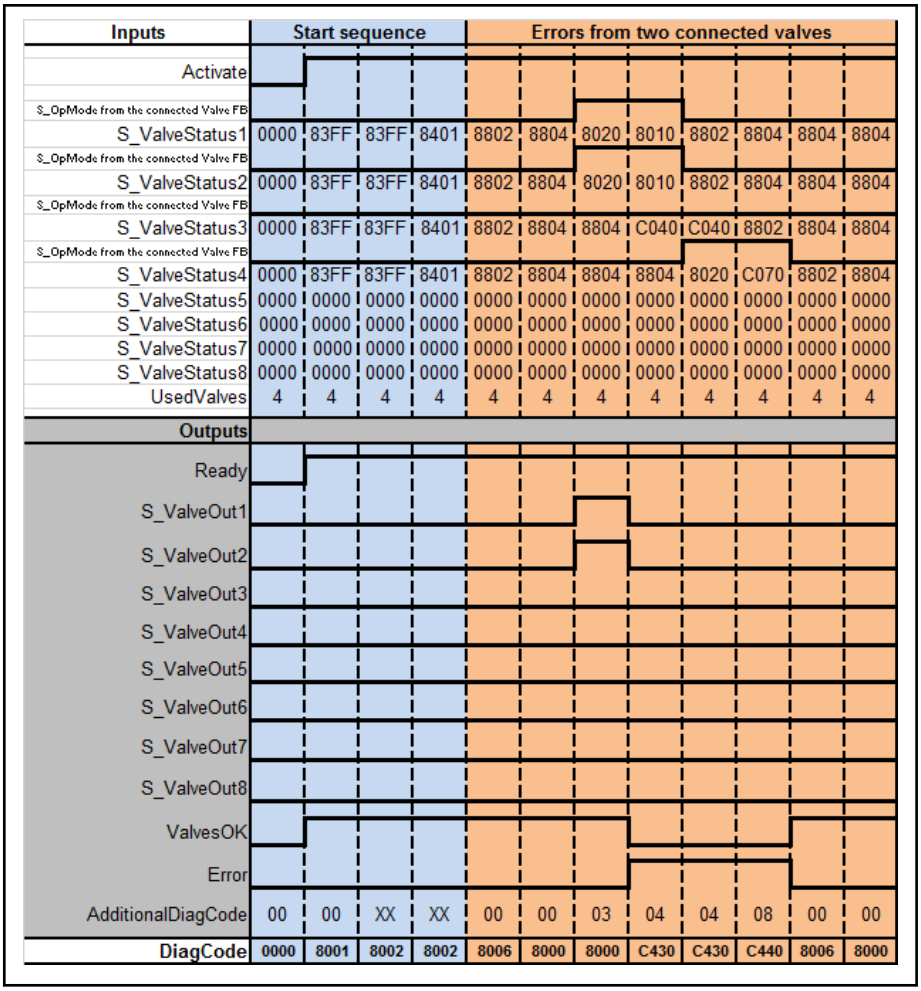


Abb. 16-116: Zeitdiagramm für SF_ValveGroupControl

Fehlererkennung

Der Funktionsbaustein erkennt einen Fehler des Parameters "UsedValves", wenn an einem aktiven "S_ValveStatusX"-Eingang ein nicht unterstützter Wert erscheint. Falls ein oder mehrere Ventile denselben Status 0x0000 gleichzeitig melden, sollte der Ausgang "DiagCode" dem Ventil mit der höchsten Nummer (z.B. 8) entsprechen, während "AdditionalDiagCode" den Status der relevanten Ventile mit dem entsprechenden Bit anzeigt.

Fehlerverhalten

Im Fall eines Fehler werden die Ausgänge "S_ValveOutX" auf FALSE gesetzt. Der Ausgang "DiagCode" zeigt den relevanten Fehlercode an und der Ausgang "Error" ist TRUE. Ein Wiederanlauf wird verhindert, bis alle Fehlerzustände beseitigt sind.

Der Funktionsbaustein besitzt den zusätzlichen Ausgang "AdditionalDiagCode", um anzuzeigen, welches Ventil nicht korrekt angemeldet wurde.

DiagCode

DiagCode	Zustandsname	Beschreibung
16#0000	Idle	Der Baustein ist nicht initialisiert. Dieser Wert wird auch am Ausgang DiagCode ausgegeben, wenn "Activate" auf FALSE geht und der Zyklus abgelaufen ist. Ready = FALSE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = FALSE AdditionalDiagCode = 16#00
16#8000	Valves OK	Die Ausgänge "S_ValveOut1" .. "S_ValveOut8" sind in Abhängigkeit vom Status der entsprechenden "S_ValveStatusX"-Eingänge aktiv Ready = TRUE S_ValveOut1 = TRUE oder FALSE S_ValveOut2 = TRUE oder FALSE S_ValveOut3 = TRUE oder FALSE S_ValveOut4 = TRUE oder FALSE S_ValveOut5 = TRUE oder FALSE S_ValveOut6 = TRUE oder FALSE S_ValveOut7 = TRUE oder FALSE S_ValveOut8 = TRUE oder FALSE S_ValveOK = TRUE Error = FALSE AdditionalDiagCode = 16#XX (Bits der Ventile gesetzt wenn der entsprechende Zustand 16#8xx0 ist)

DiagCode	Zustandsname	Beschreibung
16#8001	Init	Der Funktionsbaustein wurde aktiviert und initialisiert. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#00
16#8002	Wait for Logged On Valves	Warte auf den speziellen Code für die Anmeldung der Ventile. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#00
16#8006	Valves Logged On	Alle definierten Ventile haben sich angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#00

DiagCode	Zustandsname	Beschreibung
16#8014	Valve 1 Logged Off	Ventil 1 hat sich abgemeldet (S_ValveStatus1 = 16#0000) Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#01
16#8024	Valve 2 Logged Off	Ventil 1 hat sich abgemeldet (S_ValveStatus2 = 16#0000) Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#02
16#8034	Valve 3 Logged Off	Ventil 1 hat sich abgemeldet (S_ValveStatus3 = 16#0000) Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#04

DiagCode	Zustandsname	Beschreibung
16#8044	Valve 4 Logged Off	Ventil 1 hat sich abgemeldet (S_ValveStatus4 = 16#0000) Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#08
16#8054	Valve 5 Logged Off	Ventil 1 hat sich abgemeldet (S_ValveStatus5 = 16#0000) Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#10
16#8064	Valve 6 Logged Off	Ventil 1 hat sich abgemeldet (S_ValveStatus6 = 16#0000) Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#20

DiagCode	Zustandsname	Beschreibung
16#8074	Valve 7 Logged Off	Ventil 1 hat sich abgemeldet (S_ValveStatus7 = 16#0000) Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#40
16#8084	Valve 8 Logged Off	Ventil 1 hat sich abgemeldet (S_ValveStatus8 = 16#0000) Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = Abhängig vom Status aller relevanten Ventile Error = FALSE AdditionalDiagCode = 16#80
16#C000	Parameter Error 0	Der Eingang UsedValves befindet sich nicht im gültigen Bereich (0..8). Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#00

DiagCode	Zustandsname	Beschreibung
16#C410	Valve 1 Error	Ventil 1 hat einen Fehler erkannt. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#01
16#C420	Valve 2 Error	Ventil 2 hat einen Fehler erkannt. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#02
16#C430	Valve 3 Error	Ventil 3 hat einen Fehler erkannt. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#04

DiagCode	Zustandsname	Beschreibung
16#C440	Valve 4 Error	Ventil 4 hat einen Fehler erkannt. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#08
16#C450	Valve 5 Error	Ventil 5 hat einen Fehler erkannt. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#10
16#C460	Valve 6 Error	Ventil 6 hat einen Fehler erkannt. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#20

DiagCode	Zustandsname	Beschreibung
16#C470	Valve 7 Error	Ventil 7 hat einen Fehler erkannt. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#40
16#C480	Valve 8 Error	Ventil 8 hat einen Fehler erkannt. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#80
16#C510	Invalid Configuration Valve 1	Ventil 1 soll verwendet werden, hat sich jedoch nicht angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#01

DiagCode	Zustandsname	Beschreibung
16#C520	Invalid Configuration Valve 2	Ventil 2 soll verwendet werden, hat sich jedoch nicht angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#02
16#C530	Invalid Configuration Valve 3	Ventil 3 soll verwendet werden, hat sich jedoch nicht angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#04
16#C540	Invalid Configuration Valve 4	Ventil 4 soll verwendet werden, hat sich jedoch nicht angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#08

DiagCode	Zustandsname	Beschreibung
16#C550	Invalid Configuration Valve 5	Ventil 5 soll verwendet werden, hat sich jedoch nicht angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#10
16#C560	Invalid Configuration Valve 6	Ventil 6 soll verwendet werden, hat sich jedoch nicht angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#20

DiagCode	Zustandsname	Beschreibung
16#C570	Invalid Configuration Valve 7	Ventil 7 soll verwendet werden, hat sich jedoch nicht angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#40
16#C580	Invalid Configuration Valve 8	Ventil 8 soll verwendet werden, hat sich jedoch nicht angemeldet. Ready = TRUE S_ValveOut1 = FALSE S_ValveOut2 = FALSE S_ValveOut3 = FALSE S_ValveOut4 = FALSE S_ValveOut5 = FALSE S_ValveOut6 = FALSE S_ValveOut7 = FALSE S_ValveOut8 = FALSE S_ValveOK = FALSE Error = TRUE AdditionalDiagCode = 16#80

Tab. 16-212: Werte des Ausgangs "DiagCode"

16.5.6 SF_CamMonitoring

Kurzbeschreibung

Der Funktionsbaustein SF_CamMonitoring stellt Funktionalität zum Überwachen der Nocken einer Presse bereit.

Er überwacht die korrekte Abfolge der Nocken und gibt Informationen über die aktuelle Pressenposition wie den oberen Totpunkt (TDC) und die Bewegungsrichtung des Pressenstempels aus.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopenPress	2.6.0.2

Tab. 16-213: Bibliothekszuordnung

Schnittstellenbeschreibung

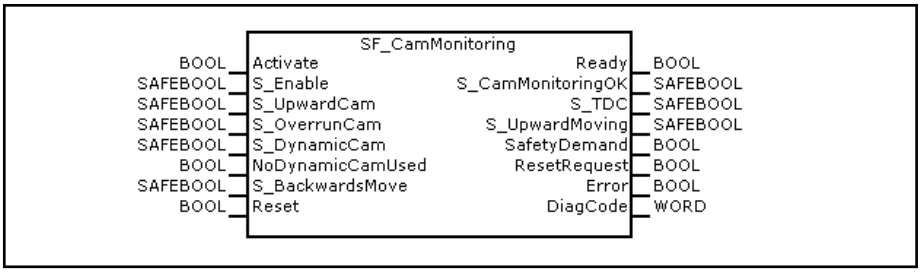


Abb. 16-117: Funktionsbaustein SF_CamMonitoring

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv
	S_Enable	SAFEBOOL	Aktiviert die Nockenüberwachung FALSE: Keine Nockenüberwachung aktiv TRUE: Nockenüberwachung aktiv
	S_UpwardCam	SAFEBOOL	Eingang für den Nocken der Aufwärtsbewegung des Pressenstempels FALSE: Nocken inaktiv TRUE: Nocken aktiv
	S_OverrunCam	SAFEBOOL	Eingang für den Nachlaufnocken (Nocken um den oberen Totpunkt) FALSE: Nocken inaktiv TRUE: Nocken aktiv
	S_DynamicCam	SAFEBOOL	Eingang für den dynamische Nocken FALSE: Nocken inaktiv TRUE: Nocken aktiv
	NoDynamicCamUsed	BOOL	Eingang zur Deaktivierung der dynamischen Nocke FALSE: Der dynamische Nocken wird verwendet TRUE: Der dynamische Nocken wird nicht verwendet
	S_BackwardsMove	SAFEBOOL	Rückwärtsbewegung der Presse FALSE: Die Presse läuft vorwärts TRUE: Die Presse läuft rückwärts
	Reset	BOOL	Rücksetzen eines Fehlers bei steigender Flanke
VAR_OUTPUT	Ready	BOOL	Funktionsbaustein ist initialisiert und betriebsbereit
	S_CamMonitoringOK	SAFEBOOL	Ausgang für die Überwachung des Nockenfolgen FALSE: Die Nockenfolge ist nicht korrekt oder Signalfehler TRUE: Die Nockenfolge ist korrekt oder "S_Enable" ist FALSE
	S_TDC	SAFEBOOL	Ausgangssignal für den oberen Totpunkt (TDC) FALSE: Nachlaufnocken inaktiv TRUE: Nachlaufnockens aktiv

E/A-Typ	Name	Datentyp	Beschreibung
	S_UpwardsMoving	BOOL	Ausgang für die Aufwärtsbewegung des Pressenstempels (vom unteren Totpunkt zum oberen Totpunkt) FALSE: Die Presse ist nicht in Aufwärtsbewegung TRUE: Die Presse ist in Aufwärtsbewegung
	SafetyDemand	BOOL	Sicherheitsanforderung. Die für den sicheren Betrieb benötigten Voraussetzungen sind nicht erfüllt. Es liegt jedoch kein Fehler vor. TRUE: Sicherheitsanforderung ist aktiv FALSE: Es wird keine Absicherung benötigt
	ResetRequest	BOOL	Es wurde ein quittierbarer Fehler festgestellt. Vor dem Weiterbetrieb muss dieser mit einer steigenden Flanke am "Reset"-Eingang quittiert werden. TRUE: Reset angefordert FALSE: Reset nicht angefordert
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Werte des Ausgangs DiagCode, Seite 495)

Tab. 16-214: Schnittstelle SF_CamMonitoring

Name	Typ	Min	Max	Default-Wert	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_Enable	SAFEBOOL			FALSE	kontinuierlich
S_UpwardCam	SAFEBOOL			FALSE	kontinuierlich
S_OverrunCam	SAFEBOOL			FALSE	kontinuierlich
S_DynamicCam	SAFEBOOL			FALSE	kontinuierlich
NoDynamicCamUsed	BOOL			FALSE	kontinuierlich
S_BackwardsMove	SAFEBOOL			FALSE	kontinuierlich
Reset	BOOL			FALSE	kontinuierlich

Tab. 16-215: Eingangsverhalten SF_CamMonitoring

Normativer Verweis

Norm	Anforderung
EN 692:2009-10	5.4.2.4 c Wenn eine Nachlaufüberwachung erforderlich ist, muss diese den folgenden Anforderungen entsprechen: 1. Handbeschickte Pressen, an denen die in 5.4.1.4 aufgelisteten Schutzeinrichtungen eingebaut sind, müssen mit Einrichtungen zur Nachlaufüberwachung versehen sein; wenn die Kurbelwelle über den normalen Anhaltepunkt um einen vom Hersteller angegebenen Wert von maximal 15° und vorzugsweise 10° nachläuft, muss sichergestellt sein, dass sofort ein Stoppsignal ausgelöst wird und die Einleitung eines weiteren Hubes nicht mehr möglich ist; 2. die Wiederaufnahme des weiteren Betriebes der Presse darf nur mit einem besonderen Mittel möglich sein, z. B. durch Werkzeug, Schlüssel oder elektronisches Passwort; 3. wenn die Nocken der Nachlaufüberwachung von einer Nockenwelle angetrieben werden, die indirekt von einer Kurbelwelle angetrieben wird, z. B. durch Duplexkettenantrieb zwischen Nockenwelle und Kurbelwelle, ist der indirekte Antrieb so zu überwachen, dass bei einem Ausfall ein Haltsignal eingeleitet wird und die Auslösung eines neuen Zyklus so lange unmöglich ist, bis der Fehler behoben ist. 5.4.2.5 Die Nocken und die entsprechenden Schalter der Nachlaufüberwachung, der Einzelzyklus-Stoppfunktion und des Mutings sollen zwangsläufig miteinander verbunden sein, so dass die Stellung der Nocken zueinander und die Stellung der Schalter untereinander nicht verändert werden kann. Bei Pressen, die mit einer veränderlichen Geschwindigkeitseinstellung ausgerüstet sind, kann jedoch der Schalter für die Einzelhub-Stoppfunktion getrennt einstellbar sein. Alle Nocken müssen formschlüssig an der Welle befestigt sein. Alle Nocken und die entsprechenden Schalter müssen sich in einem abschließbaren Gehäuse befinden. 5.4.2.6 Wenn die Hublänge verändert werden kann, muss die Verstellung der Nocken oder Nockenschalter für die Zyklussteuerung zwangsläufig so verknüpft werden, dass ihre Stellung zueinander nicht verändert werden kann, um die Möglichkeit einer Fehleinstellung durch den Bediener zu minimieren, die z. B. zum Ausgleich verminderter Bremswirkung genutzt werden könnte, jedoch durch eine Wartung der Bremse behoben werden sollte. 5.4.2.9 d Nockenscheiben müssen derart eingesetzt sein, dass ein falsches oder nicht beabsichtigtes Einstellen keine Unfälle verursachen kann, und: 1. Die Positionsschalter und die Nocken für die Steuerung der Presse müssen fest zueinander angeordnet und entsprechend gesichert sein. Ungesicherte Muttern und Schrauben sind nicht zulässig; 2. die Stellung der Positionsschalter und Nocken zueinander muss gekennzeichnet sein; 3. die Möglichkeit einer Nachstellung von Schaltern oder Nocken muss durch feste Endanschläge begrenzt sein, so dass die Übernahmezeit (Muting-Zeit) während der Schließbewegung der Presse die Gesamt- Ansprechzeit der Presse bei einer beliebigen Kombination aus Hubzahl und Hublänge nicht überschreiten kann; 4. beim Reversieren der Presse darf das Nockenschaltwerk nicht beschädigt werden. 5.4.7.2 Die Betätigungsmittel des Schalters und der Schalter selbst müssen so gestaltet sein, dass sie ihre richtige Zuordnung zueinander, zum Schaltnocken und besonders zum Hub beibehalten. 5.4.7.3 Der Mechanismus, z. B. Nocke und Nockenrolle, muss so gestaltet sein, dass ein Rückwärtsdrehen ohne Beschädigung möglich ist.

Norm	Anforderung
EN 693:2011-11	5.4.5 Positionsschalter (siehe EN 60204-1:1997, 10.1.4) Die Betätigungsmittel des Positionsschalters (siehe 3.16) und der Schalter selbst müssen so gestaltet sein, dass sie nach Einstellung der Schaltstellung ihre ordnungsgemäße Zuordnung zueinander, zum Schaltnocken und insbesondere zum Hub beibehalten.

Funktionsbeschreibung

Der Funktionsbaustein SF_CamMonitoring stellt Funktionalität zum Überwachen der Nocken einer Presse bereit.

Er überwacht die korrekte Abfolge der Nocken und gibt Informationen über die aktuelle Pressenposition wie den oberen Totpunkt (TDC) und die Bewegungsrichtung des Pressenstempels aus.

Activate, Ready

Ist "Activate" FALSE, so ist der Funktionsbaustein inaktiv. Es werden keine Eingänge verarbeitet. Alle Ausgänge sind FALSE. Der Ausgang "DiagCode" ist 16#0000.

Wenn "Activate" auf TRUE geht, wird der Funktionsbaustein initialisiert. Der Ausgang "Ready" wird TRUE. Eine fallende Flanke am Eingang "Activate" versetzt den Baustein wieder in den inaktiven Zustand.

Reset, Error, DiagCode, ResetRequest

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx, wobei xxx den Fehler-Code repräsentiert.

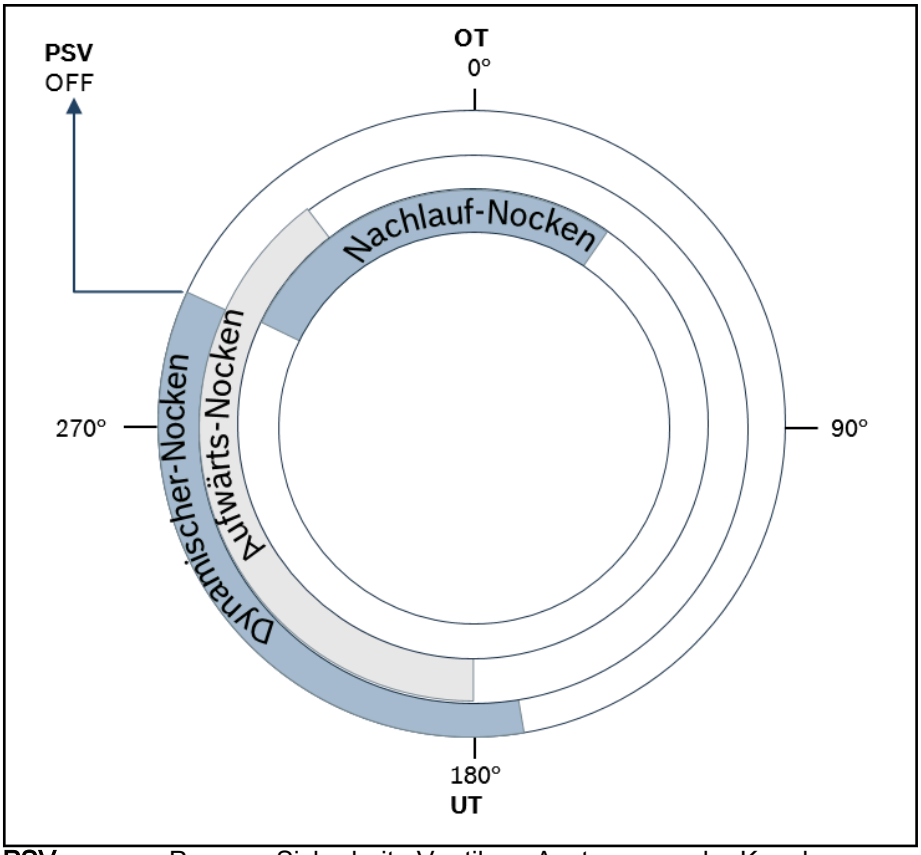
Ein Fehler wird mit einer steigenden Flanke am Eingang "Reset" zurückgesetzt. Ein rücksetzbarer Fehler wird über ein TRUE am Ausgang "ResetRequest" angezeigt. Ist der Fehler behoben, wechselt der Ausgang "Error" bei steigender Flanke am Eingang "Reset" auf FALSE und der Funktionsbaustein wartet auf den nächsten Zyklus.

S_Enable, S_CamMonitoringOK

Der Eingang "S_Enable" steuert, ob eine Überprüfung der Nockenreihenfolge stattfindet. Ist "S_Enable" FALSE, findet keine Überprüfung statt. "S_CamMonitoringOK" ist TRUE. Ist "S_Enable" TRUE, so wird die Reihenfolge der Nocken geprüft. Nur bei korrekter Nockenfolge wird "S_CamMonitoringOK" TRUE gesetzt.

S_UpwardCam, S_OverrunCam

An den Eingang "S_UpwardCam" wird der Nocken für die Aufwärtsbewegung der Presse angeschlossen. Dieser Nocken geht direkt in den Nachlaufnocken über, der den oberen Totpunktbereich abdeckt. Der Nachlaufnocken wird an den Eingang "S_OverrunCam" angeschlossen. Der Aufwärts- und der Nachlaufnocken müssen sich überlappen.



PSV	Pressen-Sicherheits-Ventil zur Ansteuerung der Kupplungs-Bremskombination
OT	oberer Totpunkt
UT	unterer Totpunkt

Abb. 16-118: Beispiel für SF_CamMonitoring, der dynamische Nocken ist optional

S_DynamicCam, NoDynamicCamUsed

Ist der Eingang "NoDynamicCamUsed" TRUE, so wird ein zusätzlicher Nocken aktiv. Dieser dynamische Nocken wird am Eingang S_DynamicCam angeschlossen und wird zur Ansteuerung des Sicherheitsventils verwendet. Der dynamische Nocken muss vor dem Aufwärtsnocken beginnen und innerhalb des Aufwärtsnockens enden (s. auch Abb. 16-118 "Beispiel für SF_CamMonitoring, der dynamische Nocken ist optional" auf Seite 492).

S_BackwardsMove

Befindet sich die Presse Rückwärtslauf, so muss dies über den Eingang "S_BackwardsMove" angezeigt werden. Auch in diesem Modus wird die Reihenfolge der Nocken überwacht.

S_CamMonitoringOK, S_TDC, S_UpwardsMoving, SafetyDemand

Die Ausgänge "S_TDC" und "S_UpwardsMoving" werden aus den Eingangssignalen "S_UpwardCam", "S_OverrunCam" und "S_DynamicCam" gebildet. Die Reihenfolge der Nocken wird überwacht. Werden alle Nocken in der richtigen Abfolge erkannt, so ist "S_CamMonitoringOK" TRUE.

Der Ausgang "SafetyDemand" wird nur während der Initialisierungsphase TRUE, wenn die Presse den oberen Totpunkt, der als Startposition dient, noch nicht erreicht hat.

Zustandsdiagramm

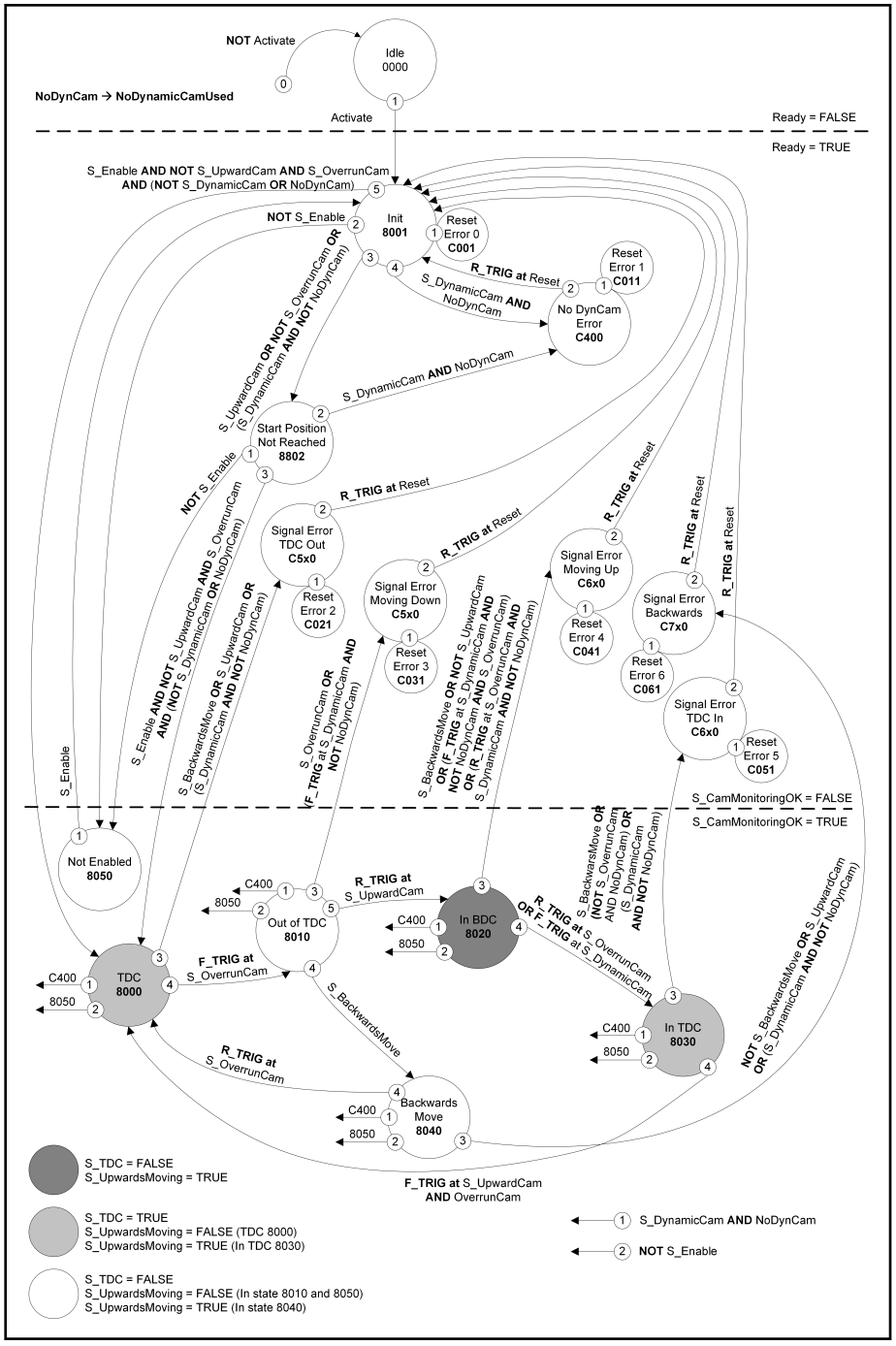


Abb. 16-119: Zustandsdiagramm SF_CamMonitoring

Zeitdiagramme

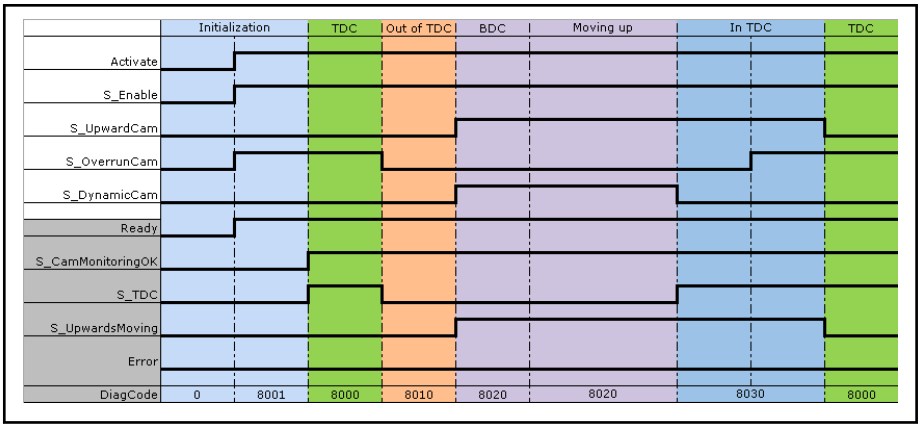


Abb. 16-120: Zeitdiagramm für SF_CamMonitoring mit dynamischen Nocken

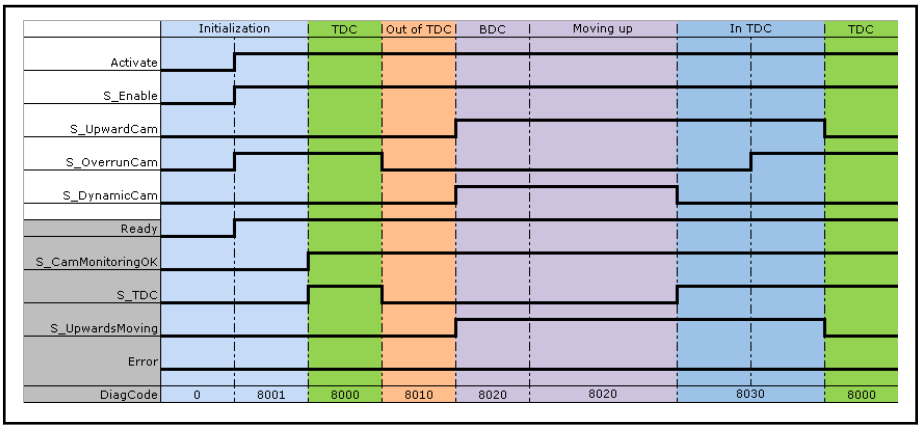


Abb. 16-121: Zeitdiagramm für SF_CamMonitoring mit dynamischen Nocken und Fehler in der Nockenfolge

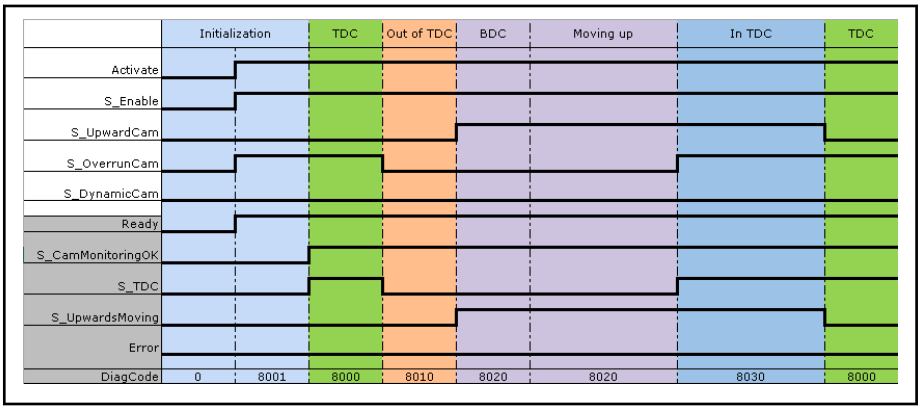


Abb. 16-122: Zeitdiagramm für SF_CamMonitoring ohne dynamischen Nocken

Fehlerbehandlung

Error:

- **TRUE:** Es ist ein Fehler aufgetreten
- **FALSE:** Es ist kein Fehler aufgetreten

DiagCode

DiagCode	Zustandsname	Beschreibung
16#0000	Idle	Der Baustein ist nicht initialisiert. Dieser Wert wird auch am Ausgang DiagCode ausgegeben, wenn "Activate" auf FALSE geht und der Zyklus abgelaufen ist. Ready = FALSE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8000	TDC	Die Presse ist im oberen Totpunktbereich Ready = TRUE S_CamMonitoringOK = TRUE S_TDC = TRUE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8001	Init	Der Funktionsbaustein wurde aktiviert und initialisiert. Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8010	Out of TDC	Die Presse bewegt sich nach unten Ready = TRUE S_CamMonitoringOK = TRUE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#8020	In BDC	Die Presse bewegt sich nach oben Ready = TRUE S_CamMonitoringOK = TRUE S_TDC = FALSE S_UpwardsMoving = TRUE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8030	In TDC	Die Presse befindet sich im oberen Totpunktbereich Ready = TRUE S_CamMonitoringOK = TRUE S_TDC = TRUE S_UpwardsMoving = TRUE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8040	Backwards Move	Die Presse bewegt sich rückwärts Ready = TRUE S_CamMonitoringOK = TRUE S_TDC = FALSE S_UpwardsMoving = TRUE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8050	Not Enabled	Die Funktion ist nicht aktiviert Ready = TRUE S_CamMonitoringOK = TRUE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8802	Start Position not reached	Die Nocken-Signale passen nicht zur Startbedingung Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#C001	Reset Error 0	statischer Reset im Zustand 16#8001 erkannt Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C011	Reset Error 1	statischer Reset im Zustand 16#C400 erkannt Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C021	Reset Error 2	statischer Reset im Zustand 16#C5x0 erkannt Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C031	Reset Error 3	statischer Reset im Zustand 16#C5x0 erkannt Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C041	Reset Error 4	statischer Reset im Zustand 16#C6x0 erkannt Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C051	Reset Error 5	statischer Reset im Zustand 16#C6x0 erkannt Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C061	Reset Error 6	statischer Reset im Zustand 16#C7x0 erkannt Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C400	No DynCam Error	Die Konfiguration für den dynamischen Nocken ist inkorrekt Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#C5x0	Signal Error TDC Out	Falsche Nockenkombination im oberen Totpunkt x=0 wenn S_BackwardsMove=TRUE x=1 wenn S_UpwardCam=TRUE x=2 wenn S_DynamicCam=TRUE und NoDynamicCamUsed=FALSE Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C5x0	Signal Error Moving Down	Flasche Nockenkombination in Abwärtsbewegung x=3 wenn S_OverrunCam=TRUE x=4 wenn fallende Flanke an S_DynamicCam und NoDynamicCamUsed=FALSE Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#C6x0	Signal Error Moving Up	Falsche Nockenkombination in Aufwärtsbewegung x=0 wenn S_BackwardsMode=TRUE x=1 wenn S_UpwardCam=FALSE x=2 wenn fallende Flanke an S_DynamicCam und NoDynamicCamUsed=FALSE und S_OverrunCam=TRUE x=3 wenn steigende Flanke an S_OverrunCam und S_DynamicCam=TRUE und NoDynamicCamUsed=FALSE x=4 wenn steigende Flanke an S_DynamicCam Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C6x0	Signal Error TDC In	Falsche Nockenkombination beim Eintritt in den oberen Totpunkt x=5 wenn S_BackwardsMove=TRUE x=6 wenn S_OverrunCam=FALSE und NoDynamicCamUsed=TRUE x=7 wenn S_DynamicCam=TRUE und NoDynamicCamUsed=FALSE Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#C7x0	Signal Error Backwards	Falsche Nockenkombination in Rückwärtsbewegung x=0 wenn S_BackwardsMode=FALSE x=1 wenn S_UpwardCam=TRUE x=2 wenn S_DynamicCam=TRUE und NoDynamicCamused=FALSE Ready = TRUE S_CamMonitoringOK = FALSE S_TDC = FALSE S_UpwardsMoving = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

Tab. 16-216: Werte des Ausgangs "DiagCode"

16.5.7 SF_CamshaftMonitor

Kurzbeschreibung

Der Funktionsbaustein SF_CamshaftMonitor stellt Funktionalität zum Überwachen einer Nockenwelle bereit. Es muss eine bestimmte Anzahl von Impulsen während einer definierten Zeit geben.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopenPress	2.6.0.2

Tab. 16-217: Bibliothekszuordnung

Schnittstellenbeschreibung

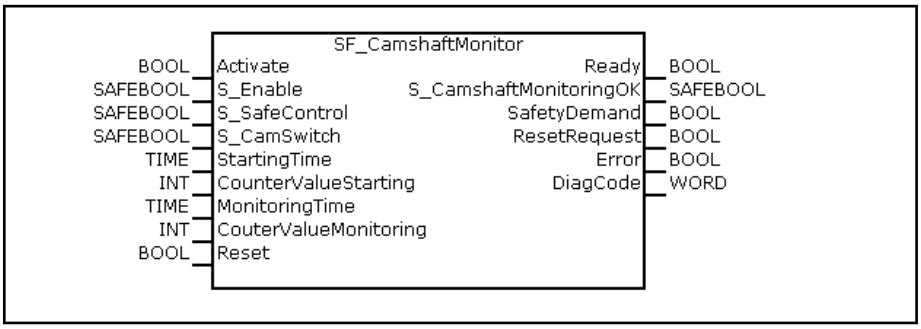


Abb. 16-123: Funktionsbaustein SF_CamshaftMonitor

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv
	S_Enable	SAFEBOOL	Aktiviert die Überwachung FALSE: Keine Überwachung aktiv TRUE: Überwachung aktiv
	S_SafeControl	SAFEBOOL	Eingangsignal, um mitzuteilen, dass sich die Presse bewegt und die Überwachung auszuführen ist. FALSE: Presse steht - keine Überwachung aktiv TRUE: Presse in Bewegung - Überwachung aktiv
	S_CamSwitch	SAFEBOOL	Eingang für den Nocken FALSE: Nocken inaktiv TRUE: Nocken aktiv
	StartingTime	TIME	Startzeit der Maschine. Zeit innerhalb der Wert des Eingang "CounterValueStarting" erreicht werden muss.
	CounterValueStarting	INT	Anzahl der Signaländerungen, die während des Starts der Maschine erreicht werden müssen.
	MonitoringTime	TIME	Überwachungszeit über den zyklischen Betrieb der Maschine. Zeit innerhalb der Wert des Eingang "CounterValueMonitoring" erreicht werden muss.
	CounterValueMonitoring	INT	Anzahl der Signaländerungen, die innerhalb der zyklischen Überwachungszeit (MonitoringTime) erreicht werden müssen.
	Reset	BOOL	Rücksetzen eines Fehlers bei steigender Flanke

E/A-Typ	Name	Datentyp	Beschreibung
VAR_OUTPUT	Ready	BOOL	Funktionsbaustein ist initialisiert und betriebsbereit
	S_CamshaftMonitoringOK	SAFEBOOL	Ausgang für die Überwachung der Nockenwelle FALSE: Die Nockenwellenüberwachung ist nicht korrekt oder Signalfehler TRUE: Die Nockenwellenüberwachung ist korrekt oder "S_Enable" ist FALSE
	SafetyDemand	BOOL	Sicherheitsanforderung. Die für den sicheren Betrieb benötigten Voraussetzungen sind nicht erfüllt. Es liegt jedoch kein Fehler vor. TRUE: Sicherheitsanforderung ist aktiv FALSE: Es wird keine Absicherung benötigt
	ResetRequest	BOOL	Es wurde ein quittierbarer Fehler festgestellt. Vor dem Weiterbetrieb muss dieser mit einer steigenden Flanke am "Reset"-Eingang quittiert werden. TRUE: Reset angefordert FALSE: Reset nicht angefordert
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Werte des Ausgangs DiagCode, Seite 506)

Tab. 16-218: Schnittstelle SF_CamshaftMonitor

Name	Typ	Min	Max	Default	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_Enable	SAFEBOOL			FALSE	kontinuierlich
S_SafeControl	SAFEBOOL			FALSE	kontinuierlich
S_CamSwitch	SAFEBOOL			FALSE	kontinuierlich
StartingTime	TIME	t#0s		t#0s	steigende Flanke an "Activate"
CounterValueStarting	INT	0	40	0	steigende Flanke an "Activate"
MonitoringTime	TIME	t#0s	< StartingTime	t#0s	steigende Flanke an "Activate"
CounterValueMonitoring	INT	0	40	0	steigende Flanke an "Activate"
Reset	BOOL			FALSE	kontinuierlich

Tab. 16-219: Eingangsverhalten SF_CamshaftMonitor

Normativer Verweis

Norm	Anforderung
EN 692:2009-10	5.4.2.4 c Wenn eine Nachlaufüberwachung erforderlich ist, muss diese den folgenden Anforderungen entsprechen: 1. Handbeschickte Pressen, an denen die in 5.4.1.4 aufgelisteten Schutzeinrichtungen eingebaut sind, müssen mit Einrichtungen zur Nachlaufüberwachung versehen sein; wenn die Kurbelwelle über den normalen Anhaltepunkt um einen vom Hersteller angegebenen Wert von maximal 15° und vorzugsweise 10° nachläuft, muss sichergestellt sein, dass sofort ein Stoppsignal ausgelöst wird und die Einleitung eines weiteren Hubes nicht mehr möglich ist; 2. die Wiederaufnahme des weiteren Betriebes der Presse darf nur mit einem besonderen Mittel möglich sein, z. B. durch Werkzeug, Schlüssel oder elektronisches Passwort; 3. wenn die Nocken der Nachlaufüberwachung von einer Nockenwelle angetrieben werden, die indirekt von einer Kurbelwelle angetrieben wird, z. B. durch Duplexkettenantrieb zwischen Nockenwelle und Kurbelwelle, ist der indirekte Antrieb so zu überwachen, dass bei einem Ausfall ein Haltsignal eingeleitet wird und die Auslösung eines neuen Zyklus so lange unmöglich ist, bis der Fehler behoben ist.

Funktionsbeschreibung

Der Funktionsbaustein SF_CamshaftMonitor stellt Funktionalität zum Überwachen einer Nockenwelle bereit. Es muss eine bestimmte Anzahl von Impulsen während einer definierten Zeit geben.

Der Funktionsbaustein kann bis zu 20 Hübe (40 Signaländerungen) verarbeiten. Hierbei wird sowohl die steigende als auch die fallende Flanke gezählt. Bei einem Zustandswechsel werden die Zähler zurückgesetzt.

Werden mehrere Nocken verwendet, müssen mehrere Instanzen des Funktionsbausteins SF_CamshaftMonitor verwendet werden.

Activate, Ready

Ist "Activate" FALSE, so ist der Funktionsbaustein inaktiv. Es werden keine Eingänge verarbeitet. Alle Ausgänge sind FALSE. Der Ausgang "DiagCode" ist 16#0000.

Wenn "Activate" auf TRUE geht, wird der Funktionsbaustein initialisiert. Der Ausgang "Ready" wird TRUE. Eine fallende Flanke am Eingang "Activate" versetzt den Baustein wieder in den inaktiven Zustand.

Reset, Error, DiagCode, ResetRequest

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx, wobei xxx den Fehler-Code repräsentiert.

Ein Fehler wird mit einer steigenden Flanke am Eingang "Reset" zurückgesetzt. Ein rücksetzbarer Fehler wird über ein TRUE am Ausgang "ResetRequest" angezeigt. Ist der Fehler behoben, wechselt der Ausgang "Error" bei steigender Flanke am Eingang "Reset" auf FALSE und der Funktionsbaustein wartet auf den nächsten Zyklus.

S_Enable, S_CamshaftMonitoringOK

Der Eingang "S_Enable" steuert, ob eine Überprüfung der Nocken stattfindet. Ist "S_Enable" FALSE, findet keine Überprüfung statt. "S_CamshaftMonitoringOK" ist TRUE. Ist "S_Enable" TRUE, so wird die Reihenfolge der Nocken geprüft. Nur bei korrekter Nockenfolge wird "S_CamshaftMonitoringOK" TRUE gesetzt.

StartingTime, CounterValueStarting

Die Nockenwelle wird während des Anlaufs der Maschine überwacht. Die zum Hochlauf der Maschine benötigte Zeit muss am Eingang "StartingTime" eingestellt werden. Die in dieser Zeit zu erwartenden Zustandswechsel am noch werden am Eingang "CounterValueStarting" angegeben.

Da der Eingang "CounterValueStarting" maximal 40 Signalwechsel vorsieht, muss gegebenenfalls die Startzeit reduziert werden.

Nach Ablauf der am Eingang StartingTime eingestellten Zeit wird davon ausgegangen, dass sich die Maschine auf ihrer Nenndrehzahl befindet. Die Überwachung erfolgt dann mit den in "MonitoringTime" und "CounterValue-Monitoring" eingestellten Werten.

Sind StartingTime und CounterValueStarting 0, so wird die Überwachung des Hochlaufs übersprungen.

MonitoringTime, CounterValueMonitoring

Die Nockenwelle wird während des Maschinenbetriebs überwacht. Die Zeit für einen Überwachungszyklus muss am Eingang "MonitoringTime" eingestellt werden. Die in dieser Zeit zu erwartenden Zustandswechsel am noch werden am Eingang "CounterValueMonitoring" angegeben. Nach dem Ablauf der "MonitoringTime" beginnt der Zyklus erneut.

Da der Eingang "CounterValueMonitoring" maximal 40 Signalwechsel vorsieht, muss gegebenenfalls der Überwachungszyklus reduziert werden.

☞ Läuft die Maschine mit verschiedenen Geschwindigkeiten, so muss der Funktionsbaustein mit den neuen Werten erneut aktiviert werden.

SafetyDemand

Der Ausgang "SafetyDemand" wird nur während der Initialisierungsphase TRUE, wenn noch kein Signalwechsel am Eingang "S_CamSwitch" erkannt wurde.

Zustandsdiagramm

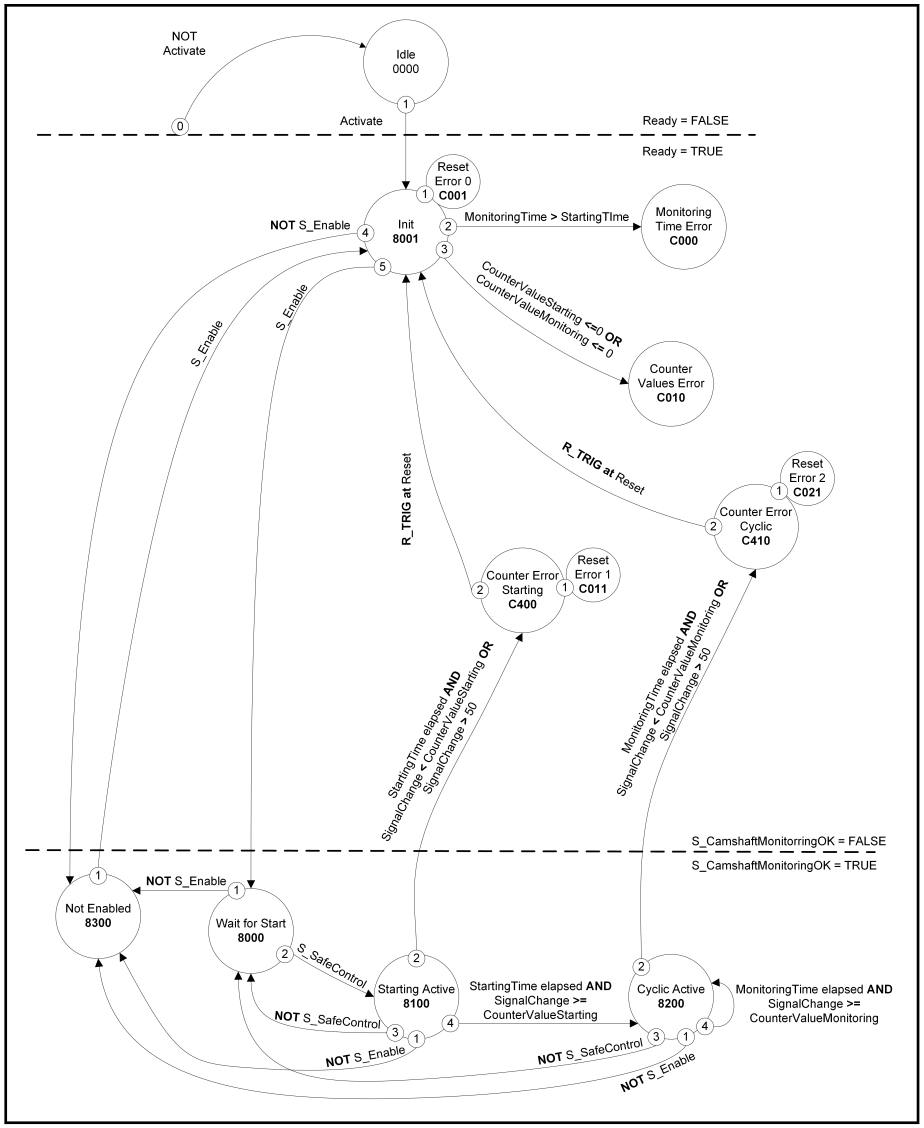


Abb. 16-124: Zustandsdiagramm SF_CamshaftMonitor

Zeitdiagramme

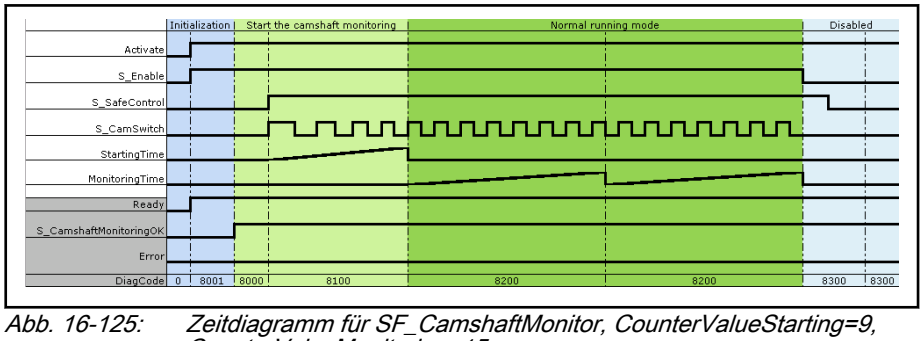


Abb. 16-125: Zeitdiagramm für SF_CamshaftMonitor, CounterValueStarting=9, CounterValueMonitoring=15

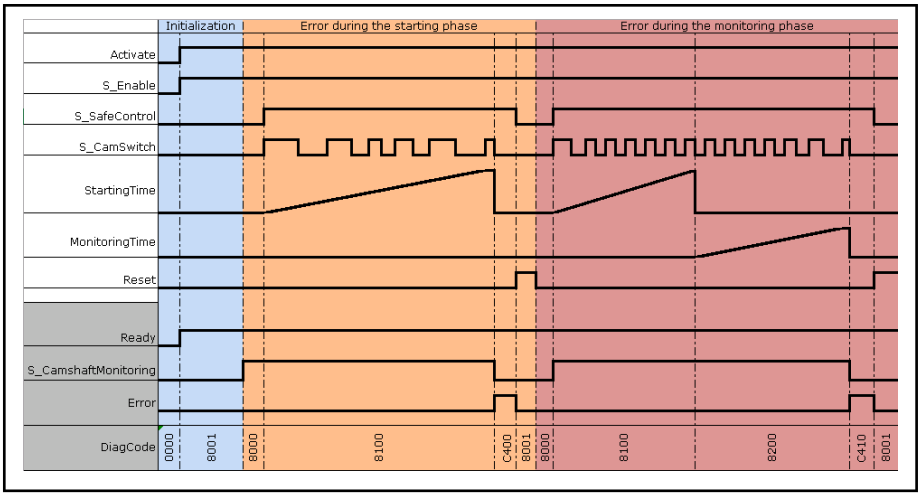


Abb. 16-126: Zeitdiagramm für SF_CamshaftMonitor, CounterValueStarting=14, CounterValueMonitoring=28

Fehlerbehandlung

Error:

- **TRUE:** Es ist ein Fehler aufgetreten
- **FALSE:** Es ist kein Fehler aufgetreten

DiagCode

DiagCode	Zustandsname	Beschreibung
16#0000	Idle	Der Baustein ist nicht initialisiert. Ready = FALSE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8000	Wait for Start	Warten auf Wechsel an S_CamSwitch Ready = TRUE S_CamshaftMonitoringOK = TRUE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8001	Init	Der Funktionsbaustein wurde aktiviert und initialisiert. Ready = TRUE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#8100	Starting Active	Die Maschine befindet sich in der Startphase Ready = TRUE S_CamshaftMonitoringOK = TRUE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8200	Cyclic Active	Die Maschine befindet sich im zyklischen Betrieb Ready = TRUE S_CamshaftMonitoringOK = TRUE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8300	Not Enabled	Die Funktion Nockenwellenüberwachung ist nicht aktiviert Ready = TRUE S_CamshaftMonitoringOK = TRUE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#C000	MonitoringTime Error	MonitoringTime ist größer als StartingTime Ready = TRUE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C001	Reset Error 0	statischer Reset im Zustand 16#8001 erkannt Ready = TRUE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C010	Counter Values Error	Die Werte in CounterValueStarting und CounterValueMonitoring sind ungültig (z.B. kleiner 0) Ready = TRUE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C011	Reset Error 1	statischer Reset im Zustand 16#C400 erkannt Ready = TRUE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C021	Reset Error 2	statischer Reset im Zustand 16#C410 erkannt Ready = TRUE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C400	Counter Error Starting	Die Anzahl der SIgnaländerungen während der Startphase war zu niedrig oder größer als 50 Ready = TRUE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE
16#C410	Counter Error Cyclic	Die Anzahl der SIgnaländerungen während des zyklischen Betriebs war zu niedrig oder größer als 50 Ready = TRUE S_CamshaftMonitoringOK = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE

Tab. 16-220: Werte des Ausgangs "DiagCode"

16.5.8 SF_CycleControl

Kurzbeschreibung Der Funktionsbaustein SF_CycleControl wird für die Kontrolle des zyklischen Modus (1 bis n Zyklen) einer Presse mit ESPE (berührungslos wirkende Schutzeinrichtung) benötigt.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopenPress	2.6.0.0

Tab. 16-221: Bibliothekszuordnung

Schnittstellenbeschreibung

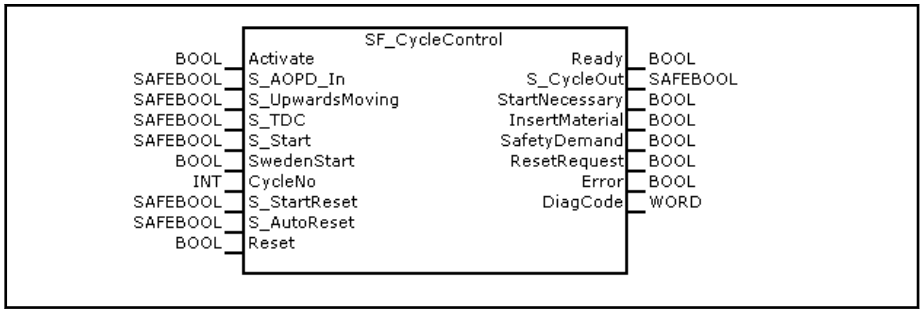


Abb. 16-127: Funktionsbaustein SF_CycleControl

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv. FALSE: Der Funktionsbaustein ist inaktiv.
	S_AOPD_In	SAFEBOOL	OSSD-Signal einer AOPD (Optische Schutzeinrichtung, die berührungslos Objekte im Schutzraum erkennt) FALSE: Der Schutzraum ist verletzt. TRUE: Der Schutzraum ist frei.
	S_UpwardsMoving	SAFEBOOL	Signal "S_UpwardsMoving" von einem vorgeschalteten sicheren Funktionsbaustein "SF_CamMonitoring" Ist der Prozess in beiden Bewegungsrichtungen gefährlich, so muss die Konstante FALSE angeschlossen werden. TRUE: Presse ist in der Aufwärtsbewegung FALSE: Presse ist nicht in der Aufwärtsbewegung
	S_TDC	SAFEBOOL	Signal "S_TDC" von einem vorgeschalteten sicheren Funktionsbaustein "SF_CamMonitoring" TRUE: Presse ist im oberen Totpunktbereich FALSE: Presse ist nicht im oberen Totpunktbereich
	S_Start	SAFEBOOL	Startet bei einer steigenden Flanke die gefahrbringende Bewegung nach einer Unterbrechung am ESPE.
	SwedenStart	BOOL	Konstante für das Startverhalten FALSE: Standard-Start TRUE: Schweden-Start
	CycleNo	INT	Konstante für die Anzahl/Summe der Unterbrechungen am Eingang S_AODP_In, um den nächsten Zyklus zu starten
	S_StartReset	SAFEBOOL	Automatische Quittierung des ersten Fehlers. Typischerweise beim Maschinenstart. Diese Funktion sollte nur aktiviert werden, wenn sichergestellt ist, das hiervon keine Gefahr ausgeht. Es sind zusätzliche Maßnahmen erforderlich um sicherzustellen, dass ein ungewollter Maschinenanlauf ausgeschlossen ist. FALSE: Manuelle Fehlerquittierung wenn das Safety-Programm gestartet wird TRUE: Automatische Fehlerquittierung beim Anlauf des Safety-Programms

E/A-Typ	Name	Datentyp	Beschreibung
	S_AutoReset	SAFEBOOL	Automatische Quittierung aller Fehler. Diese Funktion sollte nur aktiviert werden, wenn sichergestellt ist, das hiervon keine Gefahr ausgeht. Es sind zusätzliche Maßnahmen erforderlich um sicherzustellen, dass ein ungewollter Maschinenanlauf ausgeschlossen ist. FALSE: Manuelle Fehlerquittierung TRUE: Automatische Fehlerquittierung
	Reset	BOOL	Rücksetzen eines Fehlers bei steigender Flanke
VAR_OUTPUT	Ready	BOOL	Funktionsbaustein ist initialisiert und betriebsbereit
	S_CycleOut	SAFEBOOL	Sicherheitsgerichtetes Signal für die Zyklusfreigabe FALSE: Ein neuer Pressenzyklus darf nicht gestartet werden. TRUE: Ein neuer Pressenzyklus darf gestartet werden.
	StartNecessary	BOOL	Diagnosesignal dass anzeigt, dass eine Aktion für den Start des Pressenzyklus notwendig ist. FALSE: Es ist keine Aktion erforderlich TRUE und InsertMaterial=TRUE: Das Lichtgitter an S_AODP_In muss für die am Eingang CycleNo angegebene Anzahl unterbrochen werden. TRUE und InsertMaterial=FALSE: Es ist eine steigende Flanke an S_Start erforderlich.
	InsertMaterial	BOOL	Diagnosesignal dass anzeigt, dass eine Unterbrechung des am Eingang S_AODP_In erforderlich ist. Dies geschieht typischerweise beim Einlegen von neuen Material.
	SafetyDemand	BOOL	Sicherheitsanforderung. Die für den sicheren Betrieb benötigten Voraussetzungen sind nicht erfüllt. Es liegt jedoch kein Fehler vor. TRUE: Sicherheitsanforderung ist aktiv FALSE: Es wird keine Absicherung benötigt
	ResetRequest	BOOL	Es wurde ein quittierbarer Fehler festgestellt. Vor dem Weiterbetrieb muss dieser mit einer steigenden Flanke am "Reset"-Eingang quittiert werden. TRUE: Reset angefordert FALSE: Reset nicht angefordert
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Werte des Ausgangs DiagCode, Seite 517)

Tab. 16-222: Schnittstelle SF_CycleControl

Name	Typ	Min	Max	Default-Wert	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_AOPD_In	SAFEBOOL			FALSE	kontinuierlich
S_UpwardsMoving	SAFEBOOL			FALSE	kontinuierlich

Name	Typ	Min	Max	Default-Wert	Übernahme
S_TDC	SAFEBOOL			FALSE	kontinuierlich
S_Start	SAFEBOOL			FALSE	kontinuierlich
SwedenStart	BOOL			FALSE	steigende Flanke an Activate
CycleNo	INT	0		0	steigende Flanke an Activate
S_StartReset	SAFEBOOL			FALSE	kontinuierlich
S_AutoReset	SAFEBOOL			FALSE	kontinuierlich
Reset	BOOL			FALSE	kontinuierlich

Tab. 16-223: Eingangsverhalten SF_CycleControl

Normativer Verweis

Norm	Anforderung
EN 62046	4.7.4 Wiedereinleitung des zyklischen Betriebs durch die Schutzeinrichtung Bei einigen Anwendungen kann die Schutzeinrichtung zusätzlich zu ihrer Absicherungsfunktion auch zur Wiedereinleitung des Zyklusbetriebs der Maschine verwendet werden. Die folgenden Wiedereinleitungsarten sind in IEC 61496-1 definiert: • einfache Unterbrechung (Eintaktbetrieb), wenn der Maschinenbetrieb durch einmalige Betätigung und Freigabe des Sensorteils wieder eingeleitet wird; • doppelte Unterbrechung (Zweitaktbetrieb) mit Wiedereinleitung des Maschinenbetriebs durch zweimalige, aufeinanderfolgende Betätigung und Freigabe des Sensorteils.
EN ISO 61496-1	Anhang A.8 BWS zum Wiederingangsetzen einer Maschine A.8.1 Allgemeines Falls die BWS zusätzlich zu ihrer Schutzfunktion auch dazu vorgesehen ist, die Maschine wieder in Gang zu setzen, können die nachstehenden Betriebsarten angewendet werden: • Ein Ansprechen und eine Deaktivierung des Sensorteiles löst eine erneute Maschinenbewegung aus, genannt Eintakt; • ein zweifach aufeinanderfolgendes Ansprechen und Deaktivieren des Sensorteiles löst eine erneute Maschinenbewegung aus, genannt Zweitakt. Wo diese zusätzliche Funktion als Teil der BWS zur Verfügung gestellt wird, muss innerhalb der BWS ebenfalls eine Anlaufsperre, wie in Abschnitt A.5 beschrieben, und eine Wiederanlaufsperre, wie in Abschnitt A.6 beschrieben, zur Verfügung stehen.

Funktionsbeschreibung

Der Funktionsbaustein SF_CycleControl wird für die Kontrolle des zyklischen Modus (1 bis n Zyklen) einer Presse mit ESPE (berührungslos wirkende Schutzeinrichtung) benötigt. Er sichert den Gefahrenraum ab, in dem zyklisch aus Prozessgründen eingegriffen werden muss.

In solchen Pressenanwendungen ist der Arbeitsraum der Presse über ein Lichtgitter abgesichert. Nach jedem Pressenhub erfolgt eine bestimmte Anzahl von Unterbrechungen des Lichtgitters um Werkstücke zu entnehmen oder neue Werkstücke einzulegen. Während dieser Zeit darf sich die Presse unter keinen Umständen bewegen.

Activate, Ready

Ist "Activate" FALSE, so ist der Funktionsbaustein inaktiv. Es werden keine Eingänge verarbeitet. Alle Ausgänge sind FALSE. Der Ausgang "DiagCode" ist 16#0000.

Wenn "Activate" auf TRUE geht, wird der Funktionsbaustein initialisiert. Der Ausgang "Ready" wird TRUE. Eine fallende Flanke am Eingang "Activate" versetzt den Baustein wieder in den inaktiven Zustand.

Reset, S_StartReset, S_AutoReset, Error, DiagCode, ResetRequest

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx, wobei xxx den Fehler-Code repräsentiert.

Ein Fehler wird mit einer steigenden Flanke am Eingang "Reset" zurückgesetzt. Ein rücksetzbarer Fehler wird über ein TRUE am Ausgang "ResetRequest" angezeigt. Ist der Fehler behoben, wechselt der Ausgang "Error" bei steigender Flanke am Eingang "Reset" auf FALSE und der Funktionsbaustein wartet auf den nächsten Zyklus.

Soll ein Fehler beim ersten Start des Funktionsbausteins automatisch quittiert werden, so kann der Eingang "S_StartReset" auf TRUE gesetzt werden. Dieser automatische Reset wirkt jedoch nur einmalig.

In einigen Ausnahmefällen ist es sinnvoll, auftretende Fehler automatisch zu quittieren. Hierfür kann der Eingang "S_AutoReset" verwendet werden.

⚠ WARNUNG **Ungewollter Maschinenanlauf**

Werden "S_StartReset" oder "S_AutoReset" auf TRUE gesetzt, kann dies zu ungewolltem Maschinenanlauf führen.

Stellen Sie sicher, dass die Maschine durch andere geeignete Maßnahmen vor automatischem An- und Wiederanlauf geschützt ist.

S_Start, SwedenStart, S_CycleOut

Der Eingang "S_Start" startet einen neuen Zyklus. vorher werden alle sicherheitskritischen Voraussetzungen für den Maschinenanlauf durch den Funktionsbaustein geprüft. Der Eingang "SwedenStart" ermöglicht das automatische Auslösen des Pressenzyklusses nach dem Werkstückwechsel. Für den ersten Anlauf muss aber auch in diesem Modus "S_Start" auf TRUE wechseln.

der Ausgang S_CycleOut gibt den Start des Pressenzyklus frei. Er wird, verknüpft mit anderen Vorbedingungen mit den Eingang "S_CycleStart" des Funktionsbausteins SF_PressControl verbunden (SF_PressControl, Seite 522).

Startbedingungen wenn "SwedenStart"=FALSE:

- Die Presse befindet sich im oberen Totpunkt (Ausgangsposition).
- "S_CycleOut" ist FALSE.
- Das Lichtgitter muss für die am Eingang "CycleNo" angegebene Anzahl unterbrochen werden. Zwischen den Unterbrechungen dürfen nur 30 Sekunden liegen.
- "S_CycleOut" ist immer noch FALSE.
- Der Bediener muss den Eingang "S_Start" innerhalb von 30s von FALSE auf TRUE setzen.

Startbedingung wenn "SwedenStart"=TRUE:

- Die Presse befindet sich im oberen Totpunkt (Ausgangsposition).
- "S_CycleOut" ist FALSE.
- Der Bediener muss den Eingang "S_Start" von FALSE auf TRUE setzen. Hierdurch wird der 30s-Zähler gestartet.

- "S_CycleOut" ist immer noch FALSE.
- Das Lichtgitter muss für die am Eingang "CycleNo" angegebene Anzahl unterbrochen werden. Zwischen den Unterbrechungen dürfen nur 30 Sekunden liegen.
- Ist die am Eingang "CycleNo" angegebene Anzahl erreicht, wird der Pressenzyklus gestartet.

CycleNo, StartNecessary, InsertMaterial, S_AOPD_In

Die BWS (meist ein Lichtgitter), wird am Eingang "S_AOPD_In" angeschlossen. Liegt keine Unterbrechung des Lichtgitters vor, so muss dieser Eingang TRUE sein. Bei einer Unterbrechung des Lichtgitters wird der Eingang FALSE. Der Eingang "CycleNo" bestimmt Die Anzahl der notwendigen Unterbrechungen am Lichtgitter vor dem nächsten Pressenzyklus. Diese Unterbrechungen werden in der Regel vom Werkstückwechsel hervorgerufen. Die Anzahl der notwendigen Eingriffe in dem Gefahrenbereich ist von der Bauart der Maschine bzw. der zu bearbeitenden Werkstücke abhängig. Bei jedem Eingriff muss "S_AOPD_In" für mindestens 100ms unterbrochen werden, um eine Zählung auszulösen.

Wird das Einlegen eines neuen Werkstücks erwartet, so ist der Ausgang "InsertMaterial" TRUE. Gleichzeitig wechselt der Ausgang "StartNecessary" auf TRUE. "StartNecessary" wird ebenfalls TRUE, wenn eine steigende Flanke am Eingang "S_Start" erwartet wird.



Für das korrekte Verschalten des Eingangs "S_AOPD_In" ist der Maschinenhersteller verantwortlich.

S_UpwardsMoving, S_TDC, SafetyDemand

Die Eingänge "S_UpwardsMoving" und "S_TDC" beinhalten Informationen über Die aktuelle Position der Presse und werden typischerweise mit den entsprechenden Aufgängen eines vorgeschalteten SF_CamMonitoring-Funktionsbausteins verbunden (SF_CamMonitoring, Seite 487).

Befindet sich die Presse nach dem Aktiveren des Funktionsbausteins (Eingang "Activate" auf TRUE) nicht im oberen Totpunkt, so ist "SafetyDemand" TRUE. Ein Pressenzyklus kann in diesem Zustand nicht ausgelöst werden. Erst nach Erreichen des oberen Totpunkts wechselt "SafetyDemand" auf FALSE und der normale Betrieb der Presse ist möglich.



Für das korrekte Verschalten der Eingänge "S_UpwardsMoving" und "S_TDC" ist der Maschinenhersteller verantwortlich.

Zustandsdiagramm

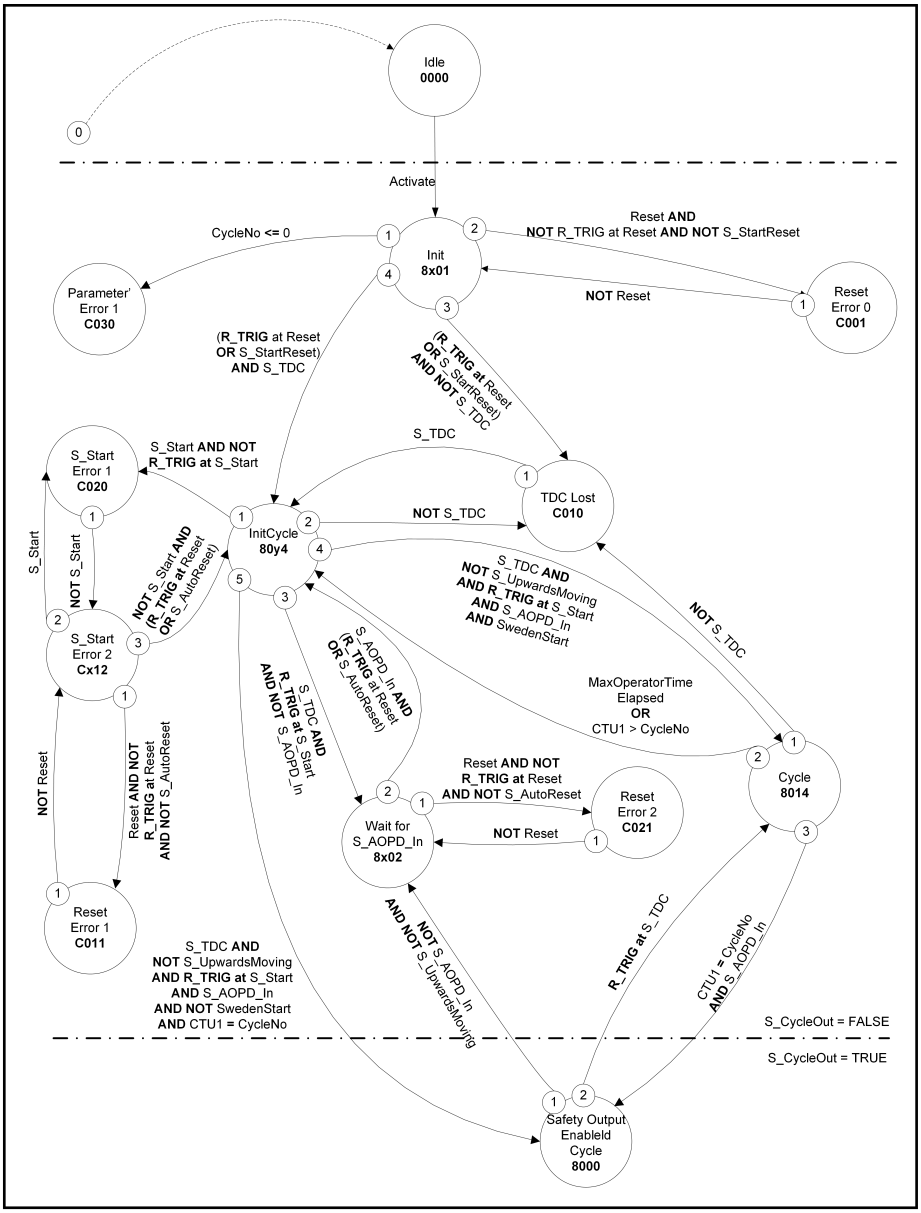
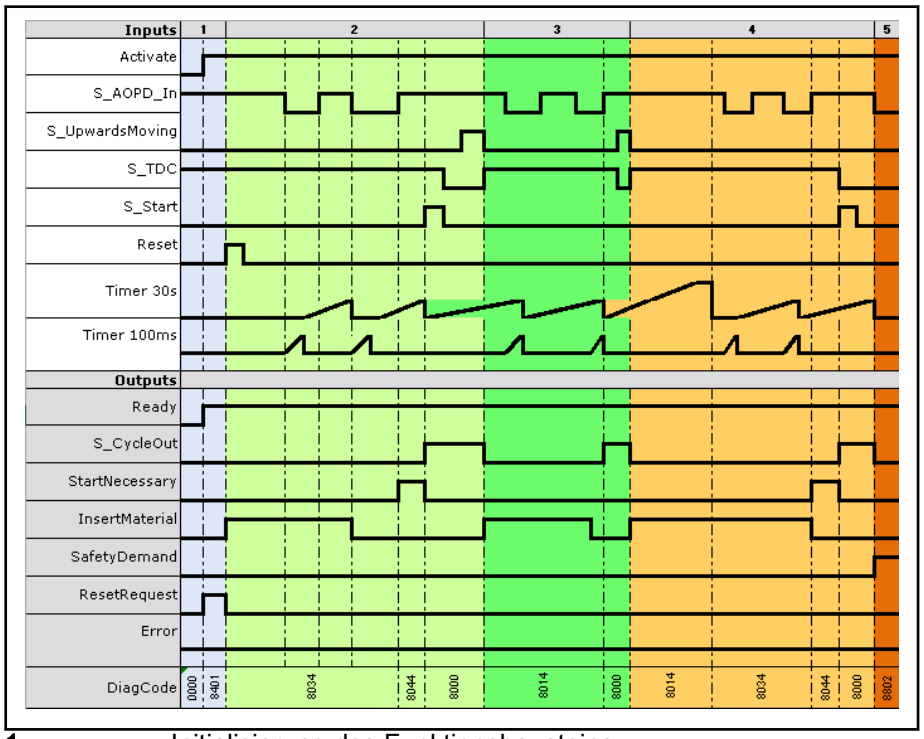


Abb. 16-128: Zustandsdiagramm SF_CycleControl

Zeitdiagramme

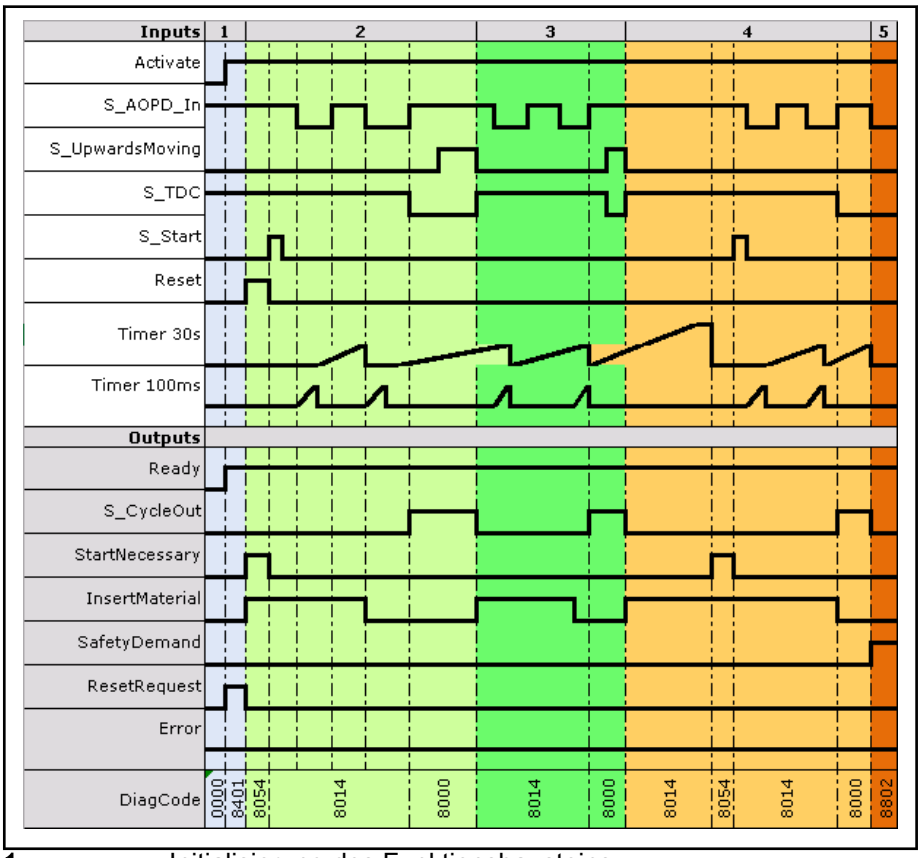
S_StartReset = FALSE; SwedenStart = FALSE; CycleNo = 2:



1 Initialisierung des Funktionsbausteins
2 Initialisierung des ersten Zyklus mit Standardanlauf
3 Normaler Betrieb
4 MaxOperatorTime (30s) abgelaufen; eine Neuinitialisierung ist notwendig
5 S_AOPD_In wurde zwischen den Pressenzyklen verloren

Abb. 16-129: Zeitdiagramm für SF_CycleControl ohne SwedenStart

S_StartReset = FALSE; SwedenStart = TRUE; CycleNo = 2:



1 Initialisierung des Funktionsbausteins
2 Initialisierung des ersten Zyklus mit Swedenstart
3 Normaler Betrieb
4 MaxOperatorTime (30s) abgelaufen; eine Neuinitialisierung ist notwendig
5 S_AOPD_In wurde zwischen den Pressenzyklen verloren

Abb. 16-130: Zeitdiagramm für SF_PressControl in der manuellen Betriebsart

Fehlerbehandlung

Error:

- **TRUE:** Es ist ein Fehler aufgetreten
- **FALSE:** Es ist kein Fehler aufgetreten

DiagCode

DiagCode	Zustandsname	Beschreibung
16#0000	Idle	Der Baustein ist nicht initialisiert. Dieser Wert wird auch am Ausgang DiagCode ausgegeben, wenn "Activate" auf FALSE geht und der Zyklus abgelaufen ist. Ready = FALSE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = FALSE
16#8000	Safety Output Enable Cycle	Der Funktionsbaustein erlaubt Die Zykluskontrolle Ready = TRUE S_CycleOut = TRUE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = FALSE Wenn S_Upwardsmoving = FALSE: Der Zyklenzähler (CTU1) wird zurückgesetzt. S_AOPD_In = FALSE ist eine nicht erlaubte Unterbrechung. Wenn S_Upwardsmoving = TRUE: Jede Signaländerung von FALSE auf TRUE am Eingang S_AOPD_In wird als Zyklus gezählt (CTU1). Das FALSE-Signal jeder Unterbrechung am Eingang S_AOPD_In muss mindestens 100ms andauern. Kürzere FALSE-Impulse sind keine Unterbrechungen. Die Zeit zwischen zwei Eingriffen/ Unterbrechungen oder zwei Zyklusinitialisierungen und dem nachfolgenden benötigten Eingriff darf 30s (T1) nicht überschreiten. Bei Überschreitung der 30s zwischen zwei Unterbrechungen, wird der Zähler (CTU1) auf 0 gesetzt.

DiagCode	Zustandsname	Beschreibung
16#8x01	Init	Die Funktionsbaustein ist aktiviert und initialisiert. Es ist eine steigende Flanke am Eingang Reset notwendig. Zähler CTU1 und Timer T1 werden zurückgesetzt. x=0 wenn S_StartReset=TRUE: Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = FALSE x=4 wenn S_StartReset=FALSE: Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = TRUE Error = FALSE
16#8x02	Wait for S_AOPD_In	Das Schutzfeld wurde oder ist unerwartet unterbrochen. Zähler CTU1 und Timer T1 werden zurückgesetzt. x=4 wenn S_AutoReset=FALSE und S_AOPD_In=TRUE: Ready = TRUE S_CycleOut = FALSE SafetyDemand = TRUE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = FALSE x=8 sonst: Ready = TRUE S_CycleOut = FALSE SafetyDemand = TRUE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = TRUE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#80y4	Init Cycle	Die Zykluskontrolle kann gestartet werden. Im ersten Zyklus wird der Zähler CTU1 beim Erreichen des Zustands 16#80y4 zurückgesetzt. y=3 wenn SwedenStart=FALSE und CTU1<CycleNo: Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = TRUE StartNecessary = FALSE ResetRequest = FALSE Error = FALSE Jede Signaländerung von FALSE auf TRUE am Eingang S_AOPD_In wird als Zyklus gezählt (CTU1). Das FALSE-Signal jeder Unterbrechung am Eingang S_AOPD_In muss mindestens 100ms andauern. Kürzere FALSE-Impulse sind keine Unterbrechungen. Die Zeit zwischen zwei Eingriffen/Unterbrechungen oder zwei Zyklusinitialisierungen und dem nachfolgenden benötigten Eingriff darf 30s (T1) nicht überschreiten. Bei Überschreitung der 30s zwischen zwei Unterbrechungen, wird der Zähler (CTU1) auf 0 gesetzt. y=4 wenn SwedenStart=FALSE und CTU=CycleNo: Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = TRUE ResetRequest = FALSE Error = FALSE DIE Zeit zwischen der letzten Unterbrechung, Die für Die Initialisierung am Eingang S_AOPD_In (CTU1=CycleNo) benötigt wird, und dem Signalwechsel von FALSE auf TRUE am Eingang S_Start darf 30s nicht überschreiten. Wenn sie 30s überschreitet, wird der Zähler CTU1 auf 0 gesetzt (DiagCode 16#8034). In diesem Fall muss Die Initialisierung noch einem durchgeführt werden. Jede Signaländerung von FALSE auf TRUE am Eingang S_AOPD_In wird als Zyklus gezählt (CTU1). Das FALSE-Signal jeder Unterbrechung am Eingang S_AOPD_In muss mindestens 100ms andauern. Kürzere FALSE-Impulse sind keine Unterbrechungen. Wenn CTU1 > CycleNo, wird CTU1 auf 0 gesetzt und der DiagCode 16#8034 ausgegeben. In diesem Fall muss Die Initialisierung erneut durchgeführt werden. y=5 wenn SwedenStart=TRUE: Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = TRUE StartNecessary = TRUE ResetRequest = FALSE Error = FALSE DIE Zeit zwischen der letzten Unterbrechung, Die für Die Initialisierung am Eingang S_AOPD_In (CTU1=CycleNo) benötigt wird, und dem Signalwechsel von FALSE auf TRUE am EINGANG S_Start darf 30s nicht überschreiten.

DiagCode	Zustandsname	Beschreibung
16#8014	Cycle	Die Zykluskontrolle ist aktiv. Die Maschine befindet sich in der Ausgangsposition. Es muss Material eingelegt werden. Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = TRUE StartNecessary = FALSE ResetRequest = FALSE Error = FALSE Jede Signaländerung von FALSE auf TRUE am Eingang S_AOPD_In wird als Zyklus gezählt (CTU1). Das FALSE-Signal jeder Unterbrechung am Eingang S_AOPD_In muss mindestens 100ms andauern. Kürzere FALSE-Impulse sind keine Unterbrechungen. Die Zeit zwischen zwei Eingriffen/Unterbrechungen oder zwei Zyklusinitialisierungen und dem nachfolgenden benötigten Eingriff darf 30s (T1) nicht überschreiten. Bei Überschreitung der 30s zwischen zwei Unterbrechungen, muss Die Zykluskontrolle neu initialisiert werden.
16#C001	Reset Error 0	Statischer Reset wurde im Zustand 16#8001 erkannt. Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = TRUE
16#C011	Reset Error 1	statischer Reset wurde im Zustand 16#C012 erkannt. Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = TRUE
16#C021	Reset Error 2	statischer Reset wurde im Zustand 16#8x02 erkannt. Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#C010	TDC lost	Die Ausgangsposition wurde unerlaubt verlassen. Der Zähler CTU1 und der Timer T2 werden zurückgesetzt. Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = TRUE
16#C020	S_Start Error 1	statischer S_Start wurde im Zustand 16#80y4 erkannt. S_Start ist TRUE. S_Start muss auf FALSE gesetzt werden, um in den Zustand 16#Cx12 zu gelangen. Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx12	S_Start Error 2	statischer S_Start wurde im Zustand 16#8004 erkannt. S_Start ist FALSE. Es wird eine steigende Flanke am Eingang Reset erwartet, um in diesen Zustand in Richtung 16#80y4 zu verlassen. x=0 wenn S_AutoReset=TRUE: Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = TRUE x=4 wenn S_AutoReset=FALSE: Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = TRUE Error = TRUE
16#C030	Parameter Error	Der Wert am Eingang CycleNo ist nicht erlaubt. (<=0). Der Wert muss >0 sein. Ready = TRUE S_CycleOut = FALSE SafetyDemand = FALSE InsertMaterial = FALSE StartNecessary = FALSE ResetRequest = FALSE Error = TRUE

Tab. 16-224: Werte des Ausgangs "DiagCode"

16.5.9 SF_PressControl

Kurzbeschreibung

Der Funktionsbaustein SF_PressControl stellt drei Betriebarten für Pressen zur Verfügung. Er wird immer zusammen mit den Funktionsbausteinen SF_ModeSelector und SF_CamMonitoring verwendet.

Siehe auch:

- SF_ModeSelector, Seite 347
- SF_CamMonitoring, Seite 487

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyPLCopenPress	2.6.0.0

Tab. 16-225: Bibliothekszuordnung

Schnittstellenbeschreibung

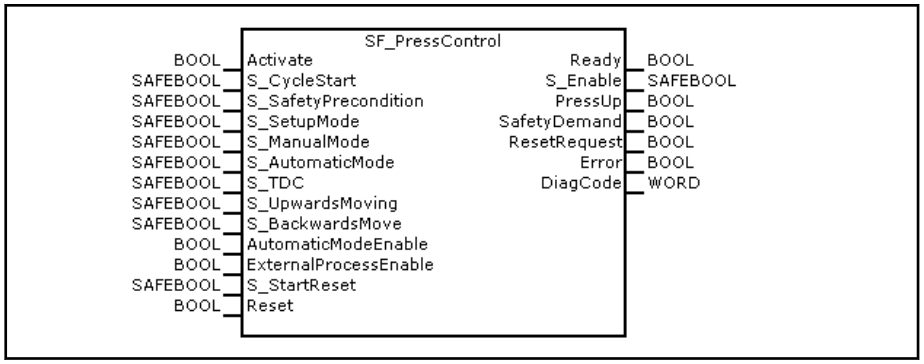


Abb. 16-131: Funktionsbaustein SF_PressControl

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	Activate	BOOL	Bearbeitungsfreigabe TRUE: Der Funktionsbaustein ist aktiv FALSE: Der Funktionsbaustein ist inaktiv
	S_CycleStart	SAFEBOOL	Startet einen Zyklus unabhängig vom Modus bei steigender Flanke FALSE: Keine Anforderung zum Prozessstart TRUE: Anforderung zum Prozessstart
	S_SafetyPrecondition	SAFEBOOL	Safety-Freigabe von vorgeschalteten Sicherheitsfunktionsbausteinen FALSE: Sicherheitsrelevante Voraussetzungen sind nicht erfüllt TRUE: Sicherheitsrelevante Voraussetzungen sind erfüllt
	S_SetupMode	SAFEBOOL	Wählt den Inbetriebnahmemodus TRUE: Inbetriebnahmemodus ist aktiviert FALSE: Inbetriebnahmemodus ist nicht aktiviert
	S_ManualMode	SAFEBOOL	Wählt den manuellen Modus TRUE: manuellen Modus ist aktiviert FALSE: manuellen Modus ist nicht aktiviert
	S_AutomaticMode	SAFEBOOL	Wählt den Automatikmodus TRUE: Automatikmodus ist aktiviert FALSE: Automatikmodus ist nicht aktiviert
	S_TDC	SAFEBOOL	Signal "S_TDC" von einem vorgeschalteten sicheren Funktionsbaustein "SF_CamMonitoring" TRUE: Presse ist im oberen Totpunktbereich FALSE: Presse ist nicht im oberen Totpunktbereich
	S_UpwardsMoving	SAFEBOOL	Signal "S_UpwardsMoving" von einem vorgeschalteten sicheren Funktionsbaustein "SF_CamMonitoring" TRUE: Presse ist in der Aufwärtsbewegung FALSE: Presse ist nicht in der Aufwärtsbewegung

E/A-Typ	Name	Datentyp	Beschreibung
	S_BackwardsMove	SAFEBOOL	Erlaubt die Rückwärtsbewegung des Stempels FALSE: Keine Anforderung für Rückwärtsbewegung TRUE: Anforderung für Rückwärtsbewegung
	AutomaticModeEnable	BOOL	Steuersignal von der Standard-Steuerung für den Automatikmodus beim nächsten oberen Totpunkt FALSE: Automatikmodus beim Beenden des Zyklus am oberen Totpunkt nicht aktivieren TRUE: Automatikmodus ist erlaubt
	ExternalProcessEnable	BOOL	Steuersignal von der Standard-Steuerung als Vorbedingung zum Aktivieren aller Betriebsmodi. Neustart nur bei steigender Flanke an "S_CycleStart". FALSE: Anforderung für "S_Enable" auf FALSE TRUE: Anforderung für "S_Enable" auf TRUE
	S_StartReset	SAFEBOOL	Automatische Quittierung des ersten Fehlers. Typischerweise beim Maschinenstart
	Reset	BOOL	Rücksetzen eines Fehlers bei steigender Flanke
VAR_OUTPUT	Ready	BOOL	Funktionsbaustein ist initialisiert und betriebsbereit
	S_Enable	SAFEBOOL	Sicherheitsgerichtetes Signal für angeschlossene Aktoren FALSE: Alle verbundenen Ventile abschalten TRUE: Alle verbundenen Ventile aktivieren
	PressUp	BOOL	Ausgangssignal für Diagnosen FALSE: Presse nicht in Aufwärtsbewegung TRUE: Presse in Aufwärtsbewegung
	SafetyDemand	BOOL	Sicherheitsanforderung. Die für den sicheren Betrieb benötigten Voraussetzungen sind nicht erfüllt. Es liegt jedoch kein Fehler vor. TRUE: Sicherheitsanforderung ist aktiv FALSE: Es wird keine Absicherung benötigt
	ResetRequest	BOOL	Es wurde ein quittierbarer Fehler festgestellt. Vor dem Weiterbetrieb muss dieser mit einer steigenden Flanke am "Reset"-Eingang quittiert werden. TRUE: Reset angefordert FALSE: Reset nicht angefordert
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Werte des Ausgangs DiagCode, Seite 532)

Tab. 16-226: Schnittstelle SF_PressControl

Name	Typ	Min	Max	Default-Wert	Übernahme
Activate	BOOL			FALSE	kontinuierlich
S_CycleStart	SAFEBOOL			FALSE	kontinuierlich
S_SafetyPrecondition	SAFEBOOL			FALSE	kontinuierlich

Name	Typ	Min	Max	Default-Wert	Übernahme
S_SetupMode	SAFEBOOL			FALSE	kontinuierlich
S_ManualMode	SAFEBOOL			FALSE	kontinuierlich
S_AutomaticMode	SAFEBOOL			FALSE	kontinuierlich
S_TDC	SAFEBOOL			FALSE	kontinuierlich
S_UpwardsMoving	SAFEBOOL			FALSE	kontinuierlich
S_BackwardsMove	SAFEBOOL			FALSE	kontinuierlich
AutomaticModeEnable	BOOL			FALSE	kontinuierlich
ExternalProcessEnable	BOOL			FALSE	kontinuierlich
S_StartReset	SAFEBOOL			FALSE	kontinuierlich
Reset	BOOL			FALSE	kontinuierlich

Tab. 16-227: Eingangsverhalten SF_PressControl

Normativer Verweis

Norm	Anforderung
EN 692:2009-10	3.1.6 Automatik-Betrieb 3.1.8 Einzelhub-Betrieb 3.1.25 Einzelhubfunktion 5.3.16 Bedingungen für die Verriegelung von Motor und Kupplung 5.3.17 Einzelhubsicherung 5.4.1.2 Wiederanlauf Pressenhub 5.4.2.4 Nachlaufüberwachung
EN 693:2011-11	3.4 Einzelhub-Betrieb 3.19 Einzelhubfunktion

Funktionsbeschreibung

Der Funktionsbaustein SF_PressControl stellt drei Betriebarten für Pressen zur Verfügung. Er wird immer zusammen mit den Funktionsbausteinen SF_ModeSelector und SF_CamMonitoring verwendet. Der Funktionsbaustein implementiert die Einzelhubfunktion in den Betriebsarten "Manual" (manueller Betrieb) und "Setup" (Einrichtbetrieb).

Siehe auch:

- SF_ModeSelector, Seite 347
- SF_CamMonitoring, Seite 487

S_SetupMode, S_ManualMode, S_AutomaticMode, AutomaticModeEnable, ExternalProcessEnable

Über die Eingänge S_SetupMode, S_ManualMode und S_AutomaticMode wird die Betriebsart des Funktionsbausteins SF_PressControl ausgewählt. Sie sollten durch einen vorgeschalteten SF_ModeSelector-Funktionsbaustein abgesichert werden. Am Funktionsbaustein SF_ModeSelector kann nur eine der drei Betriebarten gleichzeitig angewählt werden. Um systematische Programmierfehler im Safety-Programm aufzudecken, wird dies durch den Funktionsbaustein geprüft.

Die folgenden Betriebarten werden unterstützt:

- Manuelle Betriebsart mit Einzelhubfunktion:

  Diese Betriebsart wird aktiviert wenn der Eingang "S_ManualMode" TRUE ist.

  Der Eingang "ExternalProcessEnable" muss TRUE sein.

  Der Hub startet bei einer steigenden Flanke am Eingang "S_CycleStart" wenn die Eingänge"S_TDC" und "S_SafetyPrecondition" TRUE sind. Nach dem Start muss "S_CycleStart" bis zur steigenden Flanke an "S_UpwardsMoving" TRUE sein. Andernfalls wird "S_Enable" auf FALSE gesetzt und die Bewegung somit gestoppt.

  Wenn der Pressenzyklus gestartet wurde und "S_CycleStart" zwischen dem oberen Totpunktbereich und dem unteren Totpunkt auf FALSE gesetzt wurde kann im oberen Totpunktbereich wieder gestartet werden. Die Nachlaufkontrolle prüft, dass ein erneutes Schalten nach dem Erreichen des oberen Totpunkts nur im oberen Totpunkt möglich ist. Läuft die Presse zu weit, wird der Neustart verhindert. Ein Nachlauffehler kann nur im Einrichtbetrieb (Setup) zurückgesetzt werden.

  Beim Erreichen des oberen Totpunkts wir der Ausgang "S_Enable" auf FALSE gesetzt. Der Hub wurde vollständig beendet und der Prozess kann erneut gestartet werden.

- Einrichtbetrieb im Uhrzeigersinn:

  Diese Betriebsart wird aktiviert wenn "S_SetupMode" TRUE und "S_BackwardsMove" FALSE ist.

  Der Eingang "ExternalProcessEnable" muss TRUE sein.

  Der Hub kann aus jeder Position heraus gestartet werden. Die Einzelhubsicherung ist aktiv (s. Manuelle Betriebsart). Ein Nachlauffehler wird nur zu Diagnosezwecken erkannt. Die Übernahme bei steigende Flanke an S_UpwardsMoving ist nicht möglich (obwohl "S_CycleStart" TRUE sein muss).

- Einrichtbetrieb entgegen dem Uhrzeigersinn:

  Diese Betriebsart wird aktiviert wenn sowohl "S_SetupMode" als auch "S_BackwardsMove" TRUE sind.

  Der Hub kann aus jeder Position heraus gestartet werden. Die Einzelhubsicherung ist nicht aktiv. Es wird automatisch bei steigender Flanke an "S_TDC" angehalten. Eine Bewegung ist nur mit einer Zweihand-Bedienung möglich.

- Automatikbetrieb:

  Diese Betriebsart wird aktiviert wenn "S_AutomaticMode" TRUE ist.

  Der Eingang "ExternalProcessEnable" muss TRUE sein.

  Der Hub kann aus jeder Position heraus gestartet werden. Die Einzelhubsicherung ist inaktiv. Die Übernahme (steigende Flanke an "S_UpwardsMoving") ist nicht aktiv.

  Der erste Hub startet mit einer steigenden Flanke an "S_CycleStart" (nach der steigenden Flanke kann "S_CycleStart" FALSE sein) wenn "S_SafetyPrecondition" und "AutomaticModeEnable" TRUE sind. Gestoppt wird wenn "S_SafetyPrecondition" oder "AutomaticModeEnable" FALSE wird.

  Wird der Automatikbetrieb mit "AutomaticModeEnable" = FALSE gestoppt, wird der Zyklus weiter ausgeführt, bis der obere Totpunkt erreicht ist.

- Alle Betriebsarten:

Ist "S_SafetyPrecondition" FALSE, wird die Bewegung sofort gestoppt.

Wird "S_SafetyPrecondition" danach wieder auf TRUE gesetzt, kann ein neuer Zyklus mit einer steigenden Flanke am Eingang "Reset" gestartet werden.

Activate, Ready

Ist "Activate" FALSE, so ist der Funktionsbaustein inaktiv. Es werden keine Eingänge verarbeitet. Alle Ausgänge sind FALSE. Der Ausgang "DiagCode" ist 16#0000.

Wenn "Activate" auf TRUE geht, wird der Funktionsbaustein initialisiert. Der Ausgang "Ready" wird TRUE. Eine fallende Flanke am Eingang "Activate" versetzt den Baustein wieder in den inaktiven Zustand.

Reset, S_StartReset, Error, DiagCode, ResetRequest

Tritt während der Abarbeitung des Funktionsbausteins ein Fehler auf, so wird der Ausgang "Error" TRUE. Dieser Zustand ist auch am Ausgang "DiagCode" erkennbar. Er wechselt auf Cxxx, wobei xxx den Fehler-Code repräsentiert.

Ein Fehler wird mit einer steigenden Flanke am Eingang "Reset" zurückgesetzt. Ein rücksetzbarer Fehler wird über ein TRUE am Ausgang "ResetRequest" angezeigt. Ist der Fehler behoben, wechselt der Ausgang "Error" bei steigender Flanke am Eingang "Reset" auf FALSE und der Funktionsbaustein wartet auf den nächsten Zyklus.

Soll ein Fehler beim ersten Start des Funktionsbausteins automatisch quittiert werden, so kann der Eingang "S_StartReset" auf TRUE gesetzt werden. Dieser automatische Reset wirkt jedoch nur einmalig.

⚠ WARNUNG **Ungewollter Maschinenanlauf**

Wird S_StartReset auf TRUE gesetzt, kann dies zu ungewolltem Maschinenanlauf führen.

Stellen Sie sicher, dass die Maschine durch andere geeignete Maßnahmen vor automatischem Anlauf geschützt ist.

S_Enable, S_SafetyPrecondition, SafetyDemand

Der Ausgang "S_Enable" steuert die Freigabe des Pressenzyklus. Ist "S_Enable" FALSE, darf sich die Presse unter keinen Umständen bewegen.

Über den Eingang "S_SafetyPrecondition" wird dem Funktionsbaustein mitgeteilt, dass alle Schutzeinrichtungen aktiviert sind und somit ein gefahrloser Betrieb der Presse möglich ist. Das heißt insbesondere, dass alle Schutztüren geschlossen sind und das Lichtgitter nicht unterbrochen wurde. Ein TRUE am Eingang "S_SafetyPrecondition" ist Voraussetzung für die Bewegungsfreigabe der Presse über den Ausgang "S_Enable".

Benötigt der Prozess Maschinensicherheit, um die Presse gefahrlos bewegen zu können, so wird dies über den Ausgang "SafetyDemand" angezeigt. Wird "S_SafetyPrecondition" FALSE während "SafetyDemand" TRUE ist, so wechselt der Funktionsbaustein in den Zustand 16#8808. Dies wird auch am Ausgang "DiagCode" angezeigt. Der Ausgang "S_Enable" wechselt auf FALSE und führt somit zum Maschinenhalt.

☞ Für das korrekte Verschalten des Ausgangs "S_Enable" ist der Maschinenhersteller verantwortlich.



Für das korrekte Verschalten des Eingangs "S_SafetyPrecondition" ist der Maschinenhersteller verantwortlich.

S_TDC, S_UpwardsMoving, S_BackwardsMove, PressUp

Der Eingang "S_TDC" wird TRUE, wenn sich die Presse im oberen Totpunkt befindet. Er wird normalerweise mit dem Ausgang "S_TDC" des Funktionsbausteins SF_CamMonitoring verbunden.

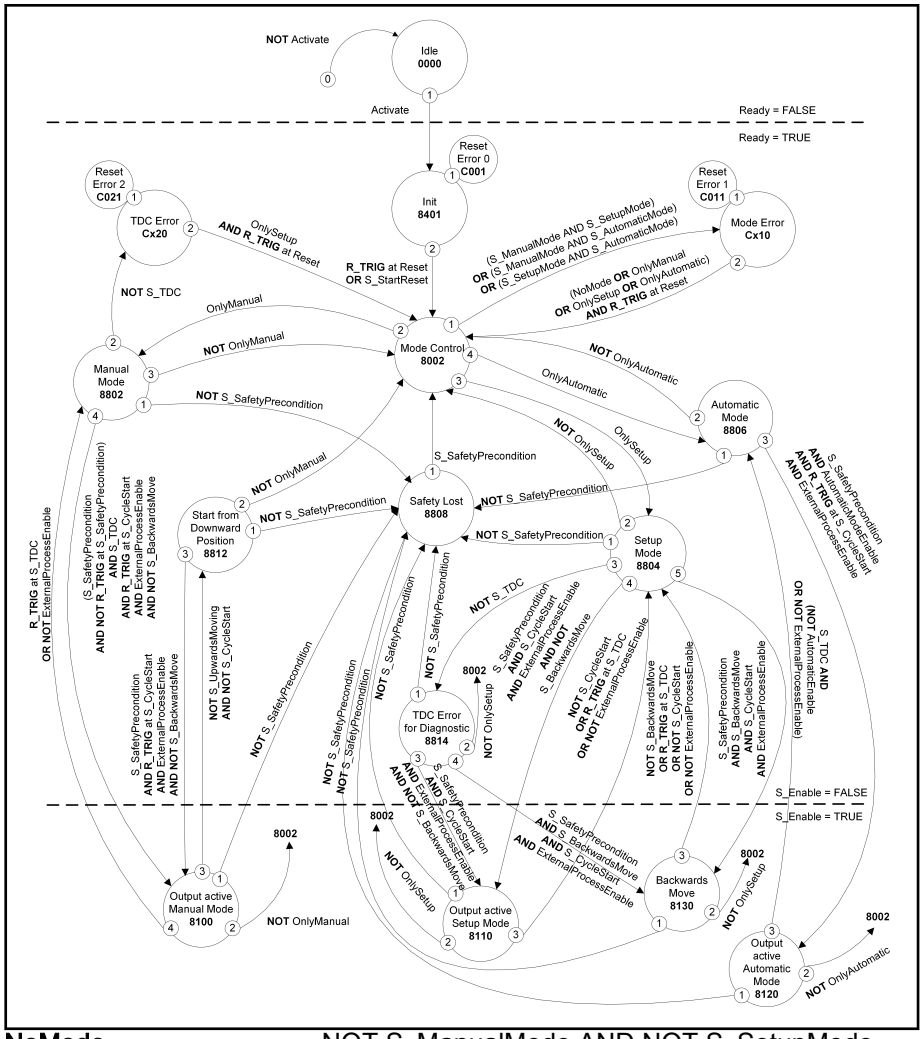
Über den Eingang "S_UpwardsMoving" angezeigt, dass sich die Presse in der Aufwärtsbewegung befindet. Er wird in der Regel mit dem Ausgang "S_UpwardsMoving" eines vorgeschalteten SF_CamMonitoring-Funktionsbausteins verbunden. Von der Aufwärtsbewegung geht in der Regel keine Gefahr aus. Dies wird auch über den Ausgang "PressUp" angezeigt, wenn eine Betriebsart aktiv ist.

Über den Eingang "S_BackwardsMove" wird eine rückwärts laufende Pressenbewegung angezeigt. Der Eingang "S_BackwardsMove" wird mit dem entsprechenden sicheren Steuersignal für die Pressensteuerung verbunden. Eine Rückwärtsbewegung der Presse ist nur im Einrichtbetrieb erlaubt. Dies wird vom Funktionsbaustein überwacht.

Siehe auch:

- SF_ModeSelector, Seite 347
- SF_CamMonitoring, Seite 487

Zustandsdiagramm



NoMode	NOT S_ManualMode AND NOT S_SetupMode AND NOT S_AutomaticMode
Inbetriebnahme	S_SetupMode AND NOT S_ManualMode AND NOT S_AutomaticMode
Manueller Betriebsmodus	S_ManualMode AND NOT S_SetupMode AND NOT S_AutomaticMode
Automatikbetrieb	S_AutomaticMode AND NOT S_ManualMode AND NOT S_SetupMode

Abb. 16-132: Zustandsdiagramm SF_PressControl

Zeitdiagramme Einrichtbetrieb (S_SetupMode = TRUE):

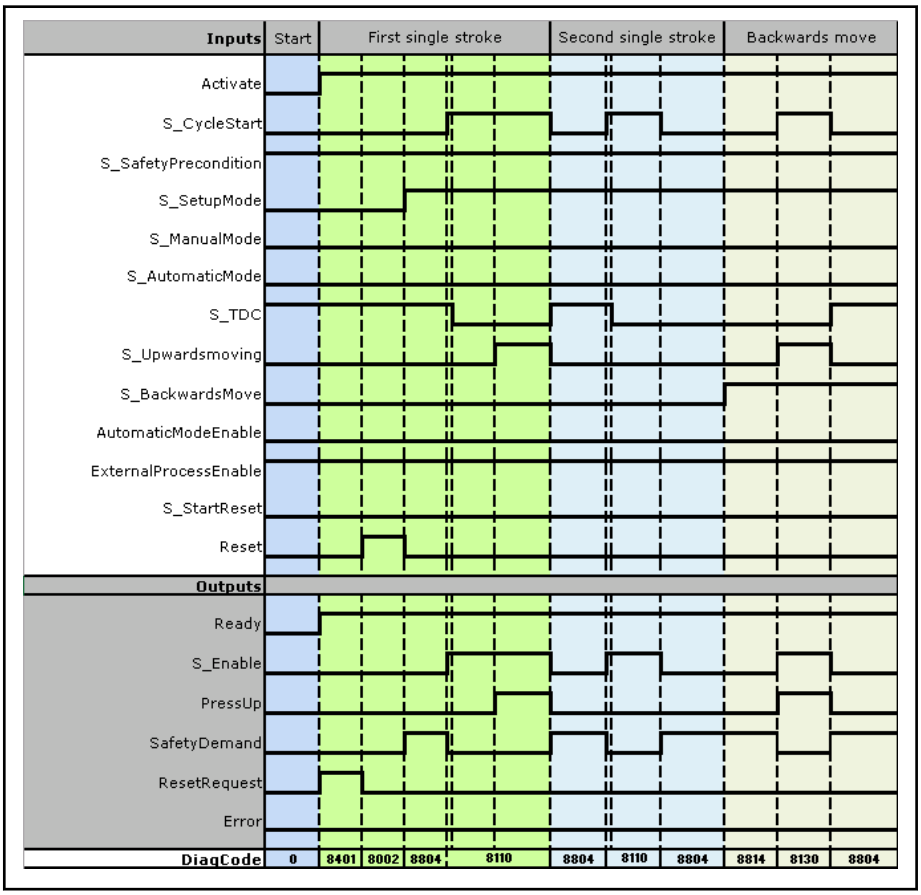


Abb. 16-133: Zeitdiagramm für SF_PressControl im Einrichtbetrieb

Manuelle Betriebsart (S_ManualMode = TRUE):

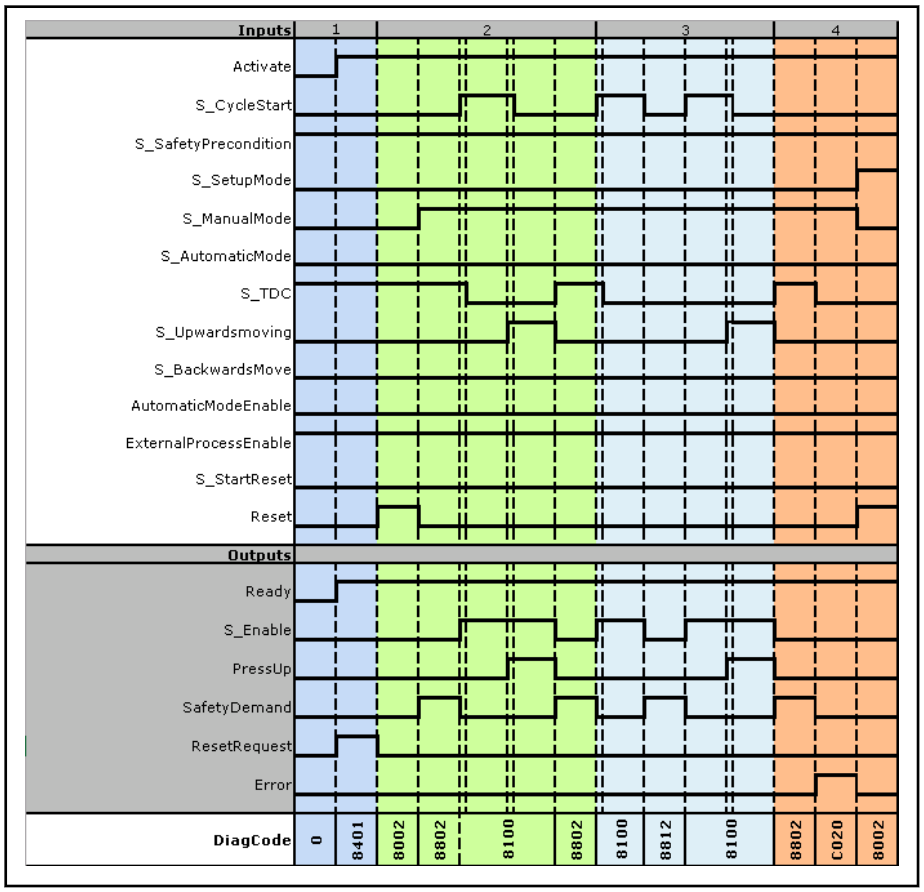


Abb. 16-134: Zeitdiagramm für SF_PressControl in der manuellen Betriebsart

Automatische Betriebsart (S_AutomaticMode = TRUE)

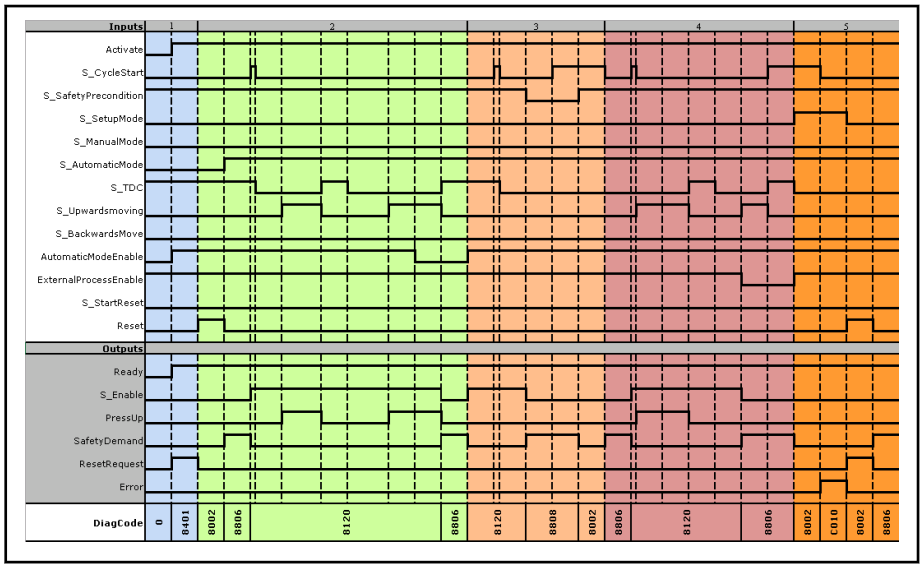


Abb. 16-135: Zeitdiagramm für SF_PressControl in der automatischen Betriebsart

Fehlerbehandlung

Error:

- **TRUE:** Es ist ein Fehler aufgetreten
- **FALSE:** Es ist kein Fehler aufgetreten

DiagCode

DiagCode	Zustandsname	Beschreibung
16#0000	Idle	Der Baustein ist nicht initialisiert. Dieser Wert wird auch am Ausgang DiagCode ausgegeben, wenn "Activate" auf FALSE geht und der Zyklus abgelaufen ist. Ready = FALSE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8401	Init	Der Funktionsbaustein wurde aktiviert und initialisiert. Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = FALSE
16#8002	Mode Control	Die gewählte Betriebsart wird ausgewertet. Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8802	Manual Mode	Die Manuelle Betriebsart wurde angewählt. Der Funktionsbaustein wartet auf den Start des Zyklus. Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8804	Setup Mode	Der Einrichtbetrieb wurde angewählt. Der Funktionsbaustein wartet auf den Start des Zyklus. Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#8806	Automatic Mode	Der Automatikbetrieb wurde angewählt. Der Funktionsbaustein wartet auf den Start des Zyklus. Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8808	SafetyLost	S_SafetyPrecondition ist nicht TRUE. Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8812	Start from Downward Position	Die Presse hat den oberen Totpunkt überschritten und das S_CycleStart-Signal ging verloren, bevor die Presse in die Aufwärtsbewegung ging. Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8814	TDC Error for Diagnostic	Nachdem TDC im Pressenzyklus nicht aktiv war (Nachlauf). Wird dies im Einrichtbetrieb nicht als Fehler, sondern als Diagnose angezeigt. Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = TRUE ResetRequest = FALSE Error = FALSE
16#8100	Output active Manual Mode	Der Pressenzyklus wurde im manuellen Betriebsmodus gestartet. Ready = TRUE S_Enable = TRUE PressUp = FALSE ⇒ TRUE wenn S_UpwardsMoving SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE

DiagCode	Zustandsname	Beschreibung
16#8110	Output active Setup Mode	Der Pressenzyklus wurde im Einrichtbetrieb gestartet. Ready = TRUE S_Enable = TRUE PressUp = FALSE ⇒ TRUE wenn S_UpwardsMoving SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8120	Output active Setup Mode	Der Pressenzyklus wurde im Automatikbetrieb gestartet. Ready = TRUE S_Enable = TRUE PressUp = FALSE ⇒ TRUE wenn S_UpwardsMoving SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#8130	Backwards Move	Der Pressenzyklus wurde im Einrichtbetrieb rückwärts gestartet. Ready = TRUE S_Enable = TRUE PressUp = FALSE ⇒ TRUE wenn NOT S_UpwardsMoving SafetyDemand = FALSE ResetRequest = FALSE Error = FALSE
16#C001	Reset Error 0	Statischer Reset wurde im Zustand 8401 erkannt Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx10	Mode Error	Es wurde mehr als eine Betriebsart angewählt. x=4 (16#C410) wenn NoMode OR OnlyManual OR OnlySetup OR OnlyAutomatic: Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = TRUE Error = TRUE x=0 (16#C010) sonst: Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C011	Reset Error 1	statischer Reset wurde im Zustand Cx10 erkannt Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

DiagCode	Zustandsname	Beschreibung
16#Cx20		Nach dem Pressenzyklus war TDC nicht aktiv (Nachlauf). x=4 (16#C420) S_SetupMode=TRUE: Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = TRUE x=0 (16#C020) sonst: Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE
16#C021	Reset Error 2	statischer Reset wurde im Zustand Cx20 erkannt Ready = TRUE S_Enable = FALSE PressUp = FALSE SafetyDemand = FALSE ResetRequest = FALSE Error = TRUE

Tab. 16-228: Werte des Ausgangs "DiagCode"

16.6 SafetyDriveMotion.library

Die Funktionsbausteine der Bibliothek SafetyDriveMotion ermöglichen die Ansteuerung von sicheren Bewegungsbetriebsarten von Rexroth IndraDrives. Sie sind so implementiert, dass der Anwender kein Wissen über die ausgewählten Safe Motion-Profile oder Assemblies und deren Bitbelegung benötigt.

Anwendungsgebiete und Funktionsbausteine von SafetyDriveMotion

Anwendungsgebiet (Ordner in der Bibliothek)	
Funktionsbausteine	**Beschreibung**
SF_MotionEmergencyStop, Seite 539	steuert und überwacht die Emergency-Stop-Funktionalität des Antriebs
SF_MotionSafeStop, Seite 545	steuert und überwacht die SafeStop 1 (SS1) und SafeStop 2 (SS2) Funktionalität des Antriebs
SF_SMMSelector, Seite 552	fordert eine von acht möglichen im Antrieb konfigurierten sicheren Bewegungsbetriebsarten an
SF_AxisGroup, Seite 567	definiert eine Achsgruppe

Tab. 16-229: SafetyDriveMotion-Bibliothek

Durch die Verwendung der Funktionsbausteine wird das Control-Wort der entsprechenden Achse überschrieben.

Die Bits im entsprechenden Assembly des Antriebs haben dann keine Wirkung mehr.

Um die sicheren Bewegungsbetriebsarten über die bereitgestellten Funktionsbausteine ansprechen zu können, muss mindestens der Funktionsbaustein SF_MotionEmergencyStop mit der entsprechenden AxisID verwendet werden. Die anderen beiden Funktionsbausteine sind optional.

AxisID

Der anzusteuernde Antrieb wird über die SDID der CSos-Verbindung festgelegt. Für Achsgruppen wird mittels dem Funktionsbaustein "SF_AxisGroup" eine AxisGroupID erzeugt. Die SDID, oder die AxisGroupID dient als AxisID.

Mehrere Funktionsbausteine können sich auf eine AxisID verbinden. Es muss jedoch sichergestellt werden, dass pro Achse von jedem Funktionsbaustein nur eine Instanz vorhanden ist.

Prioritäten der sicheren Betriebsarten

Die Anwahl der sicheren Bewegungsbetriebsart des Antriebs unterliegt einer Priorisierung.

Werden in mehreren Funktionsbausteinen einer Achse eine sichere Betriebsart angewählt, so wird die sichere Betriebsart mit der höchsten Priorität aktiviert.

Priorität	sichere Betriebsart	Beschreibung
1	SMES	"Safe Motion Emergency Stop" Nothalt, Abschaltung der Leistung
2	SMST1, SMST2	"Safe Motion Safe Stop 1" "Safe Motion Safe Stop 2" Sicherer Stillstand mit (Safe Stop 2) und ohne Leistung (Safe Stop 1)
3	SMM1 .. SMM8	"Safe Motion Mode" Parametrierbare sichere Betriebsart. Dies kann z.B. SLS (Safe Limited Speed) oder SDI (Safe Direction) sein.

Tab. 16-230: Prioritäten der sicheren Betriebsarten

Unterstützte CSos-Profile

Profil	Eingangs-Assembly	Ausgangs-Assembly
1 Byte Bit-SMM	778	782
1 Byte Binär-SMM	779	783
2 Byte Bit-SMM	780	784
2 Byte Binär-SMM	781	785
2 Byte Binär-SMM, Safe-Homing, Bin-SCA	787	788
2 Byte Binär-SMM, Safe-Homing, Funktionale Eingänge	787	790

Profil	Eingangs-Assembly	Ausgangs-Assembly
2 Byte Bit-SMM, SafeHoming, Bin-SCA	786	789
2 Byte Bit-SMM, SafeHoming	786	784

Tab. 16-231: Unterstützte Profile

⚠ GEFAHR **Gefahrbringende Bewegung durch Änderung der Bitbelegung der Assemblies im Antrieb**

Die Belegung der Assemblies darf im Antrieb nicht geändert werden. Insbesondere darf kein Bit eines Assemblies durch die antriebsintegrierte Verknüpfungslogik beeinflusst werden.

Der Parameter P-0-3340.0.1 muss den Wert $FFFF_{HEX}$ bzw. 65335 haben.

Normativer Verweis

Norm	Anforderung
DIN EN 61800-5-2:2008	4.2 Sicherheitsfunktionen
DIN EN 60204-1/ A1:2009	9.2.1 Kategorien der Stopp-Funktionen Stopp-Kategorie 0: Stillsetzen durch sofortiges Unterbrechen der Energiezufuhr zu den Maschinen- Antriebselementen (d. h. ein ungesteuertes Stillsetzen, siehe 3.65); Stopp-Kategorie 1: ein gesteuertes Stillsetzen (siehe 3.15), bei dem die Energiezufuhr zu den Maschinen-Antriebselementen beibehalten wird, um das Stillsetzen zu erzielen. Die Energiezufuhr wird erst dann unterbrochen, wenn der Stillstand erreicht ist; Stopp-Kategorie 1b: ein gesteuertes Stillsetzen (siehe 3.15), bei dem die Energiezufuhr zu den Maschinen-Antriebselementen beibehalten wird, um das Stillsetzen zu erzielen. Die Aufrechterhaltung der Stoppbedingung wird überwacht, und bei Erkennung eines Ausfalls wird die Energiezufuhr unterbrochen, ohne dabei eine gefahrbringende Situation zu erzeugen. Stopp-Kategorie 2: ein gesteuertes Stillsetzen, bei dem die Ene Antriebselementen erhalten bleibt. 9.2.2.9 Zustimmeinrichtung Eine Zustimmeinrichtung (siehe auch 10.9) ist eine manuell aktivierte Verriegelungsfunktion der Steuerung, die: a) wenn sie betätigt wird, es erlaubt, mit einer separaten Startsteuerung einen Maschinenbetrieb einzuleiten; und b) wenn sie nicht betätigt wird; - eine Stoppfunktion einleitet, und - die Einleitung des Maschinenbetriebs verhindert.

DIN EN ISO 13849-1:2016	5.2.2 Manuelle Rückstellungsfunktion
DIN EN ISO 12100:2011	6.2.11.2 Ingangsetzung einer internen Energiequelle/Einschalten einer äußeren Energiequelle 6.2.11.4 Wiederanlauf nach Energieausfall

16.6.1 SF_MotionEmergencyStop

Kurzbeschreibung

Der Funktionsbaustein SF_MotionEmergencyStop steuert und überwacht die Emergency-Stop-Funktionalität eines oder mehrerer Antriebe (bei Verwendung von Achsgruppen).

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyDriveMotion	2.8.0.0

Tab. 16-232: Bibliothekszuordnung

Schnittstellenbeschreibung

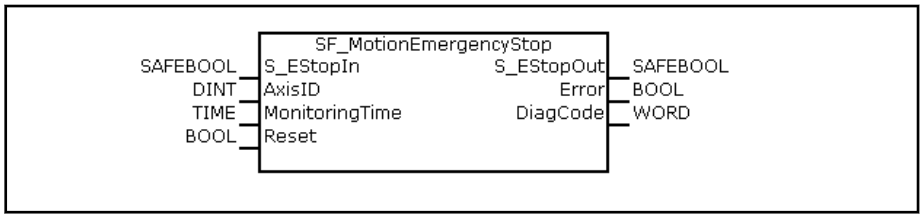


Abb. 16-136: Funktionsbaustein SF_MotionEmergencyStop

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	S_EStopIn	SAFEBOOL	Anforderung Emergency Stop (low aktiv) FALSE: Anforderung einer Sicherheitsfunktion (z.B. NOT-Aus wurde betätigt). TRUE: keine Anforderung einer Sicherheitsfunktion (z.B. NOT-Aus wurde nicht betätigt).
	AxisID	DINT	SDID des zugehörigen Antriebs oder AxisGroupID einer Achsgruppe
	MonitoringTime	TIME	Zeitspanne, in der der Antrieb den sicheren Zustand angenommen haben muss
	Reset	BOOL	Zurücksetzen anstehende Fehler und Statusmeldungen über den Diagnose-Code, wenn die Fehlerursache beseitigt wurde.
VAR_OUTPUT	S_EStopOut	SAFEBOOL	Ausgang für die Sicherheitsanforderung. FALSE: Der Antrieb bzw. die Achsgruppe ist nicht in Emergency Stop TRUE: Der Antrieb bzw. die Achsgruppe ist in Emergency Stop
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Zustand des Funktionsbausteins bzw. Diagnose-Code (siehe Zustandsbeschreibung, Seite 543)

Tab. 16-233: Schnittstelle SF_MotionEmergencyStop

Name	Typ	Min	Max	Default-Wert	Übernahme
S_EStopIn	SAFEBOOL			FALSE	kontinuierlich
AxisID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
MonitoringTime	TIME	t#0s		T#0s	Bei erstem Aufruf des Bausteins
Reset	SAFEBOOL			FALSE	kontinuierlich

Tab. 16-234: Eingangsverhalten SF_MotionEmergencyStop

Normativer Verweis

Norm	Anforderung
DIN EN 61800-5-2:2008	4.2 Sicherheitsfunktionen

Funktionsbeschreibung

Der Funktionsbaustein SF_MotionEmergencyStop schaltet einen oder mehrere Antriebe bei einem FALSE am Eingang "S_EStopIn" in die sichere Betriebsart SMES (Safe Motion Emergency Stop).

Nach Auslösen der Funktion überwacht der Funktionsbaustein, ob der Zustand SMES in allen angesteuerten Antrieben innerhalb der am Eingang "MonitoringTime" angegebenen Zeit tatsächlich erreicht wurde und während der gesamten Anforderungszeit aktiv bleibt.

Der Funktionsbaustein SF_MotionEmergencyStop wird in der Regel zusammen mit den Funktionsbausteinen SF_MotionSafeStop und SF_SMMSelector betrieben.

Der Funktionsbaustein SF_MotionEmergencyStop muss immer vorhanden und aktiviert sein, wenn die sichere Bewegungsbetriebsart über einen der PLCopenMotion-Funktionsbausteine angesteuert werden soll.

Siehe auch:

- SF_MotionSafeStop, Seite 545
- SF_SMMSelector, Seite 552



Die Ansteuerung der sicheren Bewegungsbetriebsart SMES hat Priorität (Safe Motion Emergency Stop).

Ist der Eingang "S_EStopIn" FALSE, so wird immer SMES angesteuert, egal was an den anderen PLCopenMotion-Funktionsbausteinen kommandiert wird (siehe auch Prioritäten der sicheren Betriebsarten, Seite 537).

Reset

Der Eingang "Reset" quittiert aufgetretene Fehler. Der Funktionsbaustein bleibt so lange im Fehlerzustand, bis die Ursache des Fehlers beseitigt und der Fehler durch eine steigende Flanke am Eingang "Reset" quittiert wurde.

S_EStopIn, S_EStopOut

Der low-aktive Eingang "S_EStopIn" wählt die sichere Antriebsbetriebsart SMES (Safe Motion Emergency Stop) an. In dieser Betriebsart wird überwacht, dass die Leistung des Antriebs abgeschaltet ist. Ist "S_EStopIn" FALSE, so wird die sichere Betriebsart SMES angewählt.

Die Bestätigung, dass der Antrieb sich tatsächlich in SMES befindet, wird am Ausgang "S_EStopOut" angezeigt. Ist "S_EStopOut" TRUE, befindet sich der Antrieb in SMES.

AxisID

Am Eingang AxisID wird die SDID des anzusteuernden Antriebs angegeben. Sie bildet die logische Adresse aus Sicht der SafeLogic auf einen realen Antrieb ab. Anstelle der SDID einer Achse kann auch eine zuvor mit dem Funktionsbaustein Kap. 16.6.4 "SF_AxisGroup" auf Seite 567 definierte AxisGroupID angelegt werden. Hierdurch werden alle Achsen einer Achsgruppe angesteuert.



Eine einmal am Eingang AxisID angelegte SDID oder AxisGroupID darf im weiteren Programmverlauf nicht mehr geändert werden!

MonitoringTime

Der Eingang "MonitoringTime" definiert die maximale Zeit zwischen der Kommandierung einer sicheren Bewegungsbetriebsart in den Antrieben und der Bestätigung der Anwahl von den Antrieben. Wird die angeforderte sichere Betriebsart nicht innerhalb der von "MonitoringTime" definierten Zeitspanne erreicht, so geht der Funktionsbaustein in den Fehlerzustand.

Der Eingang "MonitoringTime" wird normalerweise auf die für die zum Erreichen des sicheren Zustands maximal erlaubte Zeit gesetzt. Diese ergibt sich aus den Anforderungen an die Sicherheitsfunktion.

Wird die "MonitoringTime" auf T#0s gesetzt, so findet keine Überwachung statt.



In der SafeMotion des Antriebs wird auch eine maximale Überführzeit definiert. Diese wird unabhängig von der am Eingang "MonitoringTime" eingestellten Zeitspanne vom Antrieb selbständig überwacht.

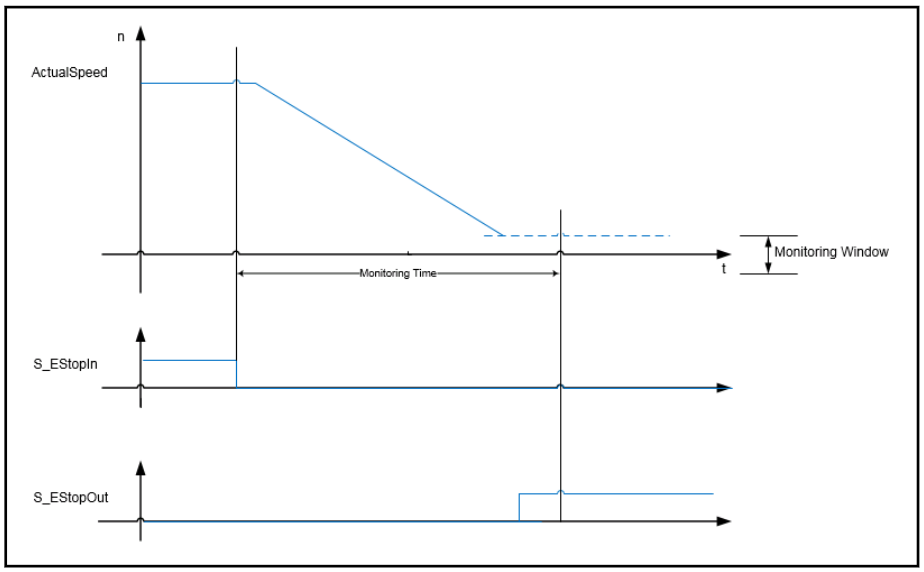


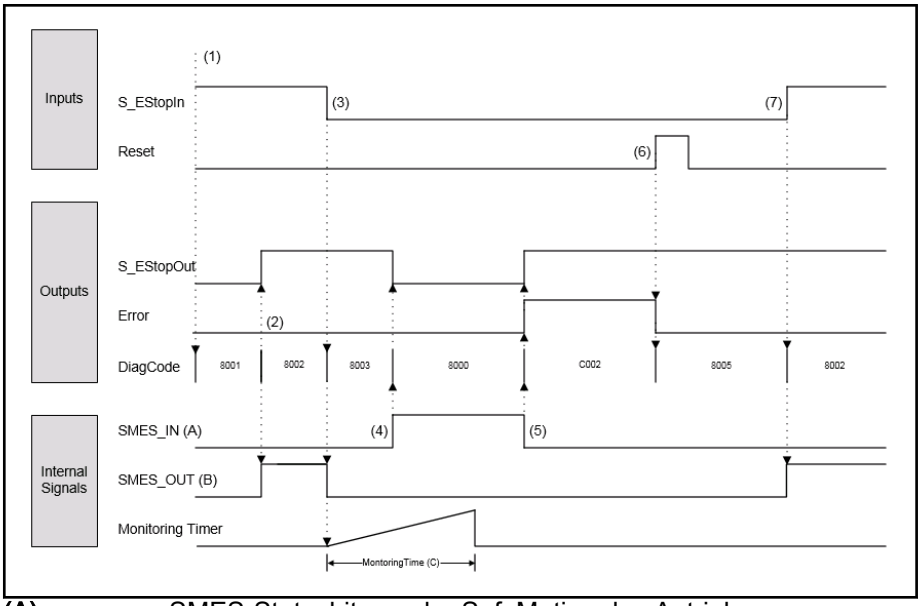
Abb. 16-137: SF_MotionEmergencyStop Prinzipieller Ablauf

Zeitgenauigkeit

Die zeitliche Genauigkeit der am Eingang "MonitoringTime" angelegten Überwachungszeit wird in Zeitmessungen erläutert.

Genauigkeit von Zeitmessungen, Seite 229

Zeitdiagram



(A) SMES-Statusbit von der SafeMotion des Antriebs

(B) SMES-Steuerbit an die SafeMotion des Antriebs

(C) Entspricht der Zeit am Eingang "MonitoringTime"

(1) Nach dem Start der Applikation wird der Baustein initialisiert (Zustand 8001).

(2) Nach erfolgreicher Initialisierung wechselt der Baustein in den Zustand 8002 und das Eingangssignal "S_EStopIn" wird in das Steuerwort des Antriebs geschrieben.

(3) Durch eine Abwärtsflanke an "S_EStopIn" wechselt der Baustein in den Zustand 8003 "Wait for Confirmation". Gleichzeitig wird der neue Zustand in das Steuerwort des Antriebs geschrieben und die "MonitoringTime" gestartet, wenn diese größer T#0s ist.

(4) Durch die Rückmeldung SMES des Antriebs wechselt der Baustein in den Zustand 8000 "Drive in Emergency Stop" und das Ausgangssignal "S_EStopOut" wechselt auf TRUE.

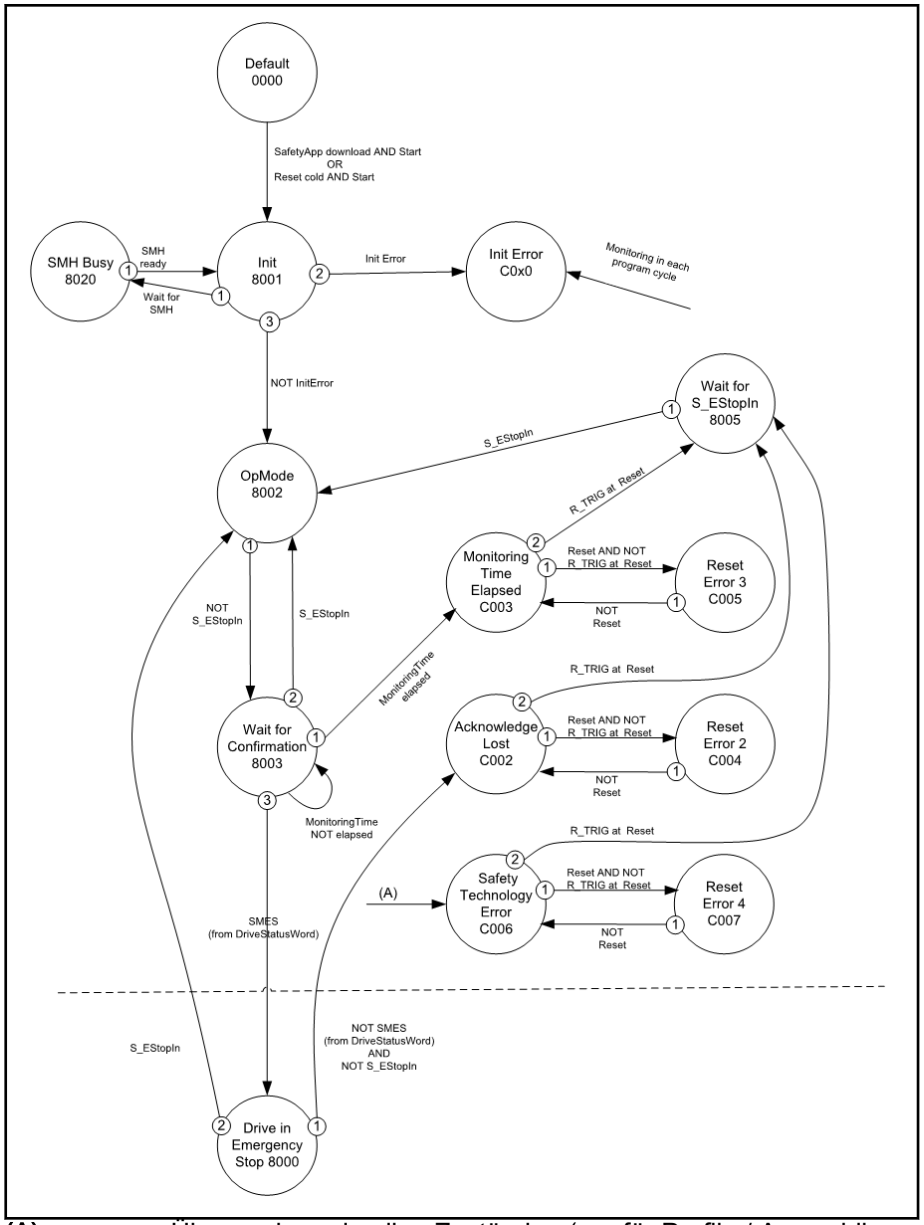
(5) Die Rückmeldung SMES des Antriebs, dass sich der Antrieb nicht mehr im Zustand SMES befindet, obwohl "S_EStopIn" immer noch FALSE ist, also eine Anforderung für SMES ansteht, führt dazu, dass der Baustein in den Zustand C002 "Acknowledge Lost" wechselt. Außerdem geht das Ausgangssignal "S_EStopOut" auf FALSE und "Error" auf TRUE.

(6) Durch eine Aufwärtsflanke an "Reset" wird der Fehler quittiert. Der Ausgang "Error" geht auf FALSE und der Baustein wechselt in den Zustand 8005 "Wait for S_EStopIn".

(7) Das Signal "S_EStopIn" wechselt auf TRUE. Damit wechselt der Baustein in den Zustand 8002 "OpMode". Das TRUE an "S_EStopIn" wird direkt in die SafeMotion des Antriebs geschrieben.

Abb. 16-138: SF_MotionEmergencyStop Timing Diagramm

Zustandsdiagramm



(A) Überwachung in allen Zuständen (nur für Profile / Assemblies 783, 785, 788, 789, 790)

S_EStopOut In den Zuständen 8001 Init und den Init Errors C0x0 wird auf den Ausgang "S_EStopOut" FALSE geschrieben. Der aktuelle Zustand des Antriebs kann nicht ermittelt werden. In allen anderen Zuständen wird der Zustand SMES des DriveStatus-Word auf "S_EStopOut" geschrieben. Wenn SMES aktiv ist, dann ist S_EStopOut FALSE. Wenn SMES nicht aktiv ist, dann ist S_EStopOut TRUE.

Init Error Aus dem Zustand "Init Error" kommt man nur durch Fehlerkorrektur und SafetyApp download und Start.

Abb. 16-139: Zustandsdiagramm SF_MotionEmergencyStop

Zustandsbeschreibung

DiagCode	Zustand	Beschreibung	Ausgänge
16#0000	Default	FB ist nicht aktiv.	S_EStopOut = FALSE(2) Error = FALSE
16#8000	Drive in Emergency Stop	Antrieb ist in Emergency Stop	S_EStopOut = TRUE Error = FALSE
16#8001	Init	Zustand, nachdem der FB aktiviert wurde.	S_EStopOut = FALSE(2) Error = FALSE
16#8002	Operation Mode	Normalbetrieb, es liegt keine Anforderung vor. Der Antrieb darf drehen.	S_EStopOut = (1) Error = FALSE
16#8003	Wait for Confirmation	Warten auf die Bestätigung des Antrieb. Beim Wechsel in diesen Zustand wird die Monitoring-Zeit neu gestartet.	S_EStopOut = (1) Error = FALSE
16#8005	Wait for S_EStopIn	Ein Fehler wurde zurückgesetzt. "S_EStopIn" muss auf TRUE gesetzt werden, um wieder in den Betriebszustand 8002 zu wechseln.	S_EStopOut = (1) Error = FALSE
16#8020	Wait for SMH	Warten auf die Initialisierung des SafeMotion-Handlers. In jedem Zyklus wird ein Funktionsbaustein abgearbeitet.	S_EStopOut = FALSE(2) Error = FALSE
16#C002	Acknowledge Lost	Der Antrieb wurde gestoppt und die Stop Anforderung steht noch an, das Bestätigungssignal ist jedoch nicht mehr vorhanden.	S_EStopOut = (1) Error = TRUE
16#C003	Monitoring Time Elapsed	Der Antrieb wurde nicht innerhalb der Monitoring Zeit angehalten.	S_EStopOut = (1) Error = TRUE
16#C004	Reset Error 2	Statischer Reset erkannt im Zustand C002.	S_EStopOut = (1) Error = TRUE
16#C005	Reset Error 3	Statischer Reset erkannt im Zustand C003.	S_EStopOut = (1) Error = TRUE
16#C006	Sicherheitstechnik Error	Die Sicherheitstechnik im Antrieb meldet einen Fehler (wird nur in Profilen mit den Assemblies 783, 785, 788, 789, 790 unterstützt).	S_EStopOut = (1) Error = TRUE
16#C007	Reset Error 4	Statischer Reset erkannt im Zustand C006.	S_EStopOut = (1) Error = TRUE
16#C010	AxisID nicht vorhanden	Kein Antrieb mit dieser AxisID konfiguriert. Dieser Fehler wird im "Init"-Zustand erkannt.	S_EStopOut = FALSE(2) Error = TRUE
16#C020	Nicht unterstütztes Profil	In der Antriebsparametrierung wurde ein nicht unterstütztes SafeMotion-Profil konfiguriert. Dieser Fehler wird im "Init"-Zustand erkannt.	S_EStopOut = FALSE(2) Error = TRUE
16#C030	Instanz mit dieser AxisID schon vorhanden	Die Instanz des Funktionsbausteins wurde schon einmal mit einer anderen AxisID verwendet. Dieser Fehler wird im "Init"-Zustand erkannt.	S_EStopOut = FALSE(2) Error = TRUE

DiagCode	Zustand	Beschreibung	Ausgänge
16#C040	Eine Drive-Verbindung fehlt	Für die angelegte AxisID wurde keine CSos-Verbindung zum Antrieb gefunden. Dieser Fehler wird im "Init"-Zustand erkannt.	S_EStopOut = FALSE(2) Error = TRUE
16#C060	AxisGroupID defined twice	Die angelegte AxisGroupID wurde zwei mal definiert und ist somit ungültig. Dieser Fehler wird im "Init"-Zustand erkannt.	S_EStopOut = FALSE(2) Error = TRUE
16#C070	AxisID already used in axis group	Die angelegte AxisID wurde auch als AxisID an einem SF_AxisGroup-Funktionsbaustein verwendet und ist somit ungültig. Dieser Fehler wird im "Init"-Zustand erkannt.	S_EStopOut = FALSE(2) Error = TRUE

(1) Die Antriebsrückmeldung für den Zustand SMES wird auf den Ausgang "S_EStopOut" geschrieben.

(2) Bei einem SPS-Programm-Neustart gilt: In den Zuständen 0000 Idle, 8001 Init und den Init-Fehlern C0x0 wird FALSE auf den Ausgang "S_EStopOut" geschrieben, da die Statusinformation vom Antrieb nicht verfügbar ist.

Tab. 16-235: Zustandsbeschreibung SF_MotionEmergencyStop

16.6.2 SF_MotionSafeStop

Kurzbeschreibung

Der Funktionsbaustein SF_MotionSafeStop steuert und überwacht die SafeStop 1 (SS1) und SafeStop 2 (SS2) Funktionalität eines oder mehrerer Antriebe (bei Verwendung von Achsgruppen).

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyDriveMotion	2.6.0.0

Tab. 16-236: Bibliothekszuordnung

Schnittstellenbeschreibung

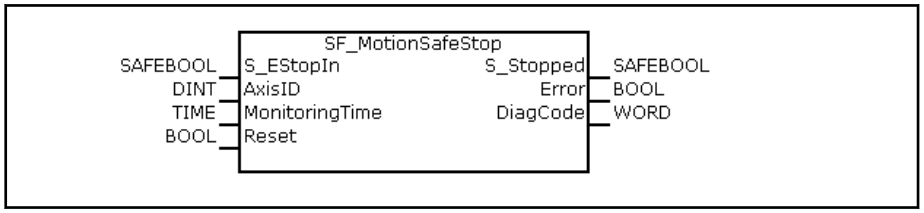


Abb. 16-140: Funktionsbaustein SF_MotionSafeStop

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	S_StopIn	SAFEBOOL	Anforderung Emergency Stop (low aktiv) FALSE: Anforderung einer Sicherheitsfunktion SafeStop. TRUE: keine Anforderung einer Sicherheitsfunktion.
	AxisID	DINT	SDID des zugehörigen Antriebes oder AxisGroupID einer Achsgruppe
	MonitoringTime	TIME	Zeitspanne in der der Antrieb den sicheren Zustand angenommen haben muss
	Reset	BOOL	Zurücksetzen anstehende Fehler und Statusmeldungen über den Diagnose-Code, wenn die Fehlerursache beseitigt wurde.

E/A-Typ	Name	Datentyp	Beschreibung
VAR_OUTPUT	S_Stopped	SAFEBOOL	Ausgang für die Sicherheitsanforderung. FALSE: Der Antrieb bzw. die Achsgruppe ist nicht im sicheren Stillstand TRUE: Der Antrieb bzw. die Achsgruppe ist im sicheren Stillstand
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Zustand des Funktionsbausteins bzw. Diagnose-Code (siehe Zustandsbeschreibung, Seite 550)

Tab. 16-237: Schnittstelle SF_MotionSafeStop

Name	Typ	Min	Max	Default-Wert	Übernahme
S_StopIn	SAFEBOOL			FALSE	kontinuierlich
AxisID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
MonitoringTime	TIME	t#0s		T#0s	Bei erstem Aufruf des Bausteins
Reset	SAFEBOOL			FALSE	kontinuierlich

Tab. 16-238: Eingangsverhalten SF_MotionSafeStop

Normativer Verweis

Norm	Anforderung
DIN EN 61800-5-2:2008	4.2 Sicherheitsfunktionen

Funktionsbeschreibung

Der Funktionsbaustein SF_MotionSafeStop schaltet einen oder mehrere Antriebe bei einem FALSE am Eingang "S_StopIn" in die sichere Betriebsart SMST1 (Safe Motion Safe Stop 1) oder SMST2 (Safe Motion Safe Stop 2).

Nach Auslösen der Funktion überwacht der Funktionsbaustein, ob der Zustand SMSSx in allen angesteuerten Antrieben innerhalb der am Eingang "MonitoringTime" angegebenen Zeit tatsächlich erreicht wurde und während der gesamten Anforderungszeit aktiv bleibt.

Nach Erreichen des Zustands wird im Fall von SMST1 die Leitung im Antrieb abgeschaltet.

Bei SMST2 verbleibt der Antrieb in Regelung. Der sichere Stillstand wird in beiden Fällen überwacht und über den Ausgang "S_Stopped" angezeigt.

Der Funktionsbaustein SF_MotionSafeStop wird in der Regel zusammen mit den Funktionsbausteinen SF_MotionEmergencyStop und SF_SMMSelector betrieben.

Siehe auch:

- SF_MotionEmergencyStop, Seite 539
- SF_SMMSelector, Seite 552

☞ Die Ansteuerung der sicheren Bewegungsbetriebsart SMSSx hat Priorität über der Betriebsart SMMx. Sie hat aber eine geringere Priorität als ein SMES (siehe auch Prioritäten der sicheren Betriebsarten, Seite 537).

Reset

Der Eingang "Reset" quittiert aufgetretene Fehler. Der Funktionsbaustein bleibt so lange im Fehlerzustand, bis die Ursache des Fehlers beseitigt und der Fehler durch eine steigende Flanke am Eingang "Reset" quittiert wurde.

S_StopIn, S_Stopped

Der low-aktive Eingang "S_StopIn" wählt die sichere Antriebsbestriebsart SMSS1 (Safe Motion Safe Stop 1) oder SMSS2 (Safe Motion Safe Stop 2) an. Welche der beiden Betriebsarten angewählt wird, kann in der SafeMotion des Antriebs parametriert werden. In diesen Betriebsarten wird überwacht, dass der Antriebs steht ist. Ist "S_StopIn" FALSE, so wird die sichere Betriebsart SMST1 oder SMST2 angewählt.

Die Bestätigung, dass der Antrieb sich tatsächlich in SMST1 oder SMST2 befindet, wird am Ausgang "S_Stopped" angezeigt. Ist "S_Stopped" TRUE, befindet sich der Antrieb im sicher überwachtem Stilstand.

AxisID

Am Eingang AxisID wird die SDID des anzusteuernden Antriebs angegeben. Sie bildet die logische Adresse aus Sicht der SafeLogic auf einen realen Antrieb. Anstelle der SDID einer Achse kann auch eine zuvor mit dem Funktionsbaustein Kap. 16.6.4 "SF_AxisGroup" auf Seite 567 definierte AxisGroupID angelegt werden. Hierdurch werden alle Achsen einer Achsgruppe angesteuert.



Eine einmal am Eingang AxisID angelegte SDID oder AxisGroupID darf im weiteren Programmverlauf nicht mehr geändert werden!

MonitoringTime

Der Eingang "MonitoringTime" definiert die maximale Zeit zwischen der Kommandierung einer sicheren Bewegungsbetriebsart in den Antrieben und der Bestätigung der Anwahl von den Antrieben. Wird die angeforderte sichere Betriebsart nicht innerhalb der von "MonitoringTime" definierten Zeitspanne erreicht, so geht der Funktionsbaustein in den Fehlerzustand.

Der Eingang "MonitoringTime" wird normalerweise auf die für die zum Erreichen des sicheren Zustands maximal erlaubte Zeit gesetzt. Diese ergibt sich aus den Anforderungen an die Sicherheitsfunktion.



Wird die "MonitoringTime" auf `T#0s` gesetzt, so findet keine Überwachung statt.

Es wird empfohlen eine Monitoring Time zu verwenden!



In der SafeMotion des Antriebs wird auch eine maximale Überführzeit definiert. Diese wird unabhängig von der am Eingang "MonitoringTime" eingestellten Zeitspanne vom Antrieb selbständig überwacht.

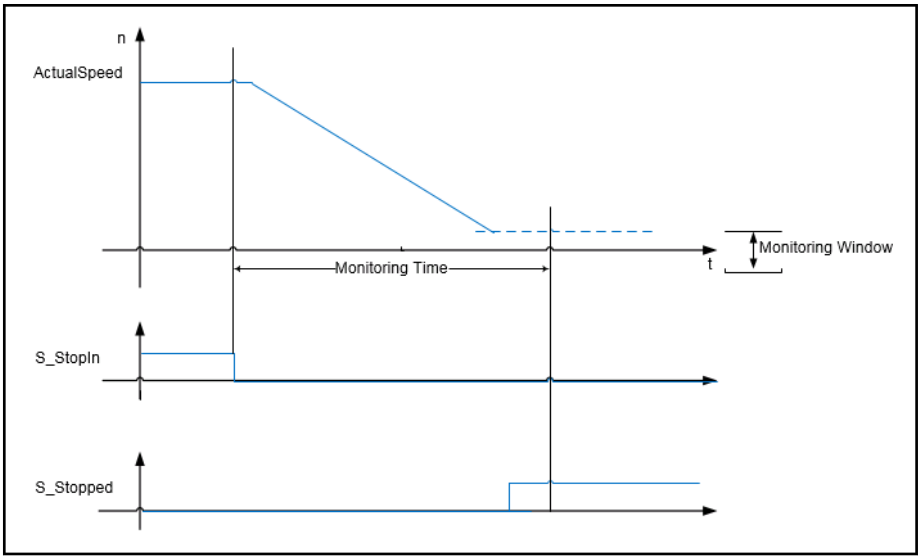


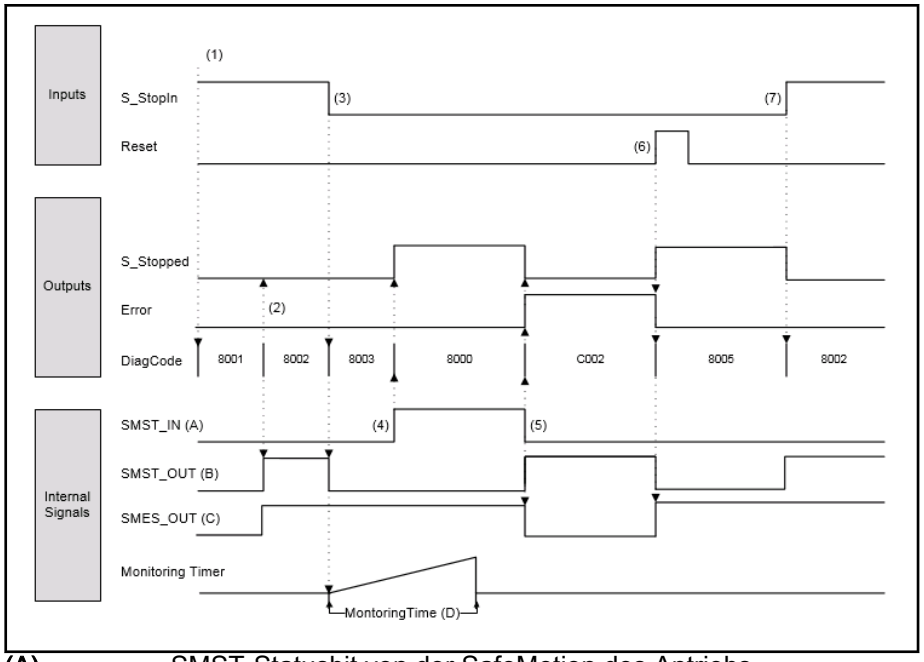
Abb. 16-141: SF_MotionSafeStop Prinzipieller Ablauf

Zeitgenauigkeit

Die zeitliche Genauigkeit der am Eingang "MonitoringTime" angelegten Überwachungszeit wird in Zeitmessungen erläutert.

Genauigkeit von Zeitmessungen, Seite 229

Zeitdiagram



(A) SMST-Statusbit von der SafeMotion des Antriebs

(B) SMST-Steuerbit an die SafeMotion des Antriebs

(C) SMES-Steuerbit an die SafeMotion des Antriebs

(D) Entspricht der Zeit am Eingang "MonitoringTime"

(1) Nach dem Start der Applikation wird der Baustein initialisiert (Zustand 8001).

(2) Nach erfolgreicher Initialisierung wechselt der Baustein in den Zustand 8002 und das Eingangssignal "S_EStopIn" wird in das Steuerwort des Antriebs geschrieben.

(3) Durch eine Abwärtsflanke an "S_StopIn" wechselt der Baustein in den Zustand 8003 "Wait for Confirmation". Gleichzeitig wird der neue Zustand in das Steuerwort des Antriebs geschrieben und die "MonitoringTime" gestartet, wenn diese größer T#0s ist.

(4) Durch die Rückmeldung SMSS des Antriebs wechselt der Baustein in den Zustand 8000 "Drive Stopped" und das Ausgangssignal "S_Stopped" wechselt auf TRUE.

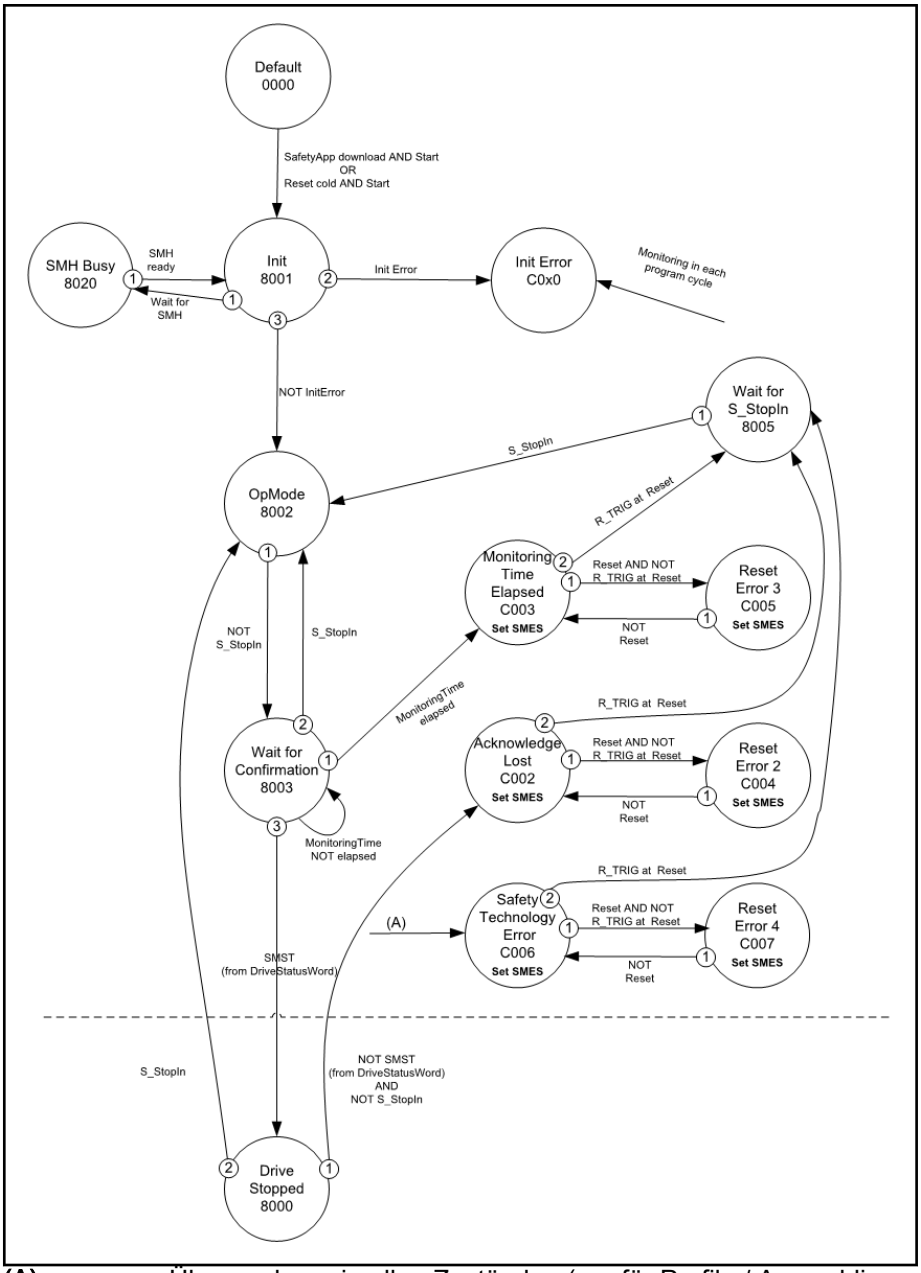
(5) Die Rückmeldung SMST des Antriebs, dass sich der Antrieb nicht mehr im Zustand SMST befindet, obwohl "S_StopIn" immer noch FALSE ist, also eine Anforderung für SMST ansteht, führt dazu, dass der Baustein in den Zustand C002 "Acknowledge Lost" wechselt. Außerdem geht das Ausgangssignal "S_Stopped" auf FALSE und "Error" auf TRUE.

(6) Durch eine Aufwärtsflanke an "Reset" wird der Fehler quittiert. Der Ausgang "Error" geht auf FALSE und der Baustein wechselt in den Zustand 8005 "Wait for S_StopIn".

(7) Das Signal "S_StopIn" wechselt auf TRUE. Damit wechselt der Baustein in den Zustand 8002 "OpMode". Das TRUE an "S_StopIn" wird direkt in die SafeMotion des Antriebs geschrieben.

Abb. 16-142: SF_MotionSafeStop Timing Diagramm

Zustandsdiagramm



(A) Überwachung in allen Zuständen (nur für Profile / Assemblies 783, 785, 788, 789, 790)

S_Stopped In den Zuständen 8001 Init und den Init Errors C0x0 wird auf den Ausgang S_Stopped FALSE geschrieben. Der aktuelle Zustand des Antriebs kann nicht ermittelt werden. In allen anderen Zuständen wird der Zustand SMST des DriveStatus-Word auf S_Stopped geschrieben.

Init Error Aus dem Zustand Init Error kommt man nur durch Fehlerkorrektur und SafetyApp download und Start.

Abb. 16-143: Zustandsdiagramm SF_MotionSafeStop

Zustandsbeschreibung

DiagCode	Zustand	Beschreibung	Ausgänge
16#0000	Default	FB ist nicht aktiv.	S_Stopped = FALSE(2) Error = FALSE
16#8000	Drive in Emergency Stop	Antrieb ist gestoppt	S_Stopped = TRUE Error = FALSE
16#8001	Init	Zustand, nachdem der FB aktiviert wurde.	S_Stopped = FALSE(2) Error = FALSE
16#8002	Operation Mode	Normalbetrieb, es liegt keine Anforderung vor. Der Antrieb darf drehen.	S_Stopped = (1) Error = FALSE
16#8003	Wait for Confirmation	Warten auf die Bestätigung des Antrieb. Beim Wechsel in diesen Zustand wird die Monitoring-Zeit neu gestartet.	S_Stopped = (1) Error = FALSE
16#8005	Wait for S_StopIn	Ein Fehler wurde zurückgesetzt. "S_StopIn" muss auf TRUE gesetzt werden, um wieder in den Betriebszustand 8002 zu wechseln.	S_Stopped = (1) Error = FALSE
16#C002	Acknowledge Lost	Der Antrieb wurde gestoppt und die Stop Anforderung steht noch an, das Bestätigungssignal ist jedoch nicht mehr vorhanden.	S_Stopped = (1) Error = TRUE
16#C003	Monitoring Time Elapsed	Der Antrieb wurde nicht innerhalb der Monitoring Zeit angehalten.	S_Stopped = (1) Error = TRUE
16#C004	Reset Error 2	Statischer Reset erkannt im Zustand C002.	S_Stopped = (1) Error = TRUE
16#C005	Reset Error 3	Statischer Reset erkannt im Zustand C003.	S_Stopped = (1) Error = TRUE
16#C006	Sicherheitstechnik Error	Die Sicherheitstechnik im Antrieb meldet einen Fehler (wird nur in Profilen mit den Assemblies 783, 785, 788, 789, 790 unterstützt).	S_Stopped = (1) Error = TRUE
16#C007	Reset Error 4	Statischer Reset erkannt im Zustand C006.	S_Stopped = (1) Error = TRUE
16#C010	AxisID nicht vorhanden	Kein Antrieb bzw. keine Achsgruppe mit dieser AxisID konfiguriert. Dieser Fehler wird im "Init"-Zustand erkannt.	S_Stopped = FALSE(2) Error = TRUE
16#C020	Nicht unterstütztes Profil	In der Antriebsparametrierung wurde ein nicht unterstütztes SafeMotion-Profil konfiguriert. Dieser Fehler wird im "Init"-Zustand erkannt.	S_Stopped = FALSE(2) Error = TRUE
16#C030	Instanz mit dieser AxisID schon vorhanden	Die AxisID wurde schon einmal mit einer anderen Instanz des Funktionsbausteins verwendet. Dieser Fehler wird im "Init"-Zustand erkannt.	S_Stopped = FALSE(2) Error = TRUE
16#C040	Eine Drive-Verbindung fehlt	Für die angelegte AxisID wurde keine CSos-Verbindung zum Antrieb gefunden. Dieser Fehler wird im "Init"-Zustand erkannt.	S_Stopped = FALSE(2) Error = TRUE

DiagCode	Zustand	Beschreibung	Ausgänge
16#C050	EStop für diese AxisID nicht programmiert	Für diese AxisID wurde kein Funktionsbaustein "SF_MotionEmergencyStop" angelegt. Siehe Kap. 16.6.1 "SF_MotionEmergencyStop" auf Seite 539. Dieser Fehler wird im "Init"-Zustand erkannt.	S_Stopped = FALSE(2) Error = TRUE
16#C060	AxisGroupID defined twice	Die angelegte AxisGroupID wurde zwei mal angemeldet und ist somit ungültig. Dieser Fehler wird im "Init"-Zustand erkannt.	S_Stopped = FALSE(2) Error = TRUE
16#C070	AxisID already used in axis group	Die angelegte AxisID wurde auch als AxisID an einem SF_AxisGroup-Funktionsbaustein verwendet und ist somit ungültig. Dieser Fehler wird im "Init"-Zustand erkannt.	S_Stopped = FALSE(2) Error = TRUE

(1) Die Antriebsrückmeldung für den Zustand SMST wird auf den Ausgang "S_Stopped" geschrieben.

(2) Bei einem SPS-Programm-Neustart gilt: In den Zuständen 0000 Idle, 8001 Init und den Init-Fehlern C0x0 wird FALSE auf den Ausgang "S_Stopped" geschrieben, da die Statusinformation vom Antrieb nicht verfügbar ist.

Tab. 16-239: Zustandsbeschreibung SF_MotionSafeStop

16.6.3 SF_SMMSelector

Kurzbeschreibung

Der Funktionsbaustein SF_SMMSelector fordert eine von acht möglichen im Antrieb konfigurierten sicheren Bewegungsbetriebsarten an.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyDriveMotion	2.6.0.0

Tab. 16-240: Bibliothekszuordnung

Schnittstellenbeschreibung

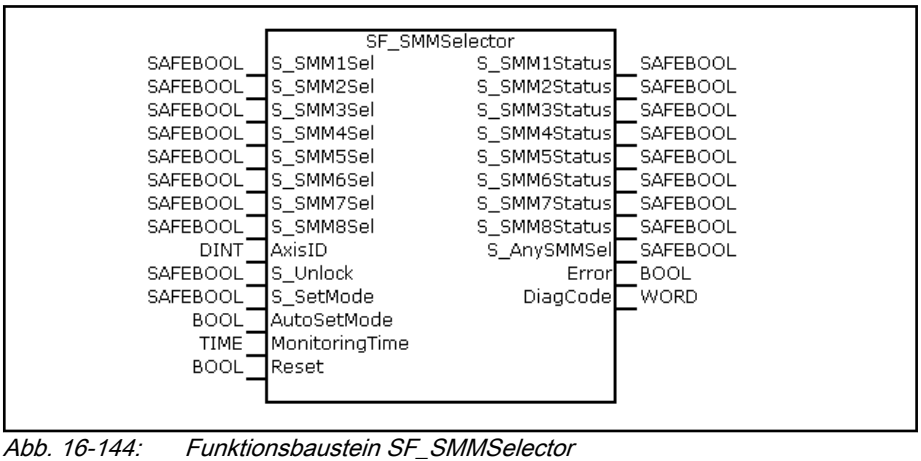


Abb. 16-144: Funktionsbaustein SF_SMMSelector

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	S_SMM1Sel	SAFEBOOL	Anforderung der sicheren Bewegungsart SMM1
	S_SMM2Sel	SAFEBOOL	Anforderung der sicheren Bewegungsart SMM2
	S_SMM3Sel	SAFEBOOL	Anforderung der sicheren Bewegungsart SMM3
	S_SMM4Sel	SAFEBOOL	Anforderung der sicheren Bewegungsart SMM4
	S_SMM5Sel	SAFEBOOL	Anforderung der sicheren Bewegungsart SMM5
	S_SMM6Sel	SAFEBOOL	Anforderung der sicheren Bewegungsart SMM6
	S_SMM7Sel	SAFEBOOL	Anforderung der sicheren Bewegungsart SMM7
	S_SMM8Sel	SAFEBOOL	Anforderung der sicheren Bewegungsart SMM8
	AxisID	DINT	SDID des zugehörigen Antriebs oder AxisGroupID einer Achsgruppe
	S_Unlock	SAFEBOOL	Verriegelt die Anwahl an den S_SMMxSel-Eingängen FALSE: Ein Wechsel an den Eingängen S_SMMxSel wird ignoriert, wenn eine steigende Flanke an S_SelMode anliegt. TRUE: Die Auswahl an den S_SMMxSel-Eingängen ist nicht verriegelt. Änderungen sind möglich.
	S_SetMode	SAFEBOOL	Übernahme der an den S_SMMxSel-Eingängen anliegenden Anwahl in den Antrieb bei steigender Flanke
	AutoSetMode	BOOL	Sofortige Übernahme der Anwahl an den S_SMMxSel-Eingängen in den Antrieb FALSE: Die SMM-Auswahl muss durch eine steigende Flanke an S_SetMode bestätigt werden. TRUE: Eine Änderung an den S_SMMxSel-Eingängen wird sofort in den Antrieb übernommen.
	MonitoringTime	TIME	Zeitspanne in der der Antrieb den sicheren Zustand angenommen haben muss
	Reset	BOOL	Zurücksetzen anstehende Fehler und Statusmeldungen über den Diagnose-Code, wenn die Fehlerursache beseitigt wurde.

E/A-Typ	Name	Datentyp	Beschreibung
VAR_OUTPUT	S_SMM1Status	SAFEBOOL	SMM1 im Antrieb bzw. in der Achsgruppe aktiv
	S_SMM2Status	SAFEBOOL	SMM2 im Antrieb bzw. in der Achsgruppe aktiv
	S_SMM3Status	SAFEBOOL	SMM3 im Antrieb bzw. in der Achsgruppe aktiv
	S_SMM4Status	SAFEBOOL	SMM4 im Antrieb bzw. in der Achsgruppe aktiv
	S_SMM5Status	SAFEBOOL	SMM5 im Antrieb bzw. in der Achsgruppe aktiv
	S_SMM6Status	SAFEBOOL	SMM6 im Antrieb bzw. in der Achsgruppe aktiv
	S_SMM7Status	SAFEBOOL	SMM7 im Antrieb bzw. in der Achsgruppe aktiv
	S_SMM8Status	SAFEBOOL	SMM8 im Antrieb bzw. in der Achsgruppe aktiv
	S_AnySMMSel	SAFEBOOL	Eine der acht SMMx ist im Antrieb bzw. in der Achsgruppe aktiv.
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Zustand des Funktionsbausteins bzw. Diagnose-Code (siehe Zustandsbeschreibung, Seite 565)

Tab. 16-241: Schnittstelle SF_SMMSelector

Name	Typ	Min	Max	Default-Wert	Übernahme
S_SMM1Sel	SAFEBOOL			FALSE	kontinuierlich
S_SMM2Sel	SAFEBOOL			FALSE	kontinuierlich
S_SMM3Sel	SAFEBOOL			FALSE	kontinuierlich
S_SMM4Sel	SAFEBOOL			FALSE	kontinuierlich
S_SMM5Sel	SAFEBOOL			FALSE	kontinuierlich
S_SMM6Sel	SAFEBOOL			FALSE	kontinuierlich
S_SMM7Sel	SAFEBOOL			FALSE	kontinuierlich
S_SMM8Sel	SAFEBOOL			FALSE	kontinuierlich
AxisID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
S_Unlock	SAFEBOOL			FALSE	kontinuierlich (in den Zuständen 0x8000 und 0x8004)
S_SetMode	SAFEBOOL			FALSE	kontinuierlich (im Zustand 0x8002)
AutoSetMode	BOOL			FALSE	kontinuierlich (im Zustand 0x8002)
MonitoringTime	TIME	t#0s		T#0s	kontinuierlich (im Zustand 0x8002)
Reset	SAFEBOOL			FALSE	kontinuierlich

Tab. 16-242: Eingangsverhalten SF_SMMSelector

Normativer Verweis

Norm	Anforderung
DIN EN 61800-5-2:2008	4.2 Sicherheitsfunktionen

Funktionsbeschreibung

Der Funktionsbaustein SF_SMMSelector aktiviert eine von acht möglichen sicheren Bewegungsbetriebsarten SMMx (Safe Motion Mode x) in einem oder mehreren Antrieben (bei Verwendung von Achsgruppen). Nach Auslösen der Funktion überwacht der Funktionsbaustein, ob der Zustand SMMx in allen angesteuerten Antrieben innerhalb der am Eingang "MonitoringTime" angegebenen Zeit tatsächlich erreicht wurde und während der gesamten Anforderungszeit aktiv bleibt.

Die sichere Betriebsart wird im Antrieb konfiguriert. Auf eine SMM können mehrere SafeMotion-Betriebsarten gleichzeitig konfiguriert werden. So kann SMM1 beispielsweise auf SLS (Safe Limited Speed) und SDI (Safe Direction) parametriert werden. SMM2 kann dann z.B. auf eine andere sichere Geschwindigkeit ohne Richtungsabhängigkeit parametriert werden. Dies ermöglicht große Flexibilität und Erweiterbarkeit, da bei einer Änderung der Parametrierung der SafeMotion des Antriebs das SPS-Program nicht angepasst werden muss.

Der Funktionsbaustein SF_SMMSelector wird in der Regel zusammen mit den Funktionsbausteinen SF_MotionSafeStop und SF_MotionEmergencyStop betrieben.

Siehe auch:

- SF_MotionSafeStop, Seite 545
- SF_MotionEmergencyStop, Seite 539



Die Ansteuerung einer sicheren Bewegungsbetriebsart SMMx hat die geringste Priorität. Liegt bei einem der beiden anderen Funktionsbausteine SF_MotionSafeStop oder SF_MotionEmergencyStop eine Anforderung vor, so wird diese aktiv (siehe auch Prioritäten der sicheren Betriebsarten, Seite 537).

Reset

Der Eingang "Reset" quittiert aufgetretene Fehler. Der Funktionsbaustein bleibt so lange im Fehlerzustand, bis der Fehler durch eine steigende Flanke am Eingang "Reset" quittiert wurde. Eine angewählte SMM wird auch im Fehlerzustand und nach der Quittierung des Fehlers weiter kommandiert. Es erfolgt jedoch keine Überwachung mehr. Eine erneute Auswahl eines SMMs ist erst nach Rücknahme aller "S_SMMxSel"-Eingänge möglich.

S_SMMxSel, S_SMMxStatus, S_AnySMMSel, S_Unlock, S_SetMode, AutoSetMode

Die acht Eingänge "S_SMM1Sel" bis "S_SMM8Sel" ermöglichen die Anwahl einer im Antrieb konfigurierten sicheren Betriebsart. Die zugeschaltete sichere Bewegungsbetriebsart hängt von der SafeMotion-Parametrierung des Antriebs ab.

Die "S_SMMxSel"-Eingänge werden übernommen, wenn entweder "AutoSetMode" TRUE ist oder eine steigende Flanke an "S_SetMode" erkannt wird. Die Übernahme der "S_SMMxSel"-Eingänge kann mit dem Eingang "S_Unlock" unterbunden werden. Ist "S_Unlock" FALSE, so wird die Anwahl auf keinen Fall übernommen.

Die Bestätigung, dass sich der Antrieb tatsächlich in einer SMM befindet, wird an den Ausgängen "S_SMMxStatus" angezeigt. Ist einer der acht Aus-

gänge "S_SMM1Status" bis "S_SMM8Status" TRUE, befindet sich der Antrieb in der entsprechenden SMM. Zusätzlich wird der Ausgang "S_AnySMMSel" TRUE. Er fungiert als Sammelstatus.

Die Rücknahme der SMM-Anwahl ist nur möglich, wenn der Eingang "S_Unlock" TRUE ist. Andernfalls bleibt eine einmal aktivierte SMM-Anwahl aktiv, egal was an den "S_SMMxSel"-Eingängen passiert.

☞ Bei der Verwendung binär codierter Assemblies können am Funktionsbaustein SF_SMMSelector die Ausgänge S_SMM1Status und S_AnySMMSel fälschlicherweise `TRUE` werden, wenn der Antrieb in den Parametriermode (PM) wechselt. Das kann passieren, obwohl der Eingang S_SMM1Sel `FALSE` ist.

Ursache ist, dass bei binär codierten Assemblies (783, 785, 788 und 790) der Zustand "SMM1 ist aktiv" im DriveStatus-Word mit binär `0` codiert wird.

Workaround: Verwenden Sie SMM1 nicht bei binär codierten Assemblies!

AxisID

Am Eingang AxisID wird die SDID des anzusteuernden Antriebs angegeben. Sie bildet die logische Adresse aus Sicht der SafeLogic auf einen realen Antrieb. Anstelle der SDID einer Achse kann auch eine zuvor mit dem Funktionsbaustein Kap. 16.6.4 "SF_AxisGroup" auf Seite 567 definierte AxisGroupID angelegt werden. Hierdurch werden alle Achsen einer Achsgruppe angesteuert.

☞ Eine einmal am Eingang AxisID angelegte SDID oder AxisGroupID darf im weiteren Programmverlauf nicht mehr geändert werden!

Achsgruppen

Bei Verwendung von Achsgruppen wird "S_SMMxStatus" durch `UND`-Verknüpfung der Status "SMMx" der einzelnen Achsen gebildet.

Direkte Umschaltung von SMMx nach SMMy

☞ Die direkte Umschaltung von SMMx nach SMMy wird von der SafeLogic ab SMFW 02V08.643 unterstützt.

Bei der direkten Umschaltung von SMMx nach SMMy für Achsgruppen kann es vorkommen, dass "S_AnySMMSel" zwischenzeitlich auf `FALSE` geht. Ursache ist, dass die einzelnen Achsen unterschiedlich lang für die direkte Umschaltung brauchen. Das Signal "S_SMMxStatus" wird durch `UND`-Verknüpfung der Status "SMMx" der einzelnen Achsen gebildet:

- "S_SMMxStatus" wechselt von `TRUE` auf `FALSE`, sobald eine Achse den Status "SMMx" = `FALSE` hat.
- "S_SMMyStatus" wird erst `TRUE`, wenn alle Achsen den Status "SMMy" = `TRUE` haben.

MonitoringTime

Der Eingang "MonitoringTime" definiert die maximale Zeit zwischen der Kommandierung einer sicheren Bewegungsbetriebsart in den Antrieben und der Bestätigung der Anwahl von den Antrieben. Wird die angeforderte sichere Betriebsart nicht innerhalb der von "MonitoringTime" definierten Zeitspanne erreicht, so geht der Funktionsbaustein in den Fehlerzustand.

☞ Der Eingang "MonitoringTime" wird normalerweise auf die für die zum Erreichen des sicheren Zustands maximal erlaubte Zeit gesetzt. Diese ergibt sich aus den Anforderungen an die Sicherheitsfunktion.

☞ Wenn die "MonitoringTime" auf T#0s gesetzt wird, findet keine Überwachung statt.

Es wird empfohlen eine Monitoring Time zu verwenden!

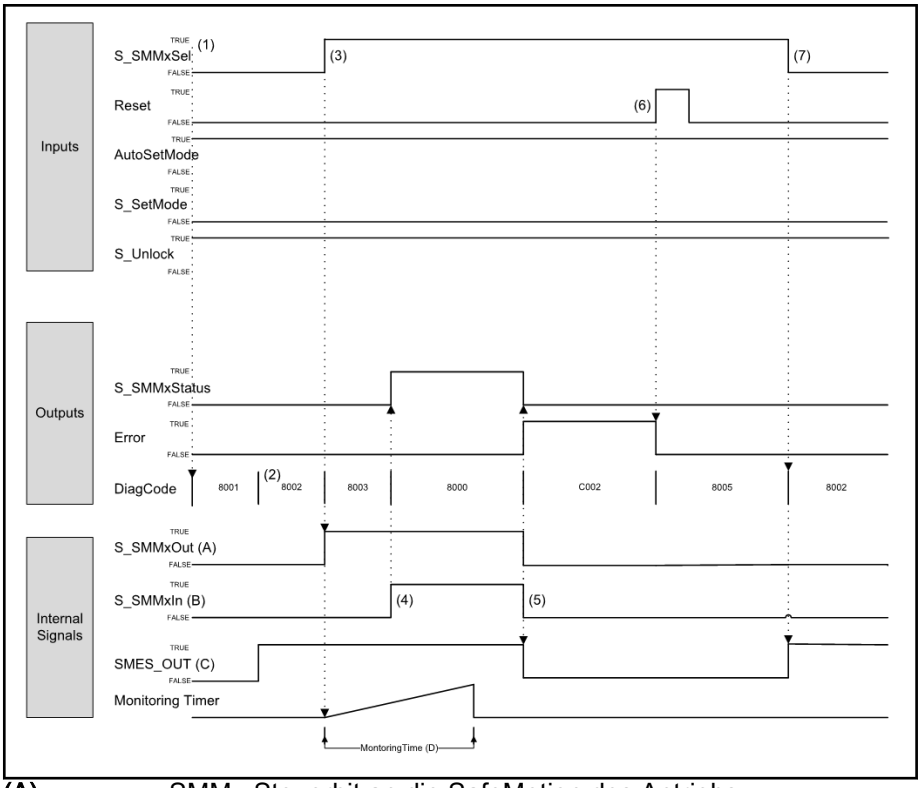
☞ In der SafeMotion des Antriebs wird auch eine maximale Überführzeit definiert. Diese wird unabhängig von der am Eingang "MonitoringTime" eingestellten Zeitspanne vom Antrieb selbständig überwacht.

Zeitgenauigkeit

Die zeitliche Genauigkeit der am Eingang "MonitoringTime" angelegten Überwachungszeit wird in Zeitmessungen erläutert.

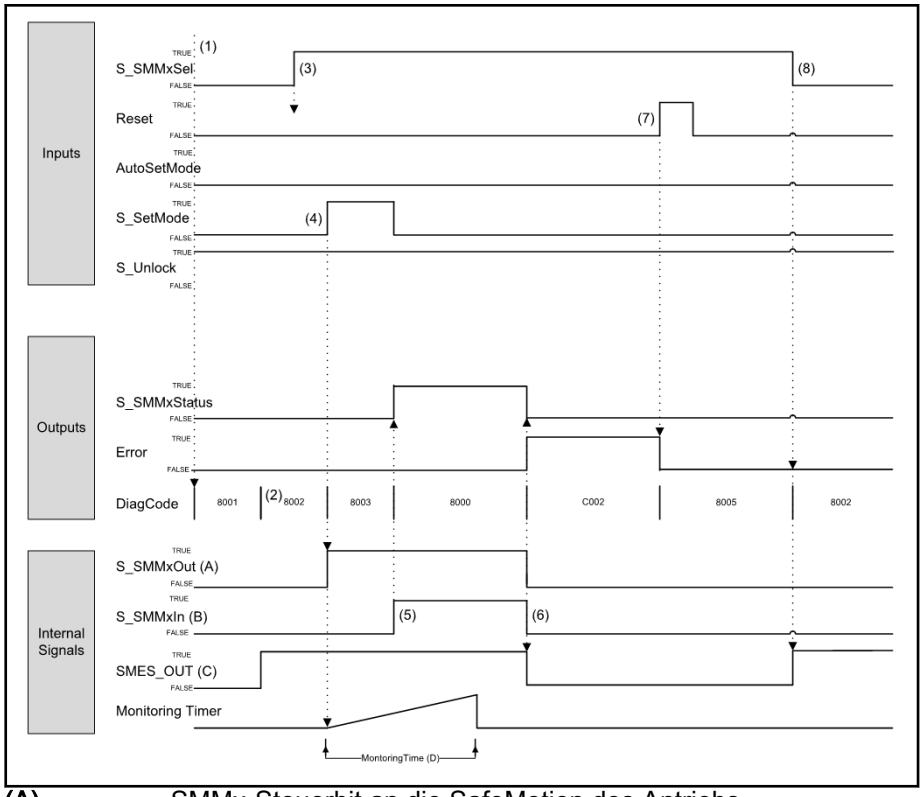
Genauigkeit von Zeitmessungen, Seite 229

Zeitdiagram



(A) SMMx-Steuerbit an die SafeMotion des Antriebs
(B) SMMx-Statusbit von der SafeMotion des Antriebs
(C) SMES-Steuerbit an die SafeMotion des Antriebs
(D) Entspricht der Zeit am Eingang "MonitoringTime"
(1) Nach dem Start der Applikation wird der Baustein initialisiert (Zustand 8001).
(2) Nach erfolgreicher Initialisierung wechselt der Baustein in den Zustand 8002 "OpMode".
(3) Durch eine steigende Flanke an "S_SMMxSel" wechselt der Baustein in den Zustand 8003 "Wait for Confirmation". Gleichzeitig wird der neue Zustand in das Steuerwort des Antriebs geschrieben und die "MonitoringTime" gestartet.
(4) Durch die Rückmeldung SMMx des Antriebs wechselt der Baustein in den Zustand 8000 "Drive in SMM" und das entsprechende Ausgangssignal "S_SMMxStatus" sowie das Sammelsignal "S_AnySMMSel" wechseln auf TRUE.
(5) Die Rückmeldung des Antriebs, dass sich der Antrieb nicht mehr im Zustand SMMx befindet, obwohl "S_SMMxSel" immer noch TRUE ist, also eine Anforderung für SMMx ansteht, führt dazu, dass der Baustein in den Zustand C002 "Acknowledge Lost" wechselt. Außerdem gehen die Ausgangssignale "S_SMMxStatus" und "S_AnySMMSel" auf FALSE und der Ausgang "Error" auf TRUE.
(6) Durch eine Aufwärtsflanke an "Reset" wird der Fehler quittiert. Der Ausgang "Error" geht auf FALSE und der Baustein wechselt in den Zustand 8005 "Wait for NOT S_SMMxSel".
(7) Durch die Wegnahme der SMM-Anforderung am Eingang "S_SMMxSel" wechselt der Baustein in den Zustand 8002 "OpMode". Die Anforderung der SMM im Antrieb wird zurück gesetzt.

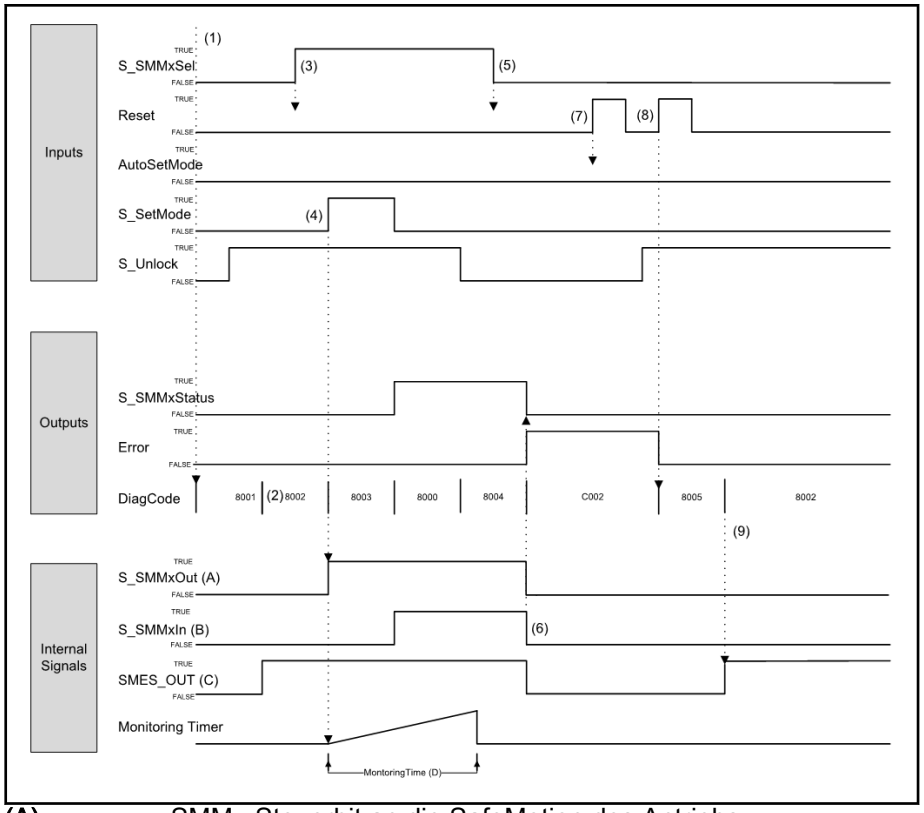
Abb. 16-145: SF_SMMSelector Timing Diagramm mit AutoSetMode



(A) SMMx-Steuerbit an die SafeMotion des Antriebs
(B) SMMx-Statusbit von der SafeMotion des Antriebs
(C) SMES-Steuerbit an die SafeMotion des Antriebs
(D) Entspricht der Zeit am Eingang "MonitoringTime"
(1) Nach dem Start der Applikation wird der Baustein initialisiert (Zustand 8001).
(2) Nach erfolgreicher Initialisierung wechselt der Baustein in den Zustand 8002 "OpMode".
(3) Die Aufwärtsflanke an einem "S_SMMxSel" bewirkt nichts, weil der Eingang "AutoSetMode" FALSE ist.
(4) Durch eine steigende Flanke an "S_SetMode" wird der angewählte "S_SMMxSel"-Eingang übernommen und der Baustein wechselt in den Zustand 8003 "Wait for Confirmation". Gleichzeitig wird der neue Zustand in das Steuerwort des Antriebs geschrieben und die "MonitoringTime" gestartet.
(5) Durch die Rückmeldung SMMx des Antriebs wechselt der Baustein in den Zustand 8000 "Drive in SMM" und das entsprechende Ausgangssignal "S_SMMxStatus" sowie das Sammelsignal "S_AnySMMSel" wechseln auf TRUE.
(6) Die Rückmeldung des Antriebs, dass sich der Antrieb nicht mehr im Zustand SMMx befindet, obwohl "S_SMMxSel" immer noch TRUE ist, also eine Anforderung für SMMx ansteht, führt dazu, dass der Baustein in den Zustand C002 "Acknowledge Lost" wechselt. Außerdem gehen die Ausgangssignale "S_SMMxStatus" und "S_AnySMMSel" auf FALSE und der Ausgang "Error" auf TRUE.
(7) Durch eine Aufwärtsflanke an "Reset" wird der Fehler quittiert. Der Ausgang "Error" geht auf FALSE und der Baustein wechselt in den Zustand 8005 "Wait for NOT S_SMMxSel".
(8) Durch die Wegnahme der SMM-Anforderung am Eingang "S_SMMxSel" wechselt der Baustein in den Zustand 8002 "Op-

Mode". Die Anforderung der SMM im Antrieb wird zurück gesetzt.

Abb. 16-146: SF_SMMSelector Timing Diagramm mit S_SetMode

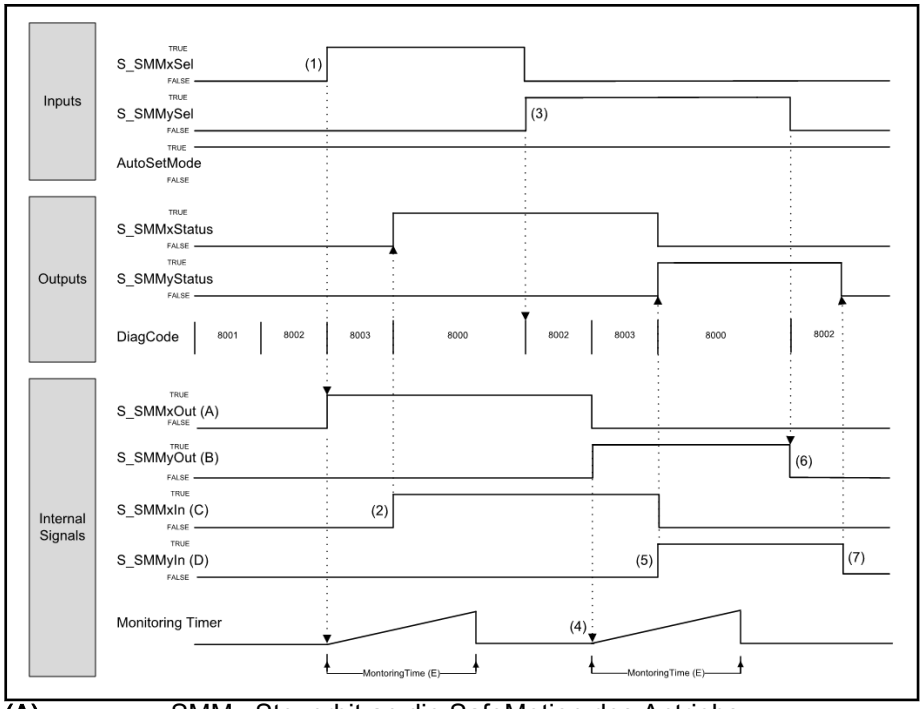


(A) SMMx-Steuerbit an die SafeMotion des Antriebs
(B) SMMx-Statusbit von der SafeMotion des Antriebs
(C) SMES-Steuerbit an die SafeMotion des Antriebs
(D) Entspricht der Zeit am Eingang "MonitoringTime"
(1) Nach dem Start der Applikation wird der Baustein initialisiert (Zustand 8001).
(2) Nach erfolgreicher Initialisierung wechselt der Baustein in den Zustand 8002 "OpMode".
(3) Die Aufwärtsflanke an einem "S_SMMxSel" bewirkt nichts, weil der Eingang "AutoSetMode" FALSE ist.
(4) Durch eine steigende Flanke an "S_SetMode" wird der angewählte "S_SMMxSel"-Eingang übernommen und der Baustein wechselt in den Zustand 8003 "Wait for Confirmation". Gleichzeitig wird der neue Zustand in das Steuerwort des Antriebs geschrieben und die "MonitoringTime" gestartet.
(5) Das "S_SMMxSel"-Signal am Eingang des Bausteins geht auf FALSE. Dieser Wechsel bewirkt jedoch nichts, weil zu diesem Zeitpunkt der Eingang "S_Unlock" FALSE ist. Die letzte "S_SMMxStatus"-Meldung am Ausgang des Bausteins bleibt erhalten.
(6) Die Rückmeldung des Antriebs, dass sich der Antrieb nicht mehr im Zustand SMMx befindet, obwohl "S_SMMxSel" immer noch TRUE ist, also eine Anforderung für SMMx ansteht, führt dazu, dass der Baustein in den Zustand C002 "Acknowledge Lost" wechselt. Außerdem gehen die Ausgangssignale "S_SMMxStatus" und "S_AnySMMSel" auf FALSE und der Ausgang "Error" auf TRUE.
(7) Die steigende Flanke am Eingang "Reset" bewirkt nichts, weil "S_Unlock" noch FALSE ist.
(8) S_Unlock hat auf TRUE gewechselt. Dadurch wird die S_SMMxSel-Anwahl in die SMMx-Anforderung des Antriebs

übernommen. Zu diesem Zeitpunkt sind alle S_SMMxSel-Eingänge FALSE. Durch die steigende Flanke an Reset wechselt der Baustein in den Zustand 8005 "Wait for NOT S_SMMxSel".

(9) Durch die steigende Flanke an Reset wechselt der Baustein in den Zustand 8005 "Wait for NOT S_SMMxSel". Da kein "S_SMMxSel"-Eingang gesetzt ist, wechselt der Baustein in den Zustand 8002 "OpMode". Die Anforderung der SEMS wird zurückgenommen.

Abb. 16-147: SF_SMMSelector Timing Diagramm mit S_SetMode und S_Unlock

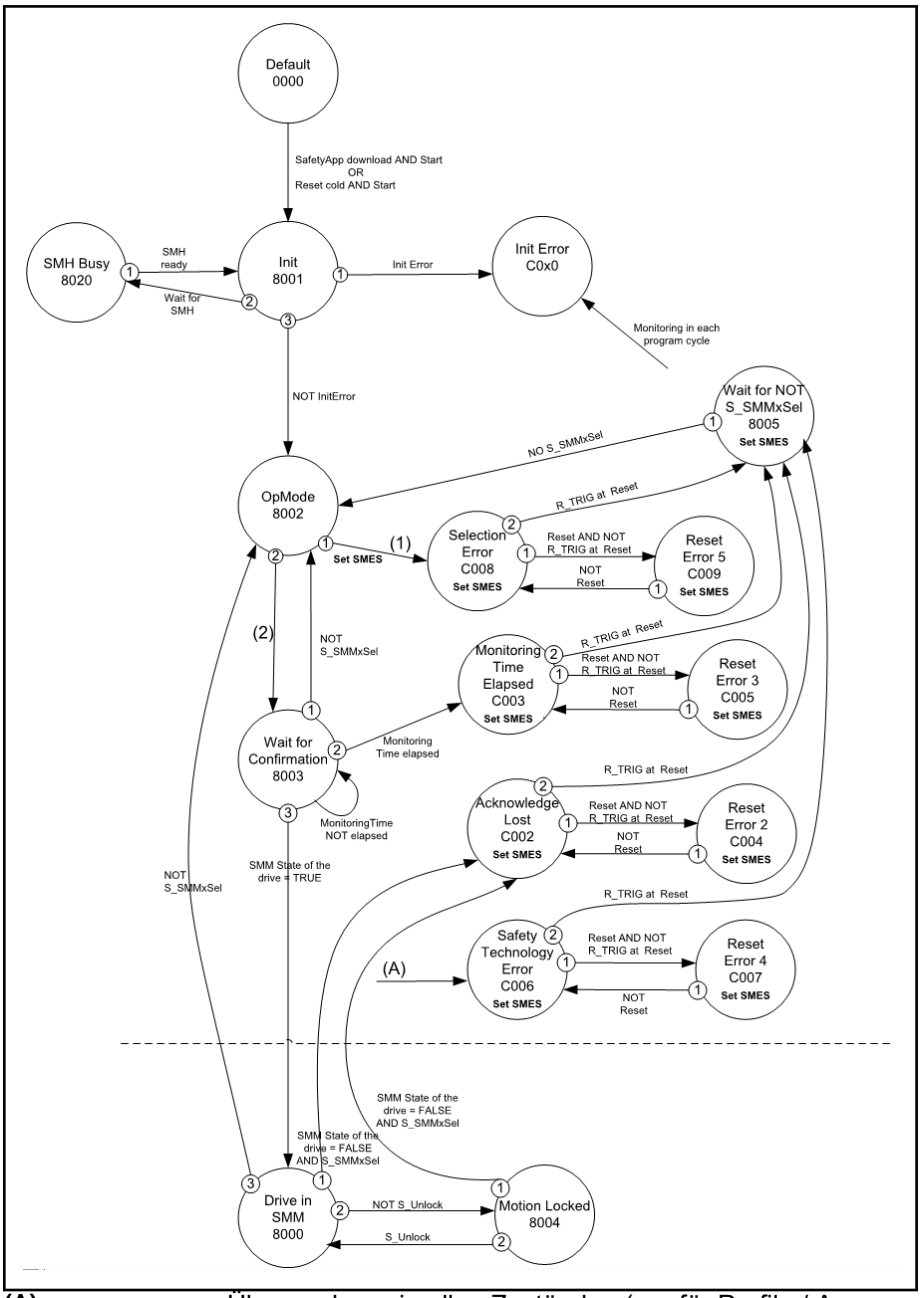


(A) SMMx-Steuerbit an die SafeMotion des Antriebs
(B) SMMy-Steuerbit an die SafeMotion des Antriebs
(C) SMMx-Statusbit von der SafeMotion des Antriebs
(D) SMMy-Statusbit von der SafeMotion des Antriebs
(E) Entspricht der Zeit am Eingang "MonitoringTime"
(1) Im Zustand "8002 OpMode" wird erkannt, dass am Bausteineingang die Anforderung für S_SMMxSel ansteht. Diese Anforderung wird ins DriveControl-Word eingetragen und an den Antrieb geschickt. Es wird die Monitoring Time gestartet und in den Zustand "8003 Wait for Confirmation" gewechselt.
(2) Bevor die Monitoring Time abgelaufen ist, meldet der Antrieb im Statuswort, dass die Funktion SMMx erreicht wurde. Dies wird am Statusausgang des Bausteins S_SMMxStatus angezeigt und es erflogt der Wechsel in den Zustand "8000 Drive in SMM".
(3) Im Zustand "8000 Drive in SMM" wird die Anforderung auf S_SMMxSel zurückgenommen und zeitgleich wird die Anforderung auf S_SMMySel am Bausteineingang gesetzt. Es erfolgt der Wechsel in den Zustand "8002 OpMode".
(4) Im Zustand "8002 OpMode" wird die Anforderung auf SMMx im DriveControl-Word zurückgenommen und zeitgleich die Anforderung SMMy in das DriveControl-Word geschrieben. Die Monitoring Time wird gestartet und es wird in den Zustand "8003 Wait for Confirmation" gewechselt.
(5) Bevor die Monitoring Time abgelaufen ist, meldet der Antrieb im Statuswort, dass die Funktion SMMy erreicht wurde. Das Erreichen wird am Statusausgang des Bausteins S_SMMyStatus angezeigt und es wird in den Zustand "8000 Drive in SMM" gewechselt.
(6) Im Zustand "8000 Drive in SMM" wird die Anforderung auf S_SMMySel am Bausteineingang zurückgenommen. Es erfolgt der Übergang in den Zustand "8002 OpMode" und die Anforderung im DriveControl-Word für SMMy wird zurückgenommen.

(7) Der Antrieb meldet über das DriveStatus-Word, dass der Zustand SMMy beendet wurde, dies wird am Statusausgang S_SMMyStatus des Bausteins angezeigt.

Abb. 16-148: SF_SMMSelector Timing Diagramm – Direkte Umschaltung von SMMx nach SMMy für einen Antrieb

Zustandsdiagramm



(A) Überwachung in allen Zuständen (nur für Profile / Assemblies 783, 785, 788, 789, 790)

(1) (AutoSetMode OR R_TRIG at S_SetMode) AND S_Unlock=TRUE AND (mehrere S_SMMxSel=TRUE)

(2) (AutoSetMode OR R_TRIG at S_SetMode) AND S_Unlock=TRUE AND (ein S_SMMxSel=TRUE)

NOT S_SMMxSel Genau die "S_SMMxSel", welche in den Zustand geführt hatte, wurde zurückgenommen (FALSE).

NO S_SMMxSel Alle "S_SMMxSel" (x: 1..8) sind FALSE

Abb. 16-149: Zustandsdiagramm SF_SMMSelector



Der Zustand "0x8002" spielt eine zentrale Rolle bei Kommandierung der SMM zum Antrieb:

- Die Anwahl einer SMMx, die Abwahl dieser SMMx als auch die direkte Umschaltung von SMMx nach SMMy wird im Zustand 0x8002 initiiert.
- Hier wird organisiert, dass eine SMM oder keine SMM aktiv ist.

Zustandsbeschreibung

DiagCode	Zustand	Beschreibung	Ausgänge
16#0000	Idle	FB ist nicht aktiv.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = FALSE
16#8000	Drive in SMMx	Antrieb ist in SMMx	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE
16#8001	Init	Zustand, nachdem der FB aktiviert wurde.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = FALSE
16#8002	Operation Mode	Normalbetrieb, es liegt keine SMM-Anforderung vor. Bei der direkten Umschaltung von SMMx nach SMMy wird der Zustand 16#8002 für einen Zyklus durchlaufen.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = FALSE
16#8003	Wait for Confirmation	Warten auf die Bestätigung des Antrieb. Beim Wechsel in diesen Zustand wird die Monitoring-Zeit neu gestartet.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = FALSE
16#8004	Motion Locked	Eine sichere Betriebsart wurde erreicht und der Anwender sperrt die Betriebsart. Unabhängig davon, was an der Eingängen des FBs anliegt, soll der Antrieb in der sicheren Betriebsart bleiben.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = FALSE
16#8005	Wait for NOT S_SMMxSel	Durch besondere Ereignisse, wie z.B. Monitoring Zeit abgelaufen, kommt man in einen Fehlerzustand. Aus dem Fehlerzustand kommt man durch eine Aufwärtsflanke an Reset. Jetzt wird gewartet bis alle Sicherheitsanforderungen zurückgenommen wurden, damit auf eine Neue reagieren werden kann.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = FALSE
16#C002	Acknowledge Lost	Der Antrieb wurde gestoppt und die Stop Anforderung steht noch an, das Bestätigungssignal ist jedoch nicht mehr vorhanden.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE

DiagCode	Zustand	Beschreibung	Ausgänge
16#C003	Monitoring Time Elapsed	Der Antrieb wurde nicht innerhalb der Monitoring Zeit in die angewählte SMM geschaltet.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE
16#C004	Reset Error 2	Statischer Reset erkannt im Zustand C002.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE
16#C005	Reset Error 3	Statischer Reset erkannt im Zustand C003.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE
16#C006	Sicherheitstechnik Error	Die Sicherheitstechnik im Antrieb meldet einen Fehler. Wird nur in Profilen mit den Assemblies 783, 785, 788, 789, 790 unterstützt.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE
16#C007	Reset Error 4	Statischer Reset erkannt im Zustand C006.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE
16#C008	Unzulässige Mehrfachselektion an S_SMMxSel	Es wurde mehr als ein S_SMMxSel gleichzeitig auf TRUE gesetzt oder keiner wenn S_SetMode TRUE wird.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE
16#C009	Reset Error 5	Statischer Reset erkannt im Zustand C008.	S_SMMxStatus = (1) S_AnySMMSel = (1) Error = TRUE
16#C010	AxisID nicht vorhanden	Kein Antrieb bzw. keine Achsgruppe mit dieser AxisID konfiguriert. Dieser Fehler wird im "Init"-Zustand erkannt.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = TRUE
16#C020	Nicht unterstütztes Profil	In der Antriebsparametrierung wurde ein nicht unterstütztes SafeMotion-Profil konfiguriert. Dieser Fehler wird im "Init"-Zustand erkannt.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = TRUE
16#C030	Instanz mit dieser AxisID schon vorhanden	Die AxisID wurde schon einmal mit einer anderen Instanz des Funktionsbausteins verwendet. Dieser Fehler wird im "Init"-Zustand erkannt.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = TRUE
16#C040	Eine Drive-Verbindung fehlt	Für die angelegte AxisID wurde keine CSos-Verbindung zum Antrieb gefunden. Dieser Fehler wird im "Init"-Zustand erkannt.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = TRUE

DiagCode	Zustand	Beschreibung	Ausgänge
16#C050	EStop für diese AxisID nicht programmiert	Für diese AxisID wurde kein Funktionsbaustein "SF_MotionEmergencyStop" angelegt. Siehe Kap. 16.6.1 "SF_MotionEmergencyStop" auf Seite 539. Dieser Fehler wird im "Init"-Zustand erkannt.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = TRUE
16#C060	AxisGroupID defined twice	Die angelegte AxisGroupID wurde zwei mal angemeldet und ist somit ungültig. Dieser Fehler wird im "Init"-Zustand erkannt.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = TRUE
16#C070	AxisID already used in axis group	Die angelegte AxisID wurde auch als AxisID an einem SF_AxisGroup-Funktionsbaustein verwendet und ist somit ungültig. Dieser Fehler wird im "Init"-Zustand erkannt.	S_SMMxStatus = FALSE(2) S_AnySMMSel = FALSE(2) Error = TRUE

(1) Die Rückmeldung von der Achse bzw. der Achsgruppe für den Zustand SMMx wird auf die Ausgänge "S_SMMxStatus" und "S_SMMAnySel" geschrieben.

(2) Bei einem SPS-Programm Neustart gilt: In den Zuständen 0000 Idle, 8001 Init und den Init-Fehlern C0x0 wird FALSE auf die Ausgänge "S_SMMxStatus" und "S_SMMAnySel" geschrieben, da die Statusinformation vom Antrieb nicht verfügbar ist.

Tab. 16-243: Zustandsbeschreibung SF_SMMSelector

16.6.4 SF_AxisGroup

Kurzbeschreibung

Der Funktionsbaustein SF_AxisGroup fasst mehrere Antriebe zu einer Achsgruppe zusammen. Diese kann dann mit den Funktionsbausteinen Kap. 16.6.1 "SF_MotionEmergencyStop" auf Seite 539, Kap. 16.6.2 "SF_MotionSafeStop" auf Seite 545 und Kap. 16.6.3 "SF_SMMSelector" auf Seite 552 verwendet werden.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyDriveMotion	2.8.0.0

Tab. 16-244: Bibliothekszuordnung

Schnittstellenbeschreibung

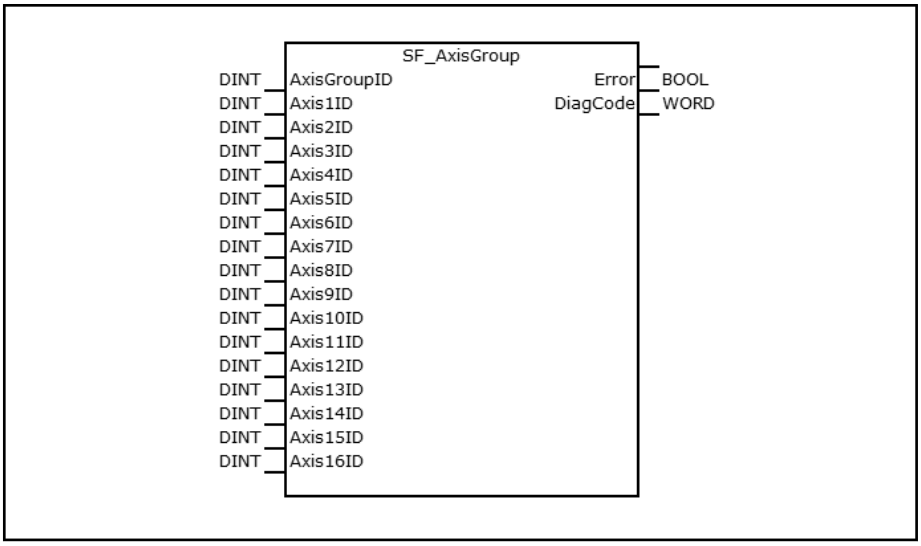


Abb. 16-150: Funktionsbaustein SF_AxisGroup

E/A-Typ	Name	Datentyp	Beschreibung
VAR_INPUT	AxisGroupID	DINT	Kennung der zur erstellenden Achsgruppe
	Axis1ID	DINT	SDID des 1. Antriebs der Achsgruppe
	Axis2ID	DINT	SDID des 2. Antriebs der Achsgruppe
	Axis3ID	DINT	SDID des 3. Antriebs der Achsgruppe
	Axis4ID	DINT	SDID des 4. Antriebs der Achsgruppe
	Axis5ID	DINT	SDID des 5. Antriebs der Achsgruppe
	Axis6ID	DINT	SDID des 6. Antriebs der Achsgruppe
	Axis7ID	DINT	SDID des 7. Antriebs der Achsgruppe
	Axis8ID	DINT	SDID des 8. Antriebs der Achsgruppe
	Axis9ID	DINT	SDID des 9. Antriebs der Achsgruppe
	Axis10ID	DINT	SDID des 10. Antriebs der Achsgruppe
	Axis11ID	DINT	SDID des 11. Antriebs der Achsgruppe
	Axis12ID	DINT	SDID des 12. Antriebs der Achsgruppe
	Axis13ID	DINT	SDID des 13. Antriebs der Achsgruppe
	Axis14ID	DINT	SDID des 14. Antriebs der Achsgruppe
	Axis15ID	DINT	SDID des 15. Antriebs der Achsgruppe
	Axis16ID	DINT	SDID des 16. Antriebs der Achsgruppe
VAR_OUTPUT	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Zustand des Funktionsbausteins bzw. Diagnose-Code (siehe Zustandsbeschreibung, Seite 571)

Tab. 16-245: Schnittstelle SF_AxisGroup

Name	Typ	Min	Max	Default-Wert	Übernahme
AxisGroupID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis1ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins

Name	Typ	Min	Max	Default-Wert	Übernahme
Axis2ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis3ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis4ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis5ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis6ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis7ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis8ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis9ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis10ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis11ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis12ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis13ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis14ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis15ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins
Axis16ID	DINT	0	32767	0	Bei erstem Aufruf des Bausteins

Tab. 16-246: Eingangsverhalten SF_AxisGroup

Normativer Verweis

Norm	Anforderung
DIN EN 61800-5-2:2008	4.2 Sicherheitsfunktionen

Funktionsbeschreibung

Der Funktionsbaustein SF_AxisGroup fasst mehrere Antriebe zu einer Achsgruppe zusammen. Diese kann dann mit den Funktionsbausteinen Kap. 16.6.1 "SF_MotionEmergencyStop" auf Seite 539, Kap. 16.6.2 "SF_MotionSafeStop" auf Seite 545 und Kap. 16.6.3 "SF_SMMSelector" auf Seite 552 verwendet werden, um die entsprechende sichere Betriebsart auf alle Antriebe der Achsgruppe zu übertragen.

Die Bildung einer Achsgruppe ist statisch und kann zur Laufzeit nicht geändert werden. Sie dient der vereinfachten Ansteuerung von mehreren Achsen mit der gleichen Sicherheitsfunktion.

Die zu einer Achsgruppe zusammengefassten Achsen müssen mit einem unterstütztes SafeMotion-Profil konfiguriert sein. Dabei müssen die Achsen einer Gruppe nicht das gleiche Profil verwenden. Achsen mit einem nicht unterstütztem Profil führen zu einem entsprechenden Fehlercode am Ausgang "DiagCode".

Der Funktionsbaustein überprüft im ersten Zyklus die folgenden Abhängigkeiten:

- Die an den Eingängen "Axis1ID" bis "Axis16ID" angelegten Achsen müssen ein tatsächlich vorhanden sein und ein unterstütztes SMO-Profil aufweisen.
- AxisIDs dürfen nicht doppelt angegeben werden.
- Dieselbe AxisID darf nicht in mehreren Achsgruppen verwendet werden.

- AxisGroupIDs dürfen sich nicht mit den AxisIDs normaler Achsen oder anderen AxisGroupIDs überschneiden.
- Achsgruppen können nicht nochmals als Achse in einen weiteren SF_AxisGroup-Funktionsbaustein angegeben werden.

Siehe auch:

- SF_MotionEmergencyStop, Seite 539
- SF_MotionSafeStop, Seite 545
- SF_SMMSelector, Seite 552

AxisGroupID

Kennung der zu erzeugenden Achsgruppe. Die Kennung darf sich nciht mit anderen Achskennungen oder bereits vorhandenen Achsgruppen überschneiden. Insgesamt können 32 Achsgruppen definiert werden. Für jede Achsgruppe wird eine Instanz des Funktionsbausteins benötigt.

Axis1ID .. Axis16ID

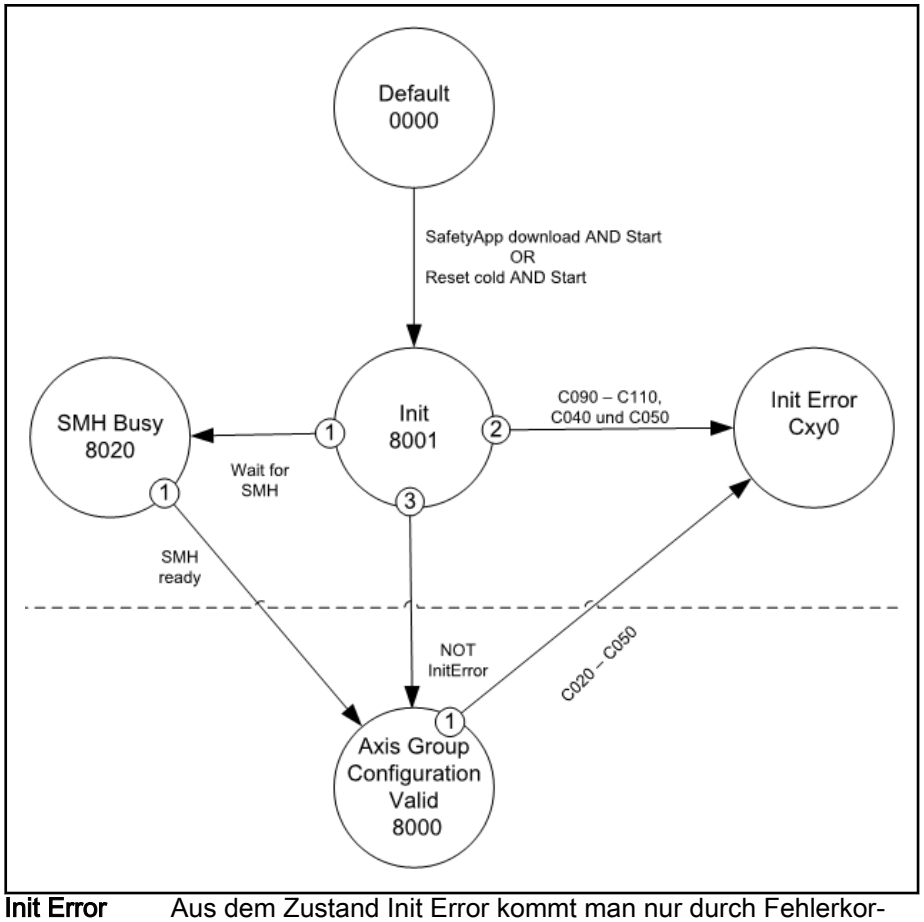
Am Eingang AxisID wird die SDID des anzusteuernden Antriebs angegeben. Sie bildet die logische Adresse aus Sicht der SafeLogic auf einen realen Antrieb. Eine einmal am Eingang AxisID angelegte SDID darf im weiteren Programmverlauf nicht mehr geändert werden.

Error, DiagCode

Tritt während der Überprüfung der Achsgruppe ein Fehler auf, so wird der Ausgang "Error" TRUE. Der Ausgang DiagCode entspricht dem derzeitig aktiven Zustand des Funktionsbausteins (s. "Zustandsdiagramm" auf Seite 571).

Ein Fehler kann nur durch Korrektur und Neustart des Safety-Programms behoben werden. In der Regel muss hierfür die Safety-Applikation geändert und neu heruntergeladen werden.

Zustandsdiagramm



Init Error Aus dem Zustand Init Error kommt man nur durch Fehlerkorrektur und SafetyApp download und Start.

Abb. 16-151: Zustandsdiagramm SF_AxisGroup

Zustandsbeschreibung

DiagCode	Zustand	Beschreibung	Ausgänge
16#0000	Default	Funktionsbaustein ist nicht aktiv.	Error = FALSE DiagCode = 16#0000
16#8000	Axis Group Configuration Valid	Alle Daten der Achsgruppe sind gültig. Die Achsgruppe kann verwendet werden.	Error = FALSE DiagCode = 16#8000
16#8001	Init	Initialisierung	Error = FALSE DiagCode = 16#8001
16#8020	Wait for SMH	Warten auf die Initialisierung des SafeMotion-Handlers. In jedem Zyklus wird ein Funktionsbaustein abgearbeitet.	Error = FALSE DiagCode = 16#8020
16#C020	Unsupported Assembly	Das im Antrieb konfiguerte SafeMotion-Profil (bestehend aus zwei Assemblies) wird nicht unterstützt.	Error = TRUE DiagCode = 16#C020
16#C040	No safe Connection	Es wurde eine AxisID angegeben für die keine gültige sichere Verbindung besteht. Dies tritt z.B. auf, wenn die SDID des logischen Status- und des Control-Geräts unterschiedlich sind.	Error = TRUE DiagCode = 16#C040

DiagCode	Zustand	Beschreibung	Ausgänge
16#C050	SF_MotionEmergencyStop FB missing	Die AxisGroupID wird von keinem SF_MotionEmergencyStop-Funktionsbaustein verwendet.	Error = TRUE DiagCode = 16#C050
16#C090	AxisGroupID is 0	Die angegebene AxisGroupID 0 ist nicht gültig.	Error = TRUE DiagCode = 16#C090
16#C0A0	No AxisIDs found	Es wurden keine gültigen AxisIDs angegeben. Alle AxisIDs sind 0	Error = TRUE DiagCode = 16#C0A0
16#C0B0	AxisGroupID already used as AxisID	Für die angegebene AxisGroupID existiert bereits eine AxisID, d.h. ein Antrieb hat bereits dieselbe AxisID (SDID) wie die AxisGroupID.	Error = TRUE DiagCode = 16#C0B0
16#C0C0	AxisID used multiple times	Die AxisID wurde mehrfach verwendet.	Error = TRUE DiagCode = 16#C0C0
16#C0D0	Too many Instances	Die maximale Anzahl von 32 AxisGroup Instanzen wurde überschritten.	Error = TRUE DiagCode = 16#C0D0
16#C0E0	AxisGroupID already used	Es wurde bereits eine Instanz mit dieser AxisGroupID angelegt.	Error = TRUE DiagCode = 16#C0E0
16#C0F0	AxisGroupID already used as AxisID	Die angegebene AxisGroupID wurde bereits als AxisID verwendet.	Error = TRUE DiagCode = 16#C0F0
16#C100	AxisID not found	Kein Antrieb mit dieser AxisID konfiguriert.	Error = TRUE DiagCode = 16#C100
16#C110	AxisID configured multiple times	Für diese AxisID exisiteren mehrere Antriebe.	Error = TRUE DiagCode = 16#C110

Tab. 16-247: Zustandsbeschreibung SF_AxisGroup

17 Feldbusse und Netzwerkvariablen

17.1 Feldbusse und Netzwerkvariablen, Allgemeines

Begriffsdefinitionen

Ausgangstelegramm ist das protokollspezifische E/A-Telegramm von der Sicherheitssteuerung in das sichere Feldgerät. Diese E/A-Telegramm beinhaltet die Ausgangsdaten an das sichere Feldgerät.

Eingangstelegramm ist das protokollspezifische E/A-Telegramm vom sicheren Feldgerät in die Sicherheitssteuerung. Diese E/A-Telegramm beinhaltet die Ausgangsdaten an das sichere Feldgerät.

Treiber-Instanz: für jedes konfigurierte logische E/A wird implizit Code für eine Treiber-Instanz des unterstützten Protokolltyps angelegt.

Für die Standardmodule und Austauschvariablen wird eine Treiber-Instanz angelegt:

- Logische E/As von Sicheren Feldgeräten (siehe Logische E/As von Sicheren Feldgeräten auf Seite 51)
- Logische E/As von Standard-Feldgeräten (siehe Logische E/As von Nicht-sicheren Feldgeräten auf Seite 51) - in Vorbereitung
- Ersatzwerte (siehe Ersatzwerte auf Seite 152)

Interface-Beschreibung

Die Sicherheitssteuerung überwacht die Übertragung der E/A-DATEN vom, bzw. zum sicheren Feldgerät.

Für jedes sichere Feldgerät wird ein logisches E/A vom Programmiersystem erzeugt.

Pro logisches Gerät wird automatisch eine Treiber-Instanz angelegt.

Mit dem Anlegen der Treiber-Instanz wird impliziter Code generiert, der für Eingangs- bzw. Ausgangsvariablen folgendes bewirkt:

- **Phase 1:** Bearbeitung der Eingangstelegramme (implizit)

  Die Treiberinstanz empfängt das Eingangstelegramm und überprüft dies gemäß dessen Protokollspezifikation, z. B. PROFIsafe.

  Die Eingangsdaten oder im Fehlerfall die Ersatzwerte werden in die gemappten Eingangsvariablen der Applikation kopiert.
- **Phase 2:** Bearbeitung der Anwenderapplikation.

  Die Ausgangsdaten werden in Abhängigkeit der Eingangsdaten und des Zustands der Applikation generiert.
- **Phase 3:** Bearbeitung der Ausgangstelegramme (implizit).

  Die gemappten Ausgangsdaten der Applikation werden der Treiberinstanz übergeben. Die Treiber-Instanz generiert das Ausgangstelegramm gemäß dessen Protokollspezifikation und schickt dieses an das sichere Feldgerät.

Default-Verhalten des Treibers

⚠ VORSICHT

Die Default-Werte der Eingänge sind so festgelegt, dass die Treiberinstanz automatisch anläuft und sofort Prozessdaten zwischen Feld und Applikation austauscht.

Durch die Implementierung anderer System und Applikationsmaßnahmen ist sicherzustellen, dass es beim Start der Applikation nicht zu einem unerwarteten (oder unbeabsichtigten) Starten der Maschine kommt.

☞ Die Wiederaufnahme der Prozessdatenübertragung nach einem Kommunikationsfehler ist nicht automatisch

Das Default-Verhalten des Treibers für das Anlaufen nach einem Reset bzw. für das Wiederanlaufen nach einem Kommunikationsfehler wird von dem Initialwert des jeweiligen Eingangs definiert. Um das Default-Verhalten zu überschreiben, muss der Baustein in der Applikation aufgerufen werden und der entsprechende Eingang entsprechend belegt werden:

Der automatische Beginn der Prozessdatenübertragung nach dem Start wird programmatisch verhindert, indem der Baustein in der Applikation aufgerufen und der Eingang <auto-Quitt-Anlauffehler> auf FALSE gesetzt wird (siehe Eingang für automatische Quittierung von Anlauffehlern auf Seite 574).

Die automatische Wiederaufnahme der Prozessdatenübertragung wird programmatisch freigegeben, indem der Baustein in der Applikation aufgerufen und der Eingang <auto-Quitt-Unterbrechung> auf TRUE gesetzt werden (siehe Eingang für automatische Quittierung nach Unterbrechung auf Seite 574).

Eingang für automatische Quittierung von Anlauffehlern

Name	Datentyp	Initialwert	Beschreibung, Parameterwerte
<auto-Quitt-Anlauffehler>	BOOL	TRUE	Anlaufverhalten nach einem Reset (Befehle: Debuggen-Befehle: Start, Stop, Reset kalt , Seite 150 und Bootapplikation starten, Seite 210) der Applikation, z. B. PowerON. TRUE: Automatische Quittierung von Fehlern während der Anlaufphase der sicheren Kommunikation bis einmal die sichere Übertragung aufgenommen worden ist. FALSE: Explizite, applikative Quittierung von aufgetretenen Fehlern während der Anlaufphase der sicheren Kommunikation erforderlich.

Tab. 17-1: Eingang für automatische Quittierung von Anlauffehler

Sicherheitshinweis für Eingang für automatische Quittierung von Anlauffehler

⚠ VORSICHT **Der Initialwert für den Eingang für das Anlaufverhalten nach einem Reset ist TRUE. Es werden sämtliche Fehler für das Anlaufverhalten nach einem Reset bestätigt.**

☞ Treten beim Anlauf keine Fehler auf, so kann der Stack automatisch anlaufen, selbst wenn auto-Quitt- Anlauffehler auf FALSE ist. Dies muss der Anwender in der Applikation berücksichtigen.

Eingang für automatische Quittierung nach Unterbrechung

Name	Datentyp	Initialwert	Beschreibung, Parameterwerte
<auto-Quitt-Unterbrechung>	BOOL	FALSE	TRUE: Automatische Quittierung nach einem Kommunikationsfehler. FALSE: Explizite, applikative Quittierung von Kommunikationsfehler erforderlich.

Tab. 17-2: Eingang für automatische Quittierung nach Unterbrechung

Sicherheitshinweis für Eingang für automatische Quittierung nach Unterbrechung

⚠ VORSICHT

Sind <auto-Quitt-Anlauffehler> **und** <auto-Quitt-Unterbrechung> **TRUE, so werden sämtliche Fehler bestätigt. Dies ist nur in absoluten Ausnahmefällen sinnvoll.**

☞ Der Initialwert für den Eingang für das Wiederanlaufverhalten nach einem Kommunikationsfehler ist FALSE, es wird der Fehler bzgl. Kommunikationsübertragung nicht automatisch bestätigt.

Es ist ein expliziter Aufruf der Bausteininstanz mit entsprechender Beschaltung der Eingänge erforderlich.

Eingang zur Quittierungsflanke (manuelle Quittierung)

Fehler können mit einer positiven Flanke am Eingang zur Quittierung bestätigt werden, sofern der Ausgang für die Quittierungsanforderung (Ausgang zur Quittierungsanforderung auf Seite 576) gesetzt ist.

Wenn sämtliche Fehler automatisch bestätigt werden (was nur in Ausnahmefällen sinnvoll ist), dann wird dieser Eingang nicht benötigt und kann unbelegt bleiben.

Wenn aktuell keine Quittierung angefordert wird, wird der Eingang ignoriert: Deshalb kann das gleiche Signal mit dem Eingang zur Quittierungsflanke aller Treiberinstanzen verbunden werden, um eine unspezifische Bestätigung von Kommunikationsproblemen zu realisieren.

Name	Datentyp	Initialwert	Beschreibung, Parameterwerte
<Quitt-Flanke>	BOOL	FALSE	Mit einer steigenden Flanke des Eingangs wird die Wiederaufnahme der Sicherheitsfunktion nach einem Fehler bestätigt.

Tab. 17-3: Eingang zur Quittierungsflanke (manuelle Quittierung)

Hinweise zu Quitt-Flanke

☞ Quitt-Flanke ist ein Eingang zur Bestätigung durch den Anwender. Er soll nicht programmatisch gesetzt, sondern mit einem Eingangssignal verbunden werden.

☞ Der Eingang Quitt-Flanke benötigt eine steigende Flanke. Er sollte, nachdem der Anwender die Bestätigung gestoppt hat, wieder auf FALSE gehen, um die Quittierung abzuschließen. Erst dann wird beim nächsten Kommunikationsfehler am Ausgang Quitt-Anf die nächste Quittierung angefordert.

Ausgang zur Quittierungsanforderung

Der Ausgang steht auf TRUE, wenn ein Kommunikationsfehler aufgetreten ist (Anlauffehler oder Unterbrechung) und dieser nur manuell bestätigt werden muss, um die Kommunikation wieder aufzunehmen.

Ob die Verbindung eine Bestätigung durch den Anwender benötigt, wird durch den Ausgang Quitt-Anf = TRUE angezeigt. Normalerweise wird es dem Anwender gemeldet, dass seine Bestätigung gebraucht wird (zum Beispiel mithilfe einer Austauschvariablen zur Hauptsteuerung und Anzeige in der Visualisierung).

Wenn der Ausgang gesetzt ist, kann der Fehler am Eingang zur Quittierung bestätigt werden.

Name	Datentyp	Initialwert	Beschreibung, Parameterwerte
<Quitt-Anf>	BOOL	FALSE	

Tab. 17-4: Ausgang zur Quittierungsanforderung



Die Anzeige am Ausgang Quitt-Anf kann nur erfolgen:

1. wenn der Kommunikationsfehler nicht automatisch quittiert wird, d. h. wenn auto-Quitt-Anlauffehler bzw. auto-Quitt-Unterbrechung FALSE ist.
2. wenn der Eingang "Quitt-Flanke" momentan FALSE ist, d. h. wenn eine vorherige manuelle Quittierung noch nicht abgeschlossen ist.

Falls also ein neues Kommunikationsproblem auftritt, während der Anwender ein vorher aufgetretenes Problem noch bestätigt, so wird das neue Kommunikationsproblem erst angezeigt und bearbeitet, nachdem der Anwender die Bestätigung des vorherigen Problems beendet hat.

Explizites Aufrufen des Bausteins

Wird der Baustein vom Anwender in der Applikation explizit aufgerufen (Phase 2), dann können die FB-Eingänge gesetzt und die FB-Ausgänge gelesen werden, wobei der FB selbst keine Operationen ausführt.

Wird der Baustein in der Applikation ein zweites Mal aufgerufen und die Eingänge werden ein zweites Mal gesetzt, so werden die Eingänge des ersten Aufrufs überschrieben, ohne dass der FB diese Eingänge ausgewertet hat, da die Auswertung der Eingänge nur in der Phase 3 und in Phase 1 stattfindet.



Die Treiber-Instanz sollte in einer Applikation nur einmal explizit aufgerufen werden, da die Eingänge ansonsten vor ihrer Auswertung überschrieben werden.

unterstützte Feldbusse

Die folgenden Kombinationen von sicheren Feldbussen werden unterstützt:

sicherer Feldbus	Feldbus	Richtung	Bemerkung
PROFIsafe	Profibus	Host	PROFIsafe V2.4
	PROFINET		
CSos	Sercos	Originator	CIP Safety Ed. 2.6 bis zu 2 Byte Nutzdaten

sicherer Feldbus	Feldbus	Richtung	Bemerkung
NVL	UDP über Ethernet	Sender	Sender für Safety Netzwerkvariablen
		Receiver	Empfänger für Safety Netzwerkvariablen

Tab. 17-5: unterstützte Feldbusse

Randbedingungen

Insgesamt sind 99 sichere Feldgeräte zulässig (PROFIsafe-, CSos-Feldgeräte, NVL).

17.2 SafetyProfisafeHost.library

Anwendbare Sicherheits-Standards

PROFIsafe ist eine international genormte (IEC 61784-3-3) Technologie für funktional sichere Kommunikation.

Grundlage ist: "Profisafe-Profile for Safety Technology on Profibus DP and Profinet IO", Version 2.4, March 2007, Order No. 3.192b

Die Erkennung des Loopback-Fehlers ist realisiert für F-Host (LP) gemäß:

"Profisafe-Profile for Safety Technology on Profibus DP and Profinet IO", Version 2.6, October 2013 Order No. 3.192b

Begriffsdefinitionen

F-Device ist ein sicheres, dezentrales Feldgerät, das die PROFIsafe-Kommunikation unterstützt.

Zur Vereinfachung der Lesbarkeit wird **F-Device** in dieser Dokumentation als Oberbegriff für F-Device, F-Slave und F-Module verwendet.

Erklärung zu Default-Verhalten des PROFIsafeHost Stack siehe "Default-Verhalten des Treibers" auf Seite 573.

⚠ VORSICHT **Kommunikationsbeziehungen pro Sicherheitsfunktion**

Für SIL3 Sicherheitsfunktionen erlaubt die PROFIsafe Norm maximal 100 Profisafe-Geräte, damit die durchschnittliche Wahrscheinlichkeit gefährlicher Fehler pro Stunde (PFH) unter dem SIL3-Grenzwert von 10^{-9} bleibt.

Für SIL2 Sicherheitsfunktionen erlaubt die PROFIsafe Norm bis zu 1000 Profisafe-Geräte, da die Wahrscheinlichkeit gefährlicher Fehler pro Stunde (PFH) um 4 x 10^{-12} für jedes zusätzliche Gerät ansteigt.

⚠ VORSICHT **SIL Monitor**

Diese Implementierung des PROFIsafe F-Host unterstützt die Variante B des SIL Monitors. Jeder CRC-, Timeout- oder Loopback-Fehler über das empfangene Telegramm führt zu einer Fehlerreaktion.

Information für Maschinenbetreiber: Für den Fall, dass ein manueller „Operator Acknowledgment“ (ausgelöst durch ein CRC-, Timeout- oder Loopback-Fehler) häufiger als 1x pro 100 Stunden angefordert wird, wird dringend empfohlen, eine Problemanalyse durch entsprechende Servicespezialisten durchzuführen. (Fehlerzustand des PROFIsafeHost 16#C103,16#C102, 16#C101, Fehlerzustand des F-Device: 16#C2XX Bit 2, Bit 3.)

Information für Betreiber und Servicetechniker: Dies stellt eine starke Beeinträchtigung der Datenübertragung innerhalb des Feldbussystems dar. Gründe für diese Störungen können sein: Änderungen in der Installation, Korrosion von Buskabelabschirmungen mit Stecker und extreme elektromagnetische Interferenzen. Die Übereinstimmung mit den entsprechenden Installationsrichtlinien sollte geprüft, oder es sollte ein EMV Experte hinzugezogen werden. Installationsrichtlinien der PNO: PROFINET Guideline: PROFINET Cabling and Interconnection Technology, Version 3.0.1, October 2011. Order-No. 2.252 , PROFIBUS Guideline: Cabling and Assembly, V1.0.6, May 2006. Order No. 8.022, PROFIsafe - Requirements for Installation, Immunity, electrical Safety and Security for PROFIBUS DP and PROFINET IO, Version 2.5, March 2007, Order No: 2.232

Funktionsweise

PROFIsafe arbeitet verbindungsorientiert:

Der F-Host unterhält zu jedem F-Device eine eigenständig lebensfähige PROFIsafe-Verbindung. Die PROFIsafe-Verbindung umfasst Eingangs- und Ausgangsdaten. Grundlage ist die zyklische Kommunikation Profibus DP V0 bzw. PROFINET IO.

Der Funktionsbaustein ProfisafeHost implementiert nach der Profisafe-Spezifikation einen Profisafe-F-Host. Der ProfisafeHost-FB überwacht die Übertragung der E/A-DATEN vom und zum sicheren PROFIsafe-F-Device.

Pro F-Device gibt es eine PROFIsafe-Verbindung, die über eine FB-Instanz des ProfisafeHost überwacht und verwaltet wird.

Die PROFIsafe-Verbindung ist durch die Kombination der Parameter F_Source_Add und F_Dest_Add gekennzeichnet.

⚠ VORSICHT **Es liegt in der Verantwortung des Anwenders, dass jede PROFIsafe-Verbindung eine eindeutige Kennzeichnung erhält.**

Beachten Sie die Hinweise zu den F-Parametern:

- F_Source_Add, Seite 587
- F_Dest_Add, Seite 588

- Der ProfisafeHost unterstützt V2 mode F-Devices bis einschließlich Profisafe-Version 2.4
- V1 mode F-Slaves werden nicht unterstützt
- Es können maximal 12 Byte Nettodaten übertragen werden
- Der ProfisafeHost unterstützt bis zu 99 F-Devices (Insgesamt sind max. 99 sichere Feldgerät zulässig: PROFIsafe- und CSos-Feldgeräte !)

- Der ProfisafeHost unterstützt F-Devices auf Basis Profibus DP und auf Basis PROFINET IO (F-Slaves, F-Devices, F-Module)
- Es dürfen nur zertifizierte PPROFIBUS / PROFINET-Geräte verwendet werden
-

Unterstützte Feldbusse

PROFIsafe wird über PROFIBUS und PROFINET unterstützt. Es ist nur der F-Host implementiert, d. h. die sichere Kommunikation geht immer von der SafeLogic-Steuerung aus.

17.2.1 ProfisafeHost

Kurzbeschreibung

Der Funktionsbaustein ProfisafeHost implementiert nach der Profisafe-Spezifikation einen Profisafe-F-Host.

Der ProfisafeHost-FB überwacht und verwaltet die Übertragung der E/A-DATEN vom und zum sicheren Profisafe-F-Device (Profisafe-Verbindung).

Wird der ProfisafeHost-FB in der Applikation SafetyApp verwendet, kann der Anwender Einfluss auf die Profisafe-Verbindung nehmen.

⚠ VORSICHT

Die Default-Werte der Eingänge sind so festgelegt, dass der ProfisafeHost Stack automatisch anläuft und die Aufnahme der Sicherheitsfunktion automatisch bestätigt.

Durch die Implementierung anderer System- und Applikationsmaßnahmen ist sicherzustellen, dass es beim Start der Applikation nicht zu einem unerwarteten (oder unbeabsichtigten) Starten der Maschine kommt.

Das automatische Anlaufen kann programmatisch verhindert werden, indem der Baustein in der Applikation aufgerufen und der Eingang StartOA auf FALSE gesetzt wird.

Die Wiederaufnahme der Prozessdatenübertragung nach einem Kommunikationsfehler ist nicht automatisch.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyProfisafeHost	1.1.1.0

Tab. 17-6: Bibliothekszuordnung

Schnittstellenbeschreibung

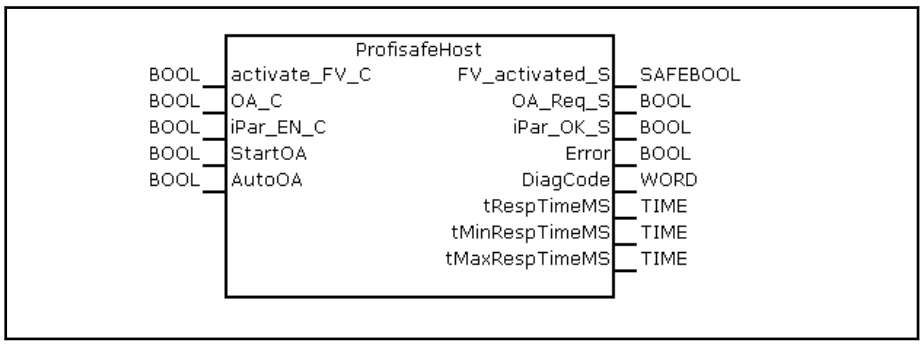


Abb. 17-1: Funktionsbaustein ProfilsafeHost

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	activate_FV_C	BOOL	(Eingang nach Profisafe-Spezifikation Kap. 6.1: Aktivierung der Fail-safe Werte) TRUE: Die Verwendung von Fail-safe Werten wird erzwungen: Der ProfisafeHost-Stack liefert als Eingangssignale die Fail-safe Werte an die Applikation und schreibt fail-safe Werte als Ausgangssignale an das F-Device. FALSE: Die Verwendung von fail-safe Werten wird nicht erzwungen. Es sollen im Normalfall Prozessdaten übertragen werden. **Vorsicht:** Das Rücksetzen dieses Eingangs von TRUE auf FALSE kann zur sofortigen Übertragung der Prozessdaten führen! Stellen Sie durch geeignete Maßnahmen sicher, dass von Ihrer Anlage/Maschine keine Gefahr ausgeht! Default-Wert: FALSE
	OA_C	BOOL	(Eingang nach Profisafe-Spezifikation, Kap. 6.1: Operator Acknowledgement) Mit einer steigenden Flanke des Eingangs wird die Wiederaufnahme der Sicherheitsfunktion nach einem Fehler bestätigt. Nach einer Bestätigung muss der Eingang mindestens einen vollen Zyklus auf FALSE stehen, bevor der nächste Fehler bestätigt werden kann. **Hinweis:** Wenn sowohl StartOA, als auch AutoOA auf TRUE gesetzt sind, wird der Eingang OA_C nicht ausgewertet, da alle Anfragen zur Aufnahme der Sicherheitsfunktion automatisch bestätigt werden. **Hinweise:** OA_C ist ein Eingang zur Bestätigung durch den Anwender. Er soll mit einem Eingangssignal verbunden werden und nicht programmatisch generiert werden. Default-Wert: FALSE **Hinweis:** Wenn aktuell keine Quittierung (OA_Req_S ist FALSE) angefordert ist, wird der Eingang ignoriert: Deshalb kann das gleiche Signal mit dem Eingang zur Quittierungsflanke aller FB-Instanzen verbunden werden, um eine unspezifische Bestätigung von Kommunikationsproblemen zu realisieren.
	iPar_EN_C	BOOL	(Eingang nach Profisafe-Spezifikation, Kap. 6.1: Aktivierung der I-Parametrierung des F-Device) Dieser Eingang wird im Kontext einer gerätespezifischen I-Parametrierung unter Nutzung eines Proxy-FBs verwendet. Falls erforderlich, sollte auch der Eingang acitvate_FV_C auf 1 gesetzt werden. Details entnehmen Sie der Dokumentation des F-Device. TRUE: Die I-Parametrierung des F-Device wird aktiviert. FALSE: Die I-Parametrierung des F-Device wird nicht aktiviert. Default-Wert: FALSE

E/A-TYP	Name	Datentyp	Beschreibung
	StartOA	BOOL	Verhalten nach einem Fehler während der Anlaufphase des F-Device. Die Wirksamkeit des Eingangs StartOA erstreckt sich vom erster Start der Applikation bis das F-Device einmal die Sicherheitsfunktion aufgenommen hat. (z.B. nach einem Reset kalt oder einem Neustart der Bootapplikation bzw. PowerON) TRUE: Automatische Bestätigung zur Aufnahme der Sicherheitsfunktion nach einem Fehler. FALSE: Manuelle Bestätigung am Eingang OA_C zur Aufnahme der Sicherheitsfunktion des F-Device nach einem Fehler notwendig. Default-Wert: TRUE Sicherheitshinweis zu StartOA siehe Sicherheitshinweis zu StartOA auf Seite 582
	AutoOA	BOOL	Wiederanlaufverhalten nach einem Fehler. Vorbedingung: Das F-Device ist erfolgreich angelaufen und hat mindestens einmal die Sicherheitsfunktion aufgenommen. TRUE: Automatische Bestätigung zur Wiederaufnahme der Sicherheitsfunktion nach einem Fehler. FALSE: Manuelle Bestätigung am Eingang OA_C zur Wiederaufnahme der Sicherheitsfunktion des F-Device nach einem Fehler notwendig. Default-Wert: FALSE Sicherheitshinweis zu AutoOA siehe Sicherheitshinweis zu AutoOA auf Seite 582
VAR_OUTPUT	FV_activated_S	SAFEBOOL	(Ausgang nach Profisafe-Spezifikation, Kap. 6.1: Anzeige Aktivierung der Fail-safe Werte) TRUE: Der ProfisafeHost-Stack liefert Fail-safe Werte. FALSE: Der ProfisafeHost-Stack liefert Prozessdaten. **Hinweis:** Der Ausgang wechselt auf TRUE, entweder auf Anweisung des Stacks oder aufgrund Übertragungsfehler zum F-Device bzw. interner Fehler im F-Device oder falls der Eingang **activate_FV_C** gleich TRUE ist.
	OA_Req_S	BOOL	(Ausgang nach Profisafe-Spezifikation, Kap.6.1: Anzeige Operator Acknowledgement Request) TRUE: Das F-Device benötigt eine Bestätigung durch den Bediener (Operator Acknowledgement), um die Sicherheitsfunktion wieder aufzunehmen. FALSE: Das F-Device benötigt derzeit keine Bestätigung durch den Bediener (Operator Acknowledgement). **Hinweis:** Dass eine PROFIsafe-Verbindung eine Bestätigung durch den Bediener benötigt, wird durch OA_Req_S = TRUE signalisiert. Normalerweise wird dies dem Bediener gemeldet, Die entsprechende Logic muss in der Applikation programmiert werden.

E/A-TYP	Name	Datentyp	Beschreibung
	iPar_OK_S	BOOL	(Ausgang nach Profisafe-Spezifikation, Kap. 6.1: Anzeige Ende I-Parametrierung des F-Device) Dieser Ausgang wird im Kontext einer gerätespezifischen I-Parametrierung unter Nutzung eines Proxy-FBs verwendet. Details entnehmen Sie der Dokumentation des F-Device. TRUE: Das F-Device hat neue I-Parameter erhalten. FALSE: Das F-Device hat keine neuen I-Parameter erhalten.
	Error	BOOL	Fehleranzeige TRUE: Es liegt ein Fehler vor (DiagCode = 16#Cxxx). FALSE: Es liegt kein Fehler vor (DiagCode = 16#8xxx).
	DiagCode	WORD	Verwendung des Ausgangs DiagCode als Diagnose-Code, Initialwert 16#0000 (siehe Diagnose-Code auf Seite 584)
	tRespTimeMS	TIME	Ausgang, der die aktuelle Durchlaufzeit für eine Nachricht angibt (siehe Zeitmessung auf Seite 586).
	tMinRespTimeMS	TIME	Ausgang, der die minimale Durchlaufzeit für eine Nachricht angibt (siehe Zeitmessung auf Seite 586). Ermittelt seit der letzten steigenden Flanke von OA_C oder Systemstart.
	tMaxRespTimeMS	TIME	Ausgang, der die maximale Durchlaufzeit für eine Nachricht angibt (siehe Zeitmessung auf Seite 586). Ermittelt seit der letzten steigenden Flanke von OA_C oder Systemstart.

Tab. 17-7: Schnittstelle ProfisafeHost

Name	Typ	Default-Wert	Übernahme
activate_FV_C	BOOL	FALSE	kontinuierlich
OA_C	BOOL	FALSE	kontinuierlich (mit Einschränkungen)
iPar_EN_C	BOOL	FALSE	kontinuierlich
StartOA	BOOL	TRUE	kontinuierlich
AutoOA	BOOL	FALSE	kontinuierlich

Tab. 17-8: Eingangsverhalten ProfisafeHost

Sicherheitshinweis zu StartOA

⚠ VORSICHT **StartOA**

Sind StartOA und AutoOA TRUE, so werden sämtliche Fehler bestätigt. Dies ist nur in absoluten Ausnahmefällen sinnvoll.

Der Initialwert für StartOA nach einem Reset ist TRUE. Es werden sämtliche Fehler für das Anlaufverhalten nach einem Reset bestätigt.

Treten beim Anlauf keine Fehler auf, so kann der Stack automatisch anlaufen, selbst wenn StartOA FALSE ist. Dies muss der Anwender in der Applikation berücksichtigen.

Sicherheitshinweis zu AutoOA

⚠ VORSICHT **AutoOA**

Sind StartOA und AutoOA TRUE, so werden sämtliche Fehler bestätigt. Dies ist nur in absoluten Ausnahmefällen sinnvoll.

Funktionsbeschreibung

Je F-Device gibt es eine Profisafe-Verbindung, die über eine FB-Instanz des ProfisafeHost überwacht und verwaltet wird.



Für jedes Profisafe-F-Device wird automatisch vom Programmiersystem ein logisches E/A und eine Instanz des Funktionsbausteins ProfisafeHost (FB-Instanz) erzeugt. Das entspricht dem üblichen Vorgehen bei logischen E/A.

Beim Anlegen der FB-Instanz vom Typ ProfisafeHost wird impliziter Code generiert, der für Eingangs- bzw. Ausgangsvariablen folgendes bewirkt:

- **Phase 1:** Bearbeitung der Eingangstelegramme (implizit):

  Das Eingangstelegramm des ProfisafeHost-Stacks ist die Antwort des F-Device an den F-Host.

  Die FB-Instanz empfängt das Eingangstelegramm vom Feldgerät und überprüft dies gemäß dessen PROFIsafe-Spezifikation.

  Die Eingangsdaten oder im Fehlerfall die Ersatzwerte werden in die gemappten Eingangsvariablen der Applikation kopiert.

- **Phase 2:** Bearbeitung der Anwenderapplikation:

  Die Ausgangsdaten werden in Abhängigkeit der Eingangsdaten und des Zustands der Applikation generiert.

- **Phase 3:** Bearbeitung der Ausgangstelegramme (implizit):

  Die gemappten Ausgangsdaten der Applikation werden der FB-Instanz übergeben. Die FB-Instanz generiert das Ausgangstelegramm gemäß der PROFIsafe-Spezifikation und schickt dieses an das sichere Feldgerät.

  Das Ausgangstelegramm des ProfisafeHost-Stacks ist die Anfrage des F-Hosts an das F-Device.

Wird der Funktionsbaustein ProfisafeHost vom Anwender in der Applikation explizit aufgerufen (Phase 2), dann können die FB-Eingänge gesetzt und die FB-Ausgänge gelesen werden, wobei der FB selbst keine Operationen ausführt.

Wird der Funktionsbaustein in der Applikation ein zweites Mal aufgerufen und die Eingänge werden ein zweites Mal gesetzt, so werden die Eingänge des ersten Aufrufs überschrieben, ohne dass der FB diese Eingänge ausgewertet hat. Ursache ist, dass die Auswertung der Eingänge nur in der Phase 3 und in der Phase 1 stattfindet.



Aufruf von Funktionsbausteinen unter Safety-Bedingungen siehe FB-Aufrufe auf Seite 112



Funktionsbausteine vom Typ ProfisafeHost sollten in einer Applikation nur einmal explizit aufgerufen werden, da die Eingänge ansonsten vor ihrer Auswertung überschrieben werden.

☞ Der Default-Wert des Eingangs "StartOA" ist mit "TRUE" so festgelegt, dass der ProfisafeHost-Stack automatisch anläuft. D. h. dass evtl. notwendige Bestätigungen (OA) für die Aufnahme der Sicherheitsfunktion nach Fehlern automatisch im FB generiert werden.

Der Default-Wert des Eingangs "AutoOA" ist "FALSE": Die Bestätigung zur Wiederaufnahme der Sicherheitsfunktion nach Fehlern muss manuell erfolgen.

Erkennung Loopback-Fehler

Loopback-Fehler werden vom ProfisafeHost erkannt und mit Diagnose-Code 16#C101 angezeigt.

Für die Wiederaufnahme der Sicherheitsfunktion nach einen Loopback-Fehler ist eine Bestätigung (**O**perator **A**cknowledgement) notwendig.

Verhalten des ProfiSafeHost-FB bezüglich "Device_Fault"-Bit

Das PROFIsafe F-Device kann mit dem Bit "Device_Fault" gleich TRUE eine Fehlersituation signalisieren. Der PROFIsafe F-Host erkennt, dass das Bit "Device_Fault" TRUE ist und aktiviert für die PROFIsafe-Verbindung Failsafe-Werte ("FV_activated_S" ist TRUE).

Sobald das Bit "Device_Fault" von TRUE auf FALSE wechselt, werden auf der PROFIsafe-Verbindung wieder Prozesswerte übertragen ("FV_activated_S" ist FALSE). Der PROFIsafe F-Host erzwingt hierfür gemäß PROFIsafe-Spezifikation Version 2.4 keine Quittierung!

☞ Stellen Sie durch geeignete System- und Applikationsmaßnahmen sicher, dass es nicht zu einem unerwarteten (oder unbeabsichtigten) Wiederanlauf der Maschine kommt!

Als geeignete Applikationsmaßnahme kann das Signal "FV_activated_S" in die Nothalt-Kette aufgenommen und gegen Wiederanlauf verriegelt werden.

Verwendung der ProfisafeHost-FB-Instanz in der Applikation

Der Anwender sollte den Funktionsbaustein in der Sicherheitsapplikation verwenden zur

- Änderung der Default-Werte (StartOA, AutoOA)
- manuellen Bestätigung von Fehlern (OA_C, OA_Req_S)
- Diagnose
- gezieltes Versetzen eines F-Device in den sicheren Zustand (activate_FV_C)
- gerätespezifische Verwaltung der I-Parameter unter Nutzung eines Proxy-FBs (iPar_EN_C, iPar_OK_S)

Diagnose-Code

Der Parameter DiagCode ist in zwei Kategorien aufgeteilt.

- 16#8XXX für Zustände der Kommunikation mit Übertragung (siehe Zustände der Kommunikation mit Übertragung auf Seite 584)
- 16#CXXX Fehlerzustände (siehe Fehlerzustände auf Seite 585)

Zustände der Kommunikation mit Übertragung

Der Diagnose-Code kann in den Zuständen der Kommunikation mit Übertragung folgende Werte annehmen:

DiagCode = **16#80XX**, XX enthält den Zustand des Profisafe-Hosts gemäß "F-Host state diagram" der Profisafe-Spezifikation:

DiagCode	Beschreibung
16#8001	(Initialisierung) "Systemstart": Anfangszustand des Profisafe-Hosts nach Systemstart.
16#8002	(Initialisierung) "Prepare Message": Vorbereitung der nächsten Nachricht für das F-Device.
16#8003	(Initialisierung) "Await Device Ack": Warten auf die Antwort des F-Device
16#8004	(Initialisierung) "Check Device Ack": Überprüfung der Antwort des F-Device (CRC, Consecutive number, F-Device Fehler)
16#8005	(Datenaustausch) "Prepare Message": Vorbereitung der nächsten Nachricht für das F-Device.
16#8006	(Datenaustausch) "Await Device Ack": Warten auf die Antwort des F-Device
16#8007	(Datenaustausch) "Check Device Ack": Überprüfung der Antwort des F-Device (CRC, Consecutive number, F-Device Fehler)

Fehlerzustände

Liegen im Profisafe-Host gleichzeitig zwei Fehler vor, wird immer nur der Fehler mit der höchsten Priorität angezeigt. Die Prioritäten der Fehler sind nach der Fehlerkategorie zugeteilt:

1. Fehlerkategorie 16#C0XX (höchste Priorität)
2. Fehlerkategorie 16#C1XX
3. Fehlerkategorie 16#C2XX (niedrigste Priorität)

Der Diagnose-Code kann im Fehlerzustand folgende Werte (aufgeteilt in 3 Fehlerkategorien) annehmen:

Konfigurations- und Initialisierungsfehler des ProfisafeHost

DiagCode = 16#C0XX (Konfigurations-und Initialisierungsfehler des ProfisafeHost), XX enthält den Fehler-spezifischen Fehlercode:

DiagCode	Beschreibung
16#C000	Ungültiger Aufruf des ProfisafeHost-Stacks
16#C001	Zeiger auf die F-Parameter ist ungültig
16#C002	Ungültige Werte der Instanzdaten
16#C003	Anzahl der F-Parameter ist nicht korrekt
16#C004	Konfigurierte Länge der CRC wird nicht unterstützt (es wird nur Profisafe V2 unterstützt)
16#C005	Anzahl der F-Parameter ungültig
16#C006	Länge der F-Parameter entspricht nicht der F-Parameter F_Block_ID
16#C007	Profisafe-Host-Version wird nicht unterstützt (es wird nur Profisafe V2 unterstützt)
16#C008	Ungültiger Wert für F-Parameter F_Par_CRC
16#C020	Ungültige Länge der E/A-Telegramme (4 <= Telegrammlänge <= 12)
16#C030	Das unterlagerte E/A-System konnte nicht initialisiert werden. Es trat ein interner Fehler bei den Konfigurationsdaten auf.



Zusätzlich werden Konfigurations- und Initialisierungsfehler des ProfisafeHost beim Download in Form einer Fehlermitteilung angezeigt.

Fehlerzustand des ProfisafeHost

DiagCode = 16#C1XX (Fehlerzustand des ProfisafeHost), XX enthält den Hostfehler-spezifischen Fehlercode:

DiagCode	Beschreibung
16#C101	Loopback-Fehler erkannt, ein Ausgangstelegramm des ProfisafeHost-Stacks wurde als Eingangstelegramm empfangen (Kurzschlussfehler)
16#C102	Überwachungszeitfehler, kein gültiges Telegramm innerhalb der F_WD_Time (F-Watchdog time) erhalten
16#C103	CRC-Fehler über das empfangene Telegramm
16#C104	Der ProfisafeHost-Stack wartet auf ein Operator Acknowledgement

Fehlerzustand des F-Device

DiagCode = 16#C2XX (Fehlerzustand des F-Device, der an den F-Host übertragen wurde), XX enthält das Statusbyte (vgl. Profisafe-Spezifikation, Kap. 7.1.3) des F-Device (Bit 0, Bit 5, Bit 6 und Bit 7 werden vom ProfisafeHost-Stack ausmaskiert). Es können mehrere Bits gleichzeitig gesetzt sein.

Bit in XX	Beschreibung
Bit 0	Nicht verwendet
Bit 1	Es liegt ein Fehler im F-Device vor (Device_Fault).
Bit 2	CRC Kommunikationsfehler (CE_CRC)
Bit 3	Timeout-Fehler (WD-timeout)
Bit 4	Fail-safe Werte sind aktiviert (FV-activated)
Bit 5	Nicht verwendet
Bit 6	Nicht verwendet
Bit 7	Nicht verwendet

Zeitmessung

Der Funktionsbaustein ProfisafeHost stellt drei Ausgänge zur Überwachung der Nachricht-Bearbeitungszeit zur Verfügung. Die Ausgänge dienen zur Diagnose der Profisafe-Kommunikation.

Als gemessene Durchlaufzeit einer Nachricht (F-Host -> F-Device -> F-Host) wird die abgelaufene Zeit des "host-timer" nach der Profisafe-Spezifikation verwendet:

- Gemessen wird vom Absenden einer Anfrage vom F-Host zum F-Device bis zum Erhalt der Antwort vom F-Device.
- Um die Messungen nicht zu verfälschen, werden nur gültige Antworten berücksichtigt, (d. h. es werden nur Werte an den Übergängen T6 und T16 nach Profisafe-Spezifikation "F-Host state diagram" übernommen).
- Der Funktionsbaustein liefert die aktuelle, die minimale und die maximale Durchlaufzeit einer Nachricht.
- Die minimale und die maximale Durchlaufzeit wird seit Systemstart bzw. seit der letzten steigenden Flanke von OA_C ermittelt.

17.2.2 Profisafe-F-Parameter

F-Parameter sind sicherheitsgerichtete Parameter (Konfigurationsdaten) eines Profisafe-F-Device, definiert in der Profisafe-Spezifikation.

Die F-Parameter werden über die GSD-Datei des F-Device zur Verfügung gestellt.

In der Sicherheitsapplikation **SafetyApp** befinden sich die F-Parameter im Editor der entsprechenden logischen E/As in der Registerkarte **Sichere Konfiguration**

Detailinformationen siehe Kap. "Sichere Konfiguration und Sichere Parametrierung" auf Seite 57.

Die Vorbelegung der F-Parameter, ihre Editierbarkeit und Sichtbarkeit sind gerätespezifisch.

Nachfolgend ein Überblick, um vor allem die Zusammenhänge aufzuzeigen.

Gerätespezifische Details entnehmen Sie der Dokumentation des F-Device.

Die F-Parameter werden durch eine Prüfsumme (CRC) gesichert, die über die F-Parameter des F-Device berechnet wird.

Diese Prüfsumme wird automatisch vom Editor berechnet und im Parameter **F_Par_CRC** abgelegt.

Eine Änderung der F-Parameter führt zur Änderung dieser Prüfsumme (CRC).

F_Check_SeqNr
V2 mode: Die fortlaufende Nummer ist immer in die CRC2-Generation einzubeziehen

nicht editierbar / nicht ausgewertet

F_Check_iPar
Herstellerspezifische Verwendung innerhalb eines homogenen Systems

nicht editierbar / nicht ausgewertet

F_SIL
SIL (Safety Integration Level), den der Anwender vom jeweiligen F-Device benötigt. Er wird mit der Angabe des Geräteherstellers im F-Device verglichen.

editierbar

F_CRC_Length
CRC-Länge, benötigter Wert: 3 Byte CRC

teilweise editierbar

F_Par_Version
Angabe der vom F-Device unterstützten Profisafe-Version (V1/ V2 mode)

Benötigter Wert: V2 mode (nur V2 mode wird vom ProfisafeHost unterstützt)

teilweise editierbar

F_Block_ID
Identifikation des Parameterblock-Typs (gibt z. B. an, ob die F_iPar_CRC im F-Parameter-Block enthalten ist.)

nicht editierbar

F_Source_Add
Eindeutige Adresse (Quelladresse) des F-Host im Profisafe-Netzwerk.

Es erfolgte keine Überprüfung auf Eindeutigkeit im Editor **Sichere Konfiguration**!

editierbar

⚠ VORSICHT **Der Anwender muss organisieren,**

dass der F-Host innerhalb der Anlage / des Profibus- /Profinet-Netzwerks eine **eindeutige** Quelladresse **F_Source_Add** hat (z. B. mittels Planungsunterlagen).



Der Anwender muss organisieren,

- dass die Quelladresse **F_Source_Add** aller F-Devices eines F-Hosts genau der Quelladresse ihres F-Hosts entspricht.

F_Dest_Add Eindeutige Adresse (Zieladresse) des F-Device im Profisafe-Netzwerk.

Es erfolgte keine Überprüfung auf Eindeutigkeit im Editor **Sichere Konfiguration**!



Empfehlung: Kommen zur Steuerung einer Anlage mehrere F-Hosts zum Einsatz, ist allen F-Devices der Anlage eine **eindeutige F_Dest_Add** zu zuordnen.

Hierzu ist es hilfreich, dass der Anwender in den Planungsunterlagen eine Aufstellung mit den für die einzelnen F-Hosts zulässigen Wertebereichen für die **F_Dest_Add** macht.

Diese Empfehlung muss realisiert werden, wenn die Voraussetzung, dass jedes F-Device über die Verkabelung mit genau einem Profibus Master / Profinet IO-Controller verbunden ist und somit seine F-Parameter nur von diesem Master / IO-Controller erhält, nicht erfüllt wird.

editierbar

Durch eine Änderung von "F_Dest_Add" ändert sich der "F_Par_CRC".

⚠ VORSICHT **F_Dest_Add**

Der Anwender muss organisieren, dass alle F-Devices eines F-Host eine **eindeutige** Zieladresse **F_Dest_Add** haben.



Der Anwender muss darauf achten, dass die **F_Dest_Add** im Editor **Sichere Konfiguration** mit der Einstellung am Adressschalter des F-Device (in der Regel DIP-Schalter) übereinstimmt.

F_WD_Time Zeitangabe in Millisekunden für Wachtdog-Timer, der die Dauer bis zum Empfang der nächsten gültigen PROFIsafe- Nachricht überwacht.

Default-Wert der GSD-Datei: Maximale Bearbeitungszeit des F-Devices

edtierbar

durch eine Änderung von "F_WD_Time" ändert sich der "F_Par_CRC"

F_iPar_CRC Prüfsumme, die aus den i-Parametern des F-Devices berechnet wird.

Durch eine Änderung von "F_iPar_CRC" ändert sich der "F_Par_CRC"



Als Wert von F_iPar_CRC muss manuell der Wert der Prüfsumme (CRC) über die I-Parameter des F-Device eingetragen werden.

Die CRC über die I-Parameter befindet sich in der Registerkarte **Sichere Parametrierung** des logischen E/As des F-Device.

Falls das F-Device nicht über diese Registerkarte verfügt, entnehmen Sie die CRC aus dem zugehörigen herstellerspezifischen Werkzeug.

F_Par_CRC Prüfsumme über alle F-Parameter.

Berechnung durch sichere Editoren.

Damit wird die fehlerfreie Übertragung der F-Parameter sichergestellt.

nicht editierbar

17.2.3 Profisafe-I-Parameter

I- Parameter sind individuelle, sicherheitsgerichtete, herstellerspezifische Geräteparameter eines Profisafe-F-Device. Details entnehmen Sie der Dokumentation des F-Device.

Für bestimmte F-Devices können die I-Parameter in der Registerkarte **Sichere Parametrierung** im Editor der logischen E/As eingesehen und editiert werden (z. B.: R-IB IL 24 PSDI 8-PAC, R-IB IL 24 PSDO 4/4-PAC).

Detailinformationen siehe Kap. "Sichere Konfiguration und Sichere Parametrierung" auf Seite 57.

Falls das F-Device nicht über die Registerkarte **Sichere Parametrierung** verfügt, verwenden Sie das zugehörige herstellerspezifischen Werkzeug zum Editieren der I-Parameter.

Die I-Parameter werden durch eine Prüfsumme (CRC) gesichert, die über die I-Parametern des F-Device berechnet wird.

Eine Änderung der I-Parameter führt zur Änderung dieser Prüfsumme (CRC).

☞ Der Wert der Prüfsumme (CRC) muss manuell in den F-Parameter **F_iPar_CRC** auf der Registerkarte **Sichere Konfiguration** eingetragen werden.

17.2.4 PROFIsafe-spezifische Nachweise für die Abnahme

Bei der Abnahme einer Sicherheitsapplikation muss ein Nachweis erbracht werden für:

- Eindeutige Quelladresse jedes F-Hosts innerhalb einer Anlage/ eines Netzwerks
- Die Quelladressen aller F-Devices eines Hosts müssen der Quelladresse ihres F-Hosts innerhalb einer Anlage/eines Netzwerks entsprechen. („F_Source_Add“ in der Registerkarte „Sichere Konfiguration“ des logischen E/As des sicheren Feldgeräts (F-Devices))
- Eindeutige Zieladresse aller F-Devices aller F-Hosts. („F_Dest_Add“ in der Registerkarte „Sichere Konfiguration“ des logischen E/As des sicheren Feldgeräts (F-Devices))
- Maximal 100 PROFIsafe-Geräte pro SIL3-Sicherheitsfunktion
- Maximal 1000 PROFIsafe-Geräte pro SIL2-Sicherheitsfunktion

17.3 SafetyCSos.library

☞ Die Anbindung von CSos-Geräten ist erst ab IndraWorks-Version 13V06 möglich. In früheren Releases ist diese Funktion nicht zertifiziert.

Anwendungsgebiete und Funktionsbausteine von SafetyCSos

Anwendungsgebiet (Ordner in der Bibliothek)	
Funktionsbausteine	**Beschreibung**
CSosOriginator, Seite 591	ist eine Kontrollinstanz für die Nutzdaten, denen man im Mapping-Dialog geeignete Namen zur Verwendung in der Sicherheitsapplikation geben kann.

Tab. 17-9: SafetyCSos-Bibliothek

Anwendbare Sicherheitsstandards

"CIP Safety on sercos" ist ein vom "sercos international e. V." verwalteter offener Standard

Grundlage ist sercos III V1.1.2, FSP "CIP Safety on sercos", Version 1.1.2.1.1, March 11th 2009

Begriffsdefinitionen

- **CSos:** Common Industrial Protocol (CIP) Safety on sercos, sicheres Protokoll über sercos
- **CSosOUNID:** CSos Originator Unique Network Identifier
- **Safety Device ID (SDID):** Geräte-ID des Zielgeräts. Die SDID muss für jedes Gerät eindeutig vergeben werden. Sie darf keine Beziehung zur sercos Adresse des Gerätes haben.
- **Safety Network Number (SNN):** Eindeutige ID des Safety-Netzwerks.

  Die SNN hat eine Länge von 6 Bytes.

  Alle CSos-Geräte müssen dieselbe SNN besitzen.
- **Originator:** Initiiert den CSos-Verbindungsaufbau.
- **Target:** Ziel einer CSos-Verbindung.

  Der Consumer befindet sich in einem sicheren sercos Slave-Gerät.
- **Producer:** versendet Daten über eine CSos-Verbindung
- **Consumer:** empfängt Daten von einer CSos-Verbindung
- **Forward Open:** Verbindungsaufbau einer sichere zyklische Übertragung
- **Forward Close:** Verbindungsabbau einer sichere zyklische Übertragung

Funktionsweise

Der Datenaustausch der sicheren E/A Daten zwischen sicherer Applikation und dem Zielgerät erfolgt verbindungsorientiert. Vor Beginn der sicheren Kommunikation muss mit einem "Forward Open" eine Verbindung aufgebaut werden.

Die sichere Applikation liest die empfangenen Daten einer Verbindung und schreibt zu sendende Daten für eine Verbindung.

In jede Datenrichtung wird eine CSos-Verbindung aufgebaut.

Jede Verbindung wird über eine Instanz des Funktionsbausteins CSosOriginator überwacht. Der erfolgreiche Verbindungsaufbau wird an den Ausgängen des CSosOriginator angezeigt (CSosOriginator, Seite 591).

☞ Die Überprüfung des Verbindungsstatus anhand von am Gerät angebrachter LEDs ist nicht zulässig.

Für den erfolgreichen Aufbau einer CSos-Verbindung, müssen beide Teilnehmer mit einer systemweit eindeutigen SDID und SNN konfiguriert sein. Weiterhin sind kommunikationsabhängige Parameter zu beachten.

Die SNN muss für jedes Sicherheitsnetzwerk und jedes Sicherheitsunternetzwerk systemweit eindeutig sein.

Die aktuelle TUNID (SDID und SNN) des Gerätes kann über den Parameter S-0-1800.x.17 geprüft werden.

Wurde eine CSos-Verbindung einmal erfolgreich aufgebaut, so wird die TUNID des Originator (OUNID) im Target gespeichert. Die OUNID kann über den Parameter S-0-1800.x.19 geprüft werden.

Wird ein bereits konfiguriertes Gerät in ein neues Netzwerk eingebaut, so ist vorher die vorhandene Konfiguration zu löschen.

Alle Sicherheitsfunktionen des Geräts müssen nachdem das Bootprojekt auf die Steuerung geladen und geprüft wurde getestet werden. Erst nach erfolgtem Test ist das Gerät als "Verified" zu betrachten.

Konfiguration Steuerung

Die CSos-Konfiguration der Sicherheitssteuerung erfolgt über die Seite "Konfiguration" der Safety-Applikation. Sie wird durch einen Doppelklick auf die Safety-Applikation geöffnet.

(siehe auch "Konfiguration" auf Seite 47).

Wird die SDID auf 0 eingestellt, so ist sicherzustellen, dass beide Kommunikationsteilnehmer korrekt konfiguriert sind.

Konfiguration Teilnehmer

Die folgenden Kommunikationsparameter werden durch die Oberfläche gerätespezifisch voreingestellt:

Kommunikationsparameter	Default-Wert	Min-Wert	Max-Wert	Erläuterung
SNN	0x600000001	0x600000001	0x605265BFF	Eindeutige ID des Safety-Netzwerks
SDID	1	1	4294967295	Eindeutige Geräte-ID des Zielgeräts.
TiMu	1	1	255	Timeout Multiplier
NTE	100 ms	10 ms	5800 ms	Network Time Expectation

Tab. 17-10: Voreingestellte Kommunikationsparameter für CSos-Verbindungen am Beispiel IndraDrive

Bei einer falschen Kombination von Verbindungsparametern (z. B. NTE zu klein im Hinblick auf gewählte Einstellungen für EPI und Safety-Zykluszeit) wird die Verbindung nicht initialisiert. Diesen Zustand kann man am DiagCode-Ausgang des Funktionsbausteins CSosOriginator erkennen. Die EPI kann unabhängig von der Safety-Zykluszeit eingestellt werden.

17.3.1 CSosOriginator

Kurzbeschreibung

Der Funktionsbaustein CSosOriginator ist eine Kontrollinstanz für die Nutzdaten, denen man im Mapping-Dialog geeignete Namen zur Verwendung in der Sicherheitsapplikation geben kann.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyCSos	2.4.0.1

Tab. 17-11: Bibliothekszuordnung

Schnittstellenbeschreibung

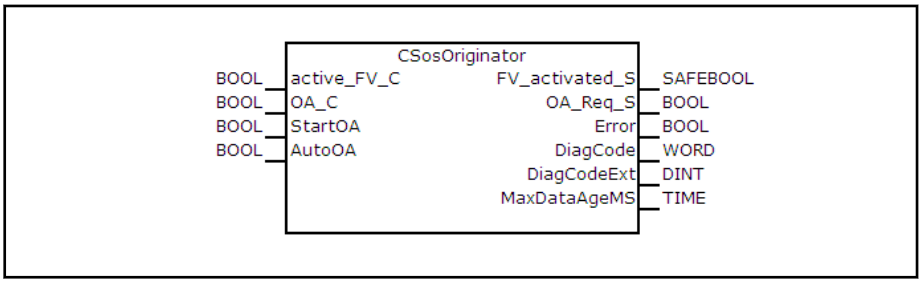


Abb. 17-2: Funktionsbaustein CSosOriginator

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	active_FV_C	BOOL	Steuert Übertragung von Failsafe-Werten TRUE: Es werden Failsafe-Werte übertragen FALSE: Wenn möglich, werden Prozessdaten übertragen, sonst Failsafe-Werte
	OA_C	BOOL	Quittierung eines Kommunikationsfehlers steigende Flanke: Ein am Ausgang OA_Req_S gemeldeter Kommunikationsfehler wird quittiert
	StartOA	BOOL	Startup-Quittierung (Operator Acknowledge) TRUE: Der Hochlauf wird einmalig quittiert FALSE: Der Hochlauf wird nicht quittiert
	AutoOA	BOOL	automatische Quittierung (Operator Acknowledge) TRUE: Kommunikationsfehler werden automatisch quittiert FALSE: Kommunikationsfehler werden nicht automatisch quittiert
VAR_OUTPUT	FV_activated_S	SAFEBOOL	Übertragung von Failsafe-Werten aktiv "Failsafe Values activated State" TRUE: Es werden Failsafe-Werte übertragen FALSE: Es werden Prozessdaten übertragen
	OA_Req_S	BOOL	Kommunikationsfehler TRUE: Es ist ein Kommunikationsfehler aufgetreten FALSE: Es liegt kein Kommunikationsfehler vor
	Error	BOOL	Es ist ein Fehler aufgetreten.
	DiagCode	WORD	Diagnose-Code (siehe Tab. 17-14 "Code-Bereiche des Ausgangs DiagCode" auf Seite 594)
	DiagCodeExt	DINT	Erweiterter DiagnoseCode (siehe Tab. 17-15 "DIagCode-Kombinationen" auf Seite 595)
	MaxDataAgeMS	TIME	Maximales Datenalter

Tab. 17-12: Schnittstelle CSosOriginator

Name	Typ	Default-Wert	Übernahme
active_FV_C	BOOL	FALSE	kontinuierlich
OA_C	BOOL	FALSE	kontinuierlich
StartOA	BOOL	TRUE	kontinuierlich
AutoOA	BOOL	FALSE	kontinuierlich

Tab. 17-13: Eingangsverhalten CSosOriginator

Funktionsbeschreibung

Der Funktionsbaustein CSosOriginator überwacht den Verbindungsaufbau und die zyklische Kommunikation eines sicheren sercos-Teilnehmers. Er stellt Diagnosen zur Verfügung und ermöglicht das Quittieren von aufgetretenen Meldungen. Mit dem logischen CSos-Gerät wird eine Kontrollinstanz dieses Funktionsbaustein automatisch der Sicherheitsapplikation hinzugefügt.

Legt man einen IndraDrive mit einem SafeLogic Safety-Profil an, werden automatisch zwei solcher Kontrollinstanzen (für Control und Status) angelegt.

Um auf die Ein- und Ausgänge einer bereits angelegten Kontrollinstanzen zugreifen zu können, müssen diese als externe Verweise in ein Safety-Programm aufgenommen werden. Werden keine Zugriffe auf die Ein- und Ausgänge benötigt, so braucht der Funktionsbaustein CSosOriginator nicht in der Sicherheitsapplikation programmiert werden.

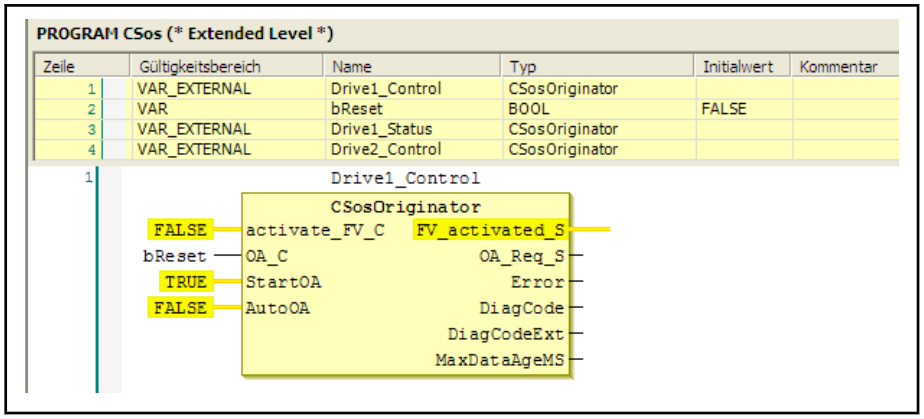


Abb. 17-3: Kontrollinstanz eines IndraDrive mit SafeLogic Safety-Profil

Es ist zulässig, an mehreren Stellen in der Sicherheitsapplikation auf die gleiche Instanz zu verweisen. Ausgangsvariablen können mehrfach anderen Variablen zugewiesen werden. Bei Eingangsvariablen "gewinnt" der im Programmablauf zuletzt zugewiesene Wert.

Die Ausgänge des Funktionsbausteins werden sowohl im gestarteten als auch im gestoppten Zustand bedient.

Es ist zulässig, an mehreren Stellen in der Sicherheitsapplikation auf die gleiche Instanz zu verweisen. Ausgangsvariablen können mehrfach anderen Variablen zugewiesen werden. Eine Mehrfachzuweisung von Eingangsvariablen führt zu einem undefinierten Zustand des betroffenen Eingangs.

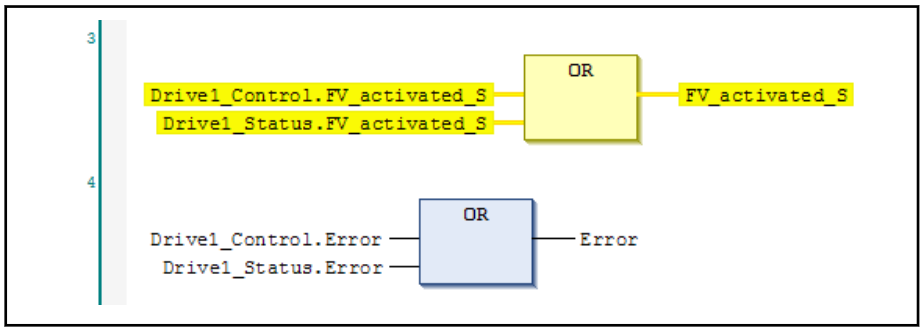


Abb. 17-4: Verknüpfung von CSoS-Instanzen

FV_activated_S

Der Ausgang FV_axtivated_S "Failsafe Values activated State" zeigt an, ob zyklische Prozessdaten oder Failsafe-Werte übertragen werden. Er ist FALSE, wenn gültige Werte übertragen werden.

Das Ausgangssignal FV_activated_S ist FALSE, wenn Prozessdaten übertragen werden. Es ist TRUE, wenn statt der Prozessdaten Failsafe-Werte übertragen werden oder keine Übertragung von oder zum Target stattfinden kann.

Failsafe-Werte werden übertragen wenn:

- die Sicherheitsapplikation nicht in RUN ist
- durch den Eingang "activate_FV_C" das Übertragen von Failsafe-Werten gefordert wird
- in der Startup-Phase (default FV_activated = TRUE)

- ein Kommunikationsfehler zwar behoben, aber noch nicht quittiert wurde
- bei einer Consumer-Verbindung das Target im IDLE-Zustand ist

Operator Acknowlegde Sequenz (OA_Req_S, OA_C, StartOA, AutoOA)

Die Quittierung eines Kommunikationsfehlers kann erst erfolgen, wenn die Kommunikation erfolgreich wieder aufgebaut wurde. Die Quittierung eines Consumer-IDLE-Fehlers kann erst erfolgen, wenn der Consumer wieder den Zustand RUN angenommen hat.

Mit einem TRUE am Ausgangssignal "OA_Req_S" (Operator Acknowledge Request State) wird dies angezeigt.

Gleichzeitig mit Setzen des OA_Req_S wird der DiagCode auf 0xC104 gesetzt. Bei erneuter Kommunikationsstörung oder einem Consumer-IDLE-Zustand wechselt das Signal OA_Req_S wieder auf FALSE.

Die Quittierung eines Kommunikationsfehlers erfolgt mit einer positiven Flanke am Eingang "OA_C" (Operator Acknowlege Control).

Mit dem Eingangssignal StartOA (Startup Operator Acknowledge) kann mit dem Wert TRUE einmalig die Startup-Quittierung automatisch ausgelöst werden (default StartOA=TRUE).

Mit dem Eingangssignal AutoOA (Automatic Operator Acknowledge) kann mit dem Wert TRUE die Quittierung automatisch ausgelöst werden (default AutoOA=FALSE). Es wirkt so, als hätte man eine Brücke zwischen OA_REQ_S und OA_C.

Ist der Zustand FV_activated_S=TRUE durch eine RUN/STOP Umschaltung oder das Eingangssignal activate_FV_C=TRUE verursacht, ist keine Quittierung bei Signalwechsel notwendig.

MaxDataAgeMS

Bei einem CSosOriginator-Baustein für eine Input-Verbindung enthält der Funktionsbausteinausgang MaxDataAgeMS das maximale Datenalter.

Bei einem Baustein für eine Output-Verbindung wird immer 0 ms angezeigt.

Die Erfassung von Zeiten erfolgt für gültige Eingangstelegramme im 128 Mikrosekunden-Raster. Zur Anzeige wird der Wert in einen Millisekundenwert gerundet.

Der erfasste Maximalwert wird mit der Operator Acknowledge Sequenz auf 0 ms zurückgesetzt. Er kann dabei den Maximalwert von mehreren Phasen gültiger Eingangstelegramme enthalten.

Fehlerbehandlung

Error:

- **FALSE:** wenn im DiagCode das Bit 14 (15..0) nicht gesetzt ist.
- **TRUE:** wenn im DiagCode das Bit 14 (15..0) gesetzt ist.

DiagCode, DiagCodeExt

DiagCode-Bereich	Beschreibung
16#8000	Normale Operation
16#8002	Initialisiere, noch nicht gestartet.
16#C002..16#C03F	Konfigurationsfehler der übergeordneten CSosOUNID-Instanz
16#C042..16#C0FF	Konfigurationsfehler der CSosOriginator Instanz

DiagCode-Bereich	Beschreibung
16#C104	Operator Acknowledge request
16#C202..16#C2FF	Fehler bei der Kommunikation mit dem Target

Tab. 17-14: Code-Bereiche des Ausgangs DiagCode

Die DiagCode-Gruppen sind in Prioritäten eingeteilt:

- 16#C0.. höchste Priorität. Es wird kein "ForwardOpen" versucht.
- 16#C1..
- 16#C2.. niedrigste Priorität.

Wurde bereits ein Fehler einer Prioritätsstufe gemeldet, so verändert ein neuer Fehler der gleichen Prioritätsstufe den DiagCode nicht mehr.

Die folgenden DiagCode/DiagCodeExt-Kombinationen sind möglich:

DiagCode	DiagCodeExt	Beschreibung
16#0000	0	Nicht initialisiert
16#8000	Zeitdauer des ForwardOpen in ms	Normale Operation, OPENED
16#8001	0: CONNSTAT_CLOSED 1: CONNSTAT_CLOSE_REQ 2: CONNSTAT_CLOSE_REQ_PENDING 3: CONNSTAT_OPENED 4: CONNSTAT_OPEN_REQ 5: CONNSTAT_OPEN_REQ_PENDING 6: CONNSTAT_STOPPED 7: CONNSTAT_OPEN_REQ_WAIT 8: CONNSTAT_OPEN_TYPE_1_REQ_WAIT	ForwardOpen wird bearbeitet. (Diese Zustände werden beim Durchlaufen der State-Maschine des CSosOpriginators nur temporär angezeigt).
16#8002	0: Initialisierung durchlaufen 1: CSAL nicht in Operating 2: IOConfigID unterschiedlich	Die Initalisierung war erfolgreich, es konnte jedoch bisher keine CIP-Verbindung aufgebaut werden. Dieser Zustand kann aufgrund eines Fehlers dauerhaft eingenommen, wenn: • dauerhaft die azyklische Aktivität blockiert ist. • dauerhaft ein falscher CCO-Status ansteht. • dauerhaft kein FO-Request-Buffer frei ist.
16#C000	0	Die CSosOUNID-Instanz ist nicht initialisiert kein SSO-Objekt (falsche Safety-FM Gerätebeschreibung, Reiter SafetyApp.Configuration fehlt)
16#C008	uiNumOfParams	Falsche Anzahl CSosOUNID Konfigurationsparameter ⇒ falsche Safety-FM Gerätebeschreibung
16#C042	0	Die CSosOriginator-Instanz hat keine IO-Konfigurationsparameter (CCO)

DiagCode	DiagCodeExt	Beschreibung
16#C044	CSosOriginator.fr0	CSosOriginator.fr0 fehlerhaft
16#C046	< 0 : CSosOriginator.fr0 > 0 : (CSosOriginator.fr0 - CSosOriginator.to0)	CSosOriginator.to0 fehlerhaft
16#C048	CSosOriginator.to0 - CSosOriginator.fr0 + 1	Falsche Anzahl CSosOriginator-Konfigurationsparameter (falsche Statuswort/Steuerwort-Gerätebeschreibung oder altes Bootprojekt)
16#C04A	intern berechneter CRC	Checksumme der CSosOriginator Konfigurationsparameter falsch
16#C050	net_OT-Wert	net_OT fehlerhaft
16#C052	net_TO-Wert	net_TO fehlerhaft
16#C054	Summe app_data_size aus net_TO und net_OT	net_OT und net_TO passen nicht zusammen
16#C056	Open connection path size	Open connection path size fehlerhaft
16#C058	connection name size	connection name size fehlerhaft
16#C05A	Index des fehlerhaften Zeichens, beginnend mit 0	connection name coding fehlerhaft
16#C05C	time	Safety Configuration ID (SCID), Kombination aus Date/Time fehlerhaft
16#C05E	time	Safety Device ID (SDID) Kombination aus Date/Time fehlerhaft
16#C060	format type	format type fehlerhaft
16#C062	Timeout multiplier	Timeout Multiplier (TiMu) außerhalb des zulässigen Bereiches Zulässig: Base Format 1..4, Extended Format 1..255
16#C064	Ping Interval EPI multiplier	Ping Interval EPI multiplier (PIEM) außerhalb des zulässigen Bereiches Zulässig: (15+TiMu) .. 1000
16#C066	Time Coord. Msg. Min. Multiplier (TCMM)	Time Coord. Msg. Min. Multiplier (TCMM) außerhalb des zulässigen Bereiches Zulässig: 0 .. 7813
16#C068	Max fault number	Max fault number zu hoch Zulässig: Base Format 0, Extended Format 0 .. 255
16#C06A	VendorID<<16 + ProductCode	Elektronisches Typenschild fehlerhaft
16#C06C	major<<8 + minor	Versionsangabe fehlerhaft
16#C092	Timeout-Wert	Timeout-Wert zu klein Der Wert muss mindestens doppelt so groß sein wie die Safety-Zykluszeit
16#C094	Timeout-Wert	Timeout-Wert zu groß
16#C096	0	Fehler bei der IO-Initialisierung

DiagCode	DiagCodeExt	Beschreibung
16#C098	Maximalwert NTE	NTE ist größer als das zulässige Maximum
16#C09A	Theoretisches minimales NTE in Millisekunden	NTE ist kleiner als der theoretische Minimalwert (EPI[ms] * TiMu) + Safety-Zykluszeit[ms] + sercos-Zykluszeit[ms] - Falls EPI 0 ist, wird intern für EPI der Wert der Safety-Zykluszeit verwendet - Wird EPI nicht angezeigt, dann ist EPI in der Regel 0
16#C09C	EPI-Wert	*Fehlerhafter EPI-Wert, zulässig:* • base format: 1.. 100ms • extended format: 1.. 1000ms (1s)
16#C09E	EPI * TiMu	EPI, TiMu und Safety-Zykluszeit passen nicht zusammen.
16#C104	0	Operator Acknowlegde request (Kodierung genauso wie PROFISAFE) Bei erneuter Kommunikationsstörung kann dieser Code anstehen, obwohl das Signal CSosOriginator.OA_Req_S wieder auf FALSE gewechselt ist.
16#C118	SSO-Status	Falscher SSO-Status
16#C202	eingestellter Timeout-Wert	Beim forward open request wurde ein timeout festgestellt Die Ursache kann z. B. eine falsche sercos-Phase (nicht 4) sein.
16#C204	IXXAT-ErrorCode 16#3118 Konfiguration passt nicht zur Antriebskonfiguration => grauer Logger für Zusatzinfo z. B.: 16#10100 Verbindung wird schon verwendet oder zweifacher Forward Open 16#10106: Originator TUNID ungleich (Einstellung der Safety-Apllikation) 16#10114 Vendor ID oder Product Code (S-0-1801.0.1 / S-0-1801.0.3) 16#10115 Product Type (S-0-1801.0.2 Device Type) 16#10117: SafetyProfil / Assembly stimmt nicht überein 16#1080D TUNID nicht gesetzt (S-0-1800.0.19) 16#1080E TUNID ungleich (S-0-1800.0.19)	Beim Forward Open Request wurde eine Fehlermeldung empfangen.
16#C206	IXXAT-ErrorCode	Beim Forward Open Request wurde eine ungültige Antwort empfangen
16#C208	eingestellter Timeout-Wert*2	Timeout beim Forward Open Request
16#C210	IXXAT-ErrorCode	Failsafe Values vom Target

DiagCode	DiagCodeExt	Beschreibung
16#C212	16#10100: Verbindung wird schon verwendet oder zweifacher Forward Open 16#10106: Originator TUNID ungleich (Einstellung der Safety-Apllikation) 16#10114: Vendor ID oder Product Code (S-0-1801.0.1 / S-0-1801.0.3) 16#10115: Product Typ (S-0-1801.0.2 Device Type) 16#10117: SafetyProfil / Assembly stimmt nicht überein 16#1080C: SCCRC ungleich (sichere Parametrierung) 16#1080D: TUNID nicht gesetzt (S-0-1800.0.19) 16#1080E: TUNID ungleich (S-0-1800.0.19)	Beim Forward Open Request wurde eine Fehlermeldung empfangen.
16#C2EE	IXXAT-ErrorCode 16#2730 Time Coordination Message Check: not received in alloted time 16#2822 Timestamp Delta greater than Network Time Expectation 16#2823 Data age of a faulty message greater than NTE 16#2824 Data age of an in other respects valid message greater than NTE 16#2832 Consumer Activity Monitor timeout	Bei einer bestehenden Verbindung hat sich der Status der Verbindung geändert (safety validator state failed)

Tab. 17-15: DIagCode-Kombinationen

17.3.2 CSos-Parameter

Die Parametrierung von CSos-Geräten erfolgt über den entsprechenden logischen Geräteknoten unterhalb des Knotens "Logische E/As" auf der Registerkarte "Sichere Konfiguration".

Die folgenden Parameter werden für CSos-Geräte verwendet:

Name	sercos Parameter	beschreibbar	Beschreibung
Vendor ID	S-0-1801.0.1	nein	Herstellerkennung
Device Type	S-0-1801.0.2	nein	Gerätetyp
Product Code	S-0-1801.0.3	nein	Gerätekennung
Major Reversion	S-0-1801.0.4	nein	Versionskennung
Minor Reversion	S-0-1801.0.4	nein	Versionskennung
Connection name		nein	Name der Verbindungsinstanz
Safety Configuration CRC (SCCRC)	S-0-1800.0.16	ja	
Safety Configuration Time Stamp (SCTS)	S-0-1800.0.16	ja	
Safety Network Number (SNN)	S-0-1800.0.19	ja *	Netzwerknummer
Safety Device ID (SDID)	S-0-1800.0.19	ja *	eindeutige Geräte-ID

Name	sercos Parameter	beschreibbar	Beschreibung
Ping Interval EPI Multiplier (PIEM)	S-0-1800.0.8	ja **	Häufigkeit der Zeitkoordinierungs-nachrichten
Time Coord. Msg. Min. Multiplier (TCMM)	S-0-1800.0.5	ja **	Kleinste Übertragungsverzögerung einer Ping-Nachricht (in 128µs)
Timeout Multiplier	S-0-1810.0.7	ja **	Anzahl der verlorenen Nachrichten vor Verbindungsabbruch
Max fault number		ja	Anzahl der verlorenen Pakete vor Verbindungsabbruch
Expected Packet Interval (EPI)		ja	Zyklische Nachrichtenrate in µs
Timeout		ja	Timeout für Verbindungsaufbau in µs

* Muss identisch zum entsprechenden sercos Parameter sein.
** Wird beim Schalten nach P4 in den Antrieb übernommen

Tab. 17-16: Einstellungen der "Sicheren Konfiguration" für CSos



Csos-Geräte mit automatischer SNN-Vergabe dürfen diese Funktion nur solange nutzen, wie das Gesamtsystem noch nicht abgenommen wurde.

17.4 SafetyNetVar.library

Diese Bibliothek enthält die Treiberbausteine zum Austausch von Netzwerk-Daten zwischen einem Sender und einem oder mehreren Empfängern.

Anwendungsgebiete und Funktionsbausteine von SafetyNetVar

Anwendungsgebiet (Ordner in der Bibliothek)	
Funktionsbausteine	**Beschreibung**
NetVarReceiver, Seite 605	Der Funktionsbaustein Implementiert eine Safety Netzwerk-variablenliste (Receiver-NVL) und verwaltet die Übertragung der Variablen aus der Safety Netzwerkvariablenliste. Der Funktionsbaustein stellt die Verbindung zu einer Safety Netzwerkvariablenliste (Sender-NVL) her. Die Verbindung wird überwacht.
NetVarSender, Seite 603	Der Funktionsbaustein Implementiert eine Safety Netzwerk-variablenliste (Sender-NVL) und verwaltet die Übertragung der Variablen aus der Safety Netzwerkvariablenliste. Der Funktionsbaustein stellt von sich aus keine Verbindung her. Angenommene Verbindungen werden nicht überwacht. Wird der NetVarSender-Funktionsbaustein in der Applikation SafetyApp verwendet, kann der Anwender Einfluss auf die übertragenen Werte nehmen, in dem er Failsafe-Werte aktiviert.

Tab. 17-17: SafetyNetVar-Bibliothek

Begriffsdefinitionen

Die **Safety Address** ist eine eindeutige Kennung der Netzwerksenderliste.

⚠ VORSICHT **Safety Address**

Die Adresse ("Safety Address") jeder Safety-Netzwerkvariablenliste muss projektweit, im gesamten Netzwerk und in allen per UDP/IP erreichbaren Steuerungen eindeutig sein.

Die **Connection ID** ist die Kennung der Netzwerkempfangsliste.

⚠ VORSICHT **Connection ID**

Die Verbindungskennung ("Connection ID") zur Safety-Netzwerkvariablenliste muss projektweit eindeutig sein.

Bei Topologie T3:

Die Eindeutigkeit der Netzwerkvariablenlisten-Connection IDs aller Sicherheitssteuerungen im Netzwerk der ganzen Maschine muss gewährleistet sein.

☞ Jede Connection ID darf nur einmal in einem Projekt für eine Empfangsliste vergeben werden.

Max. Receivers ist die maximale Anzahl von Safety Netzwerkvariablenlisten (Empfänger), die gleichzeitig von dieser Safety Netzwerkvariablenlisten (Sender) Werte erhalten können.

Die **Watchdog Time** ist die Zeit, innerhalb der die Netzwerkvariablen einer Verbindung aktualisiert werden müssen. Ein Überschreiten der Watchdog Time, z.B. durch Ausfall von UDP-Telegrammen, führt zum Verbindungsabbruch.

Systemanforderungen

⚠ VORSICHT **Restfehlerrate**

Von der Treiberinstanz gemeldete Kommunikationsfehler dürfen nicht öfter als 1 Mal in 5 Stunden auftreten, damit die Restfehlerrate pro Stunde für die sicherheitsbezogenen Signale unter dem SIL3-Grenzwert von 10-9 bleibt.

Das Netzwerk, in welchem eindeutige Safety-Adressierungen zu gewährleisten sind, muss eindeutig eingrenzbar sein und darf nicht ohne erneute Eindeutigkeitsprüfung verändert werden.

Inbetriebnehmer und Betreiber dürfen zwischen dem Maschinennetz N1, über das Sicherheitssteuerungen kommunizieren, und einem anderen Netzwerk N2 nur dann eine physikalische Verbindung oder Verbindung durch Netzwerkgeräte (Switches, Router, PCs) herstellen, wenn für das gesamte via N2 per UDP/IP prinzipiell erreichbare Netzwerk N* (in welchem N1 und N2 enthalten sind) folgende Voraussetzungen erfüllt sind:

- Inbetriebnehmer und Betreiber kennen alle Sicherheitssteuerungen in N*.
- Inbetriebnehmer und Betreiber kennen alle von ihnen verwendeten Safety- Adressierungen - und diese sind alle und zusammen mit denen in N1 eindeutig.
- Inbetriebnehmer und Betreiber haben durch organisatorische Maßnahmen für das Netzwerk N* dafür gesorgt, dass bei zukünftigen Änderungen in oder an N* die Eindeutigkeit bewahrt wird. Dies betrifft Änderun-

gen an einer querkommunizierenden Sicherheitssteuerung (auch außerhalb N1 und N2), Hinzufügen einer querkommunizierenden Sicherheitssteuerung unter Hauptsteuerungen in N* und die Verbindung von N* mit weiteren Maschinennetzen.

⚠ VORSICHT **Netzwerke, bei denen die genannten Voraussetzungen nicht erfüllt sind**

Netzwerke, bei denen die genannten Voraussetzungen nicht erfüllt sind, müssen vom Maschinennetzwerk N1 getrennt bleiben.

Insbesondere darf das Maschinennetzwerk nie mit dem Internet bzw. an ein mit dem Internet verbundenes Firmennetzwerk verbunden werden (neben der Eindeutigkeit der Adresse spricht auch die geforderte Industrietauglichkeit der Netzwerkgeräte dagegen).

Und auch die Maschinennetzwerke baugleicher Serienmaschinen mit Querkommunikation dürfen niemals verbunden werden, oder am gleichen übergeordneten Netzwerk hängen, da sie die identische Sicherheitsapplikationen enthalten und somit die Safety-Adressierungen nicht innerhalb N* eindeutig wären.

⚠ VORSICHT **Querkommunikation**

Wenn Querkommunikation in der Maschine verwendet wird (T3), dann muss das dazu verwendete Ethernet von anderen Netzwerken getrennt sein.

Die Trennung muss physikalisch sein.

Eine bloße Filterung des IP-Verkehrs zwischen N1 und N2, z.B. durch entsprechend konfigurierte Router oder Firewalls, stellt keine für SIL3-Anwendungen ausreichende Trennung dar.

⚠ VORSICHT **Eindeutigkeit**

Die Eindeutigkeit muss nicht nur einmal während der Entwicklung oder Inbetriebnahme der Maschine geprüft werden.

Die Prüfung muss jedes Mal wiederholt werden, wenn neue Steuerungen mit querkommunizierenden Sicherheitssteuerungen an das UDP-Netzwerk angeschlossen werden oder wenn das UDP-Netzwerk mit einem übergeordneten oder untergeordneten UDP-Netzwerk verbunden wird.

☞ Der Anwender muss die spezifischen Systemanforderungen und weitere, allgemeine Systemanforderungen beachten.

Systemgrenzen

Die Anzahl von Netzwerkvariablenlisten in einem Projekt ist durch die SafeLogic auf 100 Safety-Netzwerkvariablenlisten (Sender) und 20 Safety-Netzwerkvariablenlisten (Empfänger) pro Sender beschränkt.

☞ Man bewegt sich außerhalb der Spezifikation, wenn die Zusammenfassung aller NVL-Verbindungen zwischen den gleichen zwei Sicherheitsteuerungen mehr als 100 Kommunikationsbeziehungen zwischen Sicherheitssteuerungen ergibt.

Nachweise für Abnahme

Bei der Abnahme einer Sicherheitsapplikation muss ein Nachweis erbracht werden für:

- Eindeutige Safety-Adresse jeder Sender-Netzwerkvariablenliste (Eingabefeld "Safety-Adresse dieser Variablenliste" im Objekt "Safety-Netzwerkvariablenliste (Sender)", Registerkarte "Safety-Konfiguration") im gesamten Netzwerk
- Eindeutige Connection ID jeder Empfänger-Netzwerkvariablenliste (im Objekt "Safety Netzwerkvariablenliste (Empfänger)", Registerkarte "Safety-Konfiguration") im gesamten Netzwerk
- Die Betriebsanleitung für die Maschine verpflichtet den Betreiber die Kommunikationsfehlerrate im Betrieb zu überwachen:

  Von der Treiberinstanz NetVarReceiver gemeldete Kommunikationsfehler dürfen nicht öfter als 1 Mal in 5 Stunden auftreten, damit die Restfehlerrate pro Stunde für die sicherheitsbezogenen Signale unter dem SIL3-Grenzwert von 10^{-9} bleibt.
- Die Betriebsanleitung für die Maschine verpflichtet den Betreiber bei Update von Sicherheitsapplikationen oder Erweiterungen des Netzwerks erneut die Eindeutigkeit von Netzwerkvariablenlisten Safety-Adressen und Connection IDs nachzuweisen

Funktionsweise

Safety Netzwerkvariablenlisten (NVL) arbeiten verbindungsorientiert nach dem Subscriber-Prinzip:

Die NVL-Sendeliste hält eine bestimmte Anzahl an möglichen Verbindungen vor.

Die NVL-Empfangliste baut die Verbindung zur NVL-Sendeliste auf.

Der Funktionsbaustein NetVarSender ist die Instanz der NVL-Sendeliste. Er hat keinerlei überwachende Funktionalität.

Der Funktionsbaustein NetVarReceiver ist die Instanz der NVL-Empfangsliste und überwacht die Verbindung mittels Watchdog-Zeit. Fehler werden nur an diesem Funktionsbaustein gemeldet.

Die NVL-Verbindung ist durch die Kombination der Parameter Safety Address und Connection ID gekennzeichnet.

⚠ VORSICHT **Eindeutige Kennzeichnung von NVL-Verbindungen**

Es liegt in der Verantwortung des Anwenders, dass jede NVL-Verbindung eine eindeutige Kennzeichnung erhält.

Beachten Sie die Hinweise zur Parametrierung:

- Begriffsdefinitionen, Seite 599

- Pro NVL können maximal 32 Byte Nettodaten übertragen werden.
- Es können maximal 199 NVLs pro Steuerung angelegt werden.

  Die Praktisch nutzbare Anzahl an SNVL ist u.a. von der verfügbaren Bandbreite und der Systemauslastung anhängig.
- NVLs können nur Daten mit Rexroth SafeLogic-Steuerungen austauschen.

☞ Wenn Standard-Netzwerkvariablen und Safety-Netzwerkvariablen gleichzeitig auf der Standard-Steuerung verwendet werden, kann es zu einer doppelten Belegung des Ports "1202" kommen, da der Port "1202" die Standardkonfiguration bei beiden Netzwerkvariablen ist.

In diesem Fall funktionieren die Netzwerkvariablen nicht!

Um eine doppelte Belegung zu verhindern wird empfohlen, die Portkonfiguration der Safety-Netzwerkvariablen für den Sender und den zugeordneten Receiver auf einen anderen Port einzustellen (z. B. "1209").

Unterstützte Feldbusse

Die sicheren Netzwerkvariablen werden über alle Ethernet-basierten Bussysteme unterstützt. Es wird ein IP-Kanal vorausgesetzt.

Die folgenden Busse/Anschlüsse werden unterstützt:

- Engineering Netzwerkanschluss
- PROFINET
- ETHERNET/IP
- Sercos
- C2C (Sercos Querkommunikation)

17.4.1 NetVarSender

Kurzbeschreibung

Der Funktionsbaustein NetVarSender implementiert eine Safety Netzwerkvariablenliste (Sender-NVL)

Der NetVarSender-Funktionsbaustein verwaltet die Übertragung der Safety Netzwerklistenvariablen. Er stellt von sich aus keine Verbindung her. Angenommene Verbindungen werden nicht überwacht.

Wird der NetVarSender-Funktionsbaustein in der Applikation SafetyApp verwendet, kann der Anwender Einfluss auf die übertragenen Werte nehmen, in dem er Failsafe-Werte aktiviert.

⚠ VORSICHT **Die Netzwerkvariablenliste läuft mit Initialwerten automatisch an und bestätigt die Aufnahme der Sicherheitsfunktion**

Die Default-Werte der Eingänge sind so festgelegt, dass die Netzwerkvariablenliste automatisch anläuft und die Aufnahme der Sicherheitsfunktion automatisch bestätigt.

Durch die Implementierung anderer System- und Applikationsmaßnahmen ist sicherzustellen, dass es beim Start der Applikation nicht zu einem unerwarteten (oder unbeabsichtigten) Starten der Maschine kommt.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyNetVar	1.2.0.0

Tab. 17-18: Bibliothekszuordnung

Schnittstellenbeschreibung

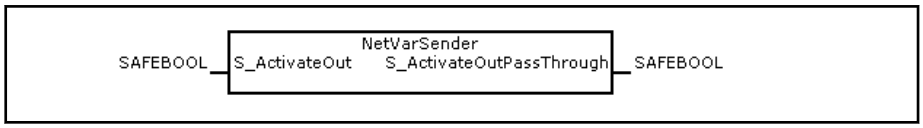


Abb. 17-5: Funktionsbaustein NetVarSender

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	S_ActivateOut	SAFEBOOL	Aktiviert die Ausgabe der Werte über die Netzwerkliste TRUE: Es werden Prozessdaten übertragen FALSE: Es werden Failsafe-Werte übertragen **Vorsicht:** Das Setzen dieses Eingangs von FALSE auf TRUE kann zur sofortigen Übertragung der Prozessdaten führen! Stellen Sie durch geeignete Maßnahmen sicher, dass von Ihrer Anlage/Maschine keine Gefahr ausgeht! Default-Wert: FALSE
VAR_OUTPUT	S_ActivatedOut-PassThrough	SAFEBOOL	Spiegelung des Eingangs S_ActivateOut TRUE: Es werden Prozessdaten übertragen FALSE: Es werden Failsafe-Werte übertragen

Tab. 17-19: Schnittstelle NetVarSender

Name	Typ	Default-Wert	Übernahme
S_ActivateOut	SAFEBOOL	FALSE	kontinuierlich

Tab. 17-20: Eingangsverhalten NetVarSender

Funktionsbeschreibung

Der Funktionsbaustein NetVarSender implementiert eine Safety Netzwerkvariablenliste (NVL)

Der NetVarSender-Funktionsbaustein verwaltet die Übertragung der Netzwerklistenvariablen. Er stellt von sich aus keine Verbindung her. Angenommene Verbindungen werden nicht überwacht.

Wird der NetVarSender-Funktionsbaustein in der Applikation SafetyApp verwendet, kann der Anwender Einfluss auf die übertragenen Werte nehmen, in dem er Failsafe-Werte aktiviert.

⚠ VORSICHT **Die Netzwerkvariablenliste läuft mit Initialwerten automatisch an und bestätigt die Aufnahme der Sicherheitsfunktion**

Die Default-Werte der Eingänge sind so festgelegt, dass die Netzwerkvariablenliste automatisch anläuft und die Aufnahme der Sicherheitsfunktion automatisch bestätigt.

Durch die Implementierung anderer System- und Applikationsmaßnahmen ist sicherzustellen, dass es beim Start der Applikation nicht zu einem unerwarteten (oder unbeabsichtigten) Starten der Maschine kommt.



Funktionsbausteine vom Typ NetVarSender sollten in einer Applikation nur einmal explizit aufgerufen werden, da die Eingänge ansonsten vor ihrer Auswertung überschrieben werden.

Verwendung der NetVarSender Funktionsbaustein-Instanz in der Applikation

Der Anwender sollte den Funktionsbaustein in der Sicherheitsapplikation verwenden zur

- Änderung der Default-Werte (ActivateOut)
- gezieltes Versetzen der sicheren NVL in den sicheren Zustand (ActivateOut)

17.4.2 NetVarReceiver

Kurzbeschreibung

Der Funktionsbaustein NetVarReceiver implementiert eine Safety Netzwerkvariablenliste (Receiver-NVL) und verwaltet die Übertragung der Netzwerklistenvariablen. Er stellt die Verbindung zu einer Safety Netzwerkvariablen-Sendeliste her. Die Verbindung wird überwacht.

Zielsystem	Bibliothek	Version
SafeLogic 15VRS	SafetyNetVar	1.2.0.0

Tab. 17-21: Bibliothekszuordnung

Schnittstellenbeschreibung

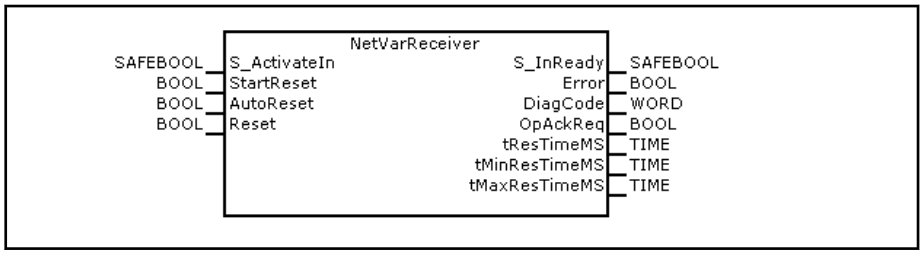


Abb. 17-6: Funktionsbaustein NetVarReceiver

E/A-TYP	Name	Datentyp	Beschreibung
VAR_INPUT	S_ActivateIn	SAFEBOOL	Aktivierung der Prozessdaten FALSE: Die Verwendung von failsafe Werten wird erzwungen: Alle Werte werden mit 0 bzw. False übertragen TRUE: Die Verwendung von failsafe Werten wird nicht erzwungen. Es sollen im Normalfall Prozessdaten übertragen werden. *Vorsicht:* Das Setzen dieses Eingangs von FALSE auf TRUE kann zur sofortigen Übertragung der Prozessdaten führen! Stellen Sie durch geeignete Maßnahmen sicher, dass von Ihrer Anlage/Maschine keine Gefahr ausgeht! Default-Wert: TRUE
	StartReset	BOOL	Verhalten nach einem Fehler während der Anlaufphase: Die Wirksamkeit des Eingangs StartReset erstreckt sich vom erster Start der Applikation bis die NVL einmal die Sicherheitsfunktion aufgenommen hat. (z.B. nach einem Reset kalt oder einem Neustart der Bootapplikation bzw. PowerON) Der Eingang "StartReset" entspricht dem Eingang <auto-Quitt-Anlauffehler> zur automatischen Quittierung von Fehlern beim Anlauf. TRUE: Automatische Bestätigung zur Aufnahme der Sicherheitsfunktion nach einem Fehler. FALSE: Manuelle Bestätigung am Eingang Reset zur Aufnahme der Sicherheitsfunktion nach einem Fehler notwendig. Default-Wert: TRUE Sicherheitshinweis zu Reset siehe Sicherheitshinweis zu StartReset, Seite 607

E/A-TYP	Name	Datentyp	Beschreibung
	AutoReset	BOOL	Wiederanlaufverhalten nach einem Fehler. Vorbedingung: Die NVL ist erfolgreich angelaufen und hat mindestens einmal die Sicherheitsfunktion aufgenommen. Der Eingang "AutoReset" entspricht dem Eingang <auto-Quitt-Unterbrechung> zur automatischen Quittierung nach Unterbrechungen. TRUE: Automatische Bestätigung zur Wiederaufnahme der Sicherheitsfunktion nach einem Fehler. FALSE: Manuelle Bestätigung am Eingang Reset zur Wiederaufnahme der Sicherheitsfunktion der NVL nach einem Fehler notwendig. Default-Wert: FALSE Sicherheitshinweis zu AutoReset siehe Sicherheitshinweis zu AutoReset, Seite 607
	Reset	BOOL	Mit einer steigenden Flanke des Eingangs wird die Wiederaufnahme der Sicherheitsfunktion nach einem Fehler bestätigt. Nach einer Bestätigung muss der Eingang mindestens einen vollen Zyklus auf FALSE stehen, bevor der nächste Fehler bestätigt werden kann. Der Eingang "Reset" entspricht dem Eingang <Quitt-Flanke> zur manuellen Quittierung. **Hinweis:** Wenn sowohl StartReset, als auch AutoReset auf TRUE gesetzt sind, wird der Eingang Reset nicht ausgewertet, da alle Anfragen zur Aufnahme der Sicherheitsfunktion automatisch bestätigt werden. **Hinweise:** Reset ist ein Eingang zur Bestätigung durch den Anwender. Er soll mit einem Eingangssignal verbunden werden und nicht programmatisch generiert werden. Default-Wert: FALSE **Hinweis:** Wenn aktuell keine Quittierung (OpAckReq ist FALSE) angefordert ist, wird der Eingang ignoriert: Deshalb kann das gleiche Signal mit dem Eingang zur Quittierungsflanke aller FB-Instanzen verbunden werden, um eine unspezifische Bestätigung von Kommunikationsproblemen zu realisieren.
VAR_OUTPUT	S_InReady	SAFEBOOL	Anzeige Aktivierung von Prozessdaten TRUE: Es werden Prozessdaten der NVL übertragen FALSE: Es werden failsafe Werte übertragen **Hinweis:** Der Ausgang wechselt auf FALSE, entweder auf Anweisung des Stacks oder aufgrund Übertragungsfehler bzw. interner Fehler in der NVL oder falls der Eingang **S_ActivateIn** gleich FALSE ist.
	Error	BOOL	Fehleranzeige TRUE: Es liegt ein Fehler vor (DiagCode = 16#Cxxx). FALSE: Es liegt kein Fehler vor (DiagCode = 16#8xxx).
	DiagCode	WORD	Verwendung des Ausgangs DiagCode als Diagnose-Code, Initialwert 16#0000 (siehe Diagnose-Code, Seite 609)

E/A-TYP	Name	Datentyp	Beschreibung
	OpAckReq	BOOL	Anzeige Operator Acknowledgement Request TRUE: Die NVL benötigt eine Bestätigung durch den Bediener (Operator Acknowledgement), um die Sicherheitsfunktion wieder aufzunehmen. FALSE: Die NVL benötigt derzeit keine Bestätigung durch den Bediener (Operator Acknowledgement). **Hinweis:** Dass eine Verbindung eine Bestätigung durch den Bediener benötigt, wird durch OpAckReq = TRUE signalisiert. Normalerweise wird dies dem Bediener gemeldet, Die entsprechende Logic muss in der Applikation programmiert werden.
	tRespTimeMS	TIME	Ausgang, der die aktuelle Durchlaufzeit für eine Nachricht angibt (siehe Zeitmessung, Seite 611).
	tMinRespTimeMS	TIME	Ausgang, der die minimale Durchlaufzeit für eine Nachricht angibt (siehe Zeitmessung, Seite 611). Ermittelt seit der letzten steigenden Flanke von Reset oder Systemstart.
	tMaxRespTimeMS	TIME	Ausgang, der die maximale Durchlaufzeit für eine Nachricht angibt (siehe Zeitmessung, Seite 611). Ermittelt seit der letzten steigenden Flanke von Reset oder Systemstart.

Tab. 17-22: Schnittstelle NetVarReceiver

Name	Typ	Default-Wert	Übernahme
S_ActivateIn	SAFEBOOL	TRUE	kontinuierlich
StartReset	BOOL	TRUE	Start des SPS-Programms
AutoReset	BOOL	FALSE	kontinuierlich
Reset	BOOL	FALSE	kontinuierlich

Tab. 17-23: Eingangsverhalten NetVarReceiver

Sicherheitshinweis zu StartReset

⚠ VORSICHT **StartReset**

Sind StartReset und AutoReset TRUE, so werden sämtliche Fehler bestätigt. Dies ist nur in absoluten Ausnahmefällen sinnvoll.

Der Initialwert für StartReset nach einem Reset ist TRUE. Es werden sämtliche Fehler für das Anlaufverhalten nach einem Reset bestätigt.

Treten beim Anlauf keine Fehler auf, so kann der Stack automatisch anlaufen, selbst wenn StartReset FALSE ist. Dies muss der Anwender in der Applikation berücksichtigen.

Sicherheitshinweis zu AutoReset

⚠ VORSICHT **AutoReset**

Sind StartReset und AutoReset TRUE, so werden sämtliche Fehler bestätigt. Dies ist nur in absoluten Ausnahmefällen sinnvoll.

Funktionsbeschreibung

Je Receiver-NVL gibt es eine Verbindung, die über eine FB-Instanz des NetVarReceiver überwacht und verwaltet wird.

Für jede Receiver-NVL wird automatisch vom Programmiersystem eine Verbindung und eine Instanz des Funktionsbausteins NetVarReceiver (FB-Instanz) erzeugt.

Beim Anlegen der FB-Instanz vom Typ NetVarReceiver wird impliziter Code generiert, der für Eingangs- bzw. Ausgangsvariablen folgendes bewirkt:

- **Phase 1:** Bearbeitung der Eingangstelegramme (implizit):

  Das Eingangstelegramm der NVL stammt von der Sender-NVL der Gegenstelle.

  Die FB-Instanz empfängt das Eingangstelegramm und überprüft dies gemäß der NVL-Spezifikation.

  Die Eingangsdaten oder im Fehlerfall die Ersatzwerte werden in die gemappten Eingangsvariablen der Applikation kopiert.
- **Phase 2:** Bearbeitung der Anwenderapplikation:

  Die Ausgangsdaten werden in Abhängigkeit der Eingangsdaten und des Zustands der Applikation generiert.
- **Phase 3:** Bearbeitung der Ausgangstelegramme (implizit):

  Die FB-Instanz generiert das Ausgangstelegramm gemäß der Spezifikation und schickt dieses an die Gegenstelle.

Wird der Funktionsbaustein NetVarReceiver vom Anwender in der Applikation explizit aufgerufen (Phase 2), dann können die FB-Eingänge gesetzt und die FB-Ausgänge gelesen werden, wobei der FB selbst keine Operationen ausführt.

Wird der Funktionsbaustein in der Applikation ein zweites Mal aufgerufen und die Eingänge werden ein zweites Mal gesetzt, so werden die Eingänge des ersten Aufrufs überschrieben, ohne dass der FB diese Eingänge ausgewertet hat. Ursache ist, dass die Auswertung der Eingänge nur in der Phase 3 und in der Phase 1 stattfindet.

Aufruf von Funktionsbausteinen unter Safety-Bedingungen siehe FB-Aufrufe, Seite 112

Funktionsbausteine vom Typ NetVarReceiver sollten in einer Applikation nur einmal explizit aufgerufen werden, da die Eingänge ansonsten vor ihrer Auswertung überschrieben werden.

Der Default-Wert des Eingangs "StartReset" ist mit "TRUE" so festgelegt, dass die Verbindung automatisch anläuft. D. h. dass evtl. notwendige Bestätigungen (Operator Acknowledge) für die Aufnahme der Sicherheitsfunktion nach Fehlern automatisch im FB generiert werden.

Der Default-Wert des Eingangs "AutoReset" ist "FALSE": Die Bestätigung zur Wiederaufnahme der Sicherheitsfunktion nach Fehlern muss manuell erfolgen.

Verwendung der NetVarReceiver-Funktionsbausteininstanz in der Applikation

Der Anwender sollte den Funktionsbaustein in der Sicherheitsapplikation verwenden zur

- Änderung der Default-Werte (StartReset, AutoReset)
- manuellen Bestätigung von Fehlern (Reset, OpAckReq)
- Diagnose
- gezieltes Versetzen einer NVL in den sicheren Zustand (S_ActivateIn)

Diagnose-Code

Der Parameter DiagCode ist in zwei Kategorien aufgeteilt.

- 16#8XXX für Zustände der Kommunikation mit Übertragung (siehe Zustände der Kommunikation mit Übertragung, Seite 609)
- 16#CXXX Fehlerzustände (siehe Fehlerzustände, Seite 609)

Zustände der Kommunikation mit Übertragung

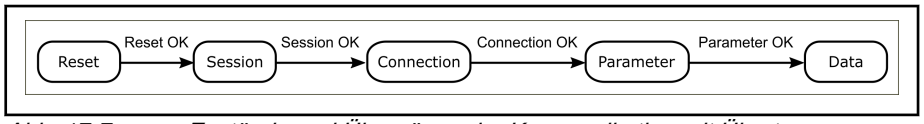


Abb. 17-7: Zustände und Übergänge der Kommunikation mit Übertragung

Der Diagnose-Code kann in den Zuständen der Kommunikation mit Übertragung folgende Werte annehmen:

DiagCode = 0x80xx, enthält den Zustand der NVL

DiagCode	Beschreibung
16#8064	(Initialisierung) "Reset": Die Verbindung ist zurückgesetzt (Ausgänge sind im sicheren Zustand)
16#8065	(Initialisierung) "Session": Es wird die Session-ID übertragen (Ausgänge sind im sicheren Zustand)
16#8066	(Initialisierung) "Connection": Es wird die Verbindungs-ID übertragen (Ausgänge sind im sicheren Zustand)
16#8067	(Initialisierung) "Parameter": Es werden die Parameter übertragen (Ausgänge sind im sicheren Zustand)
16#8068	(Datenaustausch) "Data": Prozessdaten oder Ersatzwerte werden übertragen. S_ActivateOut = TRUE : Übertragung Prozessdaten S_ActivateOut = FALSE : Übertragung Ersatzwerte

Fehlerzustände

Liegen im NetVarSender gleichzeitig zwei Fehler vor, wird immer nur der Fehler mit der höchsten Priorität angezeigt. Die Prioritäten der Fehler sind nach der Fehlerkategorie zugeteilt:

1. Fehlerkategorie 16#C0XX (Initialisierungsfehler)
2. Fehlerkategorie 16#C1XX(Empfangsfehler Receiver)
3. Fehlerkategorie 16#C2XX (vom Sender gemeldete Fehler)

Der Diagnose-Code kann im Fehlerzustand folgende Werte (aufgeteilt in 3 Fehlerkategorien) annehmen:

Konfigurations- und Initialisierungsfehler des NetVarReceiver

DiagCode = 16#C0XX (Konfigurations-und Initialisierungsfehler), XX enthält den Fehler-spezifischen Fehlercode:

DiagCode	Beschreibung
16#C001	Interne Fehler
16#C002	Interne Fehler
16#C003	Interne Fehler
16#C004	Interne Fehler
16#C005	Länge der sicheren Daten wird nicht unterstützt (unterstützt werden max. 32 Byte Daten)
16#C006	Interne Fehler
16#C00A	Interne Fehler

DiagCode	Beschreibung
16#C00B	Ungültige Überwachungszeit, Wert 0 (Parameter WatchdogTime) ist nicht erlaubt
16#C00C	Interner Fehler, ungültige logische Modul-ID



Zusätzlich werden Konfigurations- und Initialisierungsfehler der NVL beim Download in Form einer Fehlermitteilung angezeigt.

Fehlerzustand des NetVarReceiver

DiagCode = 16#C1XX (Fehlerzustand des NetVarReceiver), XX enthält den spezifischen Fehlercode:

DiagCode	Beschreibung
16#C100	Lokaler Reset oder Bestätigung eines Reset- Kommandos
16#C101	Unerwartetes Kommando im empfangenen Telegramm
16#C102	Unerwartetes Kommando im empfangenen Telegramm
16#C103	Ungültige Connection ID im empfangenen Telegramm
16#C104	CRC Fehler über das empfangene Telegramm
16#C105	Überwachungszeitfehler, kein gültiges Telegramm innerhalb von WatchdogTime erhalten
16#C106	Ungültige NVL Safety Device Adresse
16#C107	Ungültige Safety Daten im empfangenen Telegramm
16#C1FE	Fehler Reset-Signal
16#C1FF	Anforderung zur Quittierung des Wiederanlaufs

Fehlerzustand des NetVarSender

DiagCode = 16#C2XX (Fehlerzustand des NetVarReceiver, der an den NetVarReceiver übertragen wurde), XX enthält einen zusätzlichen fehlerspezifischen Fehlercode.

Bit in XX	Beschreibung
16#C200	Lokaler Reset oder Bestätigung eines Reset- Kommandos
16#C201	Unerwartetes Kommando im empfangenen Telegramm
16#C202	Unerwartetes Kommando im empfangenen Telegramm
16#C203	Ungültige Connection ID im empfangenen Telegramm
16#C204	CRC Fehler über das empfangene Telegramm
16#C205	Überwachungszeitfehler, kein gültiges Telegramm innerhalb von WatchdogTime erhalten
16#C206	Ungültige NVL Safety Device Adresse
16#C207	Ungültige Safety Daten im empfangenen Telegramm
16#C208	Ungültige Länge der Kommunikationsparameter
16#C209	Ungültige Daten der Kommunikationsparameter
16#C210	Ungültige Länge der Applikationsparameter
16#C211	Ungültige Daten der Applikationsparameter

Bit in XX	Beschreibung
16#C280	Definition der Netzwerkvariablenliste auf Seite des Safety-NVL (Empfänger) unterscheidet sich von jener auf Seite des Safety-NVL (Sender)
16#C2FE	Fehler Reset-Signal
16#C2FF	Anforderung zur Quittierung des Wiederanlaufs

Zeitmessung

Der Funktionsbaustein NetVarReceiver stellt drei Ausgänge zur Überwachung der Nachricht-Bearbeitungszeit zur Verfügung. Die Ausgänge dienen zur Diagnose der NVL-Kommunikation.

Als gemessene Durchlaufzeit einer Nachricht (NVL-Receiver -> NVL-Sender -> NVL_Receiver) wird die abgelaufene Zeit verwendet:

- Gemessen wird vom Absenden einer Anfrage vom NetVarReceiver zum NetVarSender bis zum Erhalt der Antwort vom NetVarSender.
- Um die Messungen nicht zu verfälschen, werden nur gültige Antworten berücksichtigt.
- Der Funktionsbaustein liefert die aktuelle, die minimale und die maximale Durchlaufzeit einer Nachricht.
- Die minimale und die maximale Durchlaufzeit wird seit Systemstart bzw. seit der letzten steigenden Flanke von Reset ermittelt.

17.4.3 Safety NetVar-spezifische Nachweise für die Abnahme

Bei der Abnahme einer Sicherheitsapplikation muss ein Nachweis erbracht werden für:

- Eindeutige Safety-Adresse jeder Sender-Netzwerkvariablenliste (Eingabefeld „Safety-Adresse dieser Variablenliste“ im Objekt „Safety Netzwerkvariablenliste (Sender)“ , Registerkarte „Safety Konfiguration“) im gesamten Netzwerk
- Eindeutige Connection ID jeder Empfänger-Netzwerkvariablenliste (im Objekt „Safety Netzwerkvariablenliste (Empfänger)“ , Registerkarte „Safety Konfiguration“) im gesamten Netzwerk
- Die Betriebsanleitung für die Maschine verpflichtet den Betreiber die Kommunikationsfehlerrate im Betrieb zu überwachen: Von der Treiberinstanz NetVarReceiver gemeldete Kommunikationsfehler dürfen nicht öfter als 1 Mal in 5 Stunden auftreten, damit die Restfehlerrate pro Stunde für die sicherheitsbezogenen Signale unter dem SIL3-Grenzwert von 10^{-9} bleibt
- Die Betriebsanleitung für die Maschine verpflichtet den Betreiber bei Update von Sicherheitsapplikationen oder Erweiterungen des Netzwerks erneut die Eindeutigkeit von Netzwerkvariablenlisten Safety-Adressen und Connection IDs nachzuweisen

18 Multi-Device Funktionalität mit SafeLogic

Allgemein

Maschinen bestehen häufig aus mehreren Maschinenmodulen mit vergleichbaren Aufgaben. Sowohl Hardware als auch Software dieser Module sind häufig nahezu identisch.

Die Multi-Device Funktionalität bietet die Möglichkeit, mehrere vergleichbare Steuerungen zu einer Applikation hinzuzufügen. Die Steuerungen in der Multi-Device Tabelle können einzeln ausgewählt und aktiviert werden, um auf der aktivierten Steuerung zu arbeiten (Online schalten, Einloggen, Debuggen, Bootapplikation erzeugen …).

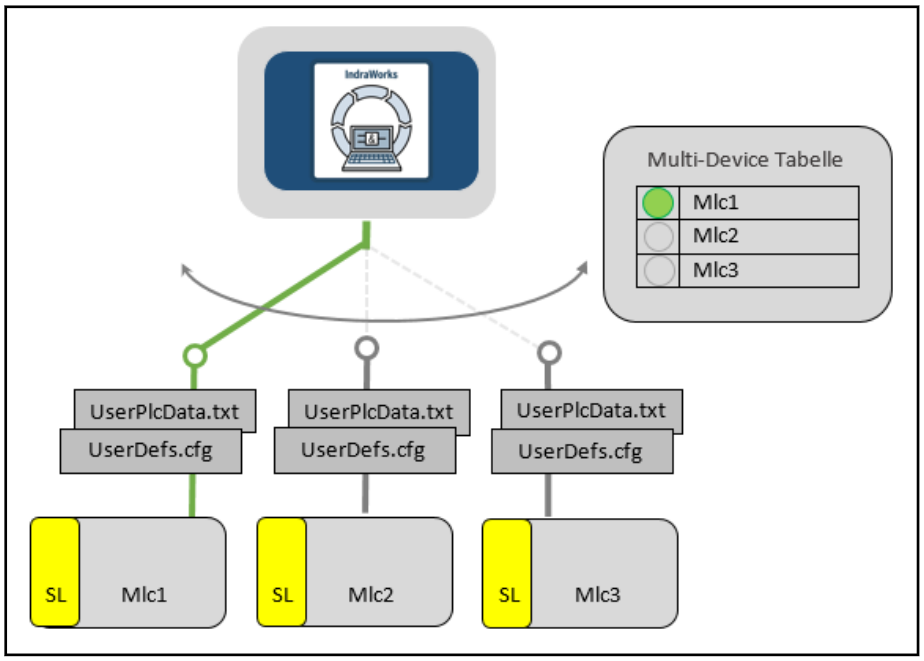


Abb. 18-1: Funktionsweise der Multi-Device Tabelle

Dieses Kapitel soll den Anwender mit folgenden Aspekten vertraut machen:

- Inbetriebnahme von Steuerungen mit SafeLogic bei Verwendung der Multi-Device Funktionalität
- Unterschiede zwischen individuellen- und einheitlichen (gemeinsamen) Steuerungseinstellungen innerhalb der Multi-Device-Funktion
- Eindeutige identifizierung von Safety-Steuerungen
- Hinweis auf wichtige Punkte im Zusammenhang mit dem Wechsel von Steuerungen über die Multi-Device Tabelle

Weiterführende Dokumentation

MLC 15VRS Funktionsbeschreibung

DOK-MLC***-FUNC****V15-APRS-DE-P, R911393304

Diese Dokumentation beschreibt allgemeine Vorgehensweisen für die Erstellung und Programmierung von Projekten auf der Standard-Steuerung

18.1 Inbetriebnahme der Multi-Device Funktionalität mit Safety-Steuerung

Erstinbetriebnahme

Für die Erstinbetriebnahme eines Projekts mit Multi-Device-Funktionalität empfiehlt sich folgende Vorgehensweise:

1. Auswahl einer Steuerung (Maschinenmodul), die als Vorlage verwendet werden soll. Diese Steuerung wird im Weiteren als Vorlagen-Steuerung bezeichnet.
2. Lauffähiges Projekt für Standard- und Safety-Steuerung erstellen, inklusive der Arbeitsschritte "Pinnen der Safety-Applikation" und "Erzeugen der Bootapplikation" für die Safety-Steuerung.

   Hierbei sind folgende Kapitel zu beachten:

   - Kap. 18.3 "Gemeinsame (einheitliche) Steuerungseinstellungen" auf Seite 618
   - Kap. 18.4 "Einschränkungen durch die Verwendung der Multi-Device Funktionalität" auf Seite 618
3. Konfiguration der Multi-Device Funktionalität (Kontextmenü der Standard-Steuerung)

☞ Bei Verwendung von UserDefs.cfg Konfigurationsdateien kann es sinnvoll sein, diese jetzt zu erstellen.

---

4. Steuerung in der Multi-Device Tabelle duplizieren
5. Gerätenamen und IP-Adressen der Standard-Steuerungen in der Multi-Device Tabelle gemäß der gewünschten Zielsteuerung anpassen

Folgende Schritte müssen in allen duplizierten Steuerungen durchgeführt werden:

1. Die gewünschte Standard-Steuerung in der Multi-Device Tabelle mittels Doppelklick auswählen
2. Über den Eigenschaften Dialog der Standard-Steuerung die Verbindung zur Steuerung testen
3. In die Standard-Steuerung Einloggen (mit Download) und Bootapplikation erzeugen
4. Gerätename für die Safety-Steuerung vergeben über das Kontextmenü der Steuerung im Projektexplorer (dabei Betätigungstaster an der gewünschten Safety-Steuerung betätigen)

   Weitere Einstellungen finden Sie im Kapitel Kap. 18.2.3 "Gerätenamen der Safety-Steuerung ändern" auf Seite 616
5. In die Safety-Steuerung einloggen
6. Bootapplikation auf der Safety-Steuerung erzeugen und aktivieren.

   Überprüfen, dass der Bootapplikations-Pin der gleiche ist, wie auf der Vorlagen-Steuerung.

Änderung eines Multi-Device Projekts

Für Änderungen eines Projekts mit Multi-Device-Funktionalität empfiehlt sich folgende Vorgehensweise:

1. Steuerung in der Multi-Device Tabelle auswählen, an der die Änderungen durchgeführt und überprüft werden soll. Diese Steuerung wird im Weiteren als Vorlagen-Steuerung bezeichnet.
2. Lauffähiges Projekt für Standard- und Safety-Steuerung erstellen, inklusive der Arbeitsschritte "Pinnen der Safety-Applikation" und "Erzeugen der Bootapplikation" für die Safety-Steuerung.

   Hierbei sind folgende Kapitel zu beachten:

   - Kap. 18.3 "Gemeinsame (einheitliche) Steuerungseinstellungen" auf Seite 618
   - Kap. 18.4 "Einschränkungen durch die Verwendung der Multi-Device Funktionalität" auf Seite 618

Folgende Schritte nacheinander in allen anderen Steuerungen durchführen:

1. Die gewünschte Standard-Steuerung in der Multi-Device Tabelle mittels Doppelklick auswählen
2. In die Standard-Steuerung Einloggen (mit Download) und Bootapplikation erzeugen
3. In die Safety-Steuerung Einloggen
4. Bootapplikation auf der Safety-Steuerung erzeugen und aktivieren.

   Überprüfen, dass der Bootapplikations-Pin der gleiche ist, wie auf der Vorlagen-Steuerung.

Der funktionale Test der Safety-Anwendung und die Abnahme muss für jede weitere Steuerung bzw. für jedes Maschinenmodul separat durchgeführt werden!

Es wird empfohlen, mit der Vorlagen-Steuerung zu beginnen.

Für weitere Steuerungen ist die Safety-Bootapplikation von der SD-Card der Vorlagen-Steuerung auf die SD-Cards der anderen Safety-Steuerungen zu kopieren.

18.2 Individuelle Steuerungseinstellungen

Für jede Steuerung, die in der Multi-Device Tabelle angelegt ist, können individuelle Einstellungen vorgenommen werden. Die Einstellungen bleiben auch nach dem Wechsel der Steuerung über die Multi-Device Tabelle erhalten.

18.2.1 Gerätenamen der Standard-Steuerung ändern

Der Gerätenamen der Standard-Steuerung kann individuell vergeben werden. Dieser Gerätename wird auch als Steuerungsname bezeichnet.

Der Gerätename wird auch im Projektexplorer angezeigt. Dadurch bekommt man eine erste Kontrolle, auf welcher Steuerung man sich gerade befindet. Das Umbenennen kann direkt in der Spalte "Steuerungsname" der Multi-Device Tabelle oder einzeln über den Projektexplorer erfolgen.

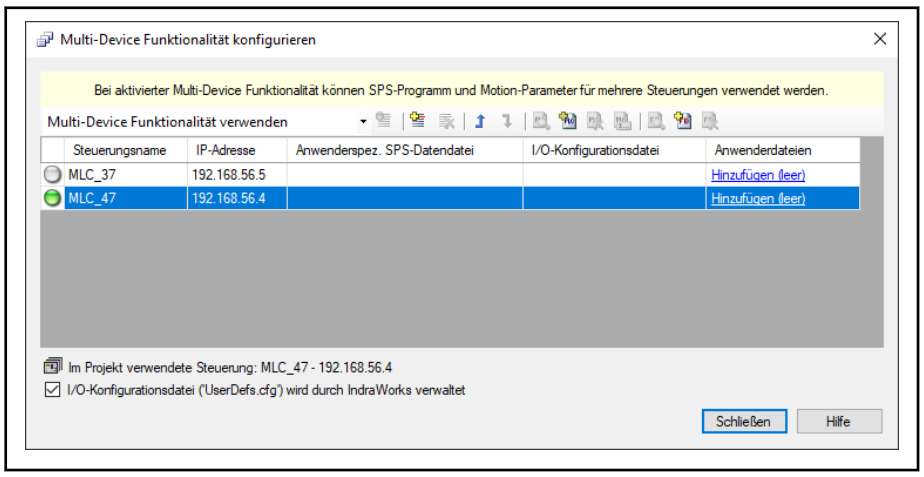


Abb. 18-2: Umbenennen der Standard-Steuerung in der Multi-Device Tabelle



Um die Steuerungen klar voneinander unterscheiden zu können, sollte der Steuerungsname direkt nach dem Anlegen der Steuerung in der Multi-Device Tabelle vergeben werden.

18.2.2 IP-Adresse der Standard-Steuerung ändern

Über die Multi-Device Tabelle lassen sich die IP-Adressen der beteiligten Standard-Steuerungen angeben.

18.2.3 Gerätenamen der Safety-Steuerung ändern

Der Gerätename der Safety-Steuerung (Safety-CPU) lässt sich individuell einstellen. Der Gerätename wird in der Safety-Steuerung (Safety-CPU) gespeichert.

Die Namensänderung erfolgt im Dialog "Verbindung zur Sicherheitssteuerung".

Der Dialog kann über das Kontextmenü der Sicherheitssteuerung im Projektexplorer aufgerufen werden → **Gerätename ändern**

Im Projektexplorer wird weiterhin das Geräteobjekt angezeigt, z. B. "Safety_CPU_XFE01_1_SY_01".

Der Gerätename der Safety-Steuerung ermöglicht die Identifizierung des Geräts in vielen Dialogen.

Siehe hierzu "Wo und Wann wird der Gerätename der Safety-CPU angezeigt?" auf Seite 621

☞ Der Gerätename der Safety-Steuerung sollte so gewählt werden, dass der Anwender auch auf die zugehörige Standard-Steuerung schließen kann.

Durch die Vergabe eines individuellen Gerätenamens kann der Anwender überprüfen, dass er auf der gewünschten Safety-Steuerung arbeitet.

Daher ist die Vergabe eines individuellen Gerätenamens durchzuführen und bei der Abnahme zu überprüfen.

18.2.4 Administrationspasswort für die Safety-Steuerung

Das Administrationspasswort lässt sich für Safety-Steuerungen individuell einstellen. Die Abfrage des Admin-Passworts erfolgt bei erstmaligem, schreibendem Zugriff und bei schreibendem Zugriff nach dem Wechsel der Steuerung über die Multi-Device Tabelle.

Nach dem Wechsel der Steuerung über die Multi-Device Tabelle kann es zu folgender Fehlermeldung kommen:

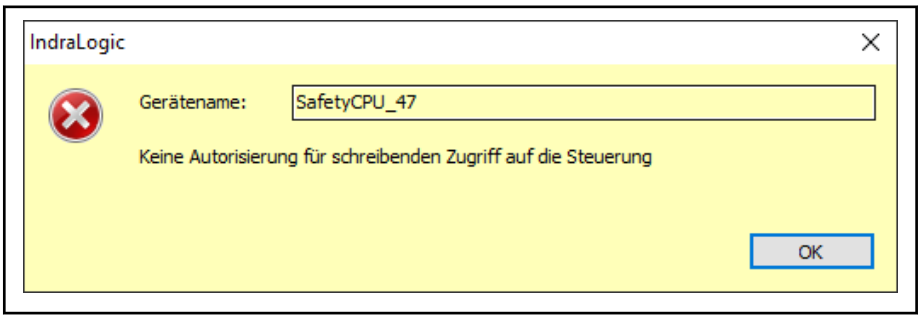


Abb. 18-3: Fehlermeldung Admin-Passwort

☞ Um das Administrationspasswort einstellen zu können, muss die Benutzerverwaltung aktiviert werden.

18.2.5 Fernwartung für die Safety-Steuerung

Die Fernwartung bietet die Möglichkeit, Zustände zu beobachten, jedoch ohne die Möglichkeit Werte zu ändern. Der Fernzugang muss für jede Safety-Steuerung individuell konfiguriert werden.

Dazu muss über die Multi-Device Tabelle eine Steuerung nach der anderen ausgewählt werden.

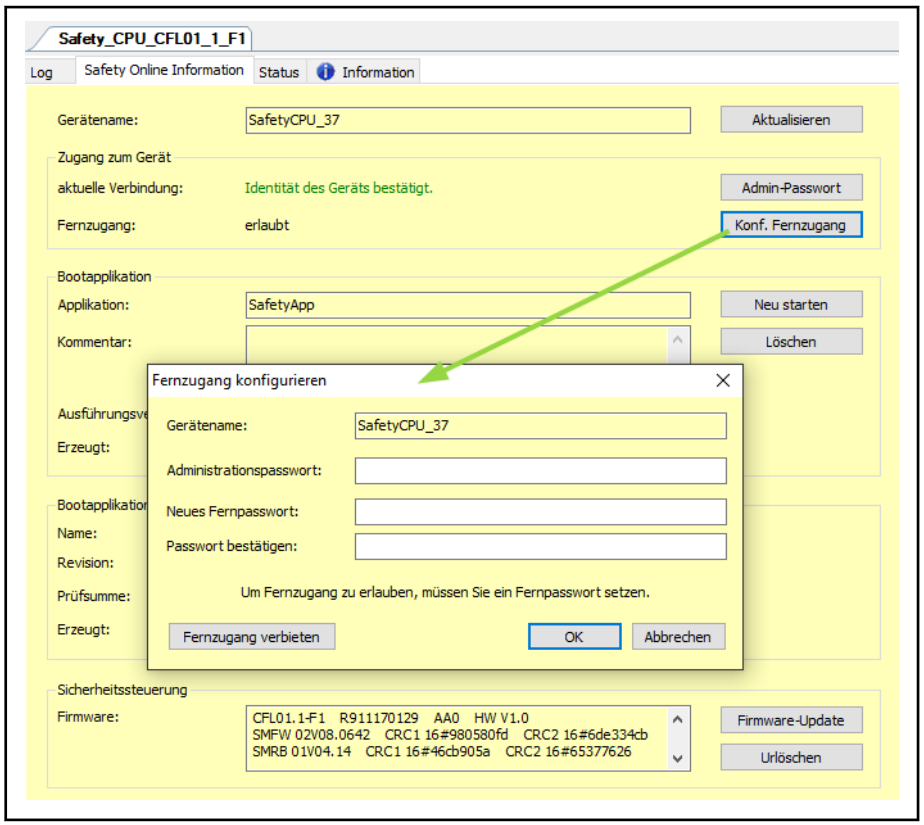


Abb. 18-4: Fernzugang einrichten

Um die Handhabung zu vereinfachen und um Verwechslungen zwischen den Steuerungen zu vermeiden empfiehlt es sich, die Passwörter der Steuerungen nach einem definierten Schema aufzubauen.

Beispielsweise aus einem gemeinsamen Passwort-Teil und einer individuellen Kennung je Steuerung:

Passwort Aufbau	
Gemeinsamer Teil für alle Steuerungen	Individuelle Endung (z. B. Steuerungsnummer)
*********	37

☞ Um den Fernzugriff konfigurieren zu können, muss die Benutzerverwaltung aktiviert sein.

☞ Zum Konfigurieren eines Fernpasswortes wird die Lizenz "IndraWorks SafetyManager Lite" benötigt.

18.3 Gemeinsame (einheitliche) Steuerungseinstellungen

Der Vorteil der Multi-Device Funktionalität liegt in der Vereinheitlichung der Steuerungseinstellungen, da hiermit der Engineering-Aufwand reduziert wird.

Die Vereinheitlichung führt teilweise aber auch zu Einschränkungen im Aufbau, siehe Kap. 18.4 "Einschränkungen durch die Verwendung der Multi-Device Funktionalität" auf Seite 618

18.3.1 Projekt pinnen

Mittels Pin-Kennung ist es möglich, einen bestimmten Stand der Safety-Applikation im Projekt, eines Objekts im Editor und einer Bootapplikation auf der Safety-Steuerung zu identifizieren. Bei der Verwendung der Multi-Device Tabelle muss die Safety-Applikation einmalig für alle Safety-Steuerungen gepinnt werden.



Der Anwender muss sicherstellen, dass auf allen Safety-Steuerungen der Multi-Device Tabelle die gepinnte Bootapplikation geladen ist, siehe "Bootapplikation ermitteln" auf Seite 619

18.3.2 Benutzerverwaltung

Die Einstellungen für die Benutzerverwaltung müssen einmalig vorgenommen werden und sind dann für alle Safety-Steuerungen identisch.

18.3.3 SafetyApp umbenennen

Es wird empfohlen, die SafetyApp umzubenennen z. B. in "SafetyAppMulti", um im Projektexplorer einen Hinweis auf die Multi-Device Funktionalität zu erhalten. Sind mehrere Safety-Steuerungen im Projekt enthalten, müssen sich die Namen der SafetyApps unterscheiden, z.B. SafetyAppMulti1 SafetyAppMulti2 usw.

18.4 Einschränkungen durch die Verwendung der Multi-Device Funktionalität

18.4.1 Sichere Feldbusse

CIP Safety on Sercos

Die Safety Network Number (SNN Alias) und die Safety Device ID (SDID) des CSoS[1)] -Originators lassen sich nicht individuell für die einzelnen Steuerungen der Multi-Device-Tabelle einstellen.

Das führt zu Einschränkungen in der Topologie. Über die Multi-Device Funktion lassen sich keine Topologien erstellen, bei denen mehrere Safety-Steuerungen miteinander kommunizieren.

Alle Steuerungen der Multi-Device-Tabelle haben die gleichen CSoS-Targets mit den gleichen Werten für die CSoS-Adressen (Safety Network Number (SNN) und die Safety Device ID (SDID).).

Dadurch können für CSoS keine individuellen CSoS-Adressen für die CSoS-Targets einzelner Steuerungen vergeben werden.

Da die CSoS-Adressen eindeutig sein müssen, müssen die Sercos-Netzwerke der Steuerungen voneinander isoliert bzw. getrennt sein.

1) *CIP Safety on Sercos*

PROFIsafe

Alle Steuerungen der Multi-Device-Tabelle haben die gleichen F-Devices mit den gleichen Werten für die F-Adressen ("F_Source_Add" und "F_Dest_Add").

Dadurch können für PROFIsafe keine individuellen F-Adressen für einzelne Steuerungen vergeben werden.

Da die F-Adressen eindeutig sein müssen, müssen die Busse der Steuerungen (z.B. PROFINET) voneinander isoliert bzw. getrennt sein.

18.4.2 Safety Netzwerkvariablen (NVL)

Um Safety Netzwerkvariablen nutzen zu können, müssen sich mindestens zwei Safety-Steuerungen im Projekt befinden. Daher ist der Datenaustausch über Safety Netzwerkvariablen zwischen Safety-Steuerungen (Topologie T3) in der Multi-Device Tabelle nicht möglich.

18.5 Maximalprojekt erstellen

Allgemein

Bei sicheren Feldbusteilnehmern kann ebenfalls ein Maximalprojekt erstellt werden.

Dabei ist folgendes zu beachten:



Projektierte Module, die aufgrund der Maschinenkonfiguration in der realen Maschine nicht vorhanden sind, müssen deaktiviert werden.

Feldbusgeräte über UserDefs.cfg deaktivieren

Alternativ zur Deaktivierung von fehlenden Teilnehmern im Projektexplorer Gerätebaum besteht die Möglichkeit, Teilnehmer über die UserDef.cfg Datei in der Multi-Device Tabelle zu deaktivieren. Dadurch wird der Aufwand bei der Projektierung reduziert.

Deaktivieren von sicheren Feldbusteilnehmern

Durch das Deaktivieren von sicheren Feldbusteilnehmern lassen sich mit einem einzigen Projekt verschiedene Maschinenkonfigurationen realisieren. Der Einlernvorgang muss für jede Steuerung einzeln vorgenommen werden. Die Kontrolle der korrekten Konfiguration kann über einen Vergleich der Prüfnummer der aktuellen Konfiguration (S_CRC) im Baustein `SF_IOConfigCntl` mit einer bereits bekannten Konfiguration durchgeführt werden.

Siehe Kap. 16.3.14 "SF_IOConfigCntl" auf Seite 266

18.6 Ergänzende Hinweise

Logbuch (Standard- und Safety-Steuerung)

Nach dem Öffnen des Geräteeditors werden die Logbücher nicht automatisch aktualisiert. Um sicher zu stellen, dass Einträge der aktuellen Steuerung vollständig vorliegen, muss der Anwender nach dem Öffnen eines Logbuchs die Einträge aktualisieren.

Siehe Kap. 13.3.2 "Logbuch" auf Seite 198

Bootapplikation ermitteln

Zur Ermittlung der Bootapplikation auf der Steuerung existieren folgende Möglichkeiten:

1. Mittels Überprüfung der Pin-Kennung in der Registerkarte "Safety Online Information" im Geräteeditor der Safety-CPU. Betätigen Sie hierzu die Schaltfläche "Aktualisieren". Hier können Sie auch den Gerätenamen der Safety-CPU einsehen.

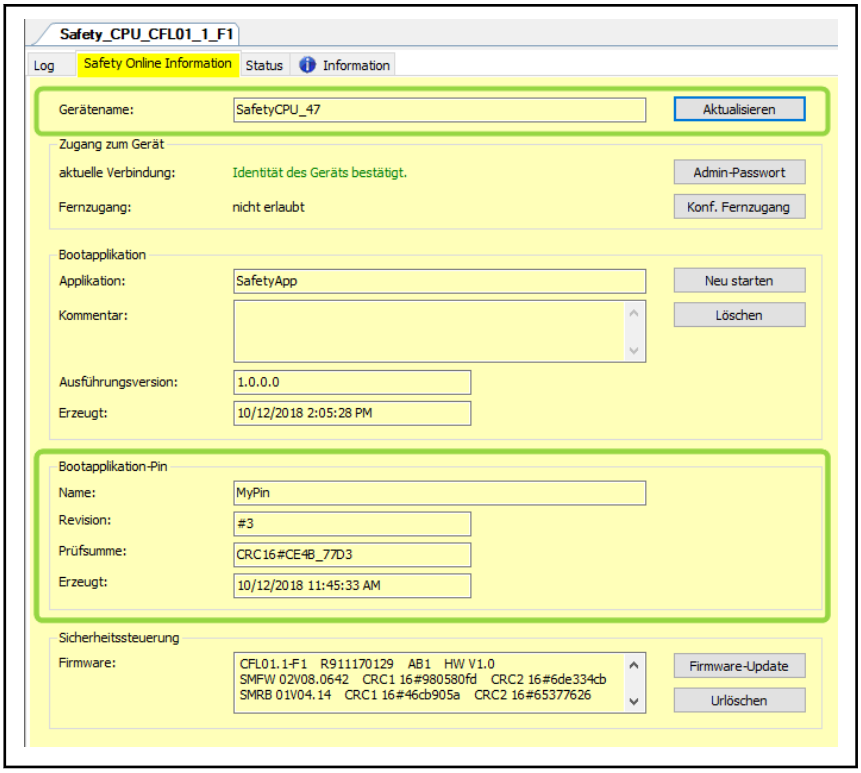


Abb. 18-5: Anzeige der Pin-Kennung in der Registerkarte "Safety Online Information"

Siehe "Safety Online Information" auf Seite 40

2. Mittels Überprüfung der Pin-Kennung beim Einloggen in die Steuerung.

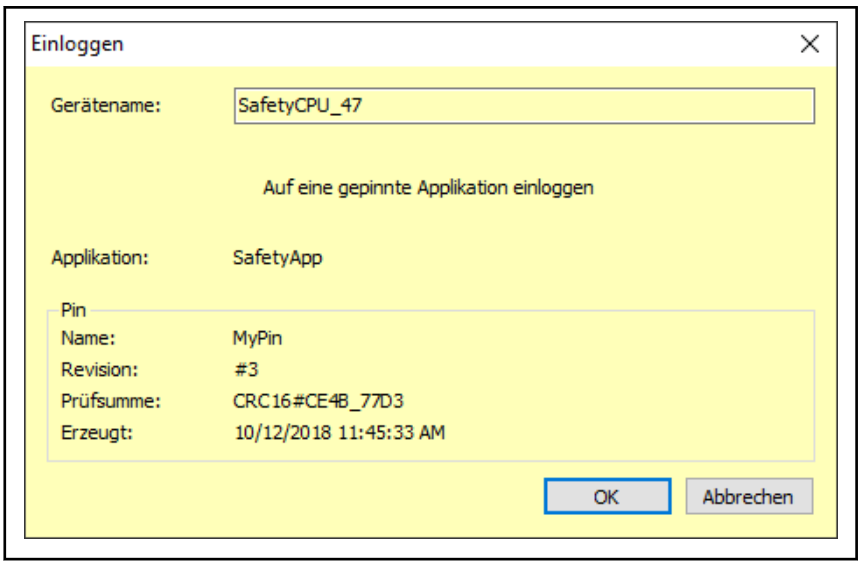


Abb. 18-6: Anzeige der Pin-Kennung beim Einloggen

3. Mittels Überprüfung der Pin-Kennung beim Ausloggen aus der Safety-Steuerung.

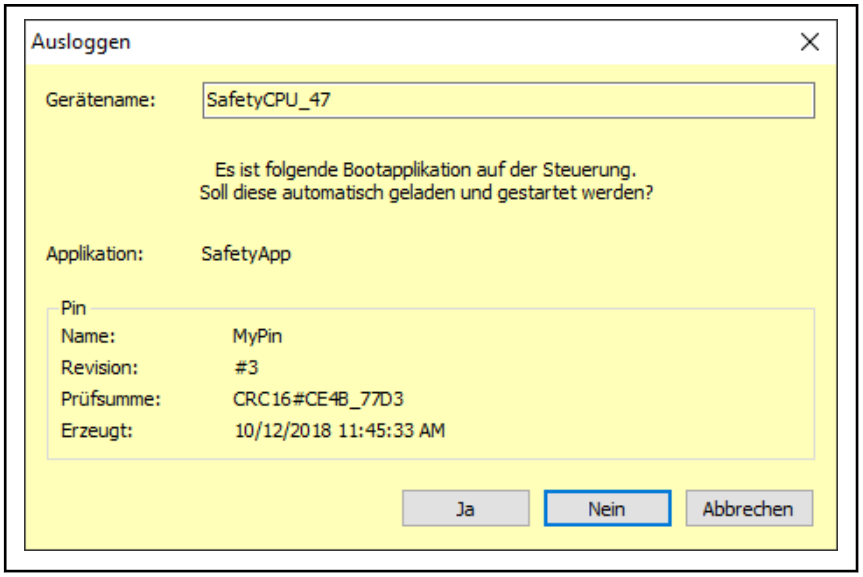


Abb. 18-7: Anzeige der Pin-Kennung beim Ausloggen

Es muss sichergestellt sein, dass auf allen Safety-Steuerungen eine Bootapplikation mit der korrekten Pin-Kennung vorhanden ist. Die Pin-Kennung muss der Pin-Kennung eines verifizierten Applikationsstandes entsprechen.

18.7 FAQ – Multi-Device

Wann muss die Safety-Steuerung per Tastendruck bestätigt werden?

- Nach dem Wechsel der Steuerung über die Multi-Device Tabelle ist bei der nächsten Bedienaktionen eine Bestätigung der Safety-Steuerung per Tastendruck notwendig.
- Beim Einloggen in die Safety-Steuerung, siehe Kap. 8.1 "Eingeloggter Zustand, Einführung" auf Seite 125
- Beim Aktualisieren des Geräteeditors der Safety-Steuerung in den Registerkarten "Safety-Logbuch" und "Safety Online Information", siehe:
  - Kap. 13.3.2 "Logbuch" auf Seite 198
  - "Safety Online Information" auf Seite 40
- Beim Ändern des Gerätenamens der Safety-Steuerung ist immer eine Bestätigung per Tastendruck notwendig, auch wenn vorher die Verbindung bereits bestätigt wurde

Wenn der Fernzugang aktiviert ist, erfolgt die Bestätigung mit dem Fernpasswort.

Wo und Wann wird der Gerätename der Safety-CPU angezeigt?

Im Baum des Projektexplorers wird immer das Geräteobjekt angezeigt, z.B. "Safety_CPU_XFE01_1_SY_01".

Der Name der Safety-CPU kann über die Registerkarte "Safety Online Information" im Geräte-Editor ausgelesen werden. Weiterhin wird der Gerätename der Safety-CPU in folgenden Situationen angezeigt:

- Bei Verbindungsbestätigung Safety, siehe Kap. 8.2 "Verbindung zur Sicherheitssteuerung" auf Seite 126
- Bei der Autorisierung des Benutzers, beim erstmaligen Einloggen nach dem Hochfahren, siehe Kap. 6.16.2 "Benutzerverwaltung im Projekt" auf Seite 78

- Beim Laden der Bootapplikation, siehe Kap. 8.5 "Erzeugen und Neustart der Bootapplikation" auf Seite 136
- Beim Einloggen (Debug-Modus, gepinnt)
- Beim Ausloggen (Debug-Modus, nach dem Erzeugen einer Bootapplikation)
- Beim Konfigurieren des Fernzugangs, siehe Kap. 18.2.5 "Fernwartung für die Safety-Steuerung" auf Seite 617
- Beim Ändern des Gerätenamens
- Bei fehlgeschlagenem Verbindungsaufbau

Wann ist ein Wechsel der Steuerung über die Multi-Device Tabelle möglich?

Für einen Wechsel der Steuerung über die Multi-Device Tabelle müssen folgende Bedingungen erfüllt sein:

1. Die Standard-Applikation ist ausgeloggt
2. Die Safety-Applikation ist ausgeloggt
3. Die Steuerung ist offline (Motion)

Wann sind Änderungen in der Multi-Device Tabelle möglich?

Für das Ändern der Einstellungen in der Multi-Device Tabelle müssen folgende Bedingungen erfüllt sein:

1. Die Standard-Applikation ist ausgeloggt
2. Die Safety-Applikation ist ausgeloggt
3. Die Steuerung ist offline (Motion)

19 Versionsliste der Safety-Bausteine

Hinweis Lib_1 (Version)

⚠ VORSICHT

Die Version jedes in die Applikation eingebundenen Bibliotheksbausteins muss mit der dokumentierten Version des Bausteins übereinstimmen (siehe SafeLogic Dokumentation und neuester Eintrag in der Spalte „Version“).

Hinweis Lib_2 (Gültig)

⚠ VORSICHT

Wenn bei einem Baustein zu einer bestimmten Version in der Spalte "Gültig" die Angabe "Nein" steht, so darf diese Version des Bausteins nicht mehr verwendet werden.

Sicherheitsbibliothek SafetyStandard

Baustein	Version	Gültig	Bemerkung
SF_CTD	1.0.0.0	Ja	
SF_CTU	1.0.0.0	Ja	
SF_CTUD	1.1.1.0	Ja	Eine steigende Flanke an einem der Eingänge CU bzw. CD bewirkt den entsprechenden Zählvorgang, auch wenn der andere Eingang bereits gleich TRUE gesetzt ist
	1.0.0.0	Nein	
SF_F_TRIG	1.0.0.0	Ja	
SF_R_TRIG	1.0.0.0	Ja	
SF_RS	1.0.0.0	Ja	
SF_SR	1.0.0.0	Ja	
SF_TOF	1.0.0.0	Ja	
SF_TON	1.0.0.0	Ja	
SF_TP	1.0.0.0	Ja	

Sicherheitsbibliothek SafetyPLCopen

Hinweis Lib_3 (Produktnormen)

☞ Beim Einsatz jedes Bausteins zur Überwachung oder Steuerung eines Sicherheitsbauteils müssen auch die Anforderungen der entsprechenden Produktnormen beachtet werden.

Baustein	Version	Gültig	Bemerkung
SF_Antivalent	1.0.0.0	Ja	
SF_EDM	1.0.0.0	Ja	
SF_EmergencyStop	1.0.0.0	Ja	
SF_EnableSwitch	1.0.0.0	Ja	
SF_Equivalent	1.0.0.0	Ja	

Baustein	Version	Gültig	Bemerkung
SF_ESPE	1.0.0.0	Ja	
SF_GuardLocking	1.0.0.0	Ja	
SF_GuardMonitoring	1.0.0.0	Ja	
SF_ModeSelector	1.1.0.0	Ja	Der FB wechselt bei einer Modusänderung innerhalb von Zustand 8004 und anschließendem Setzen von S_Unlock auf TRUE nicht mehr in Zustand 8000 (Ausgänge neu setzen), sondern in Zustand 8005 (Warten auf Aktivierung) Die abgelaufene Zeit zur Prüfung gegen ModeMonitorTime wird in Zustand 8005 mit jeder Modusänderung zurückgesetzt
	1.0.0.0	Nein	
SF_MutingPar	1.1.0.0	Ja	S_AOPD_In gleich FALSE in den Zuständen 8014, 8114, 8314 und 8414 bedingt keinen Fehler (siehe auch entsprechender Hinweis in SF_MutingPar, Seite 356) MutingEnable gleich FALSE in Zustand 8000 bedingt keinen Fehler
	1.0.0.0	Nein	
SF_MutingPar_2Sensor	1.0.0.0	Ja	
SF_MutingSeq	1.0.0.0	Ja	
SF_OutControl	1.0.0.0	Ja	
SF_SafetyRequest	1.0.0.0	Ja	
SF_TestableSafetySensor	1.0.0.0	Ja	
SF_TwoHandControlTypeII	1.0.0.0	Ja	
SF_TwoHandControlTypeIII	1.0.0.0	Ja	

Sicherheitsbibliothek SafetyProfisafeHost

Baustein	Version	Gültig	Bemerkung
ProfisafeHost	1.0.0.0	Ja	

Sicherheitsbibliothek SafetyNetVar

Baustein	Version	Gültig	Bemerkung
NetVarReceiver	1.2.0	Ja	
NetVarSender	1.2.0	Ja	
NetVarSenderStack	1.2.0	Ja	Interner Baustein, in der Applikation nicht zugreifbar

20 Spezifische Sicherheitshinweise für applikative Bibliotheksbausteine

Hinweis Lib_4 (SF_F_TRIG)

Betrifft: `SF_F_TRIG`

⚠ VORSICHT

Besonderes Verhalten des Bausteins im 1. Zyklus

Im ersten Zyklus erkennt `SF_F_TRIG` eine fallende Flanke bei anliegendem `FALSE` und setzt den Ausgang `Q` auf `TRUE`. Dieses Verhalten ist konform zum `F_TRIG` von IEC 61131-3 ab edition2. Es weicht somit vom Verhalten des Bausteins `F_TRIG` von der Standardsteuerung ab, welcher konform zu IEC 61131-3 edition1 ist.

Hinweis Lib_5 (TIME-Eingänge)

Betrifft: `SF_Equivalent`, `SF_Antivalent`, `SF_GuardMonitoring`, `SF_ModeSelector`, `SF_SafetyRequest`, `SF_EDM`, `SF_TestableSafetySensor`, `SF_MutingSeq`, `SF_MutingPar`, `SF_MutingPar_2Sensor`

⚠ VORSICHT

TIME-Eingänge

Für Entwickler im Extended Level gilt: Alle `TIME`-Eingänge dürfen nur mit konstanten Werten beschaltet werden. D.h. der Wert darf bei den Aufrufen nicht verändert werden.

Betroffene Eingänge:

- `DiscrepancyTime`
- `DiscTimeEntry`
- `DiscTime11_12`
- `DiscTime21_22`
- `MonitoringTime`
- `ModeMonitoringTime`
- `MaxMutingTime`
- `TestTime`

Hinweis Lib_6 (AutoSetMode)

Betrifft: `SF_ModeSelector`

⚠ VORSICHT

AutoSetMode

Der `AutoSetMode`-Eingang soll nur aktiviert werden, wenn sichergestellt ist, dass keine Gefährdung auftreten kann, wenn die Sicherheitssteuerung gestartet wird.

Hinweis Lib_7 (Muting-Sensorsignale)

Betrifft: `SF_MutingSeq`, `SF_MutingPar`

⚠ VORSICHT	**Ein Kurzschluss bei den Muting-Sensorsignalen, oder ein funktionaler Applikationsfehler beim Unterstützen dieser Signale wird von diesem Baustein nicht unterstützt, aber als inkorrekte Mutingsequenz interpretiert (Datentyp `BOOL` (nicht sicher), versorgt durch die funktionale Anwenderhardware oder - software). Es muss sichergestellt sein, dass so ein Fall nicht zu einem ungewollten Muting führt. Der Anwender sollte dies in seiner Risikoanalyse berücksichtigen.**

Hinweis Lib_8 (Not S_AOPD_In)

Betrifft: `SF_MutingPar`

☞ Die Implementierung des Zustandsmodells weicht von der PLCopen Spezifikation 1.0 [N2.1.1] in einem Fall ab: Mit `FALSE` am Eingang `S_AOPD_In` wird auch bei aktivem Muting (Zustände 8012, 8021, 8112, 8121) in den Zustand SafetyDemand AOPD gewechselt.

21 Liste zulässiger oder geänderter Funktionen

21.1 Zulässige Befehle

Zulässige Menübefehle

Befehle der Kategorie 'Safety Applikation und Safety Task'	
alle Befehle	
Befehle der Kategorie 'Safety Deklarationen'	
alle Befehle	
Befehl der Kategorie 'Safety FUP'	
alle Befehle	
Befehle der Kategorie 'Safety Online'	
alle Befehle	
Befehle der Kategorie 'Debuggen' // 'Online'	
Einloggen	Ausloggen
Bootapplikation erzeugen	Reset kalt
Start (aktive Applikation)	Stop (aktive Applikation)
Start (selektierte Applikation)	Stop (selektierte Applikation)
Werte schreiben (aktive Applikation)	Werte forcen (aktive Applikation)
Forcen für alle Werte aufheben	Alle Forces zur Überwachungsliste hinzufügen
Forceliste aufheben	Ablaufkontrolle (aktive Applikation)
Binär	Zur Überwachungsliste hinzufügen
Hexadezimal	Dezimal
Überwachungsliste 2	Überwachungsliste 1
Überwachungsliste 4	Überwachungsliste 3
Alle Forces anzeigen	
Befehle der Kategorie 'Datei'	
Vergleichen	Archiv extrahieren
Archiv speichern/versenden	Beenden
Neues Projekt	Projekt öffnen
Projekt schließen	Projekt speichern
Befehle der Kategorie 'Datei'	
Projekt speichern unter	Projektinformationen
Projekteinstellungen	
Befehle der Kategorie 'Drucken'	
Dokumentieren	Drucken
Befehle der Kategorie 'Quelltextnavigation'	
Querverweise ausgeben	Gehe zur Definition

Befehle der Kategorie 'Übersetzen'	
Übersetzen	Bereinigen
Alles bereinigen - Achtung!	Bibliothekskompatibilität prüfen
	Neu übersetzen
Befehle der Kategorie 'Hilfe'	
Safety-Versionsinformation anzeigen	Hilfe
Erste Schritte	Hilfe Antriebe
Rexroth Homepage	Info
Befehle der Kategorie 'Intelligente Code-Bearbeitung'	
Eingabehilfe	Variable deklarieren
Befehle der Kategorie 'Lesezeichen'	
Lesezeichen ein-/ausschalten	Lesezeichen löschen
Nächstes Lesezeichen	Vorheriges Lesezeichen
Befehle der Kategorie 'Zwischenablage'	
Alles selektieren	Ausschneiden
Einfügen	Kopieren
Löschen	
Befehle der Kategorie 'Rückgängig/Wiederherstellen'	
Rückgängig	Wiederherstellen
Befehle der Kategorie 'Suchen/Ersetzen'	
Ersetzen	Suchen
Vorheriges suchen	Vorheriges suchen (Auswahl)
Weitersuchen	Weitersuchen (Auswahl)
Befehle der Kategorie 'Meldungsfenster'	
Gehe zur Quelltextposition	Nächste Meldung
Vorherige Meldung	
Befehle der Kategorie 'Einstellungen'	
Anpassen	Optionen
Befehle der Kategorie 'Benutzerverwaltung'	
Benutzer anmelden	Benutzer abmelden
Rechte	
Befehle der Kategorie 'Ansicht'	
Werkzeuge	Querverweisliste
Safety Querverweisliste	Element-Eigenschaften
Speicher	Meldungen
Befehle der Kategorie 'Objekte'	
Objekt hinzufügen	Objekt bearbeiten

Exportieren	Importieren
Aktive Applikation setzen	Objekt offline bearbeiten
Ordner hinzufügen	
Befehle der Kategorie 'Gerätekommunikation'	
Gerätenamen ändern	
Befehle der Kategorie 'Geräte'	
Geräte-Repository	Gerät aktualisieren
Befehle der Kategorie 'Installation'	
Bibliotheks-Repository	

21.2 Zulässige Ansichten

Zulässige Editoren, ihre Registerkarten und Steuerelemente

Editor der Sicherheitssteuerung

Registerkarte	Steuerelement
Log	Stammt aus der IndraLogic-Basis und kann deshalb variieren. Es sind nur die im Folgenden aufgelisteten Steuerelemente zulässig: • Option "Offline-Aufzeichnung" • Option "UTC Zeit" • Auswahlliste für Komponenten • Auswahlliste "Logger" • Schaltflächen:  
Safety Online Information	alle Steuerelemente
Status	Stammt aus der IndraLogic-Basis und kann deshalb variieren. Es sind nur die im Folgenden aufgelisteten Steuerelemente zulässig: Steuerelement: Schaltfläche "Bestätigen"
Information	Stammt aus der IndraLogic-Basis und kann deshalb variieren. Enthält nur Informationen

Objekt Bibliotheksverwalter

Fenster	Steuerelement
Befehlsleiste oberhalb der Fenster	Schaltflächen: • Bibliothek hinzufügen • Bibliothek löschen • Eigenschaften Details Platzhalter • Bibliotheks-Repository
Liste der eingebundenen Bibliotheken	
Liste der Bausteine der selektierten Bibliothek	Sortier- und Suchfunktionen
Details der Bausteine mit den Registerkarten "Eingänge/ Ausgänge", "Grafisch", "Dokumentation"	

Vergleichseditor

Steuerelement: Schaltfläche "Vergleich drucken"	

Editoren von Safety Applikationsobjekt, Safety Task, Safety Globale Variablenliste, Safety POU,

Alle Steuerelemente	

Geräteeditor der logischen E/As

Registerkarte	Steuerelement
Sichere Konfiguration	alle Steuerelemente
Sichere Parametrierung	alle Steuerelemente
E/A-Abbild	alle Steuerelemente
Information	Stammt aus der IndraLogic-Basis und kann deshalb variieren. Enthält nur Informationen.

Editor der Safety Netzwerkvariablenliste (Empfänger)

Registerkarte	Steuerelement
Bereich oberhalb der Registerkarten	alle Steuerelemente
Safety Konfiguration	alle Steuerelemente
SPS Netzwerk	alle Steuerelemente

Editor der Safety Netzwerkvariablenliste (Sender)

Registerkarte	Steuerelement
Bereich oberhalb der Registerkarten	alle Steuerelemente
Safety Konfiguration	alle Steuerelemente
SPS Netzwerk	alle Steuerelemente

Ansicht 'Meldungen'

Steuerelemente	•  • Auswahlliste mit Meldungskategorien • Meldungstypen: Fehler, Warnung, Information

Ansicht 'Safety Querverweisliste'

Alle Steuerelemente	

Ansicht 'Eigenschaften'

Registerkarte	Steuerelement
Allgemein	Eingabefeld zur Änderung des Objektbezeichners
Zugriffskontrolle	alle Steuerelemente
Safety	alle Steuerelemente

Ansicht 'Überwachungsliste'

Überwachungsliste 1 Überwachungsliste 2 Überwachungsliste 3 Überwachungsliste 4 Alle Forces anzeigen	

Zulässige Werkzeuge

Zulässige FUP-Werkzeuge (FUP, Ansicht "Werkzeuge")

Generelles	Zuweisung Eingang Sprung Return
Boolesche Operatoren	AND (2 Eingänge) AND (3 Eingänge) OR (2 Eingänge) OR (3 Eingänge) XOR NOT

Mathematische Operatoren	ADD (2 Eingänge) ADD (3 Eingänge) SUB MUL DIV EQ NE LT LE GT GE
Sonstige Operatoren	SEL MUX
Safety Funktionsbausteine	alle
Safety Standard FBs	alle

Dialoge des Menüs 'Extras'

Alle Kategorien sind zulässig. In folgenden Kategorien müssen bestimmte Optionen aktiviert sein:

Kategorie "Geräteeditor"	Die Option "Klassische Darstellung der Kommunikationseinstellungen verwenden" muss aktiviert sein!

Dialog 'Anpassen'

Alle Steuerelemente	

21.3 Geänderte Standardfunktionen

Befehl 'Querverweise ausgeben'

Wenn der Befehl auf nonsafety-Objekte angewendet wird, zeigt der Befehl normales Verhalten, gemäß IndraLogic: Querverweise werden in der Ansicht „Querverweisliste“ aufgelistet. Wenn der Befehl auf Safety-Objekte angewendet wird, werden die Querverweise in der Ansicht „Safety Querverweisliste“ aufgelistet.

Ansicht 'Querverweisliste'

In dieser Ansicht können keine Querverweise in Safety-Objekten gefunden werden.

Im Gegensatz dazu können in der Ansicht "Safety Querverweisliste" nur Querverweise in Safety-Objekten gefunden werden, keine aus Standard-Objekten. Informationen zur Funktionsweise finden Sie in der IndraLogic Safety Onlinehilfe.

Befehl 'Dokumentieren'

Wenn das Projekt mindestens 1 Safety-Objekt enthält, wird eine bei Freigabe von IndraLogic Safety fixierte Version des IndraLogic Druck-Rahmenwerks verwendet. Es kann also zum Beispiel sein, dass neue Funktionen des Dokumentieren-Auswahldialogs aus der IndraLogic Basis nun nicht mehr zur Verfügung stehen. Die Funktion entspricht dann nicht mehr der Beschreibung in der Online- Hilfe IndraLogic, sondern nur noch der Beschreibung in der Onlinehilfe IndraLogic Safety.

Befehle 'Archiv speichern/ versenden'

Wenn das Projekt mindestens 1 Safety-Objekt enthält, wird eine bei Freigabe von IndraLogic Safety fixierte Version des Archivieren- Auswahldialogs ver-

wendet. Es kann also zum Beispiel sein, dass neue Funktionen des Archivieren-Auswahldialogs aus der IndraLogic Basis nun nicht mehr zur Verfügung stehen.

Die Funktion entspricht dann nicht mehr der Beschreibung in der Onlinehilfe IndraLogic, sondern nur noch der Beschreibung in der Onlinehilfe IndraLogic Safety.

Wenn bei dem Befehl **Datei ▸ Projektarchiv ▸ Archiv speichern /versenden** ein logisches Gerät der Sicherheitsapplikation ausgewählt ist, werden alle entsprechenden sicherheitsbezogenen Gerätebeschreibungen mit archiviert. Wenn bei der Safety-Variante der Bibliotheksverwalter mit ausgewählt ist, werden alle referenzierten Bibliotheken mit archiviert.

22 Fehler

22.1 Safety-Fehlermeldungen

Nachstehend sind die Meldungen des sicherheitsgerichteten Checkers der SafeLogic-Erweiterung des IndraWorks-Programmiersystems aufgelistet.

Über den Standardbefehl **Übersetzen** einer Applikation kann der Anwender im Fall einer Sicherheitsapplikation die sicherheitsspezifische Funktion für den Programmiersystem-Teil der Fehlerchecks manuell auslösen.

Analog zum Standardbefehl **Übersetzen** erscheinen eventuelle Fehlermeldungen im Standard-Meldungsfenster; von einer Fehlermeldung kann zu deren Position im Quellcode gesprungen werden.

Applikation

Fehler nummer	Gewichtung	Bedingung	Position	Text
[A014]	Fataler Fehler		Applikationsobjekt	`Wenigstens eine POU in Applikation erwartet` Erläuterung: Die Applikation enthält keine oder nur auskommentierte POUs
[A021]	Fataler Fehler		Applikationsobjekt	`Applikation enthält keine Task` Erläuterung: Die Applikation enthält keine Task.
[A048]	Warnung	Applikationseinstellung `Warnung unbenutzte Objekte` aktiviert	Objekt	`Unbenutzter <Objekttyp> in Applikation` *Erläuterung:* • **logisches Gerät:** Keine Variable eines nicht leeren Mappings wird in der Applikation verwendet. • **GVL:** Keine Variable einer nicht leeren GVL wird in der Applikation verwendet. • **Funktionsbaustein:** Der FB wird nicht instanziiert • **Programm:** Das Programm wird in der Task nicht aufgerufen oder ist dort auskommentiert
[A051]	Warnung		GVL-Objekt	`GVL enthält keine Deklarationen` Erläuterung: GVL enthält keine Deklarationen

Fehler nummer	Gewichtung	Bedingung	Position	Text
[A052]	Warnung		Logisches Geräteobjekt	`Logisches Gerät enthält keine Mappings` Erläuterung: Das logische Gerät enthält keine E/A-Mappings.
[A053]	Warnung	Basic-/Extended-POU	POU-Objekt	`Die POU enthält kein Netzwerk` Erläuterung: Eine Basic- oder Extended-Level POU enthält keine Netzwerke.
[A061]	Warnung	Applikationseinstellung `Warnung für fehlenden Objektkommentar` aktiviert	Objekt	`Das Objekt benötigt Objektversion und -kommentar` Erläuterung: Der Objektkommentar der Sicherheitsapplikation oder einer Safety-POU ist leer.
[A331]	Fehler		Objekt	`Namenskonflikt: Name von Objekt 'Name des Objekts' ist nicht eindeutig` Erläuterung: Die Applikation enthält zwei Objekte mit gleichem Namen. Namen muss über folgende Objekttypen hinweg eindeutig sein: Sicherheitsapplikation, Task, POU, GVL, Logisches Gerät.
[A331] Zusatzinfo	Information		Objekt	`... siehe Objekt mit gleichem Namen`
[A332]	Warnung	Applikationseinstellung `Warnung für signifikante Zeichen` aktiviert.	Objekt	`Unzureichende Signifikanz von Objektbezeichner '<Objekt>'` Erläuterung: Die Applikation enthält zwei Objekte mit Namen, welche sich in den ersten n signifikanten Zeichen nicht unterscheiden.
[A332] Zusatzinfo	Information		Objekt	`... die gleichen <Anzahl> Zeichen in Objektbezeichner '<Objekt>'`
[A351]	Fehler		Objekt Variablenddeklaration	`Namenskonflikt: Variable '<Variable>' hat den selben Namen wie eine Objekt` Erläuterung: Die Applikation enthält zwei Objekte mit Namen, welche sich in den ersten n signifikanten Zeichen nicht unterscheiden.
[A351] Zusatzinfo	Information		Objekt	`...siehe Objekt mit gleichem Namen`

Fehler nummer	Gewichtung	Bedingung	Position	Text
[A352]	Warnung	Applikationseinstellung `Warnung für signifikante Zeichen` aktiviert	Objekt Variablendeklaration	`Unzureichende Signifikanz von Variablenbezeichner '<Variable>'` Erläuterung: Die Applikation enthält eine Variable (VAR_GLOBAL, Mapping, VAR etc.), welches sich in den ersten signifikanten Zeichen nicht von einem Objekt der Applikation (Application, POU, Task, etc.) unterscheidet.
[A352] Zusatzinfo	Information		Objekt	`... die gleichen <Anzahl signifikanter Zeichen> Zeichen in Objektbezeichner '<Objekt>'`
[A388]	Fehler		Objekt Variablendeklaration.Name bzw. Sprungmarke	`Namenskonflikt: Das Symbol '<Symbol>' ist bereits definiert` Erläuterung: Das Symbol ist im Gültigkeitsbereich bereits definiert. Gültig für Variablen und Labels.
[A388] Zusatzinfo	Information	Konflikt von mehreren globalen Variablen oder einer globalen und einer lokalen Variablen	Objekt Variablendeklaration.Name	`... siehe Variable mit gleichem Namen`
[A388] Zusatzinfo	Information	Sprungmarke mit gleichem Namen wie eine Variable	Objekt Sprungmarke	`... siehe Sprungmarke mit gleichem Namen`
[A399]	Warnung	Applikationseinstellung `Warnung für signifikante Zeichen` aktiviert	Objekt Variablendeklaration.Name bzw. Sprungmarke	`Unzureichende Signifikanz von Symbol 'Symbol'` Erläuterung: Im Gültigkeitsbereich werden 2 Symbole definiert, welche sich in den ersten n signifikanten Zeichen nicht unterscheiden.
[A399] Zusatzinfo	Information	Konflikt von mehreren globalen Variablen oder einer globalen und einer lokalen Variable durch Signifikanz	Objekt Variablendeklaration.Name	`... die gleichen <Anzahl> Zeichen in Variable '<Symbol>'`
[A399] Zusatzinfo	Information	Sprungmarke mit gleichem Namen wie eine Variable durch Signifikanz	Objekt, Sprungmarke	`... die gleichen <Anzahl> Zeichen in Sprungmarke '<Symbol>'`
[A422]	Warnung		Taskobjekt	`Die Task enthält keine Aufrufe` Erläuterung: Die Task enthält keine Aufrufe

Fehler nummer	Gewichtung	Bedingung	Position	Text
[A426]	Warnung	Applikationseinstellung `Warnung auskommentierte Inhalte` aktiviert	Taskobjekt	`Die Task enthält <n> auskommentierte Aufrufe` Erläuterung: Die Task enthält wenigstens 1 auskommentierten Aufruf.
[A440]	Fehler		Taskobjekt, Programmeintrag	`Unbekanntes Programm '<Name des Programms>' in Task` Erläuterung: Die Task enthält einen Aufrufeintrag mit einem Programm, welches nicht Teil der Applikation ist.
[A450]	Fehler		Taskobjekt Programmeintrag	`Task kann Objekt '<Name des Objekts>' nicht aufrufen, da es kein Programm ist` Erläuterung: Die Task enthält einen Aufrufeintrag mit dem Namen einer POU, welche kein Programm ist.
[A450] Zusatzinfo	Information		2tes Objekt	`... siehe referenziertes Objekt`

Tab. 22-1: Fehler in der Applikation

Code-Format

Fehlernummer	Gewichtung	Position	Text
[C310]	Fataler Fehler		`Inkompatibles Format: Unbekannte Code-Elemente in Objekt der Version <SafetyErweiterungsversion>`
[C410]	Information	Objekt	`Vorwärtskompatibilität: Objekt aus '<SafetyVersion>' enthält Erweiterungen, welche in Zielsystem-Version <ExecVersion> ignoriert werden`
[C510]	Fataler Fehler		`Ausführungsversion <ExecVersion> des Zielsystems wird nicht unterstützt`

Tab. 22-2: Fehler im Code-Format

Fehlernummer	Gewichtung	Position	Text
[F000]	Fataler Fehler	Abhängig vom Fehler	`Fatal: Interner Fehler [<Id>] in Safety-Überprüfung <(Fun, Info)>`
[F100]	Fataler Fehler	Objekt	`Fatal: Objekt korrumpiert`
[F101]	Fataler Fehler	Objekt	`Fatal: Applikation korrumpiert`

Fehlernummer	Gewichtung	Position	Text
[F200]	Fataler Fehler	Objekt	`Fatal: Interner Fehler in Objekt <Id,Info)>`
[F300]	Fataler Fehler	Objekt	`Fatal: Fehler in Projektstruktur <(Id, Info)>`

Tab. 22-3: Fatale Fehler

Implementierung

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[I030]	Fehler		Objekt, Variablendeklaration	`PLCopen: Instanz '<Instanz>' des Einmaliger-Aufruf-Funktionsbausteins '<POU>' wird nicht aufgerufen` Erläuterung: Eine Instanz eines FBs mit Single-Call-Attribut wird in der Applikation nie aufgerufen
[I040]	Fehler		Objekt, Netzwerk, Call-Instanz	`PLCopen: Instanz '<Instanz>' des Einmaliger Aufruf-Funktionsbausteins '<POU>' wird mehrfach aufgerufen` Erläuterung: Eine Instanz eines FBs mit Single-Call-Attribut wird in der Applikation 2 mal aufgerufen
[I040] Zusatzinfo	Information		Objekt, Netzwerk, Call-Instanz	`... siehe weiterer Aufruf`
[I050]	Fehler		Objekt, Netzwerk, Call-Instanz	`PLCopen: Bedingter Aufruf von '<Instanz>' des Typs '<POU>' (nach bedingtem Sprung/Return) nicht zulässig` Erläuterung: Eine Instanz eines FBs mit Single-Call-Attribut wird nach einem bedingtem Sprung/Return aufgerufen
[I050] Zusatzinfo	Information		Objekt, Netzwerk, Sprung/Return	`... siehe Sprung/Return`
[I090]	Fehler		Objekt, Netzwerk, Call-Instanz	`E/A-Instanz '<Instanz>' darf nicht mehrfach aufgerufen werden` Erläuterung: Eine implizite Instanz eines Stack-FBs wird mehrfach aufgerufen

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[I090] Zusatzinfo	Information		Objekt, Netzwerk, Call-Instanz	`... siehe weiterer Aufruf`
[I120]	Fehler		Objekt, Netzwerk, L-Operand	`ISO 13849: Ausgang '<Mapping>' darf nicht mehrfach beschrieben werden` Erläuterung: Ein gemappter Ausgang wird an mehreren Stellen beschrieben
[I120] Zusatzinfo	Information		Objekt, Netzwerk, L-Operand	`... siehe weitere Zuweisung`
[I131]	Warnung	Applikationseinstellung `Warnung unbenutzte Variablen` aktiviert	Objekt, Variablendeklaration	`Unbenutzte Variable: Funktionsbausteininstanz '<Instanz>' wird nicht aufgerufen` Erläuterung: Eine deklarierte FB-Instanz wird in der Applikation nicht aufgerufen. Zugriffe auf die Komponenten können erfolgt sein
[I132]	Warnung	Applikationseinstellung `Warnung unbenutzte Variablen` aktiviert	Objekt, Variablendeklaration	`Unbenutzte Variable: '<Variable>' ist deklariert, wird aber nicht verwendet` Erläuterung: Eine deklarierte Variable wird weder gelesen noch beschrieben.
[I133]	Warnung	Applikationseinstellung `Warnung unbenutzte Variablen` aktiviert	Objekt, Variablendeklaration	`Überflüssige Variable: '<Variable>' wird beschrieben aber nicht gelesen'` Erläuterung: Eine deklarierte Variable wird zwar beschrieben, danach aber nicht mehr gelesen
[I134]	Warnung	Applikationseinstellung `Warnung unbenutzte Inhalte` aktiviert	Objekt, Variablendeklaration	`<Scope> '<Variable>' benötigt eine Zuweisung` Erläuterung: Eine VAR_OUTPUT Variable wird nicht beschrieben. Scope = VAR_OUTPUT.
[I139]	Warnung	Applikationseinstellung `Warnung unbenutzte Inhalte` aktiviert	Objekt, Variablendeklaration	`<Scope> '<Variable>' wird nicht modifiziert, sie sollte CONSTANT deklariert werden` Erläuterung: Eine Variable wird nicht beschrieben aber gelesen. Sie kann CONSTANT deklariert werden. Scope = VAR | VAR_GLOBAL| VAR_EXTERNAL

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[I151]	Fehler		Objekt, Netzwerk, R-Operand	`Ausgang '<Variable>' wird vor dem Aufruf der Funktionsbausteininstanz '<Instanz>' gelesen` Erläuterung: Der Ausgang einer FB-Instanz wird gelesen bevor der Aufruf der Instanz erfolgt ist.
[I152]	Fehler		Objekt, Netzwerk, Sprungmarke	`Sprungmarke zwischen Aufruf und Lesen des Ausgangs nicht zulässig` Erläuterung: Es ist eine Sprungmarke zwischen dem Aufruf eines FBs und dem Auslesen eines Ausgangs dieses FBs vorhanden.
[I152] Zusatzinfo	Information		Objekt, Netzwerk, Aufruf	`... siehe vorheriger Aufruf`
[I161]	Fehler	Basic-POU	Objekt, Netzwerk, L-Operand	`Basic Level: Zuweisung zu '<Variable>' erzeugt eine Rückkopplungsschleife zu vorherigem Lesezugriff` Erläuterung: Eine Variable wird innerhalb eines Netzwerks gelesen und beschrieben.
[I161] Zusatzinfo	Information		Objekt, Netzwerk, R-Operand	`... siehe vorheriger Lesezugriff`
[I303]	Fehler		Objekt, Operand	`Bezeichner oder Wert erwartet` Erläuterung: Ein Operator-Eingang oder eine Zuweisungsquelle ist leer.
[I362]	Fehler		Objekt, Netzwerk, L-Operand	`Sprung/Return nicht letztes Element im Netzwerk` Erläuterung: Das Netzwerk enthält weitere Sprünge/ Returns, Zuweisungen nach einem Sprung/Return.
[I391]	Warnung	Applikationseinstellung `Warnung für Anzahl Netzwerke` aktiviert.	Objekt	`Die Anzahl <Anzahl> effektiver Netzwerke übersteigt das zulässige Maximum von <Maximum>` Erläuterung: Die POU enthält mehr als die im Applikationsobjekt angegeben Anzahl nicht auskommentierter Netzwerke.

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[I392]	Warnung		Objekt, Netzwerk	`Leeres Netzwerk` Erläuterung: Ein Netzwerk enthält keine Anweisungen.
[I393]	Warnung	Applikationseinstellung `Warnung für Anzahl Aufrufe` aktiviert.	Objekt, Netzwerk,	`Netzwerk mit <Anzahl> Aufrufen übersteigt das zulässige Maximum von <Maximum>` Erläuterung: Das Netzwerk enthält mehr Aufrufe von FBs und Operatoren als in den Applikationseinstellungen angegeben.
[I396]	Warnung	Applikationseinstellung `Warnung auskommentierte Inhalte` aktiviert	Objekt, Netzwerk	`Auskommentiertes Netzwerk` Erläuterung: Ein Netzwerk in der POU ist auskommentiert.
[I397]	Warnung		Objekt, Netzwerk, Operator	`Unbenutzter Operand 'Operand'` Erläuterung: Das Ergebnis eines Operators wird nicht zugewiesen
[I398]	Warnung	Applikationseinstellung `Warnung für unbenutzte Variablen` aktiviert	Objekt, Netzwerk, Label	`Die Marke '<Element>' wird nicht referenziert` Erläuterung: Die Sprungmarke wird innerhalb der POU nicht in einem Sprung referenziert.
[I400]	Fehler		Objekt, Netzwerk, R-Operand	`Funktionsbausteininstanz '<Instanz>': '<FB>' kann nicht als Wert verwendet werden` Erläuterung: Eine FB-Instanz wird als Teil einer Zuweisung verwendet.
[I431]	Fehler		Objekt, Netzwerk, R-Operand	`'<Operator>' auf Typ '<Typ>' nicht zulässig` Erläuterung: Es wird eine Operation auf einem nicht kompatiblen Typen durchgeführt, z. B. AND auf INT.
[I432]	Fehler		Objekt, Netzwerk, Operator	`Typ '<Typ1>' kann nicht mit '<Typ2>' verglichen werden` Erläuterung: Die Operanden eines Vergleichsoperators haben inkompatible Type, zum Beispiel BOOL und INT.

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[I433]	Fehler		Objekt, Netzwerk, Operator	`Unterschiedliche Typen '<Typ>1' und '<Typ2>' in Operanden von Operator '<Operator>'` Erläuterung: Es werden Operanden unterschiedlicher Typen an einem Operator verwendet, z. B. BOOL und INT an MUX oder SEL.
[I435]	Fehler		Objekt, Netzwerk, Operator	`Multiplikation/Division mit Operand vom Typ '<Typ>' nicht zulässig.` Erläuterung: Es wird eine ungültige Multiplikation/ Division durchgeführt, z. B. TIME*TIME, INT/TIME etc.
[I440]	Fehler	Basic-POU	Objekt, Netzwerk, Operator	`Basic Level: Operator OR nur für SAFEBOOL zulässig` Erläuterung: Alle Operanden des OR müssen SAFEBOOL sein.
[I610]	Fehler		Objekt, Netzwerk, Call-Instanz	`Funktionsbausteininstanz anstelle von '<Identifier>' erwartet` Erläuterung: Es wird versucht einen FB mit einer skalaren Variable als Instanz aufzurufen.
[I640]	Fehler		Objekt, Netzwerk, Call-Instanz	`'<Instanz>' ist keine Instanz von '<Typ>'` Erläuterung: Die übergebene Instanz entspricht nicht den Typen des Aufrufs.
[I660]	Fehler		Objekt, Netzwerk, Call	`'<Input>' ist kein Eingang von '<FB>'` Erläuterung: Der formelle Eingang im Call ist in der Definition des FBs nicht vorhanden.
[I660] Zusatzinfo	Information	Der FB hat keine Variable dieses Names oder die Variable ist SYSONLY	Objekt	`... siehe referenzierter Funktionsbaustein`
[I660] Zusatzinfo	Information	<Input> ist in <FB> definiert aber nicht SYSONLY und nicht VAR_INPUT	Objekt	`...siehe <Scope> '<Input>' in POU`
[I670]	Fehler		Objekt, Netzwerk, Call	`'<Output>' ist kein Ausgang von '<FB>'`

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[I670] Zusatzinfo	Information	Der FB hat keine Variable dieses Namens oder die Variable SYSONLY	Objekt	`... siehe referenzierter Funktionsbaustein`
[I670] Zusatzinfo	Information	<Output> ist in <FB> definiert aber nicht SYSONLY und VAR_OUTPUT	Objekt	`siehe <Scope> '<Output>' in POU '<POU>'`
[I711]	Fehler		Objekt, Netzwerk, L-Operand	`'<Variable>' ist kein gültiges Zuweisungsziel` Erläuterung: Das Ziel einer Zuweisung ist eine Konstante, lokaler VAR_INPUT, VAR_OUTPUT einer anderen Instanz oder ein Eingangs-Mapping.
[I731]	Fehler		Objekt, Netzwerk, L-Operand	`Eingang '<Instanz.Eingang>' ist außerhalb des Aufrufs kein gültiges Zuweisungsziel` Erläuterung: Auf den Eingang einer FB-Instanz wird außerhalb des Instanzaufrufs geschrieben.
[I750]	Fehler		Objekt, Netzwerk, Sprung	`Marke 'Label' innerhalb des Bereichs der 'JMP'-Anweisung nicht definiert` Erläuterung: Das für den Sprung angegeben Label existiert in der POU nicht.
[I750] Zusatzinfo	Information	Sprungmarke ist definiert, aber das Netzwerk auskommentiert	Objekt, Netzwerk, Sprungmarke	`... Definition von '<Label>' ist auskommentiert`
[I760]	Fehler		Objekt, Netzwerk, Sprung	`PLCopen: Rückwärtssprung zur Sprungmarke '<Label>' ist nicht zulässig` Erläuterung: Die Definition der Sprungmarke liegt vor dem Sprung.
[I760] Zusatzinfo	Information		Label: Netzwerk	`... siehe Definition von '<Label>'` Erläuterung: Die Definition der Sprungmarke liegt vor dem Sprung.

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[I770]	Fehler	Basic-POU	Objekt, Netzwerk, Sprung/Return	`Basic Level: Bedingter Sprung/ Return nicht zulässig` Erläuterung: Es wird ein Sprung/Return verwendet. Möglich durch Copy/Paste aus Extended-Level.
[I801]	Fehler		Objekt, Netzwerk, L-Operand	`Kann Typ '<Typ1>' nicht in Typ '<Typ2>' konvertieren` Erläuterung: Der Quelltyp der Zuweisung ist inkompatibel mit dem Zuweisungsziel, z. B. BOOL -> INT oder BOOL -> SAFEBOOL
[I802]	Fehler		Objekt, Netzwerk, R-Operand	`Bedingter Sprung/Return abhängig von Typ '<Typ>' nicht möglich` Erläuterung: Die Bedingung eines Sprung/Return ist nicht boolesch.
[I810]	Fehler		Objekt, Netzwerk, Operand	`PLCopen: Zugriff auf Variable '<Variable>' vom Typ '<Typ>' ist nicht zulässig` Erläuterung: Auf den Eingang oder Ausgang eines System-FBs vom Typ SAFEDWORD/ DWORD wird zugegriffen. Diese dürfen nur direkt mit anderen FB- Ausgängen/ Eingängen verknüpft werden.
[I821]	Fehler	Basic-POU	Objekt, Netzwerk, Operand	`Basic Level: Ausgang '<Ausgang>' vom Typ '<Typ>' darf nicht gelesen werden` Erläuterung: Ausgänge vom Typ TIME/SAFETIME dürfen nur direkt an andere FBs weitergereicht werden.

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[I824]	Fehler		Objekt, Netzwerk, L-Operand	`PLCopen: Eingang '<Eingang>' vom Typ '<Typ>' darf nicht geschrieben werden` Erläuterung: Eingänge anderer FBs vom Typ DWORD/SAFEDWORD dürfen nicht beschrieben werden.
[I825]	Fehler	Basic-POU	Objekt, Netzwerk, L-Operand	`Basic Level: Eingang '<Eingang>' vom Typ '<Typ>' muss mit konstantem Wert belegt werden` Erläuterung: Eingänge anderer FBs vom Typ TIME/ SAFETIME dürfen nur mit Konstanten beschrieben werden.

Tab. 22-4: Fehler in Implementierung

Lexikalische Analyse

Fehler nummer	Gewichtung	Bedingung	Position	Text Beispiel
[L011]	Fehler		Objekt, Zelle	`Bezeichner anstelle von '<Token>' erwartet` Erläuterung: Eine Zelle enthält keinen gültigen Identifier, z. B. '???' als Operand, Call-Instanz, Call-Typ oder Variablenname in Deklaration.
[L014]	Fehler		Objekt, Zelle	`Typdefinition anstelle von '<Token>' erwartet` Erläuterung: Die Typdefinition einer Variablendeklaration enthält keinen Identifier oder Basistyp, z. B. '???' oder 'a.b'
[L020]	Fehler	Basic-/Extended-POU	Objekt, Variablendeklaration, Typ	`Name '<Symbol>' reserviert für System Level` Erläuterung: Es wird ein Typ mit führendem Unterstrich verwendet.
[L031]	Fehler		Objekt, Zelle	`Reserviertes Symbol '<Symbol>' als Bezeichner verwendet` Erläuterung: Ein reserviertes Symbol wird als Identifier verwendet, z. B. Operatoren, Standard-FB-Namen etc.

Fehler nummer	Gewichtung	Bedingung	Position	Text Beispiel
[L032]	Fehler		Objekt	`Reserviertes Symbol '<Symbol>' als Objektnamen verwendet` Erläuterung: Ein reserviertes Symbol wird als Objektname verwendet, z. B. Operatoren, Standard-FB-Namen etc.
[L200]	Fehler	Basic-POU	Objekt, Variablendeklaration, Typ	`Basic Level: Funktionsbausteintyp '<Typ>' darf nur in Extended Level verwendet werden` Erläuterung: Es wird eine Variable von einem in Basic Level nicht zulässigen FB (SF_SR, SF_RS, etc.) deklariert.
[L222]	Fehler		Objekt, Variablendeklaration, Typ	`Unbekannter Funktionsbaustein-typ '<Typ>'` Erläuterung: Es wird ein Identifier als Typ verwendet, der in der Applikation als FB nicht vorhanden ist.
[L223]	Fehler		Objekt, Variablendeklaration, Typ	`Vorwärtsreferenz auf Funktionsbausteintyp '<Typ>' (Datenrekursion?)` Erläuterung: Es wird ein FB referenziert der den eigenen FB referenziert.
[L223] Zusatzinfo	Information		Objekt, Variablendeklaration, Typ	`... siehe referenzierter Funktionsbaustein`
[L231]	Fehler		Objekt, Variablendeklaration, Typ	`<Objekttyp> '<Objekt>' als Funktionsbausteintyp verwendet` Erläuterung:
[L231] Zusatzinfo	Information		Objekt	`... siehe referenziertes Objekt` Erläuterung: Es wird ein Objekt, welches kein Funktionsbaustein ist als FB verwendet.
[L310]	Fehler	Basic-/Extended-POU	Objekt, Netzwerk, R-Operand	`PLCopen: Konstante <Wert> vom Typ '<Typ>' nicht zulässig` Erläuterung:
[L340]	Fehler		Objekt, Zelle	`Konstante '<Wert>' zu groß für Typ '<Typ>'` Erläuterung: Es wird eine Konstante vom Typ (SAFE)BYTE/DWORD verwendet

Fehler nummer	Gewichtung	Bedingung	Position	Text Beispiel
[L400]	Fehler		Objekt, Netzwerk, Call	'<Operator>' nicht zulässig Erläuterung: Es wird ein nicht zulässiger Operator (ABS, LOG, etc.) verwendet.
[L411]	Fehler	Basic-POU	Objekt, Netzwerk, Call	Basic Level: Operator '<Operator>' nicht zulässig (Nur AND und OR) Erläuterung: Die Operatoren NOT, XOR, arithmetische Operatoren, Vergleichsoperatoren, MUX, SEL und Konvertierungen sind in Basic-Level nicht erlaubt.
[L421]	Fehler	Extended-POU	Objekt, Netzwerk, Call	'<Operator>' nur in Extended Level Programmen zulässig (für Konvertierung von E/A)" Erläuterung: Es wird eine Konvertierung zwischen SAFE-Typen durchgeführt. In Funktionsbausteinen darf weder Ein- noch Ausgangstyp (SAFE)BYTE/DWORD sein.
[L511]	Fehler		Objekt, Netzwerk, Operand	Bezeichner '<Identifier>' nicht definiert Erläuterung: Für den angegeben Identifier existiert keine Definition, z. B. keine Variable definiert oder Komponente in Instanz nicht definiert.
[L511] Zusatzinfo	Information	<Identifier> ist deklariert, aber auskommentiert	Objekt, Deklaration	... Definition von '<Identifier>' ist auskommentiert
[L512]	Fehler		Objekt, Netzwerk, Operand	Objekt '<Objekt>' wird wie eine Variable verwendet Erläuterung:
[L512] Zusatzinfo	Information		Objekt	... siehe Definition von '<Objekt>' Erläuterung: Der Name eines Objekts wird als Operand verwendet. Objekt = GVL, POU, Mapping.
[L520]	Fehler		Objekt, Netzwerk, Operand	Zugriff auf globale Variable '<Variable>' erfordert eine VAR_EXTERNAL Deklaration Erläuterung: Es wird eine globale Variable ohne lokale VAR_EXTERNAL Deklaration als Operand verwendet.

Fehler nummer	Gewichtung	Bedingung	Position	Text Beispiel
[L520] Zusatzinfo	Information		Objekt, Variablendeklaration	`... siehe Deklaration von '<Variable>'`
[L531]	Fehler		Objekt, Netzwerk, Operand	`Komponentenzugriff '<Komponente>': '<Instanz>' ist keine strukturierte Variable` Erläuterung: Es wird ein Komponentenzugriff auf einer Variable mit Basistyp durchgeführt.
[L541]	Fehler		Objekt, Netzwerk, L-Operand	`Zugriff auf VAR_INPUT '<Parameter>' von 'FB' außerhalb eines Aufrufs ist nicht zulässig` Erläuterung: Ein VAR_INPUT einer Instanz wird außerhalb des Calls beschrieben.

Tab. 22-5: Lexikalische Fehler

Variablen

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[V014]	Warnung	Applikationseinstellung `Warnung für Anzahl Deklarationen` aktiviert. Anzahl überschritten.	Objekt	`Die Anzahl von <Anzahl> Deklarationen übersteigt das zulässige Maximum von <Maximum>` Erläuterung: Eine GVL oder POU enthält Deklarationen als die im Applikationsobjekt angegebene maximale Anzahl.
[V016]	Warnung	Applikationseinstellung `Warnung auskommentierte Inhalte` aktiviert	Objekt	`Objekt enthält <Anzahl> auskommentierte Deklarationen` Erläuterung: Eine GVL, POU oder Mapping enthält auskommentierte Deklarationen.
[V101]	Fehler		Objekt, Variablendeklarationen	`Zugriff auf VAR_IN_OUT <Var> verboten` Erläuterung: In einem Aufruf wird eine VAR_IN_OUT Variable beschrieben. Kann durch Umwandlung des Scope im Lib-FB von VAR_INPUT nach VAR_IN_OUT entstehen.

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[V102]	Fehler		Objekt, Variablendeklarationen	'<Scope>' in diesem Kontext nicht zulässig Erläuterung: Es wird eine Variable mit einem Scope deklariert, welcher im Objekttyp nicht zulässig ist, z. B. VAR in GVL, VAR_INPUT in PRG etc. Kann durch Copy/ Paste von Deklarationen zwischen Objekten vorkommen.
[V140]	Fehler		Objekt, Variablendeklarationen	Keine globale Definition für VAR_EXTERNAL '<Variable>' gefunden Erläuterung: In der Applikation existiert keine VAR_GLOBAL Deklaration für eine VAR_EXTERNAL Variable.
[V140] Zusatzinfo	Information	Globale Deklaration von <Variable> existiert, ist aber auskommentiert	Objekt, Variablendeklaration	... Definition von '<Variable>' ist auskommentiert Erläuterung:
[V151]	Fehler		Objekt, Variablendeklaration	Falscher Modifizierer in Deklaration <Scope> '<Variable>' Erläuterung: Die VAR_GLOBAL Deklaration für eine VAR_EXTERNAL Variable hat unterschiedliche Modifizierer, z. B. VAR_EXTERNAL gegen VAR_GLOBAL CONSTANT. VAR_EXTERNAL CONSTANT gegen VAR_GLOBAL in einer GVL ist dagegen erlaubt!
[V151] Zusatzinfo	Information		Objekt, Variablendeklaration	... siehe referenzierte <Scope> '<Globale Variable>' Deklaration
[V152]	Fehler		Objekt, Variablendeklaration	Modifizierer für externe Deklarationen von E/A-Variablen nicht zulässig Erläuterung: VAR_EXTERNAL Deklarationen von E/A-Variablen dürfen keinen CONSTANT Modifizierer haben.
[V152] Zusatzinfo	Information	VAR_GLOBAL für Variable definiert	Objekt, Variablendeklaration	...siehe referenzierte <Scope> '<Globale Variable>' Deklaration

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[V160]	Fehler		Objekt, Variablendeklaration	`Falscher Typ '<Typ>' für VAR_EXTERNAL '<Variable>'` Erläuterung: Der Datentyp der VAR_EXTERNAL Deklaration stimmt nicht mit dem Datentyp der entsprechenden VAR_GLOBAL Deklarationen überein.
[V160] Zusatzinfo	Information		Objekt, Variablendeklaration	`...siehe globale Deklaration '<Globale Variable>:<Typ>'`
[V180]	Fehler	Basic-POU	Objekt, Variablendeklaration	`Basic Level: Referenz auf normale globale Variable nicht zulässig` Erläuterung: Es wird eine normale VAR_GLOBAL referenziert. In Basic-Level sind nur E/A- und Stack-Variablen als VAR_EXTERNAL erlaubt.
[V180] Zusatzinfo	Information		Objekt, Variablendeklaration	`.. siehe referenzierte <Scope> '<Globale Variable>' Deklaration.`
[V410]	Fehler		Objekt, Variablendeklaration	`Datenrekursion: <Rekursionspfad> mit einem <Rekursionspfad> der Form 'fb1' -> 'fb2'->...->'fbn'` Erläuterung: Ein FB deklariert eine Variable vom eigenen Typen. Auch über mehrere Ebenen hinweg.
[V420]	Fehler		Objekt, Variablendeklaration	`<Scope> Funktionsbausteininstanz '<Instanz>' nicht zulässig` Erläuterung: Funktionsbausteininstanzen dürfen nicht VAR_INPUT, VAR_OUTPUT oder CONSTANT sein.
[V440]	Fehler		Objekt, Variablendeklaration	`Typ '<FB>' nur zulässig in VAR_EXTERNAL Deklarationen` Erläuterung: Ein FB mit Flag IOAPI-only wird als VAR deklariert.
[V440] Zusatzinfo	Information		Objekt	`siehe referenzierter Funktionsbaustein`

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[V451]	Fehler		Objekt, Variablendeklaration	PLCopen: Instanz von Single-Call Funktionsbaustein '<FB>' nur als VAR zulässig Erläuterung: „PLCopen: Instanz von Single-Call-Funktionsbaustein 'FB' nur als VAR zulässig“
[V451] Zusatzinfo	Information		Objekt	... siehe referenzierter Funktionsbaustein
[V452]	Fehler		Objekt, Variablendeklaration	PLCopen: Instanz von Single-Call Funktionsbaustein '<FB>' nur in PROGRAM oder Single-Call FUNKTIONSBAUSTEIN Erläuterung: Ein Einmaliger-Aufruf FB darf nicht in einem nicht Einmaliger-Aufruf FB instanziert werden.
[V452] Zusatzinfo	Information		Objekt	siehe referenzierter Funktionsbaustein
[V520]	Fehler	Basic-POU	Objekt, Variablendeklaration	Basic Level: Verwendung von globaler Integer-Variable '<Variable>' nur als CONSTANT zulässig Erläuterung: Globale INT/DINT-Variablen dürfen nur als Konstante in Basic-Level verwendet werden.
[V530]	Fehler	Basic-POU	Objekt, Variablendeklaration	Basic Level: WORD nur als Diagnoseausgang zulässig ('<Variable>' muss konstant oder ein Ausgang sein) Erläuterung: Variablen mit Typ WORD müssen CONSTANT oder VAR_OUTPUT sein.
[V540]	Fehler	Basic-POU	Objekt, Variablendeklaration	Basic Level: TIME nur als konstanter Funktionsbausteineingang zulässig ('<Variable>' muss konstant oder ein Eingang sein) Erläuterung: Variablen mit Typ TIME müssen CONSTANT oder VAR_INPUT sein.

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[V552]	Fehler	Extended-POU	Objekt, Variablendeklaration	`Extended Level: WORD nur intern, als Diagnoseausgang oder E/A zulässig (Variable '<Variable>')` Erläuterung: Variablen mit Typ WORD müssen VAR, CONSTANT, VAR_OUTPUT oder VAR_EXTERNAL auf E/A sein.
[V561]	Fehler	Extended-POU	Objekt, Variablendeklaration	`Extended Level: TIME nur intern oder als konstanter Funktionsbausteineingang zulässig ('<Variable>' muss intern, konstant oder ein Eingang sein)` Erläuterung: Variablen mit Typ TIME müssen VAR, CONSTANT oder VAR_INPUT sein.
[V571]	Fehler	Basic-POU	Objekt, Variablendeklaration, Typ	`Basic Level: Variable '<Variable>' von Typ '<Typ>' nicht erlaubt.` Erläuterung: Es wird eine Variable vom Typ DWORD oder BYTE deklariert.
[V572]	Fehler	Extended-POU	Objekt, Variablendeklaration, Typ	`Extended Level: Variable <var> vom Typ <T> nur für E/A-Zugriff zulässig` Erläuterung: Es wird eine Variable vom Typ (SAFE)BYTE/DWORD explizit deklariert.
[V611]	Fehler		Objekt, Variablendeklaration	`Kein Initialwert für Variable '<Variable>' vorhanden` Erläuterung: VAR_INPUT, VAR_OUTPUT, VAR, VAR_GLOBAL von Basistypen benötigen einen Initialwert (auch CONSTANT).

Fehler nummer	Gewichtung	Bedingung	Position	Text Anmerkung/Beispiel
[V621]	Fehler		Objekt, Variablendeklaration	`Für VAR_EXTERNAL '<Variable>' ist kein Initialwert erlaubt` Erläuterung: VAR_EXTERNAL-Deklarationen für Basistypen dürfen keinen lokalen Initialwert festlegen.
[V622]	Fehler		Objekt, Variablendeklaration	`Initialwert für VAR_IN_OUT '<Variable>' nicht zulässig` Erläuterung: VAR_IN_OUT-Deklarationen dürfen keinen Initialwert festlegen.

Tab. 22-6: Fehler: Variablen

22.2 Safety-Diagnosen, CSS

22.2.1 SF_2730 Time Coordination Message Check

Time Coordination Message Check: not received in allotted time

Ursache: Die Time Coordination Message wurde nicht oder nicht im erwarteten Zeitfenster empfangen.

Lösung: Den Parameter Timeout Mutiplier (TiMu) auf einen Wert größer 2 (Standard) setzen. Maximal ist der Wert 4 möglich.

22.2.2 SF_2832 Safety Message Check

Safety Message Check: Consumer Activity Monitor timeout

Ursache: Die Safety Message für die Verbindung wurde nicht im erwarteten Zeitfenster empfangen.

Lösung:

22.2.3 SF_283A Safety Message Check

Safety Message Check: Initialization not completed within time limit

Ursache: Die Initialisierung der Safety Verbindung wurde nicht im erwarteten Zeitfenster abgeschlossen.

Lösung:

22.2.4 SF_3118 Unsuccessful Forward_Open response

Target returned unsuccessful Forward_Open response

Ursache: Die Parameter für die sichere Konfiguration der Verbindung stimmen nicht mit den Werten im Consumer überein. Der fehlerhafte Parameter wird in einem weiteren Parameter übergeben.

0x0100 - Connection in use or duplicate Forward Open

0x0110 - Target for connection not configured

0x0111 - RPI not supported

0x0113 - Out of connections

0x0114 - Vendor ID or product code mismatch

0x0115 - Product type mismatch

0x0116 - Revision mismatch
0x0117 - Invalid produced or consumed application path
0x0118 - Invalid or inconsistent configuration application path
0x080e - TUNID Mismatch

Lösung: Korrigieren Sie den entsprechenden Parameter der sicheren Konfiguration.

22.2.5 Weitere CSS-Fehlercodes

Fehlercode	Beschreibung
16#210A	Sending termination message to CSAL was not possible (HALCS_TxDataPut() returned error)
16#2302	IXSCE_CmdProcess() has received an unknown Command from HALC (0)
16#2303	Error Sending a Foward_Open Success Response (0)
16#2304	Error Sending a Forward_Close Success Response (0)
16#2305	Error Sending a Foward_Open Unsuccessful Response (0)
16#2306	Error Sending a Forward_Close Unsuccessful Response (0)
16#2380	SafetyClose referenced a connection that could not be found
16#2383	SafetyOpen message was shorter than minimum length (0 bytes)
16#2384	SafetyOpen message's application path length (0 bytes) longer than remaining length (0)
16#2385	SafetyOpen message's application path length (0 bytes) shorter than remaining length (0)
16#2387	SafetyOpen received with an invalid Application Path
16#2388	SafetyClose message was shorter than minimum length (0 bytes)
16#238D	SafetyOpen message's application path addresses invalid class
16#238E	SafetyOpen message's application path addresses a 32-bit instance
16#2391	SafetyOpen message's Safety Segment size does not match Safety Segment type
16#2392	SafetyOpen checks: CPCRC mismatch (expected 0x00000000)
16#2393	SafetyOpen checks: received SafetyOpen but our TUNID is not yet set
16#2394	SafetyOpen checks: TUNID mismatch
16#2395	SafetyOpen checks: Duplicate Forward_Open received
16#2396	SafetyOpen checks: Electronic Key mismatch: Product Code or Vendor ID
16#2397	SafetyOpen checks: Electronic Key mismatch: Device Type
16#2398	SafetyOpen checks: Electronic Key mismatch: Revision
16#2399	SafetyOpen checks: payload data length (0 bytes) less than Multicast TimeCorrection message length
16#239A	SafetyOpen checks: payload data length mismatch
16#239B	SafetyOpen checks: Network Connection Parameters mismatch (Server)
16#239C	SafetyOpen checks: Network Connection Parameters mismatch (single-cast Client)
16#239D	SafetyOpen checks: Network Connection Parameters mismatch (multi-cast Client)
16#239E	SafetyOpen checks: TimeoutMultiplier invalid (0)
16#239F	SafetyOpen checks: TransportClass/Trigger invalid (16#00), Server not supported
16#23A0	SafetyOpen checks: TransportClass/Trigger invalid (16#00), Client not supported

Fehlercode	Beschreibung
16#23A1	SafetyOpen checks: TransportClass/Trigger invalid (16#00)
16#23A2	SafetyOpen checks: TimeCorrection EPImust be zero for Single
16#23A3	SafetyOpen checks: TimeCorrection Network Connection Parameters must be zero (Server)
16#23A4	SafetyOpen checks: Ping Interval EPI mutliplier (0) not within valid range
16#23A5	SafetyOpen checks: Ping_Count_Interval greater than 100 seconds
16#23A6	SafetyOpen checks: Time Coordination Message Min Multiplier (0) greater than 7813
16#23A7	SafetyOpen checks: Network Time Expectation Multiplier (0) greater than 45313
16#23A8	SafetyOpen checks: Safety Timeout Multiplier (0) invalid
16#23A9	SafetyOpen checks: Max Consumer Number invalid
16#23AA	SafetyOpen checks: Time Correction Connection (0) ID invalid
16#23AB	SafetyOpen checks: Max Fault Number greater (0) than 255
16#23AC	SafetyOpen checks: Initial TimeStamp or Initial Rollover Value is invalid
16#23AD	SafetyOpen checks: Application Path: Configuration Class ID (0) is invalid
16#23AE	SafetyOpen checks: Application Path: Configuration Assembly Instance ID (0) may not be NULL in Type1 SafetyOpen
16#23AF	SafetyOpen checks: Application Path: Configuration Assembly Instance (0) does not exist
16#23B0	SafetyOpen checks: Application Path: For a Safety Validator Server the Produced Connection Point (0) must be NULL
16#23B1	SafetyOpen checks: Application Path: Consumed Connection Point (0) of a Safety Validator Server either not existing or is not an output
16#23B2	SafetyOpen checks: Application Path: For a Safety Validator Client the Consumed Connection Point (0) must be NULL
16#23B3	SafetyOpen checks: Application Path: Produced Connection Point (0) of a Safety Validator Client either not existing or is not an input
16#23B4	SafetyOpen checks: request was denied by the Safety Application
16#23B5	SafetyOpen checks: configuration data of Type1 SafetyOpen not applied by Safety Application
16#23B6	SafetyOpen checks: error in store operation of SCID
16#23B7	SafetyOpen checks: error in store operation of OCPUNID
16#23B8	SafetyOpen checks: Safety Supervisor Object state machine returned an error while Type1 SafetyOpen
16#23B9	SafetyOpen checks: Type1 SafetyOpen received while configuration is locked
16#23BA	SafetyOpen checks: Type2 SafetyOpen received but device is unconfigured
16#23BB	SafetyOpen checks: Type2 SafetyOpen received with non-matching SCID
16#23BC	SafetyOpen checks: Output Connection Point (0) not found
16#23BD	SafetyOpen checks: Output Connection point owned by somebody else
16#23BE	SafetyOpen checks: SCCRC mismatch in Type1 SafetyOpen
16#23BF	SafetyOpen checks: Type1 SafetyOpen received but configuration is owned by tool
16#23C0	SafetyOpen checks: Type1 SafetyOpen received but configuration is owned by somebody else

Fehlercode	Beschreibung
16#23C5	SafetyOpen checks: Safety Supervisor Object state machine returned an error while applying Type1 SafetyOpen data
16#23C6	SafetyOpen checks: TimeCorrection Network Connection Parameters must be zero (Single-cast Client)
16#23C7	SafetyOpen checks: TimeCorrection Network Connection Parameters invalid (Multi-cast Client)
16#23C8	SafetyOpen checks: error in store operation of CFUNID
16#23C9	SafetyOpen checks: message was longer than maximum length (0 bytes)
16#23CA	SafetyOpen checks: received request is for Base Format, but this is not supported
16#23CB	SafetyOpen checks: received request is for Extended Format, but this is not supported
16#23D3	SafetyOpen checks: Initial TimeStamp or Initial Rollover Value is invalid (Target Singlecast Consumer)
16#2402	IXSMR_CmdProcess() has received an unknown Command from HALC (0)
16#2403	Error sending Explicit Response message to CSAL for serice 16#00
16#2480	the received message is too short for being a valid explicit request (0 bytes)
16#2481	the request path length contained in the message (0) is more than remaining message length (0 bytes)
16#2482	request path in explicit message did not contain a Class ID
16#2483	request path in explicit message did not contain an Instance ID
16#2484	request path in explicit message contained a 32-bit Instance ID (not supported by CSS)
16#2485	request path in explicit message addresses an unkown Class
16#251C	error in communication via HALC
16#251D	error in communication via HALC
16#2538	Forward_Open request received but not allowed in Wait for TUNID state
16#2539	Forward_Open request received but not allowed in Self Testing state
16#253A	Forward_Open request received but not allowed in Self Test Exception state
16#253B	Forward_Open request received but not allowed in Abort state
16#253C	Forward_Open request received but not allowed in Critcal Fault state
16#253D	Propose_TUNID service requested in an invalid state: 0
16#253E	Apply_TUNID service requested in an invalid state: 0
16#2546	Safety Supervisor Object state machine detected an illegal operation (connection established in state 0)
16#2547	Safety Supervisor Object state machine detected an illegal operation (connection delete in state 0)
16#2607	unit received an unknown command (0)
16#2612	Safety Validator Instance ID is invalid (0)
16#2618	The passed 'instance' parameter is invalid (0)
16#2619	The passed 'instance' parameter is invalid (0)
16#261A	The passed 'instance' parameter is invalid (0)
16#2622	Allocation of an Originator Safety Validator Server struct failed (0)
16#2624	Allocation of an Originator Safety Validator Client struct failed (0)
16#2626	Allocation of a Target Safety Validator Server Instance ID failed (0)

Fehlercode	Beschreibung
16#2627	Allocation of a Target Safety Validator Client Instance ID failed (0)
16#264F	Received Safety I/O message is shorter than minimum Safety I/O message (inst=0 len=0)
16#2650	Received Cricital Link Failure message with invalid length (len=0)
16#2709	data received for Validator Client Inst (0) had invalid length
16#2710	Sending a message to CSAL was not possible (HALCS_TxDataPut() returned error) inst=0 error=0
16#2712	Sending a message to CSAL was not possible (HALCS_TxDataPut() returned error)(0)
16#2715	Sending a message to CSAL was not possible (HALCS_TxDataPut() returned error) (0)
16#271F	The passed 'Consumer Number' parameter is invalid (0)
16#2730	Time Coordination Message Check: not received in allotted time (Single-cast) (inst=0)
16#2731	Time Coordination Message Check: not received in allotted time (Multi-cast) (inst=0 cons=0)
16#2732	Time Coordination Message Check: message with same time stamp already received from this consumer (inst=0 cons=0)
16#2733	Time Coordination Message Check: parity check error (inst=0 cons=0)
16#2734	Time Coordination Message Check: Ack_Byte_2 check error (inst=0 cons=0)
16#2735	Time Coordination Message Check: not received within within the approximatley 5 second limit (inst=0 cons=0)
16#2736	Time Coordination Message Check: not received within with the same ping interval or the next ping interval (inst=0 cons=0)
16#2737	Time Coordination Message Check: CRC mismatch (inst=0 cons=0)
16#2738	Time Coordination Message Check: CRC mismatch (inst=0 cons=0)
16#2808	Data received for Validator Server Inst (0) had invalid length
16#2809	Sending a message to CSAL was not possible (HALCS_TxDataPut() returned error) inst=0 error=0
16#2811	Sending a message to CSAL was not possible (HALCS_TxDataPut() returned error)(0)
16#2812	Sending a message to CSAL was not possible (HALCS_TxDataPut() returned error) (0)
16#2820	Safety Message Check: Timestamp CRC mismatch (inst=0 expected=16#00)
16#2821	Safety Message Check: Timestamp Delta negative (inst=0 delta=0)
16#2822	Safety Message Check: Timestamp Delta greater than Network Time Expectation (inst=0 delta=0)
16#2823	Safety Message Check: Data age of a faulty message greater than Network Time Expectation (inst=0 data age=0)
16#2824	Safety Message Check: Data age of an in other respects valid message greater than Network Time Expectation (inst=0 data age=0)
16#2825	Safety Message Check: Actual Data CRC mismatch (inst=0 expected=0)
16#2826	Safety Message Check: Complemented Data CRC mismatch (inst=0 expected=0)
16#2827	Safety Message Check: Redundant bits in Mode Byte mismatch (inst=0)
16#2828	Safety Message Check: Actual Data CRC mismatch (inst=0 expected=0)
16#2829	Safety Message Check: Complemented Data CRC mismatch (inst=0 expected=0)
16#282A	Safety Message Check: Redundant bits in Mode Byte mismatch (inst=0)
16#282B	Safety Message Check: Actual versus complemented data check mismatch (inst=0)

Fehlercode	Beschreibung
16#282C	Safety Message Check: CRC mismatch (inst=0 expected=0)
16#282D	Safety Message Check: Redundant bits in Mode Byte mismatch (inst=0)
16#282E	Safety Message Check: Actual Data CRC mismatch (inst=0 expected=0)
16#282F	Safety Message Check: Complemented Data CRC mismatch (inst=0 expected=0)
16#2830	Safety Message Check: Redundant bits in Mode Byte mismatch (inst=0)
16#2831	Safety Message Check: Actual versus complemented data check mismatch (inst=0)
16#2832	Safety Message Check: Consumer Activity Monitor timeout (inst=0 expected message at=0)
16#2833	Safety Message Check: Time Correction message not received within expected time (inst=0)
16#2834	Safety Message Check: Data cross check error (inst=0)
16#2835	Safety Message Check: Time Correction message parity check error (inst=0)
16#2836	Safety Message Check: Time Correction message Mcast_Byte_2 check error (inst=0)
16#2837	Safety Message Check: Time Correction message Multi_Cast_Active_Idle transitioned to idle (inst=0)
16#2838	Safety Message Check: Time Correction message CRC mismatch (inst=0 expected=0)
16#2839	Safety Message Check: Time Correction message CRC mismatch (inst=0 expected=0)
16#283A	Safety Message Check: Initialization not completed within time limit (inst=0 expected completion at=0)
16#2E80	Unknown Segment Type found (16#00)
16#2E81	Unknown electronic key format (16#00)
16#2E82	Unknown 8-bit logical segment (16#x00)
16#2E83	Unknown 16- or 32-bit logical segment type (16#x00)
16#2E84	Unknown data segment type (16#00)
16#2E85	Unknown network segment type (16#00)
16#2E86	Uncomplete segment detected
16#2E87	Too many Attribute IDs found in EPathParseSegLogicalVal8()
16#2E88	Too many Attribute IDs found in EPathParseSegLogicalVal16_32()
16#2E89	More than one electronic key found
16#2E8A	More than one data segment found
16#2E8B	More than one safety network segment found
16#2E8C	Too many Class IDs found in EPathParseSegLogicalVal8()
16#2E8D	Too many Class IDs found in EPathParseSegLogicalVal16_32()
16#2E8E	Too many Instance IDs found in EPathParseSegLogicalVal8()
16#2E8F	Too many Instance IDs (16 bits) found in EPathParseSegLogicalVal16_32()
16#2E90	Too many Instance IDs (32 bits) found in EPathParseSegLogicalVal16_32()
16#2E91	Too many Connection Points found in EPathParseSegLogicalVal8()
16#2E92	Too many Connection Points (16 bits) found in EPathParseSegLogicalVal16_32()
16#2E93	Too many Connection Points (32 bits) found in EPathParseSegLogicalVal16_32()
16#3102	IXCCO_CmdProcess() has received an unknown Command from HALC (0)

Fehlercode	Beschreibung
16#3103	IXCCO_Create() cannot find more free instances (0)
16#3105	HALCS_TxDataPut() returned an error when CCO tried to send (0)
16#310B	received an invalid instance from CSAL in Forward_Open response (0)
16#310C	Conflict in the message format between Target and Originator (inst=0), detected on sending Forward_Open
16#3113	SafetyClose Error Response command had invalid parameters (0)
16#3114	SafetyOpen Response command had invalid length (inst=0, length=0)
16#3115	SafetyOpen Response command had invalid length (inst=0, length=0)
16#3116	SafetyOpen Response command had invalid length (inst=0)
16#3117	SafetyClose Error Response command had invalid length (inst=0)
16#3118	Target returned unsuccessful Forward_Open response (inst=0 error=0)
16#3119	Target returned an invalid Originator Connection triad in Forward_Open response (inst=0)
16#311A	Target returned an invalid Consumer Number for Single-cast in Forward_Open response (inst=0 Cons-Num=0)
16#311B	Target returned an invalid Consumer Number for Multi-cast in Forward_Open response (inst=0 ConsNum=0)
16#311C	Target returned an unexpected Vendor ID in Forward_Open response (inst=0 VendorID=0)
16#313B	received an invalid instance from CSAL in Forward_Close response (0)
16#313E	Base/Extended Format conflict detected (during reception of Forward_Open response (int=0)
16#3146	Detected an invalid I/O message length while evaluating the Forward_Open response
16#3148	SafetyOpen Response contained an invalid reply service code (inst=0, srvc=0)
16#3149	SafetyOpen Response contained an invalid value in the reserved byte (inst=0, srvc=0)
16#314A	SafetyClose Response contained an invalid reply service code (inst=0, srvc=0)
16#314B	SafetyClose Response contained an invalid value in the reserved byte (inst=0, srvc=0)
16#314C	SafetyClose Response indicated an error (inst=0, 16#00)
16#314D	SafetyOpen Response contained an invalid value in the reserved byte (srvc=0)
16#314E	SafetyOpen Response contained an Application Reply Size which is not consistent with the message length (inst=0, bytes remaining for application reply=0)
16#314F	Target returned reserved Forward_Open General Status Code D0 (inst=0 extStatus=0)
16#3150	Validate service parameter range check: CCO inst 0 attribute 1
16#3151	Validate service parameter range check: CCO inst 0 attribute 5: Timeout Multiplier
16#3152	Validate service parameter range check: CCO inst 0 attribute 5: Transport Class Trigger
16#3153	Validate service parameter range check: CCO inst 0 attribute 5: O->T RPI
16#3154	Validate service parameter range check: CCO inst 0 attribute 5: O->T RPI
16#3155	Validate service parameter range check: CCO inst 0 attribute 5: O->T Connection Parameters (Connection Size)
16#3156	Validate service parameter range check: CCO inst 0 attribute 5: T->O RPI
16#3157	Validate service parameter range check: CCO inst 0 attribute 5: T->O Connection Parameters
16#3158	Validate service parameter range check: CCO inst 0 attribute 5: O->T RPI

Fehlercode	Beschreibung
16#3159	Validate service parameter range check: CCO inst 0 attribute 5: O->T Connection Parameters
16#315A	Validate service parameter range check: CCO inst 0 attribute 5: T->O RPI
16#315B	Validate service parameter range check: CCO inst 0 attribute 5: T->O Connection Parameters
16#315C	Validate service parameter range check: CCO inst 0 attribute 5: T->O Connection Parameters (Connection Size)
16#315D	Validate service parameter range check: CCO inst 0 attribute 9 (I/O Data Mapping Invalid)
16#315E	Validate service parameter range check: CCO inst 0 attribute 12
16#315F	Validate service parameter range check: CCO inst 0 attribute 13: Ping Interval EPI Multiplier
16#3160	Validate service parameter range check: CCO inst 0 attribute 13: Time Coordination Message Min Multiplier
16#3161	Validate service parameter range check: CCO inst 0 attribute 13: Network Time Expectation Multiplier
16#3162	Validate service parameter range check: CCO inst 0 attribute 13: Timeout Multiplier
16#3163	Validate service parameter range check: CCO inst 0 attribute 13: Max Consumer Number
16#3164	Validate service parameter range check: CCO inst 0 attribute 13: Time Correction Connection Parameters
16#3165	Validate service parameter range check: CCO inst 0 attribute 16
16#3167	Validate service parameter range check: CCO inst 0 attribute 22
16#3170	Target returned an invalid O->T API in Forward_Open response (inst=0 API=0)
16#3171	Target returned an invalid T->O API in Forward_Open response (inst=0 API=0)
16#3177	received an unexpected Forward_Open response from CSAL (instance not waiting for it) (0)
16#3178	received an unexpected Forward_Close response from CSAL (instance not waiting for it) (0)

23 Referenz Safety-Benutzeroberfläche

23.1 Objekte

Objekte in SafeLogic stellen bestimmte Funktionalitäten für die Erstellung Ihrer Safety-Applikation zur Verfügung.

Beispiele: Safety Netzwerkvariablenliste (Sender), Safety Netzwerkvariablenliste (Empfänger).

Objekte werden im Projektexplorer verwaltet. Sie können ein Objekt mit dem Befehl **Hinzufügen** in den Projektexplorer einfügen.

Die Einfügemöglichkeiten hängen von der Position im Explorer ab. Jedes Objekt hat Eigenschaften, die Sie über das Kontextmenü des Objekts ansehen und bearbeiten können.

23.1.1 Objekt 'Safety Netzwerkvariablenliste (Sender)'

Symbol: 

Das Objekt Safety Netzwerkvariablenliste (Sender) dient der Deklaration der Safety-Netzwerkvariablen und der Konfiguration von Netzwerkvariablenlisten-Parametern und Netzwerkkommunikation.

Das Objekt wird im Gerätebaum unterhalb des Safety Applikationsobjekts mit dem Befehl **SafetyApp ▸ Hinzufügen ▸ Safety Netzwerkvariablenliste (Sender)** hinzugefügt.

Eine Safety Netzwerkvariablenliste (Sender) kann mit mehreren Safety Netzwerkvariablenlisten (Empfänger) verbunden sein.

Der Editor des Objekts enthält die Registerkarten Safety-Konfiguration und SPS-Netzwerk.

Der Editor überprüft, ob die Gesamt-Datengröße der angelegten Deklarationen innerhalb des für die Steuerung geltenden Limits ist. Wenn das Limit überschritten wird, wird eine Warnmeldung ausgegeben.

Registerkarte 'Safety-Konfiguration'

Die Registerkarte dient der Deklaration der Netzwerkvariablen und der Konfiguration von Netzwerkvariablenlisten--Parametern.

Pin-Informationen	Anzeige der Pin-Information der Sicherheitsapplikation, bzw. In Work, falls der Objektstand vom gepinnten Stand abweicht oder die Sicherheitsapplikation noch nicht gepinnt wurde. Pin-Information [SafetyApp I <Pin-Name> I <Revisionsnr.> I <Pin-Prüfsumme> I <Letzte Änderung>]
Safety-Adresse dieser Variablenliste	Erlaubte Werte: 0 bis 65535 Initialwert: 1000
Maximalanzahl von Empfängern	Erlaubte Werte: 1 bis 20 Initialwert: 4 Der Wert kann bei einer höheren Anzahl von Sicherheitssteuerungen im Projekt auch zwischen 4 und dem Maximalwert von 20 liegen

Empfänger anzeigen	Öffnet den Dialog "Zugehörige Empfänger"
Bereich für Variablendeklaration	Eingabe der Variablendeklarationen. Sie können Variablen über den Befehl **Bearbeiten ▸ Variablendeklaration einfügen** im Dialog "Safety-Variable deklarieren" deklarieren. *Spalten* • Gültigkeitsbereich – VAR_GLOBAL • Name: Bezeichner für Netzwerkvariable • Datentyp – SAFEBOOL – SAFEINT – SAFEWORD • Initialwert • Kommentar

Auskommentierte Variablendeklarationen einer Safety Netzwerkvariablenliste (Sender) werden nicht übertragen.

Siehe auch

- Befehl 'Alle zugehörigen Empfänger anzeigen', Seite 669

Registerkarte 'SPS Netzwerk'

Die Registerkarte dient den Konfigurationseinstellungen für die Netzwerkkommunikation der Standardsteuerungen.

Gruppe der Variablenlisten-Identifikatoren auf der Hauptsteuerung	Erlaubte Werte: 1 bis 255 SafeLogic berechnet den angezeigten Wert beim Erstellen des Safety NVL(Sender). Wenn Sie den Wert verändern, wird die Liste mit den Variablenlisten-Identifikatoren neu erstellt und die Anzeige "Reservierte Variablenlisten-Identifikatoren" aktualisiert. In diesem Zuge wird die Option zur Verwendung von Standardwerten aktiviert, wodurch manuelle Änderungen in der Liste zurückgesetzt werden.
Reservierte Variablenlisten-Identifikatoren	Angabe des Bereich: <Kleinster der Gruppe zugeordneter Wert> - <Größter der Gruppe zugeordneter Wert> der Listenidentifikatoren, die für diese Safety Netzwerkvariablenliste (Sender) reserviert sind. *Hinweis:* Werte aus diesem Bereich dürfen nicht für die Variablenlistenkennung von Standard-Netzwerkvariablen oder andere Safety-Netzwerkvariablen im gleichen Netzwerk verwendet werden!
Kommunikation mit der Hauptsteuerung des Empfängers	
Standardwerte für Zielnetzwerkadresse und Port verwenden (255.255.255.255, 1202):	☑ Rücksetzen der Zielnetzwerkadressen und des Ports der Liste auf den Standardwert
Liste mit Variablenlisten-Identifikatoren Spalten:	Anzahl der Einträge (Zeilen) entspricht dem Wert von Maximalanzahl von Empfängern der Registerkarte "Safety Konfiguration"

Listenidentifikatoren	Erster Listenidentifikator für ein Sender-Empfänger Verbindungspaar.
	Nicht editierbar. Wird abhängig vom ersten Listenidentifikator gesetzt: `Listenidentifikator + 1`
Zielnetzwerkadresse	Editierbar, wenn Standardwerte für Zielnetzwerkadresse und Port verwenden (255.255.255.255, 1202): nicht aktiviert ist. Kann verwendet werden, um eine Verbindung zu einer Safety-Netzwerkvariablenliste (Empfänger) an einer Steuerung mit einer bestimmten IP-Netzwerkadresse aufzubauen, anstatt die Broadcast-Adresse zu verwenden. In diesem Fall kann es erforderlich sein, dass auch die Gegenstelle (auf Empfängerseite) entsprechend konfiguriert wird. Für den Fall dass die IP Adressen der Steuerungen derart eingetragen werden, müssen ebenfalls die Listenidentifikatoren auf Empfängerseite manuell passend gesetzt werden.
Port	Kann verwendet werden um einen benutzerdefinierten Port für die Kommunikation mit dem Empfänger vorzunehmen, dessen Eintrag angepasst wird. Die Einstellung für den Port muss für die jeweilige Sender-Empfänger-Verbindung übereinstimmen, damit eine Kommunikation zustandekommen kann.

Objekteigenschaften Safety-Netzwerkvariablenliste (Sender)

Der Eigenschaftendialog enthält die Registerkarten "Allgemein", "Safety" und "Zugriffskontrolle".

Safety

Prüfsumme	Prüfsumme zu dieser Safety-Netzwerkvariablenliste (Sender)
Version	editierbar Die Version kann frei vergeben werden. Mithilfe der Version kann in der Objektliste des Safety-Applikationsobjekts schnell erkannt werden, um welche Version des Objekts es sich handelt.
Kommentar	

Für eine Beschreibung der Registerkarten Allgemein und Zugriffskontrolle wird auf die Hilfeseiten zu den Objekteigenschaften der Standard-Online-Hilfe verwiesen.

Siehe auch

- Kap. 23.1.2 "Objekt 'Safety Netzwerkvariablenliste (Empfänger)'" auf Seite 665
- Kap. 7.7 "Querkommunikation mit Safety Netzwerkvariablen" auf Seite 116
- Kap. 6.11 "Safety Netzwerkvariablenliste (NVL)" auf Seite 67
- Kap. 17.4 "SafetyNetVar.library" auf Seite 599

23.1.2 Objekt 'Safety Netzwerkvariablenliste (Empfänger)'

Symbol: 

Das Objekt Safety Netzwerkvariablenliste (Empfänger) dient dazu, die durch eine Safety-Netzwerkvariablenliste (Sender) im Projekt veröffentlichten Netzwerkvariablen innerhalb der Sicherheitsapplikation der eigenen Sicherheitssteuerung zur Verwendung bereitzustellen.

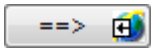
Hierfür kann der Safety Netzwerkvariablenliste (Empfänger) ein Sender zugeordnet und die Netzwerkkommunikation konfiguriert werden.

Das Objekt wird im Gerätebaum unterhalb des Safety-Applikationsobjekts mit dem Befehl **SafetyApp ▸ Objekt hinzufügen ▸ Safety Netzwerkvariablenliste (Empfänger)** hinzugefügt.

Eine Safety Netzwerkvariablenliste (Sender) kann mit mehreren Safety Netzwerkvariablenlisten (Empfänger) verbunden sein.

Der Editor enthält die Registerkarten Safety-Konfiguration und SPS-Netzwerk.

Bereich über den Registerkarten

Zugeordneter Sender	Auswahlliste mit allen im Projekt verfügbaren "Safety-Netzwerkvariablenliste (Sender)", die nicht Teil der gleichen Sicherheitsapplikation wie das Objekt "Safety-Netzwerkvariablenliste (Empfänger)" sind. *Mögliche Einträge* • Im Projekt verfügbare Objekte Safety-Netzwerkvariablenlisten (Sender) • leer • <ungültig>: Die zugeordnete "Safety-Netzwerkvariablenliste (Sender)" ist nicht mehr gültig. Wenn in der Auswahlliste ein Sender ausgewählt wird, dann werden die Variablendeklarationsliste und folgende Felder automatisch gefüllt: "Safety-Adresse des Senders", "Objektversion des Senders" Wenn der leere Eintrag oder ggf. <ungültig> ausgewählt wird, wird die Auswahl zurückgesetzt. Dabei bleiben die vorhandene Variablendeklarationsliste, die "Safety-Adresse des Senders" und die "Objektversion des Senders" unverändert.
	Öffnet den aktuell zugeordneten Sender in seinem Editor.
Aktualisieren	Aktualisiert in der Registerkarte "Safety-Konfiguration" die angezeigten Senderinformationen: "Safety-Adresse des Senders", die "Objektversion des Senders" und die "Variablendeklarationsliste" entsprechend der verknüpften Safety-Netzwerkvariablenliste (Sender). *Sonderfälle* • Wenn in der Spalte "Name" eine Umbenennung einer Variablen vorgenommen wurde, so bleibt diese bestehen, sofern eine Variable mit dem gleichem Namen ("Name im Sender") importiert wird. • Wenn in der Spalte "Kommentar" ein Wert für eine bestehende Variable definiert wurde, so bleibt dieser beim "Aktualisieren" bestehen (ein Löschen des Kommentars bewirkt einen erneuten Import vom Sender). Aktualisiert in der Registerkarte SPS-Netzwerk den Listenidentifikator entsprechend der verknüpften Safety-Netzwerkvariablenliste (Sender), falls die Option "Standardwerte für Zielnetzwerkadresse und Port verwenden" aktiviert ist. (Andernfalls werden die Listenidentifikatoren nicht beim "Aktualisieren" mit aktualisiert) Siehe Kap. 23.2.1 "Befehl 'Senderinformationen Aktualisieren'" auf Seite 668

Registerkarte 'Safety-Konfiguration'

Die Registerkarte dient der Anzeige der Netzwerkvariablen der zugewiesenen Safety-Netzwerkvariablenliste (Sender) und der Anzeige und Konfiguration der Parameter der Safety-Netzwerkvariablenliste (Empfänger).

Pin-Informationen	Anzeige der Pin-Information der Sicherheitsapplikation, bzw. In Work, falls der Objektstand vom gepinnten Stand abweicht oder die Sicherheitsapplikation noch nicht gepinnt wurde. Pin-Information [SafetyApp | <Pin-Name> | <Revisionsnr.> | <Pin-Prüfsumme> | <Letzte Änderung>]
Safety-Adresse des Senders	Nicht editierbar Initialwert: 0

Objektversion des Senders	Versionsnummer des zugeordneten Objekts Safety-NLV(Sender), die in den Objekteigenschaften des Senders: **Ansicht ▸ Eigenschaften**, Registerkarte "Safety", Eingabefeld "Version" des Senders eingetragen ist.
Connection ID	Erlaubte Eingabewerte: Ganze Zahlen im Bereich 1...65 535 Initialwert: projektweit eindeutige ganze Zahl >= 1000
Watchdog-Zeit	Angabe in ms Erlaubte Eingabewerte: 1...65 534 Initialwert: 200
Variablenliste mit den Variablen der Sender-NVL	Auflistung der Variablendeklaration der zugeordneten Sender-NVL. Die Liste enthält zusätzlich zu den Spalten der Safety Netzwerkvariablenliste (Sender) auch die Spalte "Name im Sender", jedoch keine Spalte "Initialwert". Folgende Felder können im Offline-Modus editiert werden: "Name", "Kommentar". Die empfangene Variable wird unter dem Namen in der Spalte "Name" in der Sicherheitsapplikation zur Verfügung gestellt. Initial ist der Name gleich zu der Spalte "Name im Sender". Im Feld "Name" kann ein von "Name im Sender" abweichender Name eingegeben werden.

Registerkarte 'SPS Netzwerk' Die Registerkarte dient den Konfigurationseinstellungen für die Netzwerkkommunikation der Standardsteuerungen.

Kommunikation mit der Hauptsteuerung des Senders	
Standardwerte für Zielnetzwerkadresse und Port verwenden (255.255.255.255, 1202):	☑ Rücksetzen die Zielnetzwerkadresse und Port der Liste auf den Standardwert Wenn die Safety Netzwerkvariablenliste (Empfänger) mit einer Safety Netzwerkvariablenliste (Sender) verknüpft ist und diese Option aktiviert ist, wird automatisch in der Liste ein gültiger Listenidentifikator eingetragen. Dieser Listenidentifikator ist auch in den Listenidentifikatoren des verbundenen Senders enthalten.
Spalten:	
Listenidentifikatoren	Erster Listenidentifikator, der für diese Verbindung verwendet wird. Editierbar, wenn Verwendung "Standard Zielnetzwerkadresse und Port (255.255.255.255, 1202)" nicht aktiviert ist. Leer, wenn die Safety Netzwerkvariablenliste (Empfänger) mit keiner Safety-Netzwerkvariablenliste (Sender) verknüpft ist.
	Zweiter Listenidentifikator für die Sender-Empfänger-Verbindung Wird abhängig vom ersten Listenidentifikator gesetzt: `Listenidentifikator + 1` Leer, wenn die Safety-Netzwerkvariablenliste (Empfänger) mit keiner Safety-Netzwerkvariablenliste (Sender) verknüpft ist.

Zielnetzwerkadresse	IP-Adresse des Senders Editierbar, wenn Verwendung "Standard Zielnetzwerkadresse und Port (255.255.255.255, 1202)" nicht aktiviert ist. Kann verwendet werden, um eine Verbindung zu einer Safety-Netzwerkvariablenliste (Sender) an einer Steuerung mit einer bestimmten IP-Netzwerkadresse aufzubauen, anstatt die Broadcast Adresse zu verwenden. In diesem Fall kann es erforderlich sein, dass auch die Gegenstelle (auf Senderseite) entsprechend konfiguriert wird. Für den Fall dass die IP Adressen der Steuerungen derart eingetragen werden, müssen ebenfalls die Listenidentifikatoren korrekt manuell gesetzt werden.
Port	Port der Zielnetzwerkadresse, über den die Kommunikation laufen soll. Editierbar, wenn Verwendung "Standard Zielnetzwerkadresse und Port (255.255.255.255, 1202)" nicht aktiviert ist. Kann verwendet werden um einen benutzerdefinierten Port für die Kommunikation mit dem ausgewählten Sender vorzunehmen. Die Einstellung für den Port muss auf beiden Seiten der Sender-Empfänger-Verbindung übereinstimmen, damit eine Kommunikation zustandekommen kann.

Objekteigenschaften Safety Netzwerkvariablenliste (Empfänger)

Der Eigenschaftendialog enthält die Registerkarten "Allgemein", "Safety" und "Zugriffskontrolle".

Safety

Prüfsumme	Prüfsumme zu dieser Safety-Netzwerkvariablenliste (Empfänger)
Version	editierbar Die Version kann frei vergeben werden. Mithilfe der Version kann in der Objektliste des Safety-Applikationsobjekts schnell erkannt werden, um welche Version des Objekts es sich handelt.
Kommentar	

Für eine Beschreibung der Registerkarten Allgemein und Zugriffskontrolle wird auf die Hilfeseiten zu den Objekteigenschaften der Standard-Online-Hilfe verwiesen.

Siehe auch

- Kap. 23.1.1 "Objekt 'Safety Netzwerkvariablenliste (Sender)'" auf Seite 663
- Kap. 7.7 "Querkommunikation mit Safety Netzwerkvariablen" auf Seite 116
- Kap. 6.11 "Safety Netzwerkvariablenliste (NVL)" auf Seite 67
- Kap. 17.4 "SafetyNetVar.library" auf Seite 599

23.2 Befehle

Standardmäßig sind die wichtigsten Befehle in der Benutzeroberfläche von IndraLogic/ SafeLogic verfügbar.

Wenn Sie die Menükonfiguration individuell anpassen möchten, wählen Sie den Befehl **Extras ▸ Anpassen ▸ Befehle**.

23.2.1 Befehl 'Senderinformationen Aktualisieren'

Funktion:

Der Befehl aktualisiert die im Objekt "Safety-Netzwerkvariablenliste (Empfänger)" hinterlegte "Safety-Adresse", "Objektversion" des Senders und die Variablenliste des Objekts "Safety-Netzwerkvariablenliste (Empfänger)".

Aufruf: Schaltfläche Aktualisieren im Editor der Safety-Netzwerkvariablenliste (Empfänger).

Voraussetzungen:

- Ein Objekt "Safety-Netzwerkvariablenliste (Empfänger)" einer Sicherheitsapplikation ist geöffnet
- Eine gültiges Objekt "Safety-Netzwerkvariablenliste (Sender)" ist dem Objekt "Safety-Netzwerkvariablenliste (Empfänger)" zugeordnet.
- Die Applikation befindet sich nicht aktuell im Online-Betrieb

Siehe auch

- Objekt 'Safety Netzwerkvariablenliste (Sender)', Seite 663
- Objekt 'Safety Netzwerkvariablenliste (Empfänger)', Seite 665

23.2.2 Befehl 'Alle zugehörigen Empfänger anzeigen'

Funktion:

Der Befehl öffnet den Dialog "Zugeordnete Empfänger". Der Dialog listet alle Objekte "Safety-Netzwerkvariablenliste (Empfänger)" auf, die dem geöffneten "Safety-Netzwerkvariablenliste (Sender)" als Empfänger in diesem Projekt zugeordnet sind

Aufruf: **Bearbeiten ▸ Safety Netzwerkvariablenliste (Sender)** im Projektexplorer oder Schaltfläche im Editor des Objekts "Safety-Netzwerkvariablenliste (Sender)"

Voraussetzungen:

Der Editor eines Objekts Safety-Netzwerkvariablenliste (Sender) ist aktiv.

Dialog 'Zugehörige Empfänger'

Verbunden mit Safety-Netzwerkvariablenliste:	Name und Pfad der zugeordneten Safety Netzwerkvariablenliste(Sender).
Auflistung der Safety Netzwerkvariablenlisten (Empfänger) des Projekts mit folgenden Informationen: • **Applikation (mit Pfad)**: Pfad der Safety-Variablenliste (Empfänger): [<Haupt-SPS>:<Safety-SPS>:Safety Logic:<SafetyApp>] • **Variablenliste**: Name der Safety-Netzwerkvariablenliste (Empfänger) Wenn Sie auf eine Safety-Netzwerkvariablenliste (Empfänger) der Liste doppelklicken, wird die Safety-Netzwerkvariablenliste (Empfänger) im Editor geöffnet. Wenn im Projekt kein Empfänger existiert, in dem der Safety-Netzwerkvariablenliste (Sender) als gültiger Sender hinterlegt ist, wird in der Liste angezeigt: `keine gültigen Empfänger im Projekt .`	

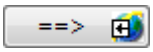
Siehe auch

- Objekt 'Safety Netzwerkvariablenliste (Sender)', Seite 663

23.2.3 Befehl 'Gehe zu Sender'

Funktion:

Der Befehl öffnet den Editor des Objekts "Safety-Netzwerkvariablenliste (Sender)", das der aktiven "Safety-Netzwerkvariablenliste (Empfänger)" aktuell zugeordnet ist.

Aufruf: **Bearbeiten**, als Schaltfläche 

im Editor der "Safety-Netzwerkvariablenliste (Empfänger)", oder Kontextmenü der Safety-Netzwerkvariablenliste (Empfänger) im Projektexplorer.

Voraussetzungen:

Der Editor einer "Safety-Netzwerkvariablenliste (Empfänger)" ist aktiv und eine gültige "Safety-Netzwerkvariablenliste (Sender)" ist der "Safety-Netzwerkvariablenliste (Empfänger)" zugeordnet.

Siehe auch

- Objekt 'Safety Netzwerkvariablenliste (Sender)', Seite 663
- Objekt 'Safety Netzwerkvariablenliste (Empfänger)', Seite 665

23.2.4 Befehl 'Querverweise ausgeben - Safety'

Symbol: 

Aufruf: Der Befehl **Ansicht ▸ Andere Fenster ▸ Safety Querverweisliste** öffnet die Safety Querverweisliste, in der die Verwendungsstellen folgender Safety-Objekte angezeigt werden können:

- Variablen (einzeln / alle)
- POUs und FB-Instanzen innerhalb des Projekts
- referenzierte Bibliotheksbausteine

Durch Auswahl des Gültigkeitsbereichs kann die die Anzeige auf folgende Elemente beschränkt werden:

- alle Querververweise
- Querverweise nur im Projekt
- Querverweise nur in der aktiven Applikation
- Querverweise nur in einem bestimmten Baustein (POU)

Siehe Tab. "Elemente der Ansicht Safety Querverweisliste" auf Seite 672

⚠ VORSICHT

Bei Verwendung der Safety Querverweisliste zur Kontrollflussanalyse, oder zur Datenflussanalyse der Sicherheitsapplikation sind folgende Punkte zu beachten:

1. Richtiges Namensformat

Im Feld "Name" dürfen nur "unqualifizierte" Bezeichner eingegeben werden. Die Suche nach einer globalen Variable erfolgt durch Eingabe von "<Variablenname>".

Die Eingabe von "<GVL-Name>.<Variablenname>" wird nicht unterstützt.

Nach FB-Eingängen und FB-Ausgängen wird durch Eingabe von "<Ein/Ausgangsname>" gesucht.

Die instanzbezogene Suche mittels "<FB Instanz-Name>.<Ein/Ausgangsname>" wird nicht unterstützt.

2. Richtiger Abschluss der Eingabe / Start der Suche

Nach Auswahl des Gültigkeitsbereichs und Eingabe des Bezeichners im Feld Name müssen diese Eingaben durch Betätigen der [Enter] - Taste abgeschlossen werden.

Die [Enter] - Taste aktiviert die Auflistung aller Querverweise in der darunter liegenden Tabelle.

Die Verwendung der Lupen-Schaltfläche sucht nur alle Verwendungsstellen im Gültigkeitsbereich "Aktive Applikation".

In der Ansicht "Safety Querverweisliste" können nur Querverweise in Safety-Objekten gefunden werden.

Querverweise aus Standardobjekten werden in dieser Ansicht nicht gefunden!

Im Gegenzug können in der Ansicht "Querverweisliste" keine Querverweise in Safety-Objekten gefunden werden.

Die Funktionen "Querverweise ausgeben" und "Gehe zur Definition" dienen in IndraLogic der Kontroll- und Datenflussanalyse bei der statischen Verifikation einer Sicherheitsapplikation, siehe Kap. 10 "Software-Verifikation" auf Seite 167

⚠ VORSICHT

Bei Verwendung der Safety Querverweisliste zur Kontrollflussanalyse bzw. zur Datenflussanalyse der Sicherheitsapplikation im Rahmen von Tätigkeiten der Verifikation, oder Einflussanalyse von Änderungen müssen alle Hinweise und Sicherheitshinweise in der Anwendungsbeschreibung beachtet werden.

Querverweise ausgeben im Safety FUP-Editor

Der Kontextmenübefehl **Durchsuchen ▸ Querverweise ausgeben** kann im Safety FUP-Editor auf Bausteine und Variablen (Bezeichner) ausgeführt werden.

Die Deklaration einer POU selbst wird in der Querverweisliste nicht als Verwendungsstelle aufgeführt.

Ansicht 'Safety Querverweisliste'

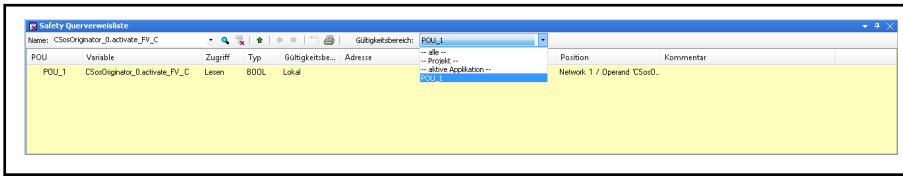


Abb. 23-1: Ansicht "Safety Querverweisliste"

Elemente der Ansicht "Safety Querverweisliste"

Name	Variablenname, POU-Name, Name der FB-Instanz Eingabemöglichkeiten: • Manuell eintippen • Durch automatische Aktualisierung, wenn die IndraWorks-Option "Querverweise automatisch bei Selektionsänderung aktualisieren" aktiviert ist Aufruf über Menü **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **SmartCoding** ▸ **Querverweise automatisch bei Selektionsänderung aktualisieren** Folgende Eingaben sind gültig, beachten Sie dabei auch jeweils den gerade eingestellten Gültigkeitsbereich: • Name • "*": IndraLogic sucht die Querverweise aller Variablen • "*<Zeichenfolge>": IndraLogic sucht nach den Querverweisen aller Variablen, deren Name diese Zeichenfolge enthält. Beispiel: "iVar*" betrifft iVar1, iVar_glob2, iVar45 etc.
	Stellt den Gültigkeitsbereich auf Aktive Applikation ein und sucht die Querverweise für diesen Gültigkeitsbereich
	Querverweise im Meldungsfenster ausgeben: IndraLogic gibt die aktuelle Querverweisliste im Meldungsfenster aus. Das kann nützlich sein, wenn die aktuelle Liste festhalten werden soll, obwohl die automatische Aktualisierung (siehe oben bei Name) aktiviert ist und sich somit die Liste im Querverweislisten-Fenster ändern kann
	Zeige Position: Wenn gerade neu nach Querverweisen gesucht wurde, zeigt nun ein grüner Pfeil auf die erste gefundene Verwendungsstelle in der Liste
	Zeige vorige Position [Umschalttaste] + [F4] : Zeigt die Position des zuvor gefundenen Querverweises
	Zeige nächste Position [F4] : Zeigt die Position des als nächstes gefundenen Querverweises
	Gehe zur Definition [F2] : IndraLogic springt zu der Stelle, an der die Variable deklariert ist. Der entsprechende Deklarationseditor wird geöffnet und die Variable dort selektiert
	Drucken: Die aktuelle Querverweisliste wird gedruckt

Gültigkeitsbereich	Auswahl des Bereichs, in dem nach Querverweisen gesucht wird: • **alle**: Projekt und referenzierte Bibliotheksbausteine • **Projekt**: Projekt ohne Bibliotheken • **aktive Applikation** • **<POU-Name>** Wenn eine POU im Editor geöffnet ist und die Variable selektiert ist, für welche die Querverweise ausgegeben werden sollen, stellt IndraLogic den Gültigkeitsbereich automatisch auf diese POU ein. Voraussetzung hierfür ist, dass die Option "Querverweise automatisch bei Selektionsänderung aktualisieren" aktiviert ist Aufruf: IndraWorks Menü **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **SmartCoding**
Die gefundenen Querverweise werden mit folgenden Informationen dargestellt:	
POU	Name des Bausteins, in dem die Variable verwendet wird
Variable	Variablenname, bei Referenzen: Bausteinname + Variablenname. Beispiele: "iVar", "xPOU.iVar1"
Zugriff	Art des Zugriffs auf die Variable an der Verwendungsstelle: **Deklaration** / **Lesen** / **Schreiben** / **Aufruf**
Typ	Datentyp der Variablen
Bereich	Gültigkeitsbereich der Variablen: **Global** / **Lokal**
Adresse	IEC-Adresse, die der Variablen gegebenenfalls zugewiesen ist
Position	Position der Verwendungsstelle der Variablen innerhalb des Editors der betroffenen POU: Beispielsweise Zeilennummer, Netzwerknummer, Deklarations- oder Implementierungsteil
Kommentar	Kommentar, falls in der Deklaration der Variable vorhanden

24 Referenz Safety-Programmierung

24.1 Programmierlevel Basic - Extended

Sprachumfang der Sicherheitsprogrammierung mit SafeLogic

Der Sprachumfang von Safety-FUP wird entsprechend der in PLCopen definierten Sprachteilmengen Basic und Extended eingeschränkt (siehe Überblick Netzwerke, Seite 97).

Die entsprechende Auswahl für den Sprachumfang Basic oder Extended wird beim Neuanlegen einer POU (Programm oder Funktionsbaustein) durch den Entwickler festgelegt (siehe POUs, Seite 96).

Im **Basic Level** kann durch Verknüpfung der bereits zertifizierten Bausteine der PLCopen-Bibliothek (SafetyPLCopen) und der Standard-Bibliothek (SafetyStandard) mit relativ geringem Aufwand eine Sicherheitsapplikation implementiert und anschließend verifiziert werden.

Extended Level bietet dem Entwickler zusätzliche Operatoren (boolesche, mathematische, und andere) und bedingte Sprünge/Returns, um komplexere Funktionsbausteine für wiederkehrende Anwendungsfälle zu erstellen. Sie erfordern im Anschluss an die Entwicklung einen entsprechend aufwändigeren Verifikationsprozess.

Je nach gewähltem Programmierlevel steht zur Code-Implementierung zur Verfügung

- entsprechende Operatoren (Operatoren, Seite 106)
- entsprechende Schlüsselwörter zur Variablendeklaration und Datentypen (Grundlegendes zu Variablen, Seite 100)
- entsprechende FB-Aufrufe (FB-Aufrufe, Seite 112)
- Sprung/Return (Sprung/Return und Sprungmarke, Seite 111)

24.2 Safety-FUP-Sprachelemente

Zur Implementierung des Programmcodes einer Sicherheitsapplikation dienen generell folgende Objekte und deren Elemente:

- Globale Variablen der Safety GVLs (Globale Variablenliste (GVL), Seite 91)
- POUs
  - Netzwerke, Seite 676
  - Variablen, Seite 676
  - Operatoren, Seite 106
  - Zuweisungen
  - Sprung/Return und Sprungmarke, Seite 111
  - FB-Aufrufe, Seite 112
- Task und Taskkonfiguration, Seite 120
  - Programmliste (siehe Safety Task)
- Logische E/As (Logische E/As - Überblick, Seite 47)
  - Mapping-Variablen (Siehe E/A Abbild)
- Bausteine der Sicherheitsbibliotheken, die im Bibliotheksverwalter angezogen werden (Bibliotheksverwalter, Seite 71)

24.2.1 Netzwerke

Programmiereinheiten in FUP sind in aufsteigend durchnummerierte Netzwerke untergliedert, die grafisch dargestellte Elemente, wie Operanden, Bausteinaufrufe, Zuweisungen, Sprünge oder Sprungmarken enthalten können. Die Netzwerke werden in aufsteigender Reihenfolge abgearbeitet.

Innerhalb eines Netzwerks muss weist der Programmcode eine Baumstruktur auf, dies bedeutet: keine Parallelverschaltung, keine Aufsplittung, keine expliziten Feedbackloops.

Die logischen Elemente eines Netzwerks werden durch Linien zu einer mit der Wurzel rechts liegenden baumartigen Formation (kurz Baum) verbunden, wobei die Boxen in diesem Baum als Knoten fungieren.

Ausnahme: Mehrfachzuweisungen sowie Sprung- und Return-Anweisungen fächern diesen Baum in entgegengesetzter Richtung auf. Die Elemente werden vom Editor automatisch, entsprechend ihrer Einfügeposition verbunden, eine freie Platzierung ist nicht möglich.

24.2.2 Variablen

Inhalt

- Kategorien für Variablen in der Eingabehilfe, Seite 676
- Schlüsselwörter der Variablendeklaration für Basic Level, Seite 676
- Schlüsselwörter der Variablendeklarationen für Extended Level, Seite 677
- Datentypen Basic Level, Seite 677
- Datentypen Extended Level, Seite 678

Die zur Auswahl stehenden Schlüsselwörter und Datentypen hängen vom Programmierlevel der POU ab (Programmierlevel Basic - Extended, Seite 675).

Im FUP-Programmcode, der im Safety FUP-Editor erstellt wird, erscheinen Variablen ausschließlich in Form von Namen (symbolische Variablen). Man kann einen einfachen Namen, einen Bezeichner (identifier) im Sinne der IEC verwenden. Ein Bezeichner, mit dem eine Variable im Implementierungsteil referenziert wird, muss im Deklarationsteil der POE deklariert sein

Kategorien für Variablen in der Eingabehilfe

In der Eingabehilfe werden für Variablen folgende Kategorien angezeigt:

- PROGRAM POU: **Lokale Variablen** und **Globale Variablen**
- FUNCTION_BLOCK POU: **Lokale Variablen**

Zur Kategorie **Lokale Variablen** gehören alle Variablen des lokalen Deklarationsteils, auch VAR_EXTERNAL.

Zur Kategorie **Globale Variablen** gehören alle Variablen der GVLs der Applikation, alle Variablen der NVLs, alle gemappten E/As und alle impliziten Instanzen der logischen E/As

Schlüsselwörter der Variablendeklaration für Basic Level

Variablen für Programm-POU Basic Level

- VAR
- VAR CONSTANT
- VAR_EXTERNAL für Kanalvariablen und Stack-Instanzen
- VAR_EXTERNAL CONSTANT nur erlaubt von VAR_GLOBAL CONSTANT

Variablen für Funktionsbaustein-POU Basic Level

- VAR

- VAR CONSTANT
- VAR_INPUT
- VAR_OUTPUT

Bedeutung der Schlüsselwörter

- VAR: Deklaration normaler interner Variablen, POU-weite Variable
- VAR_INPUT: Deklaration von Input-Variablen
- VAR_OUTPUT: Deklaration von Output-Variablen
- der Modifizierer CONSTANT dient zur Deklaration von symbolischen Konstanten

Schlüsselwörter der Variablendeklarationen für Extended Level

Variablen für Programm-POU Extended Level

- VAR
- VAR CONSTANT
- VAR_EXTERNAL
- VAR_EXTERNAL CONSTANT

Variablen für Funktionsbaustein-POU Extended Level

- VAR
- VAR CONSTANT
- VAR_INPUT
- VAR_OUTPUT

Bedeutung der Schlüsselwörter

- VAR: Deklaration normaler interner Variablen, POU-weite Variable
- VAR_INPUT: Deklaration von Eingangsvariablen
- VAR_OUTPUT: Deklaration von Ausgangsvariablen
- VAR_EXTERNAL: Deklaration von mit VAR_GLOBAL in der Applikation bereits deklarierter globaler Variablen, um sie in der POU benutzbar zu machen. Globale Variablen, die den Modifizierer CONSTANT haben, müssen als VAR_EXTERNAL CONSTANT deklariert werden.
- der Modifizierer CONSTANT dient zur Deklaration von symbolischen Konstanten.

Für die Deklaration als VAR_EXTERNAL und VAR_EXTERNAL CONSTANT stehen bereits existierende Variablen der Kategorie "Globale Variablen" zur Verfügung

- Globale Variablen des GVL-Objekts der Sicherheitsapplikation.

  Wird eine Variable des GVL-Objekts der Sicherheitsapplikation im Implementierungsteil verwendet, so wird sie automatisch im Deklarationsteil als externe Variable deklariert.

  Nach IEC ist es explizit verboten, in einer POU globale Variablen zu verwenden, ohne sie als "External" zu deklarieren.
- Mapping-Variablen der logischen E/As (Logische Austauschgeräte und sichere Feldgeräte)

Datentypen Basic Level

- BOOL
- INT: nur als konstanter Eingangsparameter für einen FB-Aufruf
- DINT: nur als konstanter Eingangsparameter für einen FB-Aufruf
- WORD: nur als Ausgang für Diagnosezwecke
- TIME: nur als konstanter Eingangsparameter in FB-Aufruf

- SAFEBOOL
- SAFEINT nur als konstanter FB-Eingang in Aufruf
- SAFEDINT: nur als konstanter Eingangsparameter in FB-Aufruf
- SAFEWORD: nur als konstanter Eingangsparameter in FB-Aufruf
- SAFETIME: nur als konstanter Eingangsparameter in FB-Aufruf

Der Datentyp REAL steht nicht zur Verfügung.

Datentypen Extended Level

Für die Implementierung einer POU im Programmierlevel Extended stehen dem Entwickler folgende Datentypen zur Verfügung:

Sicherheits-Standardtypen

- BOOL
- BYTE, nur bei VAR_EXTERNAL Deklarationen zugelassen, welche logische E/As importieren.
- DINT
- DWORD, nur bei VAR_EXTERNAL Deklarationen zugelassen, welche logische E/As importieren.
- INT
- TIME: erlaubt als konstanter Eingangsparameter und für lokale Variablen.

  Nicht zulässig sind External-Deklarationen globaler Variablen vom Typ SAFETIME
- WORD: als Ausgang für Diagnosezwecke und für lokale Variablen
- Kein Datentyp REAL
- SAFEBOOL
- SAFEBYTE, nur bei VAR_EXTERNAL Deklarationen zugelassen, welche logische E/As importieren.
- SAFEDINT
- SAFEDWORD, nur bei VAR_EXTERNAL Deklarationen zugelassen, welche logische E/As importieren.
- SAFEINT
- SAFETIME, erlaubt als konstanter Eingangsparameter und für lokale Variablen.

  Nicht zulässig sind External-Deklarationen globaler Variablen vom Typ SAFETIME, wenn sie weder symbolische Konstanten noch importierte logische E/As sind.
- Nicht zulässig: SAFE(WORD)-Eingänge (VAR_INPUT) und External-Deklarationen von normalen globalen (SAFE)WORD Variablen.

Die Datentypen BYTE, SAFEBYTE, DWORD und SAFEDWORD stehen nur für die Schnittstellen externer FBs und zum Import logischer E/As zur Verfügung.

Der Datentyp REAL/LREAL steht nicht zur Verfügung.

24.2.3 Operatoren

Als Operatoren werden die eingebauten Standardfunktionen, die eine Teilmenge der Standardfunktionen der IEC darstellen, bezeichnet. Benutzerdefinierte Funktionen können in der Safety-Programmierung nicht erstellt werden.

Die Operatoren haben die gleiche Semantik und Logik wie in Standard-IndraLogic (weitere Informationen dazu finden Sie unter Operatoren).

Sie können sowohl mit SAFExxx als auch mit Standard-Datentypen beschaltet werden.

Einfügen von Operatoren

Das Einfügen der Operatoren erfolgt über die Befehle Bausteinaufruf einfügen, bzw. Leeren Baustein einfügen. Die Operatoren können auch aus der Werkzeugbox (Toolbox) der Ansicht Werkzeuge hinzugefügt werden. In der Werkzeugbox des Safety FUP-Editor gibt es die Kategorien Boolesche Operatoren Mathematische Operatoren Sonstige Operatoren

25 Glossar, SafeLogic

Abnahme der Applikation

Ein Fall von Softwareabnahme. Teil der **Maschinenabnahme**.

Allgemein: Die Abnahme einer **Applikation**, d. h. der ablauffähigen Einheit auf der Steuerung.

Die Geräteparameter gehören nicht dazu (obwohl sie auch Software sind). Die Abnahme der Applikation kann aber abhängig sein von der Abnahme der Geräteparameter der benutzten E/A-Geräte (**Geräteparameterabnahme**), bzw. organisatorisch zusammen mit ihr erfolgen.

Die Abnahme der Applikation (im Gegensatz zur Abnahme der Geräteparameter) ist typischerweise abhängig vom Typ der Steuerung und von der Version der Firmware auf der Steuerung.

Adresse

Hier: Safety-Adresse

...eines E/A-Moduls, die über Parametrierung eingestellte Adressierung des E/A-Moduls innerhalb seines Feldbusprotokolls.

Im Fall von Profisafe: Kombination aus `F_Source_Add` und `F_Dest_Add`.

AOPD

Active opto-electronic protective device

Optische Schutzeinrichtung, die berührungslos Objekte im Schutzraum erkennt.

Beispiele:

- Lichtgitter
- Laserscanner

API

Mit Hilfe einer API erhält der Entwickler auf der Programmierebene Zugang zu jedem sicheren E/A-Modul auf Ebene der Safety-Kommunikation.

Anwender

(des IndraWorks Safety Managers): Entwickler, der Sicherheitsapplikationen bearbeitet und/oder auf Sicherheitssteuerungen ausführt.

Anwenderprogramm-Fehlerreaktion

Vom **Anwender** in der **Applikation** programmierte Fehlerreaktion.

Applikation

Ablauffähige Einheit auf einer Steuerung aus Programmcode, Bibliotheksverwaltung, Taskverwaltung und zugehöriger E/A-Konfiguration.

Es existieren:

- eine **Standardapplikation**, im Projektnavigator unterhalb **Logic ▸ Application**
- eine **Sicherheitsapplikation**, im Projektnavigator unterhalb **Safety Logik ▸ SafetyApp**

Applikationsstatus des Safety-Laufzeitsystems (S_LZS)

Zustände, in denen sich eine Sicherheitsapplikation auf dem S_LZS befinden kann, z. B. Applikation läuft, Applikation steht etc., vgl. **Betriebsmodus**.

Austauschvariablen

Variablen, deren Werte zwischen einer Sicherheitsapplikation (auf einer Sicherheitssteuerung) und einer oder mehreren Standard-Applikationen (auf verbundenen Standardsteuerungen) ausgetauscht werden.

Ausführungsversion

Kennung mittels welcher die kontrollierte Kompatibilität zwischen (abgenommener) Bootapplikation und Laufzeitsystemkern geregelt wird.

Kompatibilität mit externen POUs siehe **Implementierungsversion**

Aus Anwendersicht entspricht die Ausführungsversion der Compiler-Version aus IndraWorks.

Eine geänderte Versionsnummer bedeutet, dass sich die Abarbeitung der Applikation im Detail ändern kann.

Basic Level Gemäß PLCopen ein **Programmierlevel** mit einfachem, definiert eingeschränktem Sprachumfang.

Der Level soll dem Anwender entsprechen, der bislang die Verdrahtung von sicherheitsgerichteten Modulen vorgenommen hat.

Programmierlevel, der auf die Verwendung zertifizierter (oder validierter) Funktionsbausteine in Sicherheitsapplikation ausgerichtet ist.

BDC Bottom Dead Center

unterer Totpunkt des Exenters einer Presse

Bedienschnittstelle (User Interface) Schnittstelle zwischen Mensch und Software. Insbesondere die grafische Bedienoberfläche des **Programmiersystems** zum **Anwender**.

Zu unterscheiden von **Programmierschnittstelle**.

Betriebsmodus des Safety-Laufzeitsystems Für den Anwender wichtiger Zustand des Safety-Laufzeitsystems, z. B. Debug, System-Fehlerreaktion etc.

Je nach Modus sind unterschiedliche Aktionen aktiviert bzw. deaktiviert, vgl. **Applikationsstatus**.

Bibliotheks-Repository Datenablage von IndraWorks, in der Bibliotheken versioniert abgelegt werden.

Bibliotheksverwalter Objekt zur Verwaltung der eingebundenen Bibliotheken in IndraWorks.

BWS Berührungslos wirkende Schutzeinrichtung

Cam shaft Nockenwelle

Cam Switch Unit Nockenschaltwerk

CRC Cyclic Redundancy Check

Allgemein: Eine Prüfsumme über einen Datenstrom, die durch eine bestimmet Methode bestimmt wird. Die Methode ist parametriert durch einen Startwert und ein Polynom. Prüfsummen dienen der Absicherung übertragener oder gespeicherter Daten.

Hier auch: Prüfwert im Sicherheitstelegramm von Profisafe zur Überprüfung der Integrität der Telegrammdaten.

CSoS CIP Safety on sercos

Differenzabnahme Erleichterte Abnahme einer Applikation, die eine Änderung einer bereits verifizierten Applikation ist.

Ersatzwert Fail safe values

Wert der im Fehlerfall anstelle des programmierten / berechneten Wertes ausgegeben wird.

EDM External device monitoring

Prüfung externer sicherer Sensoren (s. auch Kap. 16.4.3 "SF_EDM" auf Seite 306)

ESPE electro-sensitive protective equipment

Berührungslos wirkende Schutzeinrichtung (s. auch Kap. 16.4.7 "SF_ESPE" auf Seite 333)

Extended Level Gemäß PLCopen ein **Programmierlevel** mit Sprachumfang zwischen **Basic Level** und **System Level**.

Externe POU Im Laufzeitsystem implementierte Bausteine

Failsafe Hier: Werte von Variablen, die die Sicherheit der Anlange gewährleisten.

Failsafe-Wert für Variablen des Datentyps SAFEBOOL muss FALSE sein.

Der Anwender muss sicherstellen, dass alle Variablen vom Datentyp SAFEBOOL die Sicherheit der Anlage gewährleisten.

Alle Variablen vom Datentyp SAFEBOOL müssen sowohl für die Initialisierung als auch für eine Sicherheitsanforderung auf den Wert FALSE gesetzt werden.

F- Präfix. Kurz für failsafe (dt. sicherheitsgerichtet).

F-Device
- In dieser Dokumentation: Sammelbezeichnung für sicherheitsbezogene Geräte (F-Slave, F-Module....)
- Profisafe-Teilnehmer auf Profinet-Basis.

F-Host Systemteil, der das Profisafe-Protokoll für alle F-Devices bedient. Dieser befindet sich typischerweise auf einem **sicherheitsgerichteten LZS**.

F-Modul Profisafe-Teilnehmer, Module eines modularen Gerätes auf Profibus- oder Profinet-Basis.

F-Parameter Sicherheitsgerichtete Parameter eines Profisafe-F-Devices, definiert in der Profisafe-Spezifikation.

Fehlerreaktion Reaktion des Laufzeitsystems (LZS), das die Anlage in den sicheren Zustand bringt. Unterschieden werden die **Anwenderprogramm-Fehlerreaktion** und die **System-Fehlerreaktion**.

Feldbus-Teilnehmer Gerät, das ein Feldbus-Protokoll (z. B. Profibus DP oder Profinet) implementiert, über welches das Prozessabbild ausgetauscht wird.

Relevant sind insbesondere:
- Koppler mit untergeordneten **E/A-Modulen**
- Antriebe, d. h. **E/A Module**, die typischerweise selbständige Feldbus-Teilnehmer sind

Firmware Systemspezifische Software für den Betrieb des Laufzeitsystems (z. B. Betriebsystem, hardwarespezifische Anpassungen) plus das Laufzeitsystem selbst.

Auf der Safety-Steuerung dementsprechend ein **sicherheitsgerichtetes Laufzeitsystem**.

Funktionsbaustein (FB) **POU** nach **IEC 61131-3**, kann andere Funktionsbausteine aufrufen und von einer Applikation und anderen Funktionsbausteinen aufgerufen werden.

Funktionsplan eine in der IEC 61131-Funktionsbausteinsprache (FBS) geschriebene Programmorganisationseinheit (POU).

Die Sprache ist zulässig für die Sicherheitsprogrammierung in **Basic Level**, **Extended Level** und **System Level**.

Gerätebeschreibung Hier: Safety-Gerätebeschreibung.

Beschreibung von Parametersätzen, mit denen ein Gerät konfiguriert werden kann.

Gerätebeschreibungspaket Hier: Safety-Gerätebeschreibungspaket.

Ein in sich abgesichertes Paket bestehend aus zwei Teilen:
- der eigentlichen, IndraWorks Safety Manager-spezifischen, generischen Safety-Gerätebeschreibung im Textformat einerseits und
- einer Absicherungs-**Prüfsumme** darüber andererseits: **Abgesicherte Ablage**

Geräteparameterabnahme Ein Fall von Softwareabnahme. Teil der **Maschinenabnahme**.

Die Abnahme der **Safety-Geräteparameter** eines E/A-Gerätes.

Die Abnahme der Geräteparameter aller verwendeter E/A-Geräte kann Voraussetzung für die Abnahme der Sicherheitsapplikation sein (**Applikationsabnahme**).

Die Abnahme der Geräteparameter ist typischerweise abhängig vom Gerätetyp.

Globale Variablenliste (GVL) Eine (benannte) Liste von globalen Variablen innerhalb einer Applikation. Es kann mehrere GVL innerhalb einer Applikation geben.

Hier: **Safety Globale Variablenliste (GVL)** dient der Darstellung, Deklaration und Bearbeitung global deklarierter Variablen, die innerhalb der ganzen Sicherheitsapplikation gültig sind. Es können einer Sicherheitsapplikation mehrere GVLs hinzugefügt werden.

Hintergrundtask Laden der Applikation und Online-Kommunikation des LZS erfolgen in einer niederprioren Task, die im Hintergrund bearbeitet.

IEC 61131-3 Internationale Norm für Speicherprogrammierbare Steuerungen.

IEC-Code Programmquelltext, konform zu IEC 61131-3, also AWL, KOP, FUP, AS oder ST.

Implementierungsversion Kennung einer externen POU mittels welcher die kontrollierte Kompatibilität zwischen (abgenommener) Bootapplikation und der Implementierung der externen POU geregelt wird. Kompatibilität mit LZS-Kern ist die **Ausführungsversion**.

IndraLogic Programmiersystem der Firma Bosch Rexroth, zur Programmierung von **SPS**.

Liegt derzeit in der zweiten Generation vor (IndraLogic 2G).

IndraWorks Automation Platform IndraWorks Automation Platform:

Eine Entwicklungsplattform zur Erweiterung des IndraWorks-Programmiersystems, zugleich ein komplettes Applikations-Framework zur Erstellung von Entwicklungsumgebungen für industrielle Steuerungsnetzwerke. Basiert auf der Microsoft-.NET-Technologie.

IndraWorks (IW) Safety Manager Erweiterung von IndraWorks ab 13VRS um Sicherheitsfunktionalität.

Technisch gesehen ist dies eine IndraWorks-Instanz mit geladenem Safety-Profil, das die Safety-Komponenten lädt.

E/A... Die Ein-/Ausgabe betreffend (englisch E/A für Input/Output.

E/A-Variable In der Sicherheitsprogrammierung werden verschiedene E/A-Aspekte durch Variablen repräsentiert.

Es gibt:

- E/A-Kanal-Variablen: Sie repräsentieren einen Eingangskanal (Eingangskanal-Variable) oder einen Ausgangskanal (Ausgangskanal-Variable) eines bestimmten E/A-Moduls.
- E/A-Stack-Variablen: Sie sind Instanzen von E/A-Stack-FBs und implementieren das Sicherheitsprotokoll und die API zu einem E/A-Modul. Die Input- und Output-Variablen der Instanz dienen zum Zugriff auf den Status des E/A-Moduls, nicht zum Zugriff auf seine E/A-Kanäle.
- E/A-Gruppen-Variablen sind Instanzen von E/A-Gruppen-Funktionsbausteinen und repräsentieren eine E/A-Gruppe als Ganzes. Die Eingangs- und Ausgangsvariablen der Instanz dienen zum Zugriff auf den Status der E/A-Gruppe, nicht zum Zugriff auf die E/A-Module in der Gruppe.

E/A-Modul Hardware-Modul mit Eingangs- und/oder Ausgangskanälen.

Es wird unterschieden zwischen:

- E/A-Modulen (auch als Klemmen oder "Scheibchen" bezeichnet) hinter einem Koppler, der Feldbus-Teilnehmer ist . Die Eingangs- bzw. Ausgangskanäle jeder Klemme hinter dem Koppler entsprechen einem Abschnitt im Prozessabbild des Kopplers.

- E/A-Modulen, die selbst Feldbus-Teilnehmer sind, z. B. der typische Antrieb. In diesem Fall bilden die Eingangs- bzw. Ausgangskanäle des E/A-Moduls das gesamte Prozessabbild des Feldbusteilnehmers.

I-Parameter Individuelle, sicherheitsgerichtete herstellerspezifische Geräteparameter eines Profisafe-F-Device.

Klemme andere Bezeichnung für **E/A-Modul** hinter einem Koppler.

Konfiguration Beschreibung des Systems aus Steuerungen und E/A-Modulen (sowohl Standard als auch sicherheitsgerichtet).

Festlegung der Projektstruktur.

Konsistenz *Eigenschaft eines Konstrukts im IEC-Code, dass Typen von Variablen zusammen passen oder Werte zum Typ passen:*

1. Eine Variable einer VAR_EXTERNAL-Deklaration ist konsistent, wenn es für sie eine gleiche Variable (namens- und typgleich, vom Attribut passend) in der VAR_GLOBAL-Deklaration gibt.
2. Das Konstrukt Initialisierung einer Variable mit einem Wert oder Zuweisung eines Wertes an eine Variable ist konsistent,
   - wenn Wert und Variable den gleichen Typ haben, oder
   - wenn der Wert einen Typ SAFE-X hat und die Variable den Typ X, oder
   - wenn der Wert den Typ INT hat und die Variable den Typ DINT, oder wenn der Wert den Typ SAFEINT hat und die Variable den Typ SAFEDINT oder DINT (implizite Typkonvertierung ohne Datenverlust).
3. Das Konstrukt FB-Aufruf (Box mit FB-Namen) ist typkonsistent, wenn der Name des FBs gleich dem Typ der darüber stehenden FB-Instanz ist und wenn die Zuweisungen der Eingangswerte auf Eingangsvariablen typkonsistent sind.
4. Die Typkonsistenz des Konstrukts Operatoraufruf (Box mit Operatornamen) hängt vom Operator ab:
   - Aufrufe einer Konvertierung (X_TO_Y) sind typkonsistent, wenn der Eingangswert den Typen X oder SAFE-X hat.
   - Aufrufe boolescher Operatoren (AND OR XOR NOT) sind typkonsistent, wenn die Eingangswerte die Typen BOOL oder SAFEBOOL haben.
   - Aufrufe mathematischer Operator (ADD SUB MUL DIV LT LE GT GE) sind typkonsistent, wenn die Eingangswerte die Typen INT, DINT, SAFEINT oder SAFEDINT haben; darüber hinaus ist ein Aufruf von MUL und DIV auch dann typkonsistent, wenn der erste Eingangswert den Typ TIME oder SAFETIME hat und nur die anderen Eingangswerte den Typ INT, DINT, SAFEINT oder SAFEDINT haben.
   - Aufrufe der allgemeinen Vergleichsoperatoren (EQ NE) sind typkonsistent, wenn die Eingangswerte untereinander typkonsistent sind (siehe unten). Aufrufe des Auswahloperators SEL sind typkonsistent, wenn der erste Eingangswerte den Typ BOOL oder SAFEBOOL hat, und die anderen Eingangswerte untereinander typkonsistent sind.
   - Aufrufe des Auswahloperators MUX sind typkonsistent, wenn der erste Eingangswerte den Typ INT, DINT, SAFEINT oder SAFEDINT hat und die anderen Eingangswerte untereinander typkonsis-

tent sind. Mehrere Eingangswerte sind untereinander typkonsistent, a. wenn sie den gleichen Typ haben, oder b. wenn sie teils den Typ T, teils den Typ SAFE-T haben, oder c. wenn sie die Typen INT, DINT, SAFEINT oder SAFEDINT haben.

5. Die Konstrukte bedingter Sprung und bedingtes Return sind typkonsistent, wenn der Eingangswert BOOL oder SAFEBOOL ist.

Koppler andere Bezeichnung für Slave am Feldbus (z. B. ein DP-Slave am Profibus), verbindet einzelne E/A-Module mit dem Feldbus.

Laufzeitsystem (LZS) Softwarekomponente der **SPS**, Das LZS ist als Teil der **Firmware** verantwortlich für die Implementierung der Steuerungslogik.

LZS-Software, synonym für Laufzeitsystem

Maschinenabnahme Die Abnahme einer Maschine oder Anlage bestehend aus Hardware und Software.

Aufbauend auf der Abnahme der **Sicherheitsapplikation** und der **Safety-Geräteparameter** sind dabei die Verdrahtung, die Standardapplikation (z. B. Reset) und die Gesamt-Reaktionszeiten der Sicherheitsfunktionen zu beurteilen.

Vgl. **Applikationsabnahme**

Objekt Ein Element der Projektstruktur, welches eine Strukturierungseinheit des Projekts repräsentiert.

OEM Original Equipment Manufacturer.

Hersteller, der sein Gerät zusammen mit Software von Bosch Rexroth als Gesamtsystem anbietet.

Im Gegensatz zu einem Endkunden, der Software von Bosch Rexroth direkt nutzt, verkauft ein OEM die Software lediglich weiter.

OEM-LZS, auf eine spezifische Sicherheitssteuerung angepasstes Sicherheitslaufzeitsystem auf Basis des Bosch Rexroth-LZS.

OSSD Output Signal Switching Device

Sicherheitsrelevantes Ausgangsschaltelement einer berührungslos wirkenden Schutzeinrichtung, das mit der Maschinensteuerung verbunden ist und das in den AUS-Zustand übergeht, wenn der Sensorteil während des bestimmungsgemäßen Betriebes anspricht.

Parametrierung Konfiguration der Parameter des Bussystems und der E/A-Module.

Pinnen ... einer **Sicherheitsapplikation**

(Begriff aus VisualSourceSafe, deutsch "festhalten").

Der aktuelle Stand der Applikation wird "festgehalten", Änderung an der Applikation ist zwar noch möglich - so dass der aktuelle Stand im Projekt vom gepinnten Stand abweichen kann - aber die abweichenden Objekte, sowie der Ausgangsstand (Pin) für die Abweichung, können identifiziert werden.

Pinnen ist kombiniert mit der Vergabe einer Bezeichnung für diesen Stand.

PLCopen Die PLCopen ist ein firmen- und produktunabhängiger weltweiter Interessenverband zur Verbreitung der Norm **IEC 61131-3**.

PLCopen TC5 Technisches Komitee TC5 der PLCopen, das Standards für die Programmierung von sicherheitsgerichteten Steuerungen definiert.

Profibus-DP Process Field Bus mit Decentralized Peripherals.

Feldbussystem zum Anschluss von E/A-Modulen (entweder direkt als Profibus-Teilnehmer oder als Klemme unter einem Profibus-Teilnehmer als Koppler).

Profisafe Protokoll zur sicherheitsgerichteten Kommunikation auf Profibus DP und Profinet nach IEC 61783-3.

Überwachungszeit Allgemein: Maximale Zeitdauer, die für eine fehlerfreie Kommunikation zwischen zwei Teilnehmern zur Verfügung steht.

Hier: Profisafe-Überwachungszeit **F_WD_Time**, Überwachungszeit für die sicherheitsgerichtete Kommunikation zwischen F-Host und F-Device.

Programm (PRG) Programmorganisationseinheit.

Programmierlevel Es gibt drei verschiedene Programmierlevel für unterschiedliche Sicherheitseinschränkungen während der Entwicklung: **Basic-Level**, **Extended-Level** und **System-Level**.

Basic- und Extended Level sind von der PLCopen bzw. den entsprechenden Spezifikationen des IndraWorks Safety Managers definiert.

Programmierschnittstelle Programmierschnittstelle (API) - Schnittstelle zwischen Softwarebausteinen:

Ein Baustein exportiert eine Schnittstelle. Bausteine, die diese Schnittstelle verwenden wollen, werden im Sinne dieser Schnittstelle programmiert

Im Fall eines **Funktionsbausteins** der IEC 61131-3 besteht die exportierte Programmierschnittstelle aus seinen Eingabe- und Ausgabe-Variablen. Zu unterscheiden von **Bedienschnittstelle**.

Programmiersystem Entwicklungsumgebung auf einem Windows-PC für das Erstellen von Programmen nach **IEC 61131-3** sowie softwaretechnischer Zugang zu Steuerungen mit IndraWorks-LZS.

Programmorganisationseinheit Objekt in der Projektstruktur, welches Software repräsentiert (und nicht Geräte/Module) (POU, POE)

Projekt Ein Projekt kann eine oder mehrere Steuerungen mit ihren untergeordneten Objekten beinhalten.

Dies können Sicherheits- und/oder Standard-Steuerungen sein, die eine Anlage oder Maschine steuern. Wird in IndraWorks 15VRS.x in der **Projektdatei** gespeichert.

Projektdatei Datei mit den Daten des Projekts, aber ohne Bibliotheken und Gerätebeschreibungen (xxx.project).

Prüfsumme Prüfsummen (**CRC**) dienen der Absicherung übertragener oder gespeicherter Daten.

Sie werden über einen Datenstrom, mittels einer dafür geeigneten Methode berechnet.

Die Methode ist parametriert durch einen Startwert und ein Polynom.

PS Programmiersystem.

Quellcodeverwaltung Externes Programm zur Verwaltung von Objektdaten außerhalb von IndraWorks. Diese können neben den eigentlichen, für den Anwender sichtbaren Quelldaten auch Daten zur Verwaltung dieser Quellen innerhalb von IndraWorks enthalten.

Synonym mit dem Begriff Versionsverwaltung verwendet.

Rückwirkungsfreiheit im Kontext der IndraWorks Safety Manager-Dokumentation die Fähigkeit, mögliche Rückwirkungen von sicherheitsgerichteten Softwarefunktionen zu erkennen und zu beherrschen.

SafeLogic Produktname für das sicherheitsgerichtete Laufzeitsystem (LZS) von Bosch Rexroth.

Safety-DCC Sicherheits-DDC (Device Description Converter), überführt eine Gerätebeschreibung in ein generisch abgesichertes Format (**SDD**).

Safety-Geräteparameter Parametersatz, der in ein Gerät geladen und dort verarbeitet wird. Im Profisafe-Kontext als **I-Parameter** bekannt.

Safety-Identifikation (eines E/A-Moduls in der Sicherheitsprogrammierung) bezeichnet innerhalb eines Projekts die Kombination aus eindeutigem Protokolltyp und der eindeutigen, protokollspezifischen Identifizierung.

Safety-Konfigurationsdaten Zusammenfassender Begriff für diejenigen Konfigurationsdaten eines E/A-Moduls, die für die Sicherheitsapplikation bzw. Sicherheitssteuerung relevant sind.

Die Safety-Geräteparameter werden nicht als Safety-Konfigurationsdaten bezeichnet.

SAFExxx Familie von sicherheitsgerichteten Datentypen mit Präfix SAFE gemäß PLCopen. z. B. SAFEBOOL, SAFEINT.

SDD Safety Devicefacet Description. **Prüfsumme**-gesicherte Gerätebeschreibung für sicherheitsgerichtete Geräteinformationen

Sicheres E/A-Modul ein **E/A-Modul**, das ein sicheres E/A-Protokoll implementiert und in der Sicherheitsprogrammierung verwendet werden kann.

In einer Sicherheitsapplikation sind die Nutzdaten der E/A-Kanäle von sicheren E/A-Modulen von einem der Datentypen **SAFExxx**.

Sicheres E/A-Protokoll Protokoll zur Verwendung sicherer E/A-Telegramme über Standard-Feldbusse, so dass eine eindeutige Identifikation des E/A-Moduls möglich ist (z. B. Profisafe Vx.y).

Sicheres E/A-Telegramm (z. B. Profisafe-Telegramm), die Kodierung der Werte der E/A-Kanäle eines einzelnen, sicheren E/A-Moduls zusammen mit Zusatzinformation als Abschnitt in das Prozessabbild eines Standard-Feldbus-Teilnehmers (z. B. ein Profibus-DP oder Profinet-Koppler).

Sicherheitsapplikation auch Safety-Applikation.

Sicherheitsgerichtete **Applikation** zur Ausführung auf einer **sicherheitsgerichteten** Steuerung.

Sicherheitsgerichtet (en: fail-safe), Fähigkeit eines Systems, beim Auftreten eines Ausfalls im sicheren Zustand zu bleiben bzw. unmittelbar in den sicheren Zustand überzugehen.

Sicherheitsgerichtete Firmware Firmware für den Betrieb eines sicherheitsgerichteten Laufzeitsystems. Besteht aus dem Safety-Funktionsmodul und betriebssystem-ähnlichen Basisfunktionen.

Sicherheitsgerichtetes Laufzeitsystem **Laufzeitsystem** auf der Sicherheitssteuerung zur Ausführung von Sicherheitsapplikationen und Unterstützung von Sicherheitsprogrammierung

Sicherheitsgerichtetes Objekt Objekt im **Sicherheits-PS**, das eine Sicherheitssteuerung, ein **sicheres E/A-Modul** oder eine Einheit einer Sicherheitsapplikation repräsentiert (die beiden Erstgenannten sind **Sicherheitsgerichtete Geräte**, letztere sind sicherheitsgerichtete Logikobjekte).

Sicherheitsgerichtetes Programmiersystem Programmiersystem auf einem Arbeitsplatzrechner (Windows-PC) für die Sicherheitsprogrammierung

Sicherheitssteuerung Eine **SPS**, die (üblicherweise) durch spezielle Hardware-Erweiterungen (z. B. zwei Prozessoren mit gegenseitiger Prozessbeobachtung) und entsprechend angepasster Software eine erhöhte Fehlersicherheit bietet.

Sicherheits-PS, Sicherheitsprogrammierung Gesamtheit der Aktivitäten zur Erstellung von Sicherheitsapplikationen; Programmierung von **Sicherheitsgerichteten** Steuerungen.

S-SPS Speicherprogrammierbare Steuerung (**SPS**) (en: PLC) bezeichnet ein Gerät nach IEC 61131, das mittels spezieller Programmiersoftware zur automatisierten Steuerung von industriellen Maschinen und Anlagen verwendet wird

Hier: **S-SPS**: sicherheitsgerichtete SPS

Stack-Instanz Instanz des Protokollstacks eines sicheren Feldbusses (z. B. Profisafe, CSos) für die Kommunikation mit einem Gerät dieses Feldbusses.

Technisch gesehen handelt es sich um eine vom logischen Gerät implizit erzeugte Instanz des Funktionsbausteins, der den Protokollstack implementiert.

Standard-... *Präfix:*

- als Gegensatz zu **sicherheitsgerichtet**
- zu sicherheitsspezifisch (safety-spezifisch), z. B. Standardsoftware, Standardkomponente, Standardansicht.

Standard-IndraLogic IndraWorks ohne Zusätze, wie es ausgeliefert wird. Technisch gesehen eine IndraWorks-Instanz, in derem Profil insbesondere die Safety-Erweiterung NICHT geladen wird.

Standardapplikation **Applikation** zur Ausführung im nichtsicherheitsgerichteten Teil der Steuerung.

System-Fehlerreaktion Reaktion des Sicherheitslaufzeitsystems auf einen Systemfehler.

Task *Zeitliche Ablaufeinheit einer Software:*

- IEC-Tasks: In der IEC-Programmierung bezeichnet "Task" eine zeitliche Ablaufeinheit einer IEC-Applikation.

  Die nicht sicherheitsgerichteten Tasks werden unter **... ▸ Logic ▸ Application ▸ Taskkonfiguration** verwaltet.
- **Safety Task**: In einer sicherheitsgerichteten Applikation **SafetyApp** kann nur eine Task konfiguriert werden.
- **Laufzeitsystemtask (LZS-Task)**: Das Laufzeitsystem enthält Tasks zur Ausführung der konfigurierten IEC-Task(s) und zur Ausführung anderer Aufgaben.

  Alle im Laufzeitsystem enthaltenen Tasks werden vom OEM bereitgestellt, da das Laufzeitsystem selbst keine Taskverwaltung beinhaltet.

  Das Laufzeitsystem stellt die Einsprungspunkte für die benötigten Tasks bereit, führt die vorgesehenen Aufgaben im Kontext der jeweiligen Task aus und kehrt danach zurück.

Innerhalb einer Task wird die Abarbeitungsreihenfolge der Programme (und Funktionsbausteininstanzen) durch deren Reihenfolge im Taskobjekt, Registerblatt Konfiguration, POUs festgelegt.

Taskkonfiguration Im Objekt Taskkonfiguration der **Standardapplikation** erfolgt die Verwaltung und Überwachung der nichtsicherheitsgerichteten Tasks. Die Reihenfolge der Abarbeitung der einzelnen zur Taskkonfiguration gehörenden Tasks wird durch deren Priorität und durch ihren Typ festgelegt.

In der **Sicherheitsapplikation** existiert nur eine Task (Safety Task). Sie benötigt kein zusätzliches Verwaltungsobjekt "Taskkonfiguration".

TDC Top Dead Center

oberer Totpunkt des Exenters einer Presse

THC Two Hand Control

Zweihandsteuerung: Der Bediener muss mit beiden Händen Sicherheitsschalter bedienen, um sicherzustellen, dass sich keine Hand im Arbeitsraum befindet.

Typen *In der Sicherheitsprogrammierung werden folgende Unterarten unterschieden:*

- B-Typen: Namen von Funktionsbausteinen als Typen von Variablen (FB-Instanzen)
- Boolesche Typen: BOOL, SAFEBOOL (für Wahrheits- und Bitwerte)

- Numerische Typen: INT, DINT, SAFEINT, SAFEDINT (für Faktoren und Divisoren)
- Bitfolgen: WORD, BYTE, DWORD, SAFEWORD, SAFEBYTE, SAFEDWORD
- Numerische und Zeitdatentypen: INT, DINT, SAFEINT, SAFEDINT, TIME, SAFETIME (für Operanden arithmetischer Operationen)

Validierung Überprüfung, ob das fertig erzeugte, verifizierte Produkt den gestellten Anforderung genügt.

Verifikation Mit dem Prozess der Software-Verifikation wird nachgewiesen, dass die Spezifikation und die Programmierrichtlinien erfüllt und damit verifiziert sind. Die Software-Verifikation erfolgt durch die Kombination von statischen und dynamischen Verifikationsmethoden.

Differenzverifikation: Erleichterte Verifikation einer Applikation, die eine Änderung einer bereits verifizierten Applikation ist.

Wiederanlauf Bei Not-Aus-Anwendungen darf der Wiederanlauf nach Not-Aus nur bestimmungsgemäß unter Kontrolle der Sicherheitssteuerung erfolgen.

Zweikanaliger Online-Dienst Ein Online-Dienst, der auf beiden Laufzeitsystemkanälen (LZS-Kanälen), jedoch nicht synchron zueinander, im Kontext der Hintergrund-Task ausgeführt wird.

Zweikanalig synchroner Online-Dienst Ein Online-Dienst, der auf beiden Laufzeitsystemkanälen (LZS-Kanälen), synchron im Kontext der SSC-Task ausgeführt wird.

Zweiterzeugung ... einer Bootapplikation

Eine Applikation im Projekt, aus der schon einmal eine Bootapplikation auf einer SPS erzeugt wurde (insbesondere für Tests), wird unverändert auf einer weiteren SPS zur Bootapplikation gemacht.

26 IEC 61131-3 Compliance

26.1 Zusammenfassung der Erweiterungen zur IEC 61131-3

Es gibt nur die folgenden Erweiterungen über die IEC hinaus:

Bereich	Erweiterungen der IEC	Begründung
Datentypen	sicherheitsgerichtete Datentypen **SAFEBOOL**, **SAFEINT**, etc. für alle unterstützen IEC Datentypen.	Für PLCopen: Datentyp zur Unterscheidung sicherer Signale
generische Datentypen / Funktionen	Jeder Datentyp SAFEX wird neben X in die Hierarchie der generischen Datentypen [N1.1.3- Tab.11] eingereiht. Generischen Funktionen (AND, ADD, SEL, EQ, etc.), die u.a. auf Typ X definiert sind, sind analog auch auf Typ SAFEX definiert.	Für PLCopen: Generische Funktionen auf Typ X sind genauso sinnvoll wie auf Typ SAFEX. Es müssen keinen neuen Funktionen „erfunden“ werden.
	Zusätzliche Varianten der Funktion „AND“, die nicht dem generischen Typschema entspricht: • AND: BOOL x SAFEBOOL → SAFEBOOL • AND: SAFEBOOL x BOOL → SAFEBOOL	Für PLCopen: „confirmation functionality“ bzw. „enabeling function“
	Konvertierungsfunktionen A_TO_B sind generisch bzgl. der SAFE-Qualifikation: • A_TO_B : A → B • A_TO_B: SAFEA → SAFEB	**Analogie zu generischen Funktionen:** ADD auf INT und auf TIME wird als eine Funktion verstanden, weil das gleiche passiert. Es erspart dem Anwender, unterschiedliche Funktionsnamen hinzuschreiben (ADD_INT, ADD_TIME). Auch bei INT_TO_BOOL auf INT/BOOL wie auf SAFEINT/SAFEBOOL passiert das gleiche. Es bringt nichts, zwischen INT_TO_BOOL und SAFEINT_TO_SAFEBOOL zu unterscheiden.
implizite Konvertierungen	Daten vom Typ SAFEX können auf Variablen oder Eingänge vom Typ X zugewiesen werden („SAFE- Polymorphie“). U.a. ist auf diese Weise der Aufruf der generischen Funktion ADD mit einem INT- und einem SAFEINT-Wert möglich. SAFEBOOL → BOOL SAFEINT → INT SAFEDINT → DINT SAFETIME → TIME SAFEBYTE → BYTE SAFEWORD → WORD SAFEDWORD → DWORD	Für PLCopen: SAFEX-Daten und X-Daten sind nicht andersartige Daten, sondern Daten mit unterschiedlicher Integrität. Daten höherer Integrität können verwendet werden, wo niedrigere Integrität ausreicht.
	Daten vom Typ INT bzw. SAFEINT können auf Variablen oder Eingänge vom Typ DINT bzw. SAFEDINT zugewiesen werden („INT-Polymorphie“) INT → DINT SAFEINT → SAFEDINT	Der Wertebereich von INT bzw. SAFEINT ist enthalten im Wertebereich von DINT bzw. SAFEDINT.

Bereich	Erweiterungen der IEC	Begründung
FB-Typen	Statt den Standard-Funktionsbausteinen: **SF_XXX** Varianten mit SAFE-Varianten der Inputs/Outputs (bis auf Inputs mit Reset-Semantik: RESET, LOAD bzw. IN)	notwendig für Verarbeitung sicherer Signale mit Standard-FBs
POEs	POEs können qualifiziert werden, um ihren Sprachumfang oder ihre Verwendung einzuschränken: • Level: Basic, Extended, External • Singlecall • IOAPI-only	**Level:** zur expliziten Unterscheidung der PLCopen Programmierlevel. **Singlecall:** Kennzeichnet FBs, für die PLCopen-Regel „einmaliger Aufruf“ gilt. **IOAPI-only:** reserviert FBs für Systemzwecke (E/A-Anbindung)
Variablendeklarationen	• **In externen FBs:** neuer Modifizierer **SYSONLY**. • **In implizitem Code:** neue Modifizierer **IOIN**, **IOOUT**, **IOAPI**, **SYSONLY**.	Für Safety-spezifische Variablenarten

26.2 Compliance Liste: Unterstützte Features

26.2.1 Compliance Liste: Unterstützte Features, Allgemeines

Das IEC Compliance Statement muss auflisten, welche in den Tabellen von [N1.1.3] genannten Features unterstützt werden [N1.1.3-Kap.1.5.1].

26.2.2 Compliance Statement: Common elements

Table No.	Feature No.	Feature description	Bemerkung
1	**2**	lower case characters	
	3a	number sign	
	4a	dollar sign	
	5a	vertical bar	
2	**2**	upper and lower case, numbers, embedded underlines	führende Underscores reserviert für impliziten Code, IndraLogic-Safety-Schlüsselwörter und Standard-IndraLogic-Schlüsselwörter reserviert, siehe 2.3
3	**-**		Kommentare implizit gekennzeichnet
3a	**-**		keine Pragmas
4	**1**	integer literals	der Typ der Literale ist SAFEINT bzw. SAFEDINT, je nach Größe, statt INT bzw. DINT (Begründung: Literalkonstanten sind Werte mit der höchst-möglichen Integrität)
	4	base 2 literals	
	6	base 16 literals	
	8	Boolean FALSE and TRUE	der Typ der Literale ist SAFEBOOL statt BOOL (Begründung: Literalkonstanten sind Werte mit der höchst-möglichen Integrität)
	9	typed literals	Typpräfixe nicht ausgeweitet auf die SAFE-Varianten. Literale mit Typpräfix X werden behandelt als Werte vom Typ SAFEX (Begründung: Literalkonstanten sind Werte mit der höchst-möglichen Integrität)

Table No.	Feature No.	Feature description	Bemerkung
5	-		keine Strings
6	-		keine Strings
7	1a	duration literals without unterlines: short prefix	kein langer Prefix time#1s
	2a	duration literals with unterlines: short prefix	kein langer Prefix time#1m_1s
8	-		kein Datentyp DATE, TOD
9	keine Featuretabelle		
10	1	BOOL	
	3	INT	
	4	DINT	
	12	TIME	
	17	BYTE	
	18	WORD	
	19	DWORD	
11	keine Featuretabelle		
12	-		keine TYPE Deklarationen
13	keine Featuretabelle		
14	-		keine TYPE Deklarationen
15	-		keine direkten Variablen
16	keine Featuretabelle		
17	5	automatic memory allocation of symbolic variables	
18	5	initialization of symbolic variables	
	9	initialization of constants	
19	-		keine graphische Input/Output-Negation ‚o'
19a	-		keine textuelle Sprache
20	-		kein EN/ENO
20a	1	in-out variable declaration (textual)	nur bei externen FBs, Aufruf nur impliziter Code
21	1	overloaded functions	

Table No.	Feature No.	Feature description	Bemerkung	
22	1	*_TO_**	Keine eigenen Konvertierungsfunktionen für SAFE-Typen, dafür ist jede Konvertierungsfunktion A_TO_B generisch bzgl. SAFE:	
			• A_TO_B: A → B • A_TO_B: SAFEA → SAFEB	
			Liste der Varianten [N1.1.3-Tab.22(a)]:	
			'BOOL_TO_INT'	'INT_TO_BOOL'
			'BOOL_TO_DINT'	'DINT_TO_BOOL'
			'BOOL_TO_TIME'	'TIME_TO_BOOL'
			'BOOL_TO_WORD'	'WORD_TO_BOOL'
			'INT_TO_DINT'	'DINT_TO_INT'*
			'INT_TO_TIME'*	'TIME_TO_INT'*
			'INT_TO_WORD'	'WORD_TO_INT'
			'DINT_TO_TIME'*	'TIME_TO_DINT'*
			'DINT_TO_WORD'*	'WORD_TO_DINT'
			'TIME_TO_WORD'*	'WORD_TO_TIME'
			'BYTE_TO_INT'	'INT_TO_BYTE'*
			'BYTE_TO_DINT'	'DINT_TO_BYTE'*
			'BYTE_TO_TIME'	'TIME_TO_BYTE'*
			'BYTE_TO_WORD'	'WORD_TO_BYTE'*
				'INT_TO_DWORD'
			'DWORD_TO_DINT'	'DINT_TO_DWORD'
			'DWORD_TO_TIME'	'TIME_TO_DWORD'
				'WORD_TO_DWORD'
			* führen ggf. zu Laufzeitfehlern (siehe Kap. 26.4.3 "Zusammenfassung Laufzeitfehler" auf Seite 703)	
23	-		kein ABS, LOG, SIN, etc.	
24	12n	ADD	vollständig überladen (ANY_NUM, dazu TIME gemäß #30)	
	13n	MUL	vollständig überladen (ANY_NUM, dazu TIME gemäß #30)	
	14n	SUB	vollständig überladen (ANY_NUM, dazu TIME gemäß #30)	
	15n	DIV	vollständig überladen (ANY_NUM, dazu TIME gemäß #30)	
25	-		keine Shift-Funktionen	
26	5n	AND (named representation)	nur für BOOL/SAFEBOOL statt für alle ANY_BIT-Typen	
	6n	OR (named representation)	nur für BOOL/SAFEBOOL statt für alle ANY_BIT-Typen	
	7n	XOR (named representation)	nur für BOOL/SAFEBOOL statt für alle ANY_BIT-Typen, nicht erweiterbar	
	8n	NOT (named representation)	nur für BOOL/SAFEBOOL statt für alle ANY_BIT-Typen	

Table No.	Feature No.	Feature description	Bemerkung
27	**1**	SEL	vollständig überladen (ANY)
	4	MUX	vollständig überladen (ANY_INT, ANY)
28	**5**n	GT (named representation)	nur für ANY_MAGNITUDE, nicht für ANY_BIT nur binär, nicht erweiterbar
	6n	GE (named representation)	nur für ANY_MAGNITUDE, nicht für ANY_BIT nur binär, nicht erweiterbar
	7n	EQ (named representation)	vollständig überladen (ANY) nur binär, nicht erweiterbar
	8n	LE (named representation)	nur für ANY_MAGNITUDE, nicht für ANY_BIT nur binär, nicht erweiterbar
	9n	LT (named representation)	nur für ANY_MAGNITUDE, nicht für ANY_BIT nur binär, nicht erweiterbar
	10n	NE (named representation)	vollständig überladen (ANY)
29	**-**		keine String Funktionen
30	**1a**n	ADD (named representation)	
	4an	SUB (named representation)	
	10an	MUL (named representation)	
	11an	DIV (named representation)	
31	**-**		keine Enum Typen
32	keine Featuretabelle		
33	**4a**	VAR_IN_OUT declaration	nur bei externen FBs
	7a	Function block instance name as external variable (textual)	
34	**1***	SR	(*) nur die Safetyvariante SF_SR
	2*	RS	(*) nur die Safetyvariante SF_RS
35	**1***	R_TRIG	(*) nur die Safetyvariante SF_R_TRIG
	2*	F_TRIG	(*) nur die Safetyvariante SF_F_TRIG
36	**1a***	CTU	(*) nur die Safetyvariante SF_CTU
	2a*	CTD	(*) nur die Safetyvariante SF_CTD
	3a*	CTUD	(*) nur die Safetyvariante SF_CTUD
37	**1***	TP	(*) nur die Safetyvariante SF_TP
	2a*	TON	(*) nur die Safetyvariante SF_TON
	3a*	TOF	(*) nur die Safetyvariante SF_TOF
38	keine Featuretabelle		
39	**22a**	VAR_EXTERNAL declarations within PROGRAM type declarations	
	22b	VAR_EXTERNAL CONSTANT declarations within PROGRAM type declarations	

Table No.	Feature No.	Feature description	Bemerkung
40..46	-		kein SFC
47	keine Featuretabelle		
48	keine Featuretabelle		
49	3	RESOURCE … ON … END_RESOURCE construction	Die Applikation A unter Steuerung S entspricht „RESOURCE A ON S“.
	4	VAR_GLOBAL … END_VAR construction within RESOURCE	IndraLogic-GVL entspricht VAR_GLOBAL auf RESOURCE-Ebene in IEC 61131
	5a	Periodic TASK construction	INTERVAL über Eingabefeld konfiguriert
	6a	WITH construction for PROGRAM to TASK association	Program P in Task T entspricht „PROGRAM P WITH T;“
	12a	VAR_GLOBAL CONSTANT … END_VAR construction within RESOURCE	IndraLogic-GVL entspricht VAR_GLOBAL auf RESOURCE-Ebene in IEC 61131
50	1a	Textual declaration of periodic TASK	Tabellarische Konfiguration entspricht eher textueller Deklaration als grafischer
	3a	Textual association with PROGRAMs	Tabellarische Konfiguration entspricht eher textueller Deklaration als grafischer

Keine der Features aus Tabellen 51a bis 56 (Common textual elements, IL language elements, ST language elements) werden unterstützt, da nur FBD unterstützt wird.

26.2.3 Compliance Statement: Common graphical elements

Table No.	Feature No.	Feature description	Bemerkung
57	2	Horizontal lines: graphic or semigraphic	normale Verbindung
	4	Vertical lines: graphic or semigraphic	wenn Verbindungen beim automatischen Layout abgeknickt werden
	6	Horizontal/vertical connection	bei Mehrfachzuweisung
	12	Block with connected lines: graphic or semigraphic	Aufruf Funktion/FB
58	1	Unconditional Jump: FBD language	
	3	Conditional Jump: FBD language	
	6	Conditional Return: FBD language	

Keine der Features aus Tabellen 59 bis 62 (LD language elements) werden unterstützt, da nur FBD unterstützt wird.

Für FBD definiert [N1.1.3] keine Feature-Tabellen (FBD language elements).

26.2.4 Liste zusätzlich reservierter Schlüsselwörter

Die zusätzlichen Schlüsselworte und vorsorglich reservierten Worte von IndraLogic-Safety sind gelb hinterlegt.

Bei den zusätzlichen Schlüsselworten von Standard-IndraLogic wird unterschieden:

* die in der bevorstehenden Ausgabe 3 der IEC 61131-3 vorgesehen Schlüsselwörter sind nicht hervorgehoben;
* die Standard-IEC-Operatoren, die spezifisch für die Sprache IL sind (und von Standard-IndraLogic über alle Sprachen hinweg reserviert werden) sind grau hinterlegt
* die darüber hinausgehenden Schlüsselworte sind grün hinterlegt

Deklarationen, Typkonstrukte	Datentypen		Implementierungsteil	
'ABSTRACT' 'CLASS' , 'END_CLASS' 'EXTENDS' 'FINAL' 'IMPLEMENTS' 'INTERFACE' , 'END_INTERFACE' 'INTERNAL' 'METHOD' , 'END_METHOD' 'NAMESPACE' , 'END_NAMESPACE' 'OVERLAP' 'OVERRIDE' 'PERSISTENT' ''PROTECTED' 'PUBLIC' 'REF_TO'	'BIT'		'CONTINUE' 'NULL' 'SUPER' 'THIS'	**Konvertierungen von oder zu neuen Basis-datentypen:** 'BIT_TO_XXX' 'XXX_TO_BIT'
	'CHAR' 'LDT' 'LTIME' 'LTOD' 'WCHAR'			'CHAR_TO_XXX' 'LDT_TO_XXX' 'LTIME_TO_XXX' 'LTOD_XXX' 'WCHAR_TO_XXX' 'XXX_TO_CHAR' 'XXX_TO_LDT' 'XXX_TO_LTIME' 'XXX_TO_LTOD' 'XXX_TO_WCHAR', 'REF'
'POINTER' 'PRIVATE' 'PROPERTY' 'UNION' , 'END_UNION' 'VAR_STAT'	'SAFEBIT' 'SAFEBYTE' 'SAFEDATE' 'SF_CTD' 'SF_CTD_DINT' 'SF_CTD_LINT' 'SF_CTD_UDINT' 'SF_CTD_ULINT' 'SF_CTU' 'SF_CTU_DINT' 'SF_CTU_LINT' 'SF_CTU_UDINT' 'SF_CTU_ULINT' 'SF_CTUD' 'SF_CTUD_DINT' 'SF_CTUD_LINT' 'SF_CTUD_UDINT' 'SF_CTUD_ULINT' 'SF_SR' , 'SF_RS' 'SF_F_TRING' 'SF_R_TRING' 'SF_TOF' , 'SF_TON' 'SF_TP'	'SAFEBOOL' 'SAFECHAR' 'SAFEDA-TE_AND_TIME' 'SAFEDINT' 'SAFEDT' 'SAFEDWORD' 'SAFELDT' 'SAFELINT' 'SAFELREAL' 'SAFELTIME' 'SAFELTOD' 'SAFELWORD' 'SAFEREAL' 'SAFESINT' 'SAFESTRING' 'SAFETIME' 'SAFEUDINT' 'SAFEUINT' 'SAFEULINT' 'SAFEUSINT' 'SAFETIME' 'SAFEWCHAR' 'SAFEWORD' 'SAFEWSTRING'	'ADR' 'BITADR' 'INDEXOF' 'INI' 'SIZEOF' 'TEST_AND_SET'	**Typisierung der Stan-dardfunktionen mit neuen Basisdatenty-pen:** 'ADD_BIT'
'IOAPI' 'IOIN' 'IOOUT' 'SYSONLY'			'ANDN' 'CAL' , 'CALC', 'CALCN' 'JMP' ,'JMPC', 'JMPCN' 'LD', 'LDN' 'ORN' 'RET' , 'RETC' , 'RETCN' 'S', 'R' 'ST' , 'STN' 'XORN'	'ADD_CHAR' 'ADD_LDT' 'ADD_LTIME' 'ADD_LTOD' 'ADD_WCHAR' u.s.w.

26.3 Compliance Liste: Implementationsabhängige Parameter

Dieser Abschnitt listet die Werte der implementationsabhängigen Parameter aus [N1.1.3-Annex D] auf [N1.1.3-Kap.1.5.1.b].

„**n.a.**“ kennzeichnet Fälle, wo das Sprachkonstrukt, auf welches sich der Parameter bezieht, nicht zum Safety-Sprachumfang gehört.

Parameters	Implementierung
Maximum length of identifiers	1015 Bytes (UTF-8)
Maximum comment length	1015 Bytes (UTF-8)
Syntax and semantics of pragmas	gibt es nicht
Syntax and semantics for the use of the double-quote character when a particular implementation supports feature 4 but not feature 2 of table 5.	gibt es nicht
Range of values and precision of representation for variables of type TIME, DATE, TIME_OF_DAY and DATE_AND_TIME Precision of representation of seconds in types TIME, TIME_OF_DAY and DATE_AND_TIME	TIME-Werte gehen von 0 Sekunden bis 2^{32}-1 Millisekunden mit einer Präzision von 1 Millisekunde. Sekunden werden millisekundengenau repräsentiert.
Maximum number of enumerated values Maximum number of array subscripts Maximum array size Maximum number of structure elements Maximum structure size Maximum range of subscript values Maximum number of levels of nested structures	**n.a.** (keine Aufzählungstypen, keine Arrays, keine Strukturen, keine Bereichstypen)
Default maximum length of STRING and WSTRING variables Maximum allowed length of STRING and WSTRING variables	**n.a.** (keine Strings)
Maximum number of hierarchical levels Logical or physical mapping	**n.a.**
Initialization of system inputs	systemspezifisch (OEM)
Maximum number of variables per declaration	Keine Beschränkung
Effect of using AT qualifier in declaration of function block instances	**n.a.** (kein AT)
Warm start behavior if variable is declared as neither RETAIN nor NON_RETAIN	Alle Variablen sind NON_RETAIN
Information to determine execution times of program organization units	systemspezifisch (OEM)
Values of outputs when ENO is FALSE	**n.a.** (kein ENO)
Maximum number of function specifications	**n.a.** (keine anwenderdefinierten Funktionen)
Maximum number of inputs of extensible functions	Keine Beschränkung

Parameters	Implementierung
Effects of type conversions on accuracy Error conditions during type conversions	**Fehlerursache:** Fehler zur Laufzeit bei verschiedenen Konvertierungsfunktionen, wenn Ausgangswert ist nicht im Wertebereich des Zieltyps (siehe Kap. 26.4.3 "Zusammenfassung Laufzeitfehler" auf Seite 703) **Auswirkung:** Konvertierung nach BOOL ergibt TRUE gdw. Ausgangswert ungleich 0. Nummerische Werte bleiben unverändert. Bitmuster werden um führende 0-Bits erweitert bzw. gekürzt.
Accuracy of numerical functions	*Berechnungen erfolgen mit 32bit interner Genauigkeit für Zwischenwerte:* • Bei Overflow (nach oben wie nach unten) erfolgt ein Vorzeichenwechsel. • Bei Underflow (im Fall von Division) wird zur nächstgrößeren oder -kleineren Ganzzahl gerundet: systemspezifisch (OEM)
Effects of type conversions between time data types and other data types not defined in table 30	Bei Konvertierungen wird ein TIME-Wert als die Anzahl seiner Millisekunden behandelt. Bsp: TIME_TO_INT(t#1s) = 1000 INT_TO_TIME(1) = t#1ms
Maximum number of function block specifications and instantiations	Keine Beschränkung
Function block input variable assignment when EN is FALSE	**n.a.** (kein EN)
Pvmin, Pvmax of counters	Pvmin = 0, Pvmax = 16#7fff
Effect of a change in the value of a PT input during a timing operation	Erhöhung verlängert die Zeit bis zum Ablauf des Timer entsprechend. Erniedrigung verkürzt die Zeit. Wenn das verkürzte PT bereits verstrichen ist, läuft der Timer sofort ab.
Program size limitations	systemspezifisch (OEM)
Precision of step elapsed time Maximum number of steps per SFC	**n.a.** (kein SFC)
Maximum number of transitions per SFC and per step	**n.a.** (kein SFC)
Maximum number of action blocks per step	**n.a.** (kein SFC)
Access to the functional equivalent of the Q or A outputs	**n.a.** (kein SFC)
Transition clearing time Maximum width of diverge/converge constructs	**n.a.** (kein SFC)
Contents of RESOURCE libraries	**n.a.**
Effect of using READ_WRITE access to function block outputs	**n.a.**
Maximum number of tasks	1
Task interval resolution	systemspezifisch (OEM)
Maximum length of expressions	**n.a.** (kein ST)

Parameters	Implementierung
Maximum length of statements	**n.a.** (kein ST)
Maximum number of CASE selections	**n.a.** (kein ST)
Value of control variable upon termination of FOR loop	**n.a.** (kein ST)
Restrictions on network topology	Die Grundstruktur eines FBD-Netzwerks ist ein Baum: Die Wurzel liegt rechts und fächert sich nach links auf, nur Mehrfachzuweisungen fächern sich nach rechts auf. *Einschränkung für den Anwender:* • Keine expliziten Feedback-Schleifen. • Von einem FB-Aufruf "f" kann er auf grafische Weise nur einen Ausgang "o1" (beliebig wählbar) auf den Eingang eines anderen Auftrufs weiterverschalten. Weitere Ausgänge "o2" kann er nur textuell weiterverschalten, indem er "f.o2" auf den Eingang legt. • Den ausgewählten Ausgang "o1" eines FB-Aufrufs "f" kann er auf grafische Weise nur einmal auf den Eingang eines anderen Aufrufs schalten. "o1" kann er nur textuell auf einen weiteren Eingang (des gleichen oder eines anderen Auftrufs) schalten, indem er "f.o1" auf den Eingang legt. • Den Ausgang eines Operator-Aufrufs kann er nur einmal (grafisch) auf den Eingang eines anderen Aufrufs weiterverschalten. Um das Operator-Ergebnis mehrfach zu verwenden, müsste der Anwender eine Zwischenvariable "x" einführen und den Ausgang des Operator-Aufrufs von der grafischen Weiterverschaltung abzweigen, um ihn auf "x" zuzuweisen, und dann "x" auf den weiteren Eingang (des gleichen oder eines anderen Auftrufs) legen.
Evaluation order of feedback loops	Explizite Feedback-Loops nicht möglich, Implizite Feedback-Loops in Basic-Level verboten

26.4 Compliance Liste: Fehlerbedingungen

26.4.1 Compliance Liste: Fehlerbedingungen, Allgemeines

Die Fehlerbedingungen in der Tabelle E.1 von Annex E [N1.1.3-Tab E.1] sind auf verschiedene Arten abgedeckt:

- Die Fehlerbedingung wird beim Übersetzen oder zur Laufzeit geprüft.
- Die Fehlerbedingung bezieht sich auf Konstrukte, die gar nicht im Safety-Sprachumfang enthalten sind („**n.a.**“), was beim Editieren oder beim Übersetzen geprüft wird.
- Nicht diagnostiziert wird der Fehler, dass das Ergebnis numerischer Operationen [N1.1.3- Kap.2.5.1.5.2] und Zeit-Operationen [N1.1.3-Kap. 2.5.1.5.6] außerhalb des Datenbereichs liegt. Wie es [N1.1.3-Kap.1.5.1 d] fordert, wird darauf in einem eigenen Abschnitt hingewiesen, und es werden die Fälle aufgelistet siehe Kap. 26.4.3 "Zusammenfassung Laufzeitfehler" auf Seite 703.

Fehlerbedingungen	Lösung
Nested comments	**n.a.** (Kommentarfelder statt Kommentarzeichen)
Ambiguous enumerated value	**n.a.** (kein Enum)
Value of a variable exceeds the specified subrange	**n.a.** (kein Subrange)
Missing configuration of an incomplete address specification ("*" notation)	**n.a.** (kein *)
Attempt by a program organization unit to modify a variable which has been declared CONSTANT	Buildfehler
Declaration of a variable as VAR_GLOBAL CONSTANT in a containing element having a contained element in which the same variable is declared VAR_EXTERNAL without the CONSTANT qualifier.	Buildfehler
Improper use of directly represented or external variables in functions	**n.a.** (keine Funktionen)
A VAR_IN_OUT variable is not “properly mapped”	**n.a.** (nur in implizitem Code)
Ambiguous value caused by a VAR_IN_OUT connection	**n.a.** (nur in implizitem Code)
Type conversion errors	Laufzeitfehler
Numerical result exceeds range for data type	nicht erkannt
Division by zero	Laufzeitfehler
N input is less than zero in a bit-shift function	**n.a.** (kein Shift)
Mixed input data types to a selection function	Buildfehler
Selector (K) out of range for MUX function	Laufzeitfehler
Invalid character position specified Result exceeds maximum string length ANY_INT input is less than zero in a string function	**n.a.** (keine Strings)
bei TIME-Funktionen: Result exceeds range for data type	nicht erkannt
No value specified for a function block instance used as input variable	**n.a.** (kein FB als Input)
No value specified for an in-out variable	**n.a.** (nur impliziter Code)
Zero or more than one initial steps in SFC network User program attempts to modify step state or time	**n.a.** (kein SFC)
Side effects in evaluation of transition condition	**n.a.** (kein SFC)
Action control contention error	**n.a.** (kein SFC)
Simultaneously true, non-prioritized transitions in a selection divergence Unsafe or unreachable SFC	**n.a.** (kein SFC)
Data type conflict in VAR_ACCESS	**n.a.** (kein VAR_ACCESS)
A task fails to be scheduled or to meet its execution deadline	Laufzeitfehler
Bei Instruction List: Numerical result exceeds range for data type Current result and operand not of same data type	**n.a.** (kein IL)
Bei ST: Division by zero	**n.a.** (kein ST)
Bei ST: Numerical result exceeds range for data type	**n.a.** (kein ST)
Bei ST: Invalid data type for operation	**n.a.** (kein ST)

Fehlerbedingungen	Lösung
Return from function without value assigned	**n.a.** (keine Funktionen)
Bei While und Repeat: Iteration fails to terminate	**n.a.** (kein ST)
Same identifier used as connector label and element name	**n.a.** (kein Connector)
Uninitialized feedback variable	Buildfehler

26.4.2 Compliance Liste: Nicht erkannte Fehler

Dieser Abschnitt listet in einem eigenen Abschnitt die Fehler aus [N1.1.3-Annex E] auf, welche nicht erkannt bzw. nicht berichtet werden [N1.1.3-Kap. 1.5.1.d].

Level	Error conditions
Extended	Numerical result exceeds range for data type
Extended	bei TIME-Funktionen: Result exceeds range for data type

26.4.3 Zusammenfassung Laufzeitfehler

Level	Sprachfeature	Laufzeitfehler bei
Extended	DIV	Division durch 0
Extended	MUX	Aufruf mit erstem Input mit negativem Wert oder mit Wert N größer als die Anzahl der Eingänge minus 1 z. B. MUX(2, 16#8000, 16#8001)
Extended	DINT_TO_INT, DINT_TO_TIME, DINT_TO_WORD, DINT_TO_BYTE, INT_TO_BYTE, INT_TO_TIME, TIME_TO_BYTE, TIME_TO_DINT, TIME_TO_INT, TIME_TO_WORD, WORD_TO_BYTE	**Ausgangswert ist nicht im Wertebereich des Zieltyps:** Bei Konvertierung zwischen zwei ANY_MAGNITUDE-Typen muss der numerische Ausgangswert im Wertebereich des Zieltyps liegen (wobei TIME-Werte als Anzahl von Millisekunden gerechnet werden). **Bei Konvertierung von/zu Bitstring-Typen muss das Bitmuster des Ausgangswerts ein Bitmuster der Zieltyps sein:** *Beispiele:* • DINT_TO_INT(16#0000FFFF), weil 2^{16}-1 kein INT-Wert ist, • DINT_TO_TIME(-1), weil es keine negative TIME-Werte gibt • TIME_TO_DINT(t#365d), weil 365 Tage = 3,153,600,000ms =16#BBF81E00 ist, und damit größer als die größte DINT-Zahl 2^{31}-1 = 16#7FFFFFFF • INT_TO_BYTE(-1), da BYTE nur 0 bis 255 umfasst, WORD_TO_BYTE(0xFFFF), da BYTE nur bis 0xFF geht.

26.5 Literaturangabe

[N1.1.3] IEC 61131, 2003-1, Programmable controllers, Part 3: Programming languages

27 Service und Support

Für Ihre schnelle und optimale Unterstützung verfügen wir über ein dichtes weltweites Servicenetz. Unsere Experten stehen Ihnen mit Rat und Tat zur Seite. Sie erreichen uns täglich **rund um die Uhr - auch an Wochenenden und Feiertagen**.

Service Deutschland

Unser technologieorientiertes Competence Center in Lohr deckt alle Belange rund um den Service für elektrische Antriebe und Steuerungen ab.

Sie erreichen unsere **Service-Hotline** und unseren **Service-Helpdesk** unter:

Telefon:	**+49 9352 40 5060**
Fax:	**+49 9352 18 4941**
E-Mail:	service.svc@boschrexroth.de
Internet:	http://www.boschrexroth.com

Auf unseren Internetseiten finden Sie ergänzende Hinweise zu Service, Reparatur (z. B. Anlieferadressen) und Training.

Service weltweit

Außerhalb Deutschlands nehmen Sie bitte zuerst Kontakt mit Ihrem Ansprechpartner auf. Die Hotline-Rufnummern entnehmen Sie bitte den Vertriebsadressen im Internet.

Vorbereitung der Informationen

Wir können Ihnen schnell und effizient helfen, wenn Sie folgende Informationen bereithalten:

- Eine detaillierte Beschreibung der Störung und der Umstände
- Angaben auf dem Typenschild der betreffenden Produkte, insbesondere Typenschlüssel und Seriennummern
- Ihre Kontaktdaten (Telefon-, Faxnummer und E-Mail-Adresse)

28 Abnahmeprotokolle

28.1 Deaktivieren sicherer Teilnehmer im SafeLogic-Programm

Siehe auch Abnahme des Safety-Programms auf Seite 160.

Checkliste zur Abnahme der IOConfig-Funktionalität des Safety-Programms

Maschinentyp:			
Safety-Applikation (Pin)			
Name:		Prüfsumme:	
Revision:		Letzte Änderung:	
Bearbeiter:		Prüfer:	
Bemerkung:			

Nr.	Anforderung	Ja	Bemerkung
1	Wird genau eine Instanz des Funktionsbausteins "SF_IOConfigCntl" verwendet?		Statische Codeanalyse oder Meldung der Runtime
2	Sind der Schlüsselschalter und der Übernahmetaster direkt ohne Verknüpfungslogik mit dem Funktionsbaustein "SF_IOConfigCntl" verbunden?		Statische Codeanalyse
3	Hat jede "SF_IOConfig"-Instanz eine eineindeutige ID		Statische Codeanalyse oder Meldung der Runtime
4	Ist der "ID"-Eingang der "SF_IOConfig"-Instanzen mit einer Konstante belegt		Statische Codeanalyse
5	Besitzt jedes deaktivierbare Maschinenmodul mindestens eine Instanz des Funktionsbausteins "SF_IOConfig"?		Statische Codeanalyse
6	Sind die "FV_Activated_S"-Ausgänge der Verbindungsüberwachungsbausteine direkt mit den jeweiligen "SF_IOConfig"-Funktionsbausteinen verbunden?		Statische Codeanalyse
7	Ist das Safety-Programm so gestaltet, dass die Maschine im nicht eingelernten Zustand nicht anlaufen kann?		Statische Codeanalyse
8	Ist das Safety-Programm so gestaltet, dass die sicheren Verbindungen überwacht werden und dass eine Störung einer sicheren Verbindung zu einem Stillsetzen der Maschine führt?		Statische Codeanalyse
		Datum	Unterschrift (Bearbeiter)
		Datum	Unterschrift (Prüfer)

28.2 Maschinenabnahme "Deaktivieren sicherer Teilnehmer"

Siehe auch Maschinenabnahme auf Seite 160.

Checkliste zur Abnahme der IOConfig-Funktionalität der Maschine

Maschinentyp:		Maschinennummer:	
Safety-Applikation (Pin)			
Name:		Prüfsumme:	
Revision:		Letzte Änderung:	
Bearbeiter:		Prüfer:	
Maschinenkonfiguration			
IOConfig-CRC:			
Maschinenkonfiguration:			
Bemerkung:			

Nr.	Anforderung	Ja	Bemerkung
1	Wurde anhand der Prüfsumme verifiziert, dass die korrekte Safety-Applikation geladen ist?		
2	Wurde der Einlernvorgang durchgeführt?		
3	Stimmt die CRC mit der im Feld "IOConfig-CRC" aufgeführten bekannten CRC überein oder sind alle im Feld "Maschinenkonfiguration" aufgeführten sicheren E/As und Antriebe vorhanden und haben eine fehlerfrei sichere Verbindung?		
4	Funktionieren alle Notaus-Taster?		
5	Funktionieren alle weiteren vorhandenen sicheren E/As?		
		Datum	Unterschrift (Bearbeiter)
		Datum	Unterschrift (Prüfer)

28.3 Abnahme des Safety-Programms

Siehe auch Voraussetzungen und Nachweise für die Abnahme auf Seite 183.

Checkliste zur Abnahme des Safety-Programms

Maschinentyp:			
Safety-Applikation (Pin)			
Name:		Prüfsumme:	
Revision:		Letzte Änderung:	
Bearbeiter:		Prüfer:	
Bemerkung:			
IndraWorks			
Version IndraWorks SL:		Version Safety Manager:	
Firmwarestände			
Firmwareversion Std-Steuerung:			
Firmwareversion Safety-Steuerung:			

Nr.	Anforderung	Ja	Bemerkung
1	Liegt ein vollständiges IndraWorks-Projektarchiv vor?		
2	Ist die Benutzerverwaltung aktiv und die Rechte vergeben?		
3	Wurde die zum Projekt passende IndraWorks-Version zusammen mit dem Projekt abgelegt?		
4	Sind alle Objekte der Applikation validiert und gepinnt? Entsprechen die Objektversionen dem Auslieferungsstand?		Doppelklick auf die Safety-Applikation
5	Stimmt die Bootapplikation auf der Maschine mit dem gepinnten Projektstand überein?		Prüfung der Pin-Information über die Registerkarte Safety Online Information
6	Wurde ein Ausdruck (ggf. als PDF) der Safety-Applikation erstellt und zusammen mit dem Projekt abgelegt?		
7	Ist im Projekt eine eindeutige F_Source-Adresse für PROFIsafe-Geräte vergeben?		Doppelklick auf das logische Gerät, F_Source_Add
8	Ist im Projekt eine eindeutige OUNID für CSos-Geräte vergeben?		Doppelklick auf die Safety-Applikation, Registerkarte Konfiguration
9	Sind die F_Dest-Adressen der PROFIsafe-Geräte projektweit eindeutig?		Doppelklick auf das logische Gerät, F_Dest_Add
10	Sind die TUNID-Adressen der CSos-Geräte projektweit eindeutig?		Doppelklick auf das logische Gerät, SNN, SDID
Datum		Unterschrift (Bearbeiter)	
Datum		Unterschrift (Prüfer)	

28.4 Abnahme einer POU

Siehe auch Software-Verifikation auf Seite 167.

Checkliste zur Abnahme eines Safety Funktionsbausteins oder Teilprogramms (POU)

POU			
POU-Name:			
Version:		Prüfsumme:	
Bearbeiter:		Prüfer:	
Bemerkung:			

Nr.	**Anforderung**	**Ja**	**Bemerkung**
1	Ist die POU gepinnt?		
2	Wurden die Programmierrichtlinien eingehalten? (Variablennamen, Aufbau der POU, Komplexität des Codes, Kommentare)		
3	Erfolgt die Übersetzung ohne Warnungen? Falls Warnungen ausgegeben werden, sind diese unvermeidlich?		
4	Sind die Schnittstellen klar dokumentiert?		
5	Wurde ein Code Review durchgeführt?		
6	Wurde ein Zweigabdeckungstest durchgeführt?		
7	Wurden die Grenzwerte gestestet?		
Datum		Unterschrift (Bearbeiter)	
Datum		Unterschrift (Prüfer)	

Index

0 … 9

1oo1- und 1oo2-Eingabegeräte ... 114

A

Abkürzungen ... 6
Abnahme
  PROFIsafe-spezifische Nachweise ... 589
  Safety NetVar-spezifische Nachweise ... 611
  Safety Software / Dokumentation ... 183
  Voraussetzungen und Nachweise ... 183
Abnahme der Applikation ... 681
Abnahmefunktionen
  Archivierung ... 185
  Ausdruck Abnahmedokumentation ... 186
Abnahmeprotokolle
  Safety ... 707
Admin-Passwort löschen
  Safety ... 227
Adresse ... 681
Anregungen ... 6
ANSI Z535.6-2006 ... 3
Ansicht
  Safety Querverweisliste ... 671
Anwender ... 681
Anwenderprogramm-Fehlerreaktion ... 681
AOPD ... 681
API ... 681
Applikation ... 681
Applikation zurücksetzen ... 151
Applikationsstatus des Safety-Laufzeitsystems (S_LZS) ... 681
Archivierung
  Safety ... 185
Ausführungsversion ... 188, 681
Ausloggen
  Safety ... 129
Austausch von Safety-Informationen innerhalb eines Projektes ... 94
Austausch von Safety-Informationen über Projektgrenzen ... 95
Austauschvariablen ... 681

B

Basic Level ... 682
BDC ... 682
Bedienschnittstelle ... 682
Befehl
  Querverweise ausgeben - Safety ... 670
Befehle der Safety-Benutzeroberfläche ... 668
Begriffe
  Safety ... 125
Beschwerde ... 6
Betrieb und Administration ... 191
  Bootapplikation löschen ... 210
  Bootapplikation starten ... 210
  Debuggen der Maschine ... 212
  Firmware-Update ... 212
  Hardware-Tausch ... 213
  Undefinierter oder sicherheitsrelevanter Fehler ... 197
Betriebsmodus des Safety-Laufzeitsystems ... 682
Betriebszustand
  Safety ... 138
Bezeichnungen ... 6
Bezeichnungen und Abkürzungen ... 6
Bibliothek
  SafetyCSos.library ... 589
  SafetyDriveMotion.library ... 536
  SafetyNetVar.library ... 599
  SafetyPLCopen.library ... 292
  SafetyPLCopenPress.library ... 422
  SafetyProfisafeHost.library ... 577
  SafetyStandard.library ... 229
  SafetyUtilities.library ... 240
Bibliotheks-Repository ... 682
Bibliotheksverwalter ... 682
  Safety ... 71
BinaryDecoder_4_16 ... 282
BinaryEncoder_16_4 ... 285
BoolToByte ... 241
BoolToDWord ... 244
BoolToWord ... 242
Bootapplikation
  Safety ... 136
Bootapplikation erzeugen
  Safety ... 136
Bootapplikation löschen
  Safety ... 210
Bootapplikation urlöschen
  Safety ... 227
Bootapplikation-Info
  Sicherheitssteuerung ... 200
BWS ... 682
ByteToBool ... 253

C

Cam shaft ... 682
Cam Switch Unit ... 682
CRC ... 682
CSoS ... 682
CSos-Parameter ... 598
CSosOriginator ... 591
CSS-Fehlercodes ... 655

D

Datenaustausch Standard-/Sicherheitssteuerung ... 52
Datenflussanalyse
  Safety ... 174
Datentypen

Safety ... 100
Deaktivieren
sichere Busteilnehmer ... 153
Debug-Betrieb
Safety ... 143
Debuggen der Maschine
Safety ... 212
Diagnose Sicherheitssteuerung ... 202
Diagnose Verbindungsstatus ... 202
Differenzabnahme ... 682
Dokumentation
Änderungsverlauf ... 1
Download
Safety ... 129
Drucken Abnahmedokumentation ... 186
Drucken Projektvergleich ... 217
Drucken Querverweisliste ... 217
Drucken Vergleichsansicht ... 217
DWordToBool ... 256
Dynamische Verifikation ... 177
Online-Tests ... 179
Online-Tests im Extended Level ... 180

E
E/A-Modul ... 684
E/A-Variable ... 684
E/A-Zugriff
Safety ... 113
E/A ... 684
EDM ... 682
Einloggen
Mehrfacher Download ... 133
Safety ... 129
Einschränkungen durch die Verwendung der Multi-Device Funktionalität
Safety Netzwerkvariablen (NVL) ... 619
Sichere Feldbusse ... 618
Sichere Feldbusse - CIP Safety on Sercos ... 618
Sichere Feldbusse - PROFIsafe ... 619
Ersatzwert ... 91, 151, 682
Ersatzwerte für Austauschvariablen ... 152
Ersatzwerte für E/As ... 152
Erweiterbare Operatoren
Safety ... 106
ESPE ... 682
Extended Level ... 682
Extended Level Datenflussanalyse ... 176
Externe POU ... 682

F
F- ... 683
F-Device ... 683
F-Host ... 683
F-Modul ... 683
F-Parameter ... 683
Failsafe ... 682
FB-Aufrufe
Safety ... 112
Feedback ... 6
Fehler sicherheitsrelevant ... 197
Fehler undefiniert ... 197
Fehlerchecker ... 635
Fehlercodes
CSS ... 655
Fehlerreaktion ... 683
Feldbus-Teilnehmer ... 683
Firmware ... 683
Firmware-Info
Sicherheitssteuerung ... 200
Firmware-Update ... 212
Firmwareversion ... 188
Forcen
Safety ... 148
Funktionsbaustein (FB) ... 683
Funktionsplan ... 683
Funktionstest der Applikation ... 180
Funktionstest der fertigen Maschine ... 181
FUP-Zuweisungen
Safety ... 110

G
Gefahrenhinweise ... 3
Gehe zur Definition ... 174
Gerät aktualisieren ... 187
Geräte-Update ... 187
Gerätebeschreibung ... 683
Gerätebeschreibungspaket ... 683
Geräteparameterabnahme ... 683
Gliederung der Dokumentation ... 2
Globale Variablenliste (GVL) ... 684

H
Hardware-Info
Sicherheitssteuerung ... 204
Hardware-Tausch ... 213
Helpdesk ... 705
Hintergrundtask ... 684
Hotline ... 705

I
I-Parameter ... 685
IEC 61131-3 ... 684
IEC 61131-3 Compliance, SafeLogic ... 691
IEC-Code ... 684
Implementierungsversion ... 684
IndraLogic ... 684
IndraWorks (IW) Safety Manager ... 684
IndraWorks Automation Platform ... 684
IndraWorks SafetyManager-Hinweise ... 31
IndraWorks SafetyManager-Lizenz ... 29

K
Klemme ... 685

Kommunikation / Diagnose
Sicherheitssteuerung ... 202
Konfiguration ... 685
Konfigurationsunterschiede ... 202
Konsistenz ... 685
Kontrollflussanalyse
Safety ... 172
Koppler ... 686
Kritik ... 6
Kundenfeedback ... 6

L
Laufzeitfehler bei Bereichsüberschreitung
Safety ... 106
Laufzeitsystem ... 686
Literalkonstanten
Safety ... 100
Logbuch
Sicherheitssteuerung ... 198
Logische E/As ... 47
CSos-Parameter ... 598
GVL für logischen Austausch ... 52
Nicht-sichere Feldgeräte ... 51
Sichere Feldgeräte ... 51
Standard-Feldgeräte ... 51
Verwendung im Projekt ... 59
LZS ... 686
LZS-Software ... 686

M
Maschinenabnahme ... 686
Maximalprojekt – SafeLogic
Allgemein ... 619
Deaktivieren von sicheren Feldbusteilnehmern ... 619
Feldbusgeräte über UserDefs.cfg deaktivieren ... 619
Multi-Device ... 619
Modifizierer
Safety ... 100
Monitoring
Safety ... 146
Monitoring bei Laufzeitfehler
Safety ... 146
Multi-Device Funktionalität mit SafeLogic ... 613
Administrationspasswort für die Safety-Steuerung ... 616
Allgemein ... 613
Änderung eines Multi-Device Projekts ... 614
Benutzerverwaltung ... 618
Einschränkungen durch die Verwendung der Multi-Device Funktionalität ... 618
Ergänzende Hinweise ... 619
Erstinbetriebnahme ... 613
FAQ – Multi-Device ... 621
Fernwartung für die Safety-Steuerung ... 617
Gemeinsame (einheitliche) Steuerungseinstellungen ... 618
Gerätenamen der Safety-Steuerung ändern ... 616
Gerätenamen der Standard-Steuerung ändern ... 615
Inbetriebnahme ... 613
Individuelle Steuerungseinstellungen ... 615
IP-Adresse der Standard-Steuerung ändern ... 616
Maximalprojekt erstellen ... 619
Projekt pinnen ... 618

N
NetVarReceiver ... 605
NetVarSender ... 603

O
Objekt ... 686
Objekte der Safety-Benutzeroberfläche ... 663
OEM ... 686
OEM-LZS ... 686
Online-Modus
Safety ... 125
Operatoren
Safety ... 106
OSSD ... 686

P
Parametrierung ... 686
Pinnen ... 686
Safety ... 161
PL-e Sicherheitsapplikationen ... 168
Planung der Adressen ... 27
Planung der Geräte und Allokation ... 20
Planung der Reaktionszeiten ... 21
Bei Querkommunikation (T3) ... 23
Gefahr des Undersamplings ... 25
Reaktionszeiten bei Feldbussteuerungen ... 21
Reaktionszeitgarantie bei Querkommunikation ... 24
Überwachungszeit bei Querkommunikation ... 25
Planung des Gesamtsystems ... 19
PLCopen ... 686
PLCopen TC5 ... 686
POUs
Safety ... 96
Produktphasen ... 1
Profibus-DP ... 686
Profisafe ... 686
Profisafe I-Parameter ... 589
Profisafe-F-Parameter ... 586
ProfisafeHost ... 579
Programm (PRG) ... 687
Programmierbeispiel

Safety, 2-Hand-Steuerung mit EDM-Baustein ... 120
Programmierlevel ... 687
Programmierlevel Basic - Extended ... 675
Programmierrichtlinien
Safety ... 82
Programmierschnittstelle ... 687
Programmiersystem ... 687
Programmorganisationseinheit ... 687
Projekt ... 687
Projektdatei ... 687
Projektvergleich
Safety ... 222
Prüfsumme ... 687
PS ... 687

Q
Quellcodeverwaltung ... 687
Quellcodeverwaltung Zugriffsschutz ... 80
Querkommunikation mit Safety Netzwerkvariablen ... 116
Querverweise ausgeben ... 174
Querverweisliste ... 174

R
Reaktionszeit
Safety ... 146
Referenz Safety-Benutzeroberfläche ... 663
Reset kalt
Safety ... 150
Rückwirkungsfreiheit ... 687

S
S-SPS ... 688
SafeLogic ... 687
Safety
Automatisch geprüfte Programmierrichtlinien ... 82
Basic-POUs, Variablen ... 103
Begriffe ... 125
Benutzerverwaltung ... 77
Bibliotheken ... 75
Datenfluss ... 86
Defensives Programmieren ... 84
Dokumentierung des Codes ... 85
Extended-POUs, Variablen ... 104
FB-Aufrufe ... 112
Geänderte Standardfunktionen ... 632
Geräte-Repository ... 76
Globale Variablenliste (GVL) ... 65, 91
IEC 61131-3-Konformität ... 87
IndraWorks SafetyManager-Lizenz ... 29
Liste zulässiger Ansichten ... 629
Liste zulässiger Befehle ... 627
Liste zulässiger oder geänderter Funktionen ... 627
Manuell zu prüfende Programmierrichtlinien ... 84, 170
Namespaces ... 90
Netzwerke ... 97
Normen und Standards ... 11
Operatoren ... 106
Pin-Prüfsumme ... 41
POUs ... 60, 96
Programmierung ... 81
Projektstruktur ... 36
Prüfsumme eines Objekts ... 41
Qualifiziertes Personal ... 29
Quellcodeverwaltung ... 80
Regeln für Bezeichner ... 90
Safety Netzwerkvariablenliste (NVL) ... 67
Software-Verifikation ... 167
Sprachelemente ... 88
Sprung/Return ... 111
Statische Verifikation ... 169
Systemvoraussetzungen ... 29
Taskkonfiguration ... 120
Vergleichsansicht ... 41
Zugriffsschutz ... 77
Safety Admin-Passwort ... 80
Safety Benutzerverwaltung ... 78
Safety Error codes ... 635
Safety GVL ... 65
Safety Logik ... 41
Safety Netzwerkvariablenliste (NVL) ... 91
Safety NVL ... 67
Safety Pinnen ... 161
Safety Task ... 63
Safety Zugriffsschutz ... 80
Safety-Abnahme ... 183
Voraussetzungen und Nachweise ... 183
Safety-Bibliotheksverwalter ... 71
Safety-DCC ... 687
Safety-Diagnosen, CSS ... 654
Safety-Fehlermeldungen ... 635
Safety-FUP-Sprachelemente ... 675
Netzwerke ... 676
Operatoren ... 678
Variablen ... 676
Safety-Geräteparameter ... 687
Safety-Identifikation ... 688
Safety-Konfigurationsdaten ... 688
Safety-NVL (Empfänger) ... 665
Safety-NVL (Sender) ... 663
SafetyCSos.library ... 589
SafetyDriveMotion.library ... 536
SafetyNetVar.library ... 599
SafetyPLCopen.library ... 292
SafetyPLCopenPress.library ... 422
SafetyProfisafeHost.library ... 577
SafetyStandard.library ... 229
SafetyUtilities.library ... 240
SAFExxx ... 688
Schreiben

Safety ... 148
Schreiben auf mehrere SPS
Safety ... 143
SDD ... 688
Service-Hotline ... 705
SF_283A Safety Message Check ... 654
SF_2730, Time Coordination Message Check ... 654
SF_2832 Safety Message Check ... 654
SF_3118 Unsuccessful Forward_Open response ... 654
SF_Antivalent ... 300
SF_AxisGroup ... 567
SF_BinaryDecoder_4_16 ... 287
SF_BinaryEncoder_16_4 ... 290
SF_BoolToByte ... 247
SF_BoolToDWord ... 250
SF_BoolToWord ... 249
SF_ByteToBool ... 258
SF_CamMonitoring ... 487
SF_CamshaftMonitor ... 500
SF_ClockGenerator ... 280
SF_CTD ... 231
SF_CTU ... 234
SF_CTUD ... 232
SF_CycleControl ... 508
SF_DoubleValveMonitoring ... 453
SF_DWordToBool ... 260
SF_EDM ... 306
SF_EmergencyStop ... 314
SF_EnableSwitch ... 320
SF_Equivalent ... 327
SF_ESPE ... 333
SF_F_TRIG ... 238
SF_GuardLocking ... 335
SF_GuardMonitoring ... 341
SF_IOConfig ... 262
SF_IOConfigCntl ... 266
SF_ModeSelector ... 347
SF_MotionEmergencyStop ... 539
SF_MotionSafeStop ... 545
SF_MutingPar ... 356
SF_MutingPar_2Sensor ... 370
SF_MutingSeq ... 379
SF_OutControl ... 389
SF_PressControl ... 522
SF_R_TRIG ... 239
SF_RS ... 230
SF_SafetyRequest ... 395
SF_SingleValveCycleMonitoring ... 437
SF_SingleValveMonitoring ... 425
SF_SMMSelector ... 552
SF_SR ... 231
SF_StartupWarning ... 268
SF_TestableSafetySensor ... 401
SF_TOF ... 235
SF_TON ... 236
SF_TP ... 237
SF_TwoHandControlTypeII ... 412
SF_TwoHandControlTypeIII ... 417
SF_ValveGroupControl ... 470
SF_WordToBool ... 259
Sichere Geräte ... 47
Sichere Konfiguration ... 57
Sichere Parametrierung ... 57
Sicheres E/A-Modul ... 688
Sicheres E/A-Protokoll ... 688
Sicheres E/A-Telegramm ... 688
Sicherheits-PS, Sicherheitsprogrammierung ... 688
Sicherheitsapplikation ... 688
Erstellen ... 34
PLCopen Architekturmodell ... 33
Sicherheitsbibliotheken
Genauigkeit von Zeitmessungen ... 229
Sicherheitsgerichtet ... 688
Sicherheitsgerichtete Firmware ... 688
Sicherheitsgerichtetes Laufzeitsystem ... 688
Sicherheitsgerichtetes Objekt ... 688
Sicherheitsgerichtetes Programmiersystem ... 688
Sicherheitshinweise ... 3
Sicherheitshinweise für applikative Bibliotheksbausteine ... 625
Sicherheitssteuerung ... 38, 688
Verbindungsaufbau ... 126
Signalgrafik ... 3
Signalwörter ... 3
SIL3 Sicherheitsapplikationen ... 168
Software Aktualisierung
Firmware ... 188
Geräteversion ... 187
IndraLogic-Version ... 189
Versionierung ... 187
Software-Verifikation
Safety ... 167
Sprung/Return
Safety ... 111
Stack-Instanz ... 689
Standard- ... 689
Standard-IndraLogic ... 689
Standardapplikation ... 689
Standardsteuerung
Abstimmung mit Sicherheitssteuerung ... 151
Start
Safety ... 150
Statische Verifikation ... 169
Statische Verifikation Programmierrichtlinien
Automatische Prüfung ... 169
Manuelle Prüfung ... 170
Testen gegen die Spezifikation ... 172
Stop
Safety ... 150
Strukturvergleich
Safety ... 222
Support ... 705
Symbole ... 4
System-Fehlerreaktion ... 689

T

Task ... 689
  Safety ... 63
Taskkonfiguration ... 689
  Safety ... 120
TDC ... 689
Testen gegen den Systemplan ... 177
Testen gegen die Spezifikation ... 172
  Datenflussanalyse ... 174
  Datenflussanalyse im Extended Level ... 176
  Globale Kontrollflussanalyse ... 172
  Lokale Kontrollflussanalyse ... 174
  Schnittstellen ... 172
THC ... 689
Typen ... 689

U

Überwachungszeit ... 687
Urlöschen
  Safety ... 227
User Interface ... 682

V

Validierung ... 690
Variablen
  Basic-POUs, Safety ... 103
  Extended-POUs, Safety ... 104
Variablenarten
  Safety ... 100
Variablendeklaration
  Safety ... 73, 100
Verifikation ... 690
Verifikation der Applikation ... 180
Verifikation gegen den Systemplan ... 177
Verifikation in der fertigen Maschine ... 181
Verifikation, Vorbereitung
  Safety ... 161
Version FB ... 219
Versionsliste der Safety-Bausteine ... 623

W

Warnhinweise ... 3
Wiederanlauf ... 690
Wiederverwendung der abgenommenen Software ... 217
  Abgenommenen Safety-Projekt ... 218
  Änderungen im Projekt ... 222
  Änderungen im Projekt mit Querkommunikation ... 224
  Funktionsbausteine ... 219
WordToBool ... 254

Z

Zielgruppen ... 1
Zu dieser Dokumentation ... 1
  Gültigkeit der Dokumentation ... 1
Zweikanalig synchroner Online-Dienst ... 690
Zweikanaliger Online-Dienst ... 690
Zweiterzeugung ... 690

Notizen

rexroth
A Bosch Company

Bosch Rexroth AG
Postfach 13 57
97803 Lohr a.Main, Deutschland
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr a.Main, Deutschland
Tel. +49 9352 18 0
Fax +49 9352 18 8400
www.boschrexroth.com/electrics



R911398634

DOK-IWORKS-SL**PRJ*V15-AP02-DE-P