rexroth
A Bosch Company

IndraWorks 15VRS

Feldbusse

Anwendungsbeschreibung
R911393282

Ausgabe 05



Titel IndraWorks
15VRS
Feldbusse

Art der Dokumentation Anwendungsbeschreibung

Dokumentations-Type DOK-IWORKS-FB******V15-AP05-DE-P

Interner Ablagevermerk RS-49feab6e24cdd0340a347e865dedebd4-5-de-DE-5

Änderungsverlauf Ausgabe 05, 2021-03
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 Software und Steuerungsfirmware, Steuerungsfunktionen SPS und Feldbusse (BAn, HBu, CCl)
Technische Redaktion (MiNi)

Inhaltsverzeichnis

Seite

1 Zu dieser Dokumentation ... 1
1.1 Gültigkeit der Dokumentation ... 1
1.2 Gebrauch der Sicherheitshinweise ... 2
1.2.1 Aufbau der Sicherheitshinweise ... 2
1.2.2 Erläuterung der Signalwörter und der Signalgrafik ... 2
1.2.3 Verwendete Symbole ... 4
1.2.4 Erläuterung der Signalgrafik auf dem Gerät ... 4
1.3 Erforderliche und ergänzende Dokumentationen ... 4

2 Allgemeine Informationen ... 7
2.1 Begriffe und Abkürzungen ... 7
2.2 Übertragungsarten ... 8

3 Feldbus-Schnittstellen und Eigenschaften ... 11
3.1 Feldbusse und Feldbuskommunikation ... 11
3.1.1 Feldbus-Schnittstellen am Steuerungssystem XM ... 11
3.2 Eigenschaften der Feldbusse ... 13
3.2.1 Feldbus-Master ... 13
3.2.2 Feldbus-Slaves ... 14

4 Feldbus übergreifende Features ... 15
4.1 Gerätedatenbank ... 15
4.1.1 Gerätedatenbank, Übersicht ... 15
4.1.2 Geräte verwalten ... 15
Gerätedatenbank, Dialog ... 15
Geräte hinzufügen ... 16
Geräte entfernen ... 19
4.2 Mapping der Feldbus-Ein- und Ausgänge ... 19
4.3 Feldbusmapping, Zugriff vom Feldbusmaster auf Variablen des Feldbusslaves ... 24
4.4 Feldbuskonfiguration "Maximalprojekt" ... 28
4.5 Buszyklus-Task: Kopieren der E/A-Daten zwischen SPS und Feldbus-Stack ... 29
4.5.1 Beschreibung der Buszyklus-Task ... 29
4.5.2 Konfiguration der Buszyklus-Task ... 30
4.5.3 Bus-synchrone SPS-Verarbeitung ... 31
4.5.4 SPS-Einstellung "Variablen aktualisieren" ... 32
4.6 IP-Kanal bei PROFINET IO & EtherNet/IP (netX) ... 33
4.7 SPS steuert Antriebe über eine Feldbusverbindung (Feldbusachsen) ... 36
4.7.1 Allgemeines ... 36
4.7.2 Feldbusachsen im Projekt anlegen ... 37
4.7.3 Antriebseinstellungen für die SPS-Kommunikation ... 39
Allgemeines ... 39
Herstellen der Kommunikation zur Antriebsparametrierung ... 39
Kommunikationseinstellungen IndraDrive ... 40

Seite

4.7.4 Einbinden der notwendigen Bibliotheken ... 40
4.7.5 Adressierung von Feldbusantrieben im SPS-Projekt ... 43
AXIS_REF, Achsadressen der Feldbusantriebe ... 43
AxisData[], Achsdaten ... 45
4.7.6 E/A-Adressierung der Antriebe im SPS-Projekt ... 45

5 Profibus DP ... 47
5.1 Features des Profibus DP Master ... 47
5.2 Features des Profibus DP Slave (Steuerung als Slave) ... 47
5.3 Profibus DP, Grundlagen ... 48
5.3.1 Begriffe und Abkürzungen ... 48
5.4 Profibus DP, Konfiguration Übersicht ... 48
5.5 Profibus DP, Master ... 50
5.5.1 Profibus DP Master, konfigurieren ... 50
Allgemeines ... 50
Registerkarte 'Allgemein' ... 51
Registerkarte 'Profibus DP Master Parameter' ... 53
Registerkarte 'Feldbus-Diagnose' ... 54
Registerkarte 'E/A-Abbild' ... 57
Registerkarte 'Profibus DP Master IEC-Objekte' ... 57
Registerkarte 'Status' ... 58
Registerkarte 'Information' ... 58
5.5.2 Profibus DP Master, Slaves konfigurieren ... 58
Slaves hinzufügen ... 58
Profibus DP-Slaves aktivieren / deaktivieren ... 60
Profibus DP-Slaves deaktivieren mittels UserDefs.cfg ... 61
Slaves konfigurieren ... 63
Registerkarte 'Allgemein' ... 64
Registerkarte 'Profibus DP Parameter' ... 66
Registerkarte 'Status' ... 66
Registerkarte 'Information' ... 66
5.5.3 Profibus DP Master, Module zum Slave hinzufügen ... 67
Allgemeines ... 67
Registerkarte 'Allgemein' ... 67
Registerkarte 'E/A-Abbild' ... 68
Registerkarte 'Status' ... 69
Registerkarte 'Information' ... 69
5.6 Profibus DP, Slave (Steuerung als Slave) ... 69
5.6.1 Profibus DP Slave, konfigurieren ... 69
Allgemeines ... 69
Registerkarte 'Allgemein' ... 70
Registerkarte 'E/A-Abbild' ... 71
Registerkarte 'Profibus DP IEC-Objekte' ... 71
Registerkarte 'Feldbusmapping' ... 71
Registerkarte 'Profibus DP-Parameter' ... 72
Registerkarte 'Profibus DP Konfiguration' ... 73

Seite

Registerkarte 'Status' ... 73
Registerkarte 'Information' ... 74
5.6.2 Profibus DP Slave, Koppelmodule hinzufügen ... 74
Allgemeines ... 74
Registerkarte 'Allgemein' ... 75
Registerkarte 'DP-Module Parameter' ... 75
Registerkarte 'DP-Module E/A-Abbild' ... 75
Registerkarte 'Status' ... 76
Registerkarte 'Information' ... 76
5.6.3 Profibus DP Slave, Master-Slave-Kopplung ... 76

6 Profinet IO ... 79
6.1 Features des Profinet IO Controller ... 79
6.2 Features des Profinet IO Device (Steuerung als Slave) ... 80
6.3 Profinet IO, Grundlagen ... 81
6.3.1 Begriffe und Abkürzungen ... 81
6.3.2 Bedeutung von Gerätenamen bei Profinet IO ... 82
6.3.3 Shared Device ... 83
6.3.4 Media Redundancy Protokoll (MRP) ... 83
6.3.5 Identifizieren ... 84
6.4 Profinet IO, Konfiguration Übersicht ... 84
6.5 Profinet IO, Controller ... 88
6.5.1 Profinet IO Controller, konfigurieren ... 88
Geräteeditor ... 88
Registerkarte 'Allgemein' ... 88
Registerkarte 'Überblick' ... 91
Registerkarte 'Einstellungen' ... 92
Registerkarte 'IP-Kanal-Einstellungen' ... 93
Registerkarte 'Profinet IO Controller Parameter' ... 93
Registerkarte 'Feldbus Diagnose' ... 93
Registerkarte 'Topologie' ... 94
Registerkarte 'E/A-Abbild' ... 96
Registerkarte 'Status' ... 97
Registerkarte 'Information' ... 97
6.5.2 Profinet IO Controller, Devices hinzufügen ... 97
Allgemeines ... 97
Profinet IO-Devices aktivieren / deaktivieren ... 98
Profinet IO-Devices und Profinet IO-Module deaktivieren mittels UserDefs.cfg ... 99
Devices zum Anschluss von I/O-Modulen ... 102
Profinet IO Device - Geräteeditor ... 102
Registerkarte 'Allgemein' ... 103
Registerkarte 'Optionen' ... 105
Registerkarte 'IOxS' ... 107
Registerkarte 'PNIO Parameter' ... 108
Registerkarte 'Status' ... 109
Registerkarte 'Information' ... 109

Seite

6.5.3 Profinet IO Controller, Module zum Device hinzufügen ... 109
Allgemeines ... 109
Profinet IO-Module aktivieren / deaktivieren ... 110
Registerkarte 'Allgemein' ... 111
Registerkarte 'PNIO Parameter' ... 112
Registerkarte 'E/A-Abbild' ... 112
Registerkarte 'Status' ... 113
Registerkarte 'Information' ... 113
6.5.4 Profinet IO Controller, Geräte suchen ... 113
Geräte suchen ... 113
Konfigurationsabgleich ... 118
6.5.5 Profinet IO Controller, Topologie und Gerätetausch ... 119
Topologie und Gerätetausch, Allgemeines ... 119
Austausch eines IO-Devices ... 119
Ablauf des Gerätetausches ... 120
Unterstützung der Topologie- und Gerätetauschfunktionen durch die SPS ... 121
6.6 Profinet IO, Device (Steuerung als Slave) ... 126
6.6.1 Profinet IO Device, konfigurieren ... 126
Profinet IO Device - Geräteeditor ... 126
Registerkarte 'Allgemein' ... 128
Registerkarte 'IP-Kanal-Einstellungen' ... 128
Registerkarte 'Feldbusmapping' ... 129
Registerkarte 'PNIO Parameter' ... 129
Registerkarte 'E/A-Abbild' ... 129
Registerkarte 'Status' ... 129
Registerkarte 'Information' ... 130
6.6.2 Profinet IO Device, Koppelmodule hinzufügen ... 130
Allgemeines ... 130
Registerkarte 'Allgemein' ... 131
Registerkarte 'PNIO Module Parameter' ... 131
Registerkarte 'PNIO Module E/A-Abbild' ... 131
Registerkarte 'Status' ... 132
Registerkarte 'Information' ... 132
6.6.3 Profinet IO Device, Controller-Device-Kopplung ... 133
Allgemeines ... 133
Registerkarte 'Allgemein' ... 134
Registerkarte 'IOxS' ... 136
Registerkarte 'PNIO Parameters' ... 137
Registerkarte 'Status' ... 137
Registerkarte 'Information' ... 137
6.7 Profinet IO, Tool Calling Interface (TCI) ... 137
6.7.1 Handlungsschritte für den Aufruf eines Device Tools aus IndraWorks ... 138
6.7.2 Funktionale Eigenschaften der TCI-Schnittstelle in IndraWorks (Engineering System) ... 139
Datenübergabe von IndraWorks (Engineering System) zum Device Tool ... 139
Grundlegende Eigenschaften von TCI ... 139
Datenablage ... 139

Seite

Kommunikation vom Device Tool zum PROFINET-Device ... 140
6.7.3 Die wichtigsten UseCases im Zusammenhang mit TCI ... 140
IndraWorks-Projekt schließen / IndraWorks beenden ... 140
IndraWorks-Projekt speichern ... 140
IndraWorks-Projekt speichern unter ... 140
IndraWorks-Projekt archivieren ... 141
IndraWorks-Projekt wiederherstellen ... 141
6.8 Profinet IO, FAQs ... 141
6.8.1 Übersicht ... 141
6.8.2 Wo findet man die Profinet Gerätebeschreibungsdateien (GSDML) für die Steuerungssysteme? ... 141
6.8.3 IP-Routing zwischen Eng.-Port der Steuerung und Profinet Netzwerk funktioniert nicht? ... 142
6.8.4 SPS-Logger Meldung: "PN_Topology <station-name> IP x.x.x.x multiple No-of-Peers n" ... 142
6.8.5 Profinet IO Controller: Profinet-Device umbenennen funktioniert nicht? (im laufenden Betrieb) ... 142
6.8.6 Profinet IO Controller: wird eine Ringtopologie / Redundanz unterstützt? ... 143
6.8.7 Profinet IO Device (Steuerung als Slave): XMx oder VPx als PN-Device wird sporadisch nicht erkannt ... 143

7 EtherNet/IP ... 145
7.1 Features des EtherNet/IP Scanners ... 145
7.2 Features des EtherNet/IP Adapter ... 145
7.3 Features des EtherNet/IP Adapter (Engineering Port) ... 149
7.4 EtherNet/IP, Grundlagen ... 149
7.4.1 Allgemeines ... 149
7.4.2 Begriffe und Abkürzungen ... 150
7.5 EtherNet/IP, Scanner ... 150
7.5.1 EtherNet/IP Scanner konfigurieren ... 150
Übersicht ... 150
Scanner hinzufügen ... 151
Registerkarte 'Allgemein' ... 152
Registerkarte 'IP-Kanal-Einstellungen' ... 152
Registerkarte 'EthernetIP Scanner Parameter' ... 152
Registerkarte 'E/A-Abbild' ... 153
Registerkarte 'Status' ... 154
Registerkarte 'Information' ... 154
7.5.2 EtherNet/IP Scanner, Adapter hinzufügen ... 154
Adapter hinzufügen ... 154
Registerkarte 'Allgemein' ... 154
Registerkarte 'Verbindungen' ... 156
Registerkarte 'Assemblies' ... 158
Registerkarte 'Status' ... 158
7.5.3 Das "Generic Device" ... 158
7.6 EtherNet/IP, Adapter ... 160
7.6.1 Übersicht ... 160
7.6.2 Registerkarte 'Adapter-Einstellungen' ... 161
7.6.3 Registerkarte 'IP-Kanal-Einstellungen' ... 163
7.6.4 Registerkarte 'E/A-Abbild' ... 163

Seite

7.6.5 Registerkarte 'Feldbusmapping' 164
7.6.6 Registerkarte 'EtherNetIP Parameter' 164
7.6.7 Registerkarte 'Status' 165
7.6.8 Registerkarte 'Information' 165
7.6.9 EtherNet/IP Scanner-Adapter-Kopplung 165
7.7 EtherNet/IP, Adapter (Engineering Port) konfigurieren 166
7.7.1 Übersicht 166
7.7.2 Registerkarte 'Status' 168
7.7.3 Registerkarte 'Information' 168
7.7.4 EtherNet/IP Adapter (Engineering Port) - Module 169
Registerkarte 'E/A-Abbild' 169
Registerkarte 'Information' 169
7.8 EtherNet/IP, FAQs 170
7.8.1 Häufig auftretende Probleme bei der EtherNet/IP Konfiguration 170
7.8.2 Hinweise zur EtherNet/IP Verbindungslänge (O => T / T => O Größe) 172

8 EtherCAT 175
8.1 Features des EtherCAT Master 175
8.2 EtherCAT Grundlagen 176
8.2.1 Begriffe und Abkürzungen 176
8.2.2 Topologie 177
8.2.3 Adressierungs- und Identifikationsverfahren 179
8.2.4 Zustandsmaschine 180
8.2.5 Azyklische Kommunikation (Mailbox) 182
8.2.6 Telegramm-Aufbau 184
8.2.7 WorkingCounter 185
8.2.8 SyncUnits 185
8.2.9 Hot Connect 186
8.2.10 Diagnose 186
Allgemein 186
AL Status Codes 187
CoE SDO Abortcodes 190
SoE Fehlercodes 191
FoE Fehlercodes 193
8.2.11 Synchronisation 193
Allgemein 193
Free Run 194
Synchronisation mit SM-Event 195
Synchronisation mit DC-Sync 197
8.2.12 EtherCAT Slave Controller 203
Aufbau 203
Kommandos 204
8.3 EtherCAT Konfiguration Übersicht 205
8.4 EtherCAT Master konfigurieren 209
8.4.1 EtherCAT Master, Gerät setzen 209
8.4.2 Registerkarte 'Allgemein' 209

Seite

8.4.3 Registerkarte 'Sync-Unit-Zuordnung' 210
8.4.4 Registerkarte 'Distributed Clocks' 211
8.4.5 Registerkarte 'Busdiagnose' 214
8.4.6 Registerkarte 'EoE' 216
8.4.7 Registerkarte 'CoE' 216
8.4.8 Registerkarte 'Diagnose' 217
8.4.9 Registerkarte 'Information' 217
8.5 EtherCAT Master, Slaves konfigurieren 217
8.5.1 Slaves und Module hinzufügen 217
8.5.2 Registerkarte 'Allgemein' 219
8.5.3 Registerkarte 'SyncManager' 224
8.5.4 Registerkarte 'Prozessdaten Expertenmodus' 225
8.5.5 Registerkarte 'Prozessdaten' 227
8.5.6 Registerkarte 'Startparameter' 228
8.5.7 Registerkarte 'EoE-Einstellungen' 232
8.5.8 Registerkarte 'Online' 233
8.5.9 Registerkarte 'ESC Register' 233
8.5.10 Registerkarte 'CoE' 234
8.5.11 Registerkarte 'Diagnose' 235
8.5.12 Registerkarte 'E/A-Abbild' 236
8.5.13 Registerkarte 'Information' 238
8.5.14 SoE Parametereditor 238
8.6 EtherCAT Master, Bus scannen 239
8.7 EtherCAT Master, Bus synchrone SPS-Verarbeitung 239
8.8 EtherCAT FAQs 240
8.8.1 Übersicht 240
8.8.2 IndraDrive: Wie verbindet man IndraWorks Ds über EtherCAT? 240
8.8.3 IndraDrive: Wie konfiguriert man SoE mit Free Run? 242
8.8.4 S20-EC-BK: Buskoppler schaltet nicht in OP (Hinweise zu AL Status Codes) 243
8.8.5 S20-EC-BK: Wie erstellt man eine Anlaufparametrierung für die S20-Module? 244

9 Sercos III IO 245
9.1 Features des Sercos III-IO-Master 245
9.2 Begriffe und Abkürzungen 245
9.3 Sercos III konfigurieren 246
9.3.1 Sercos III konfigurieren, Übersicht 246
9.3.2 Sercos III-Master, Übersicht Busadressen 247
9.3.3 Sercos III-Master, Sercos-Busdiagnose 248
9.3.4 Sercos III-Master, Verkabelung und Redundanz (am Beispiel eines MLC-Systems) 255
9.3.5 Sercos III-Master, Slave hinzufügen 259
Sercos III-Master, Slave hinzufügen, Allgemeines 259
Sercos-Slaves aktivieren / deaktivieren 261
Sercos-Slaves / Sercos-Module deaktivieren mittels UserDefs.cfg 262
Inbetriebnahme von Sercos-IO-Geräten/Modulen am Sercos-Bus 264
Sercos III Master, Sercos III-Slave konfigurieren 264
Registerkarte 'Allgemein' 265

Seite

Registerkarte 'sercos Konfiguration' ... 266
Registerkarte 'Status' ... 266
Registerkarte 'Information' ... 266
9.3.6 Sercos III Master, Module zum Slave hinzufügen ... 267
Sercos III Master, Module zum Slave hinzufügen, Allgemeines ... 267
Sercos-Module aktivieren / deaktivieren ... 267
Registerkarte 'Sercos Modul' ... 268
Registerkarte 'Funktionsgruppen' ... 269
Registerkarte 'Allgemeine Ein- und Ausgänge' ... 269
Registerkarte 'Benutzerdefinierte Parameter' ... 271
Registerkarte 'Oversampling' ... 271
Registerkarte 'Sercos Module Konfiguration' ... 273
Registerkarte 'E/A-Abbild' ... 274
Registerkarte 'Information' ... 275

10 Modbus RTU ... 277
10.1 Modbus-Buszyklus-Task ... 277
10.2 Modbus-Konfigurator ... 278
10.3 Modbus Master ... 280
10.3.1 Registerkarte 'Modbus Serieller Port - Allgemein' ... 280
10.3.2 Registerkarte 'Modbus Master - Allgemein' ... 280
10.3.3 Registerkarte 'ModbusTCP Master - Allgemein' ... 280
10.3.4 Registerkarte 'Modbus Slave - Allgemein' ... 281
10.3.5 Registerkarte 'ModbusTCP Slave - Allgemein' ... 281
10.3.6 Registerkarte 'Modbus Slave - Kanal' ... 282
10.3.7 Registerkarte 'Modbus Slave Init' ... 283
10.4 Modbus Slave Device ... 284
10.4.1 Registerkarte 'Modbus Serial Device - Allgemein' ... 284
10.4.2 Registerkarte 'ModbusTCP Slave Device - Allgemein' ... 284
10.5 Registerkarte 'Modbus - Parameter' ... 285
10.6 Registerkarte 'E/A-Abbild' ... 285
10.7 Registerkarte 'Status' ... 286
10.8 Registerkarte 'Information' ... 286

11 IO-Link ... 287
11.1 Einführung und Übersicht ... 287
11.2 Use Cases für das Engineering von IO-Link-Systemen ... 287
11.2.1 Einlesen oder Ausgabe von Analogen Werten mit Hilfe eines IO-Link-Devices ... 287
11.2.2 Erstinbetriebnahme eines IO-Link-Devices ... 288
11.2.3 Prozessdaten und IO Device Description (IODD) von IO-Link-Devices ... 288
11.2.4 Serieninbetriebnahme von Maschinen mit IO-Link-Devices ... 288
11.2.5 Parametrierung von IO-Link-Devices aus dem SPS-Programm ... 289
11.2.6 Diagnose von IO-Link-Devices ... 289
11.2.7 Gerätetausch von IO-Link-Devices ... 290
11.3 Profinet IO IO-Link-Gateways ... 290

Seite

11.3.1 Einführung und Übersicht ... 290
11.3.2 Funktionsbausteine für Profinet IO IO-Link-Gateways ... 291
11.3.3 Beispielhafte Anwendung eines Profinet IO IO-Link-Gateways der Firma Balluff ... 292
11.3.4 Beispielhafte Anwendung des Rexroth Profinet IO IO-Link-Gateways S67E-PN-IOL8-DI4-M12-6P ... 296
11.4 Sercos IO-Link Gateways ... 300
11.4.1 Einführung und Übersicht ... 300
11.4.2 Funktionsbausteine für Sercos IO IO-Link-Gateways ... 300

12 Lokales S20 IO ... 303
12.1 Features des Lokalen S20 IO ... 303
12.2 S20-IO konfigurieren ... 304
12.2.1 S20-Objekt und S20-Module, Übersicht ... 304
12.2.2 S20-IO-Objekt ... 304
Registerkarte 'Einstellungen' ... 304
Registerkarte 'S20 Module Parameter' ... 307
Registerkarte 'Feldbus Diagnose' ... 307
Registerkarte 'E/A-Abbild' ... 309
Registerkarte 'Status' ... 310
Registerkarte 'Information' ... 312
12.2.3 S20-Module einfügen ... 312
12.2.4 S20-Module aktivieren / deaktivieren ... 312
12.2.5 S20-Module deaktivieren mittels 'UserDefs.cfg' ... 313
12.2.6 S20-Module konfigurieren ... 314
Allgemeines ... 314
Registerkarte 'Benutzerdefinierte Parameter' ... 315
Registerkarte 'E/A-Abbild' ... 315
Registerkarte 'Information' ... 316
12.2.7 S20-Module, Bus scannen ... 316

13 Gebäudeautomatisierung ... 319
13.1 Allgemein ... 319
13.2 BACnet ... 321
13.2.1 Features der BACnet Implementation ... 321
13.2.2 BACnet Grundlagen ... 325
Allgemeines ... 325
Weiterführende Informationen ... 325
Begriffe und Abkürzungen ... 326
Netzwerke und Topologien ... 327
Objekte und Eigenschaften ... 328
Dienste ... 332
Übersicht ... 332
Änderungsdienst (COV Notification) ... 333
BACnet COV Notification Beispiel ... 333
Auftragspriorisierung (Outputs) ... 340

Seite

Alarmbehandlung ... 341
Datenaufzeichnung (TrendLog) ... 344
Geräte-Profile ... 345
BACnet Fehler-Codes ... 346
BACnet Status-Codes ... 349
13.2.3 BACnet Server konfigurieren ... 352
BACnet Server Inbetriebnahme ... 352
BACnet Server und Objekte anlegen und BACnet Server starten ... 353
BACnet Plugin - Persistence ... 359
BACnet Plugin - Schedule ... 361
BACnet Plugin - Calendar ... 361
BACnet Plugin - Notification Class ... 361
BACnet Plugin - Trend Log ... 361
SPS-Schnittstelle zu den Objekteigenschaften ... 361
Export und Import einer BACnet-Konfiguration im EDE-Format ... 363
13.2.4 BACnet/IP Client konfigurieren ... 364
13.2.5 Hinweise zu implementierten BACnet Objekten und Diensten ... 365
Allgemein ... 365
Objekt - Property - Aktionen ... 366
Dienste mit Implementierung von Standard-Aktionen ... 369
Dienste mit internen Implementierungen ... 377
Dienste ohne Implementierung von Standard-Aktionen ... 378
Beispiele zur Implementierung von Service Indication Hooks ... 378
Implementierungsbeispiel "CreateObject" Service Hook ... 378
Implementierung von Logging-Objekten (EL / TL / TLM) ... 381

14 Service und Support ... 383

Index ... 385

1 Zu dieser Dokumentation

Ausgaben dieser Dokumentation

Ausgabe	Stand	Bemerkung
01	2019-07	Erstausgabe zur Version 15VRS **Neu**: • Kapitel BACnet, siehe Kap. 13.2 "BACnet" auf Seite 321
02	2019-10	**Neu**: • Kap. "BACnet Fehler-Codes" auf Seite 346 • Kap. "BACnet Status-Codes" auf Seite 349 • "**BACnet Server Einstellungen**" auf Seite 358 **Überarbeitung**: • Kap. 13.1 "Allgemein" auf Seite 319 • Kap. 13.2.1 "Features der BACnet Implementation" auf Seite 321 • S20 Modulbetriebsarten, siehe Kap. 12.2.2 "S20-IO-Objekt" auf Seite 304
03	2020-06	**Überarbeitung**: • Kap. 13.2.3 "BACnet Server konfigurieren" auf Seite 352 • Kap. 8.5.6 "Registerkarte 'Startparameter'" auf Seite 228
04	2020-11	**Überarbeitung**: • Kap. 7.2 "Features des EtherNet/IP Adapter" auf Seite 145 • Kap. 13.2.3 "BACnet Server konfigurieren" auf Seite 352
05	2021-03	**Neu**: • Kap. "BACnet COV Notification Beispiel" auf Seite 333 **Überarbeitung**: • Kap. "Weiterführende Informationen" auf Seite 325 • Kap. 13.2.4 "BACnet/IP Client konfigurieren" auf Seite 364

Tab. 1-1: Änderungsverlauf

1.1 Gültigkeit der Dokumentation

Zielgruppe

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.

Die Dokumentation ist die Grundlage für die Online-Hilfe.

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.

Anwendungsphasen

Diese Dokumentation unterstützt den Anwender in den Phasen

- Konfiguration der Feldbus-Komponenten im IndraWorks,
- Feldbus-Applikation im SPS-Anwenderprogramm.

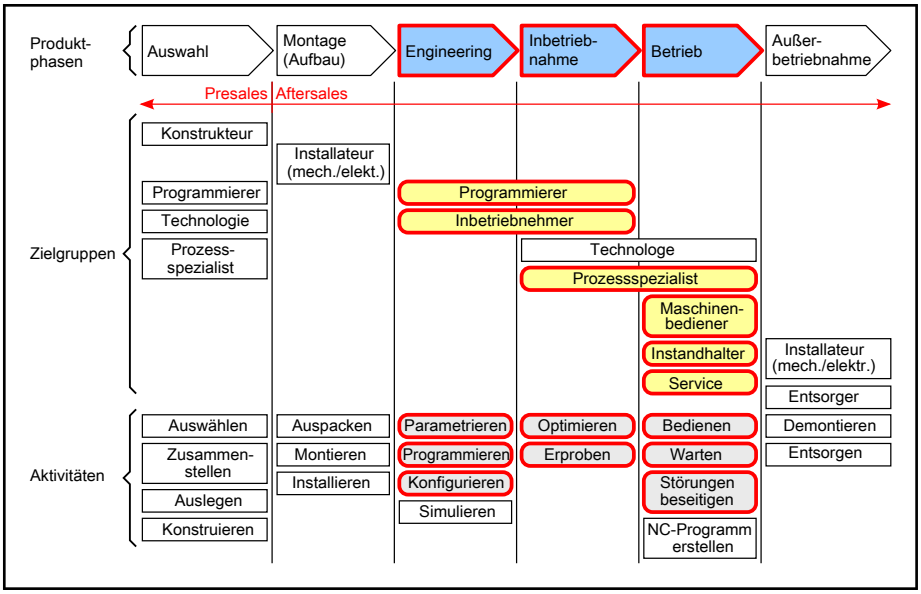


Abb. 1-1: Anwendungsphasen

1.2 Gebrauch der Sicherheitshinweise

1.2.1 Aufbau der Sicherheitshinweise

Die Sicherheitshinweise sind wie folgt aufgebaut:

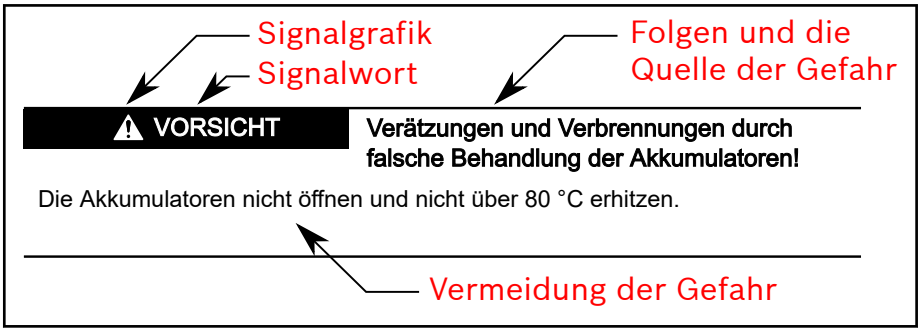


Abb. 1-2: Aufbau der Sicherheitshinweise

1.2.2 Erläuterung der Signalwörter und der Signalgrafik

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

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

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

⚠ GEFAHR

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

⚠ WARNUNG

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

⚠ VORSICHT

Bei Nichtbeachtung dieses Sicherheitshinweises können mittelschwere oder leichte Körperverletzung eintreten.

HINWEIS

Bei Nichtbeachtung dieses Sicherheitshinweises können Sachschäden eintreten.

1.2.3 Verwendete Symbole

Fingerzeige werden wie folgt dargestellt:

Dies ist ein Hinweis.

Tipps werden wie folgt dargestellt:

Dies ist ein Tipp.

1.2.4 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.

1.3 Erforderliche und ergänzende Dokumentationen

Die im Folgenden aufgeführten Dokumente beinhalten zusätzliche Informationen zu diesem Thema.

Rexroth IndraWorks 15VRS IndraLogic 2G SPS-Programmiersystem 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.
Rexroth 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.
Rexroth IndraWorks 15VRS Basis-Bibliotheken IndraLogic 2G DOK-IL*2G*-BASLIB**V15-LIRS-DE-P, R911398632 Das vorliegende Handbuch beschreibt die systemübergreifenden SPS-Bibliotheken.
Rexroth IndraMotion MLC 15VRS PLCopen-Bibliotheken DOK-XLCMLC-FUNLIB**V15-LIRS-DE-P, R911393306 Diese Dokumentation beschreibt die Funktionsbausteine, Funktionen und Datentypen der Bibliotheken RIL_CommonTypes, ML_Base und ML_PLCopen. Die Dokumentation enthält ferner die Fehlerreaktion der Bausteine.
Die Automatisierungsklemmen der Produktfamilie Rexroth Inline DOK-CONTRL-ILSYSINS***-AWRS-DE-P, R911317017 Diese Dokumentation beschreibt die Automatisierungsklemmen der Produktfamilie Rexroth Inline.
Rexroth Inline-Buskoppler für Sercos mit digitalen Ein- und Ausgängen R-IL S3 BK DI8 DO4-PAC DOK-CONTRL-ILS3BKDI8DO-KBRS-DE-P, R911170559 Der Buskoppler ist zum Einsatz innerhalb eines Sercos® Netzwerks der dritten Generation vorgesehen und stellt das Bindeglied zum Inline-Installationssystem dar. Zusätzlich dient er zur Erfassung und Ausgabe digitaler Signale.

Rexroth Inline-Buskoppler für PROFIBUS-DP mit digitalen Ein- und Ausgängen R-IL PB BK DI8 DO4-PAC DOK-CONTRL-ILPBBKDI8DO-AWRS-DE-P, R911170475 Der Buskoppler stellt das Bindeglied zwischen dem PROFIBUS-DP und dem Inline-Installationssystem dar. Zusätzlich dient er zur Erfassung und Ausgabe digitaler Signale.
Rexroth Inline-Buskoppler für PROFINET mit digitalen Eingängen und Ausgängen R-IL PN BK DI8 DO4-PAC DOK-CONTRL-ILPNBKDIO**-DARS-DE-P, R911328681 Der Buskoppler stellt das Bindeglied zwischen einem PROFINET-Netzwerk und dem Inline-Installationssystem dar. Zusätzlich dient er zur Erfassung und Ausgabe digitaler Signale.
Rexroth Inline-Buskoppler für Ethernet mit digitalen Ein- und Ausgängen R-IL ETH BK DI8 DO4 2TX-PAC DOK-CONTRL-ILETHBKDIO*-DARS-DE-P, R911328678 Der Buskoppler stellt das Bindeglied zwischen einem Ethernet-Netzwerk und dem Inline-Installationssystem dar. Zusätzlich dient er zur Erfassung und Ausgabe digitaler Signale.
Rexroth IndraControl S67 Profibus-Koppler 8 digitale Eingänge (M8) DOK-CONTRL-S67PBBKDI8*-APRS-DE-P, R911329565
Rexroth IndraControl S67 Ethernet/IP-Koppler 8 digitale Eingänge (M8) DOK-CONTRL-S67ETBKDI8*-APRS-DE-P, R911329563
Rexroth IndraControl S67 Profinet-Koppler 8 digitale Eingänge (M8) DOK-CONTRL-S67PNBKDI8*-APRS-DE-P, R911329567
Rexroth IndraControl S67 sercos Koppler 8 digitale Eingänge (M8) DOK-CONTRL-S67S3BKDI8*-APRS-DE-P, R911338400
Rexroth IndraControl S20-Buskoppler für PROFINET IO-Anschaltung modular erweiterbar mit S20-Modulen DOK-CONTRL-S20*PN*BK**-DARS-DE-P, R911335967
Rexroth IndraControl S20-Buskoppler für sercos Anschaltung modular erweiterbar mit S20-Modulen DOK-CONTRL-S20*S3*BK**-DARS-DE-P, R911335969

2 Allgemeine Informationen

2.1 Begriffe und Abkürzungen

Die folgende Abbildung stellt die feldbusabhängigen Bezeichnungen der Master- bzw. Slave-Schnittstellen der Feldbusse dar:

Begriffe der Feldbus-Schnittstellen

Feldbus	Master-Schnittstelle	Slave-Schnittstelle
Profibus DP	Master	Slave
Profinet IO	Controller	Device
EtherNet/IP	Scanner	Adapter
EtherCAT	Master	Slave
Sercos III	Master	Slave

Begriffe der Datenübertragungsmechanismen

Feldbus	zyklische Übertragung	azyklische Übertragung
Profibus DP	DP/V0	DP/V1
Profinet IO	RTC (real-time cyclic)	RTA (real-time acyclic)
EtherNet/IP	implicit messaging (polled I/O)	explicit messaging
EtherCAT	zyklische Daten / Prozessdaten	verschiedene Mailbox Protokolle
Sercos III	zyklische Daten	Service-Kanal + IP-Kanal

Eine Besonderheit stellt der lokale I/O-Bus (Inline oder S20) an der Steuerung dar. Er ist vergleichbar mit einem Feldbus-Master mit genau einem Feldbus-Slave, an dem I/O-Module anschließbar sind.

Gerätebeschreibungsdateien

Die Gerätebeschreibungsdatei beschreibt die Eigenschaften des Feldbusgeräts. In dieser Datei sind alle relevanten Daten enthalten, die sowohl für das Engineering als auch für den Datenaustausch mit dem I/O-Gerät von Bedeutung sind. Für die verschiedenen Feldbusse sind folgende Dateiformate festgelegt:

GSD-Datei

Dies ist die Gerätebeschreibung, mittels derer IndraWorks bzw. ein anderes Engineeringsystem die Profibus DP-Slaves bekannt gemacht werden.

GSDML-Datei

Dies ist die Gerätebeschreibung, mittels derer IndraWorks bzw. ein anderes Engineeringsystem die Profinet IO-Device bekannt gemacht werden.

EDS-Datei

Dies ist die Gerätebeschreibung, mittels derer IndraWorks bzw. ein anderes Engineeringsystem die EtherNet/IP Adapter bekannt gemacht werden (EDS: Electronic Data Sheet).

ESI-Datei

Dies ist die Gerätebeschreibung, mittels derer IndraWorks bzw. ein anderes Engineeringsystem die EtherCAT Slaves bekannt gemacht werden (ESI: EtherCAT Slave Information).

SDDML-Datei

Dies ist die Gerätebeschreibung, mittels derer IndraWorks bzw. ein anderes Engineeringsystem die Sercos III-Slaves nach IO-Profil bekannt gemacht werden (SDDML: Sercos Device Description Markup Language).

Feldbus-Anschaltung

Buskoppler

Feldbus-Anschaltung (Slave, Device, Adapter),

kompakt

- Bei einem kompakten Slave ist der Modulaufbau fest vorgegeben.

  Die Module unterhalb des Slave-Objektknotens sind bereits vollständig vorhanden. In der Bibliothek sind keine weiteren Module vorhanden

modular

- Der Modulaufbau des Slaves ist variabel. Die Module (Klemmen) können individuell – aber nach Bestückungsvorschrift des Gerätes – angeordnet werden.

Modul, Klemme

Kleinste I/O-Einheit zur Bestückung einer Feldbus-Anschaltung und/oder des Inlinebusses.

2.2 Übertragungsarten

Die Übertragungsarten lassen sich in folgende Kategorien einteilen:

- zyklischer Übertragungskanal
- azyklischer Übertragungskanal

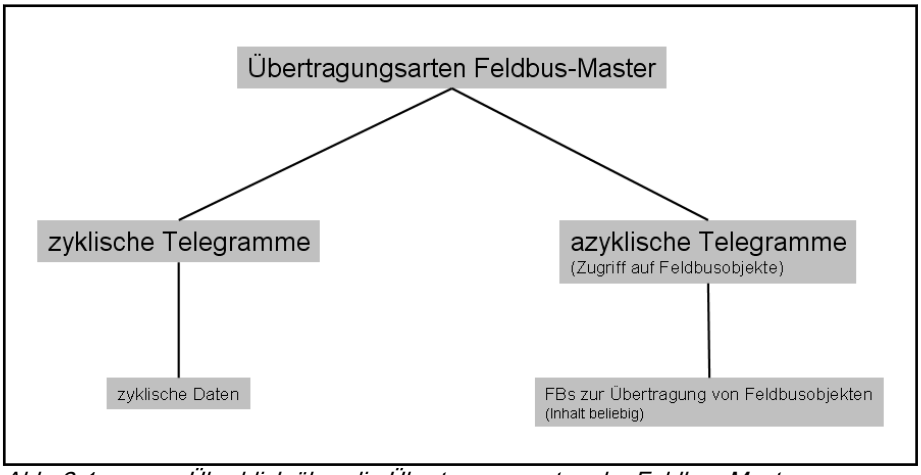


Abb. 2-1: Überblick über die Übertragungsarten der Feldbus-Master

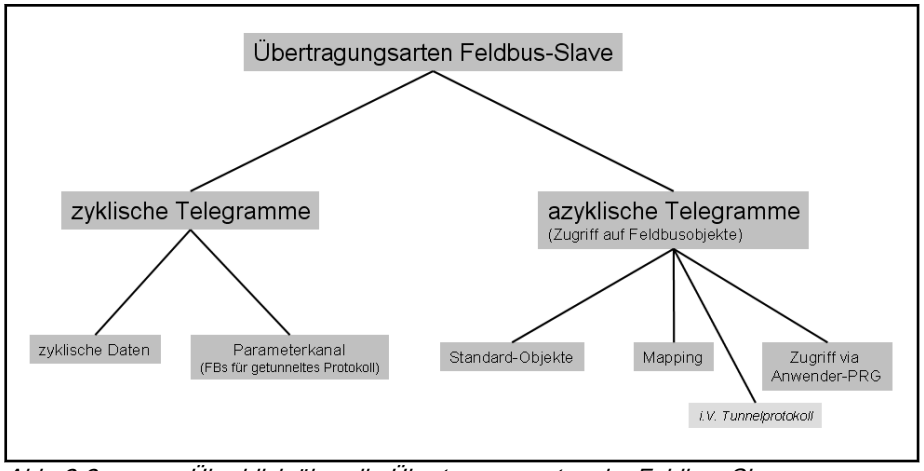


Abb. 2-2: Überblick über die Übertragungsarten der Feldbus-Slaves

Die Feldbusobjekte werden dabei mittels folgender Elemente beschrieben:

- Profibus: Slot (8 Bit), Index (8 Bit)
- Profinet IO: Slot (16 Bit), Subslot (16 Bit), Index (16 Bit)
- EtherNet/IP: Class (8 Bit), Instance (8/16/32 Bit), Attribute (16 Bit)
- Sercos III: EIDN (32 Bit) mit Parameternummer (16 Bit), Strukturelement SE (8 Bit), Strukturinstanz SI (8 Bit)

3 Feldbus-Schnittstellen und Eigenschaften

3.1 Feldbusse und Feldbuskommunikation

Als Plattform für die Systeme MLC und MTX bieten die Steuerungen der XM-Baureihe verschiedene Schnittstellen zum Anschluss von IO-Komponenten und Feldbusgeräten. Die Verwendung der Schnittstellen kann während der Projektierung über das IndraWorks Engineering vorgegeben werden.

☞ Bei der Verwendung von Safety nutzen Sie bitte die "Erste Schritte"-Dokumentation zu SafeLogic und SafeLogic Compact.

Rexroth IndraWorks SafeLogic 14VRS, Die ersten Schritte, DOK-IWORKS-SL*STEP*V14-CO02-DE-P, R911341519

Rexroth IndraWorks SafeLogic Compact 14VRS, Die ersten Schritte, DOK-CONTRL-SLC*F*STEP*-CO02-DE-P, R911346743

3.1.1 Feldbus-Schnittstellen am Steuerungssystem XM

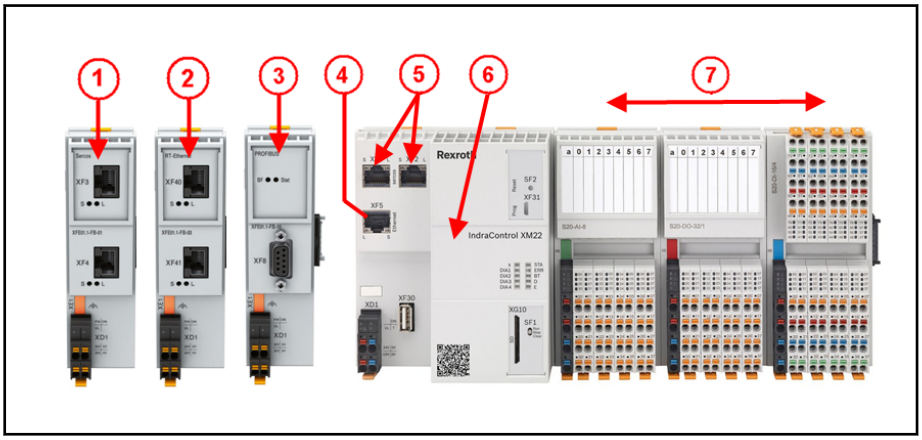


Abb. 3-1: XM2-Steuerung mit Feldbus und IO-Kopplung

Nr.	Funktion	Beschreibung
1, 2, 3	Erweiterungsmodule	An der linken Seite der XM2-Steuerung können über den dort angebrachten Funktionsmodul-Stecker bis zu drei Erweiterungsmodule angeschlossen werden *(1) Erweiterungsmodul Sercos III (XFE01.1-FB01)* • in Vorbereitung *(2) Erweiterungsmodul RT-Ethernet (XFE01.1-FB03)* • Profinet IO-Controller oder -Device, EtherNet/IP Scanner oder Adapter *(3) Erweiterungsmodul Profibus (XFE01.1-FB10)* • Profibus DP-Master oder -Slave
4	Ethernet-Schnittstelle XF5	*Standard-Ethernet-Schnittstelle für TCP/IP-Netzwerk* • Programmier- und Visualisierungsgeräteschnittstelle • BACnet Server und Client • Ethernet/IP Adapter am Engineering Port
5	Ethernet-Schnittstellen XF1/XF2	Ethernet-Schnittstellen zum Anschließen der Sercos III-Antriebe und -IO Alternativ können diese Ethernet-Schnittstellen auch für EtherCAT verwendet werden

Nr.	Funktion	Beschreibung
6	XM2	Steuerung mit Spannungsversorgung (XD1), Diagnose-LEDs und Speicherkarte (SD), USB Host (XF30), USB Device (XF31)
7	Rexroth S20 IO Module	Im IO-Bereich der XM2-Steuerung können bis zu 63 S20-Module über S20-Bussockelmodule angeschlossen werden. Die Anzahl der anschließbaren Module ist abhängig von der Gesamtstromaufnahme der S20-Module. Die XM2-Steuerung kann maximal 2 A für die Stromversorgung der S20 Module bereitstellen. Weitere Informationen zum Anschluss von Rexroth S20 Klemmen finden Sie in der Dokumentation" Die Automatisierungsklemmen der Produktfamilie Rexroth S20", DOK-CONTRL-S20*SYS*INS-APxx-DE-P, R911335987, xx - Ausgabe

Tab. 3-1: Anschlüsse und Bedienelemente der XM2-Steuerung

XM-Steuerung - Feldbusse und Lokaler Bus

- Ein Profibus DP-Master kann auf einem aufgesteckten Erweiterungsmodul (XFE01.1-FB10) aktiviert werden

  Kap. 5.5.1 "Profibus DP Master, konfigurieren" auf Seite 50
- Ein Profibus DP-Slave kann auf einem aufgesteckten Erweiterungsmodul (XFE01.1-FB10) aktiviert werden

  Kap. 5.5.2 "Profibus DP Master, Slaves konfigurieren" auf Seite 58
- Ein Profinet IO-Controller kann auf einem aufgesteckten Erweiterungsmodul (XFE01.1-FB03) aktiviert werden

  Kap. 6.5.1 "Profinet IO Controller, konfigurieren" auf Seite 88
- Ein Profinet IO-Device kann auf einem aufgesteckten Erweiterungsmodul (XFE01.1-FB03) aktiviert werden

  Kap. 6.5.2 "Profinet IO Controller, Devices hinzufügen" auf Seite 97
- Ein EtherNet/IP Scanner kann auf einem aufgesteckten Erweiterungsmodul (XFE01.1-FB03) aktiviert werden

  Kap. 7.5 "EtherNet/IP, Scanner" auf Seite 150
- Ein EtherNet/IP Adapter kann auf einem aufgesteckten Erweiterungsmodul (XFE01.1-FB03) aktiviert werden

  Kap. 7.6 "EtherNet/IP, Adapter" auf Seite 160
- Eine Sercos III Master-Schnittstelle für IO bzw. Feldgeräte steht die Onboard-Sercos III-Schnittstelle zur Verfügung; ein Sercos III-Erweiterungsmodul ist in Vorbereitung

  Kap. 9 "Sercos III IO " auf Seite 245
- S20 IO-Module können an der rechten Seite der XM2-Steuerung angeschlossen werden.

  Kap. 12 "Lokales S20 IO" auf Seite 303
- Der EtherCAT Master kann auf der onboard Ethernet-Schnittstelle XF1/XF2 alternativ zum Sercos verwendet werden

  Kap. 8.1 "Features des EtherCAT Master" auf Seite 175

3.2 Eigenschaften der Feldbusse

3.2.1 Feldbus-Master

XM2-Steuerung

	Profibus DP Master [8]	Profinet IO Controller [9]	EtherNet/IP Scanner [9]	EtherCAT Master	Sercos III-Master	S20 lokal
Max. Anzahl Teilnehmer	125	64 / 128 [1, 2]	64 [3]	128	256[5]	64 Module
Max. Anzahl zyklischer Eingangsdaten des Masters	3584 Byte	5652 Byte	4096 Byte	2048 Byte [10]	5600 Byte	1536 Byte [7]
Max. Anzahl zyklischer Ausgangsdaten des Masters	3584 Byte	5700 Byte	4096 Byte	2048 Byte [10]	5600 Byte	1536 Byte [7]
Max. Anzahl zyklischer Eingangsdaten pro Slave	244 Byte	1440 Byte	480 Byte [4]	2048 Byte [10]	5600 Byte	
Max. Anzahl zyklischer Ausgangsdaten pro Slave	244 Byte	1440 Byte	480 Byte [4]	2048 Byte [10]	5600 Byte	
Max. Telegrammdaten pro azyklischem Telegramm	240 Byte	1392 Byte	450 Byte		1500 Byte	

1 Details siehe Kap. 6.1 "Features des Profinet IO Controller" auf Seite 79

2 Die Zykluszeit (best case: 1 ms) hängt von der Anzahl der Devices und der Anzahl der E/A-Daten ab. Bei einer Überbelastung des PNIO Controllers kommt es zu Verbindungsabbrüchen in der zyklischen Kommunikation, was an einem ansteigenden "ErrorCount" im PNIO Controller "Status"-Dialog ersichtlich wird. In diesem Fall muss dann die Zykluszeit ("Send Clock" oder "Reduction Ratio") der Devices erhöht werden.

3 Wenn nur eine Verbindung pro Adapter eingerichtet wird, sind 64 Adapter möglich. Mehrere Verbindungen zu einem Adapter können eingerichtet werden, dann reduziert sich die maximale Anzahl der möglichen Adapter entsprechend.

4 Pro Verbindung, mehrere Verbindungen pro Adapter sind möglich.

5 Der Treiber unterstützt 256 IO-Ressourcen. Diese Zahl wird abhängig von der Steuerungs-Hardware durch die einzelnen Systemoberflächen reduziert.

7 Eingangsdaten und Ausgangsdaten liegen zusammen. Die Maximalzahl beträgt 1536 Byte

8 Profibus (XFE01.1-FB10)

9 RT-Ethernet (XFE01.1-FB03)

10 Die max. Anzahl an Prozessdaten des EtherCAT Masters ist 2048 Bytes (= Summe aus zyklischen Ein-/Ausgangsdaten). Es gibt keine Einschränkung hinsichtlich Ein-/Ausgangsdaten eines Slaves.

Tab. 3-2: Features der Feldbus-Master Schnittstellen

3.2.2 Feldbus-Slaves

XM2-Steuerung

	Profibus DP Slave [1]	Profinet IO Device [2]	EtherNet/IP Adapter [2]	EtherNet/IP Adapter (Eng.-Port)
Max. Anzahl zyklischer Eingangsdaten des Slaves	244 Byte	1024 Byte [3]	480 Byte	128 Byte
Max. Anzahl zyklischer Ausgangsdaten des Slaves	244 Byte	1024 Byte [3]	480 Byte	128 Byte + 4 Byte Header
Konsistenz der zyklischen Daten	128 Byte	128 Byte	128 Byte	128 Byte
Max. Telegrammdaten pro azyklischem Telegramm	240 Byte	1092 Byte [3]	450 Byte	

1 Profibus (XFE01.1-FB-10 oder XFE01.1-FB-11)
2 RT-Ethernet (XFE01.1-FB-03 oder XFE01.1-FB-04)
3 Details siehe Kap. 6.2 "Features des Profinet IO Device (Steuerung als Slave)" auf Seite 80

Tab. 3-3: Features der Feldbus-Slave-Schnittstellen

4 Feldbus übergreifende Features

4.1 Gerätedatenbank

4.1.1 Gerätedatenbank, Übersicht

Die Gerätedatenbank ist eine Ergänzung zur Gerätebibliothek und enthält die Gerätebeschreibungen, die auf dem lokalen System installiert sind, um sie für Projekte in IndraWorks verfügbar zu machen. In den Gerätebeschreibungsdateien werden Eigenschaften (z.B. Parameter, Hersteller, Gerätename usw.) des jeweiligen Gerätes hinterlegt.

Für unterschiedliche Geräte existieren verschiedene Formate

- **Profibus DP:** GSD-Datei, *.gs?,
- **Profinet IO:** GSD-Datei, GSDML*.xml,
- **EtherNet/IP:** EDS-Datei, *.xml,
- **EtherCAT:** ESI-Datei, *.xml
- **Sercos III IO:** SDDML-Datei, *.xml.
- **S20-IO:** SDDML-Datei (entspricht Sercos S20-Modul), *.xml

Die Gerätedatenbank wird bei der IndraWorks-Installation mit installiert.

Das Installieren zusätzlicher Geräte und das Deinstallieren nicht benötigter Geräte kann in der Gerätedatenbank selbst vorgenommen werden.

Die Gerätebeschreibungsdateien werden bei einem IndraWorks Projekt Archiv automatisch mit gesichert.

4.1.2 Geräte verwalten

Gerätedatenbank, Dialog

Symbol: 

Mit dem Befehl "Gerätedatenbank" können Sie Gerätebeschreibungsdateien installieren und deinstallieren. Klicken Sie im Hauptmenü auf **Extras ▸ Gerätedatenbank...**, um den Dialog "Gerätedatenbank" zu öffnen:

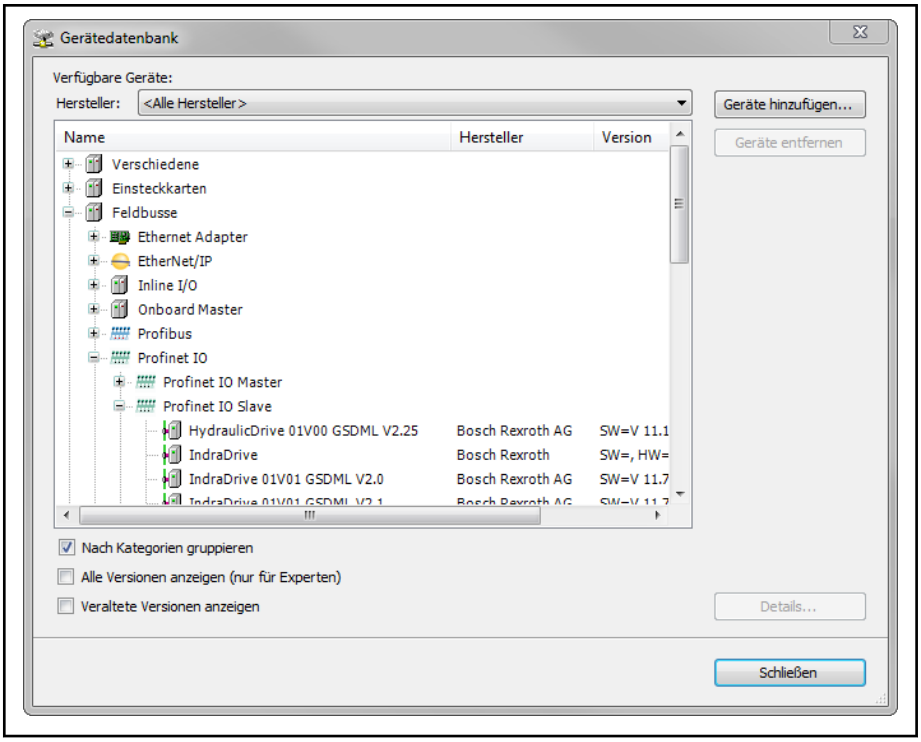


Abb. 4-1: Gerätedatenbank

Verfügbare Geräte

Im Bereich "Verfügbare Geräte" sind die Geräte aufgelistet, die aktuell installiert sind.

Die Liste kann durch Wahl des Filters "Hersteller" eingeschränkt werden.

Nach Kategorien gruppieren: Die Geräte sind hierarchisch geordnet, d.h. Steuerungen erscheinen in der obersten Ebene, darunter Funktionsmodule, integrierte Schnittstellen usw., anderenfalls werden die Einzelobjekte alphabetisch angeordnet.

Normalerweise werden die zu Ihrer IndraWorks-Version passenden Geräte gezeigt. Sie können jedoch auch den kompletten Überblick durch **Alle Versionen anzeigen (nur für Experten)** erhalten. Zusätzlich können Sie auch **Veraltete Versionen anzeigen**.

Weiterhin können Sie im Bereich "Verfügbare Geräte" Geräte hinzufügen und Geräte entfernen.

Haben Sie ein konkretes Gerät ausgewählt, stehen unter "Details" weitere Informationen zur Verfügung.

Geräte hinzufügen

Um Geräte in IndraWorks hinzuzufügen, führen Sie bitte folgende Schritte aus:

1. Entpacken Sie das komplette ZIP Archiv mit den enthaltenen Gerätebeschreibungsdateien und zugehörigen Icons.

⚠ WARNUNG	**Alle Gerätebeschreibungsdateien inklusive der iParSDD-Dateien, sofern verfügbar, sind zu entpacken!** **Wird dies nicht gemacht, ist in Safety-Applikationen der Editor „Sichere Parametrierung“ nicht verfügbar und bestehende Safety-E/A-Konfigurationen sind evtl. nicht mehr funktionsfähig.**

2. Klicken Sie im Hauptmenü auf **Extras ▸ Gerätedatenbank...**, um den Dialog "Gerätedatenbank" zu öffnen.
3. Klicken Sie im Dialog "Gerätedatenbank" auf die Schaltfläche "Geräte hinzufügen".

   Es öffnet sich der Dateiauswahl-Dialog "Gerätebeschreibung installieren":

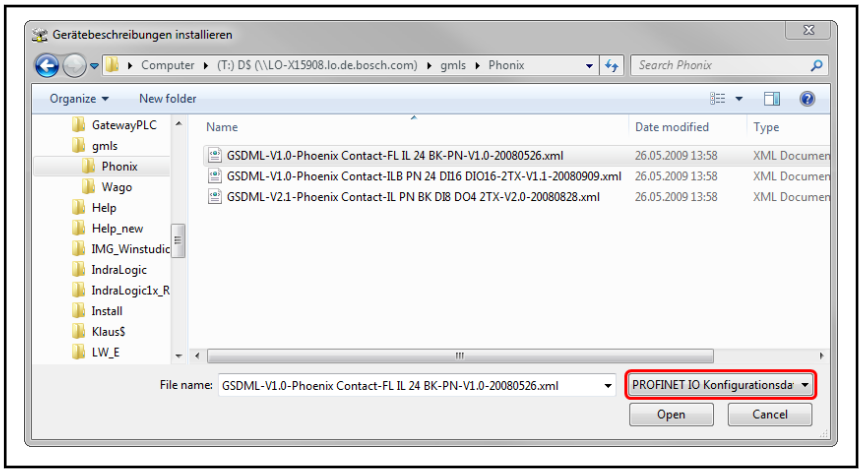


Abb. 4-2: Gerätebeschreibungen installieren

4. Wählen Sie die gewünschten Gerätebeschreibungsdateien, die Sie installieren möchten.

☞ Achten Sie auf den richtigen Dateityp im Auswahlfeld "Dateityp".

Alle zugehörigen Dateien z. b. Geräte-Icons werden automatisch hinzugefügt.

5. Starten Sie den Installationsvorgang, indem Sie auf "Öffnen" klicken.

   Nach dem Installationsvorgang erscheinen die importierten Geräte erscheinen unter "Verfügbare Geräte" und sind dort markiert:

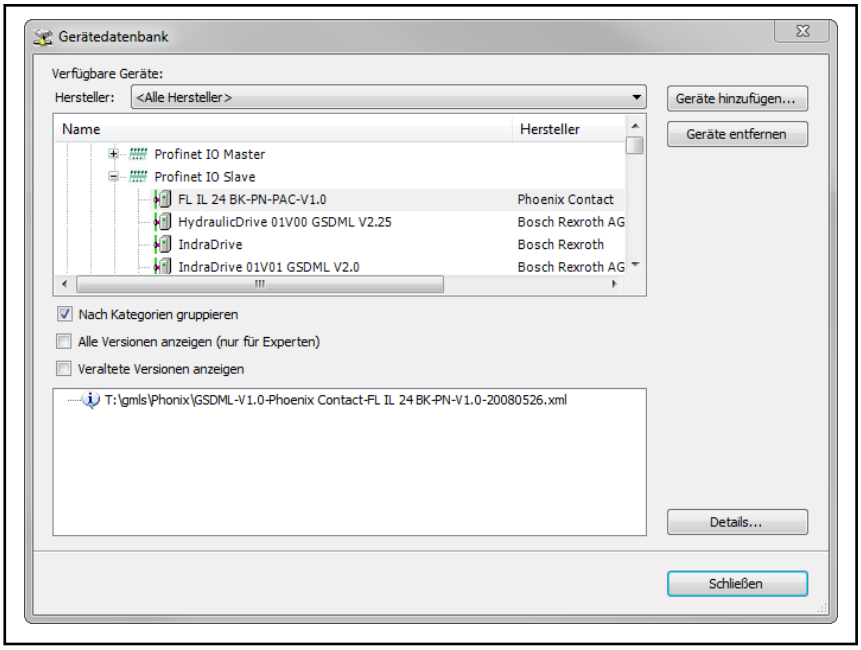


Abb. 4-3: Gerätebeschreibungen installiert

6. Wählen Sie das gewünschte Gerät, und Sie erhalten unter "Details" weitere Informationen.



Abb. 4-4: Details zur installierten Gerätebeschreibungsdatei

7. Beenden Sie den Dialog "Gerätedatenbank" mit "Schließen".
   Die neu installierten Geräte werden auch in der Bibliothek angezeigt:

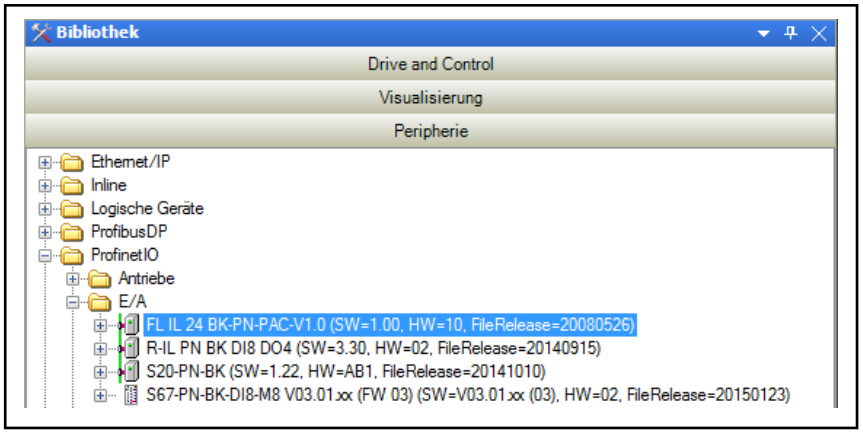


Abb. 4-5: Importiertes Gerät in der Bibliothek

Geräte entfernen

Um Geräte aus IndraWorks zu entfernen, führen Sie bitte folgende Schritte aus:

1. Klicken Sie im Hauptmenü auf **Extras ▸ Gerätedatenbank...**, um den Dialog "Gerätedatenbank" zu öffnen.
2. Markieren Sie im Gerätebaum die Geräte, die Sie entfernen möchten.
3. Klicken Sie auf die Schaltfläche "Geräte entfernen".

   Es öffnet sich der Dialog "Entfernen von Geräten bestätigen"

4. Klicken Sie auf "Ja", um das markierte Gerät zu entfernen.

   Das Gerät wird aus der Gerätedatenbank und aus der Bibliothek entfernt. (Dauert etwas...)

5. Beenden Sie den Dialog "Gerätedatenbank" mit "Schließen".

☞ S20-IO Module befinden sich in der Gerätedatenbank in folgendem Verzeichnis: ▸Feldbusse ▸Sercos ▸Modul ▸S20*.

Bitte beachten Sie, dass entfernte Module auch am Sercos Buskoppler S20-S3-BK+ fehlen.

4.2 Mapping der Feldbus-Ein- und Ausgänge

Die Erläuterungen gelten einheitlich für folgende Bus-Systeme:

- Inline (über Feldbuskoppler)
- S20
- Profibus DP
- Profinet IO
- EtherNet/IP
- Sercos III IO

Das erste Modul beginnt mit der Adresse "%IB0" bzw. "%IW0" für die Eingänge und analog "%QB0" bzw. "%QW0" für die Ausgänge.

Die weiteren Module folgen entsprechend, auch wenn sie bereits zum im Projektnavigator nachfolgenden Bus gehören.

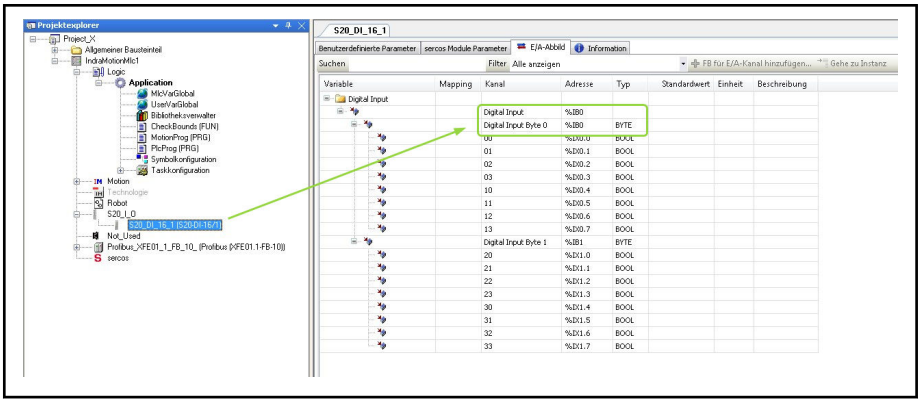


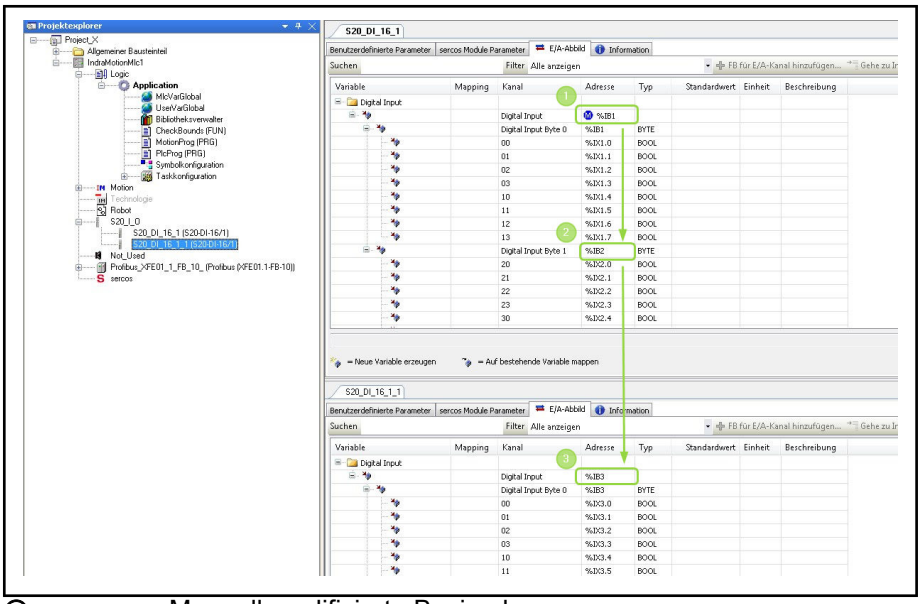
Abb. 4-6: Adressvergabe

Der Anwender kann jedoch die Vergabe der Adressen nach seinen Wünschen gestalten.

Ein Klick in das jeweilige Adressfeld erlaubt das Editieren der Adresse für das aktuelle Modul. Die übrigen folgen entsprechend.

Eine so modifizierte Adresse wird durch das weiße "M" auf blauem Grund gekennzeichnet.

Die weiteren Module folgen entsprechend, auch wenn sie bereits zum im Projektnavigator nachfolgenden Bus gehören.



① Manuell modifizierte Basisadresse
② Automatisch vergebene Folgeadresse
③ Automatisch vergebene Adresse des nächsten E/A-Moduls

Abb. 4-7: Modifizierte Basisadresse

☞ Bei mehrfachem Mapping besteht die Gefahr, dass sich Adressräume überschneiden.

Dies wird erst bei der **Code-Erzeugung** erkannt.

Die Fehlerausschrift enthält nur die zweite Vergabestelle....

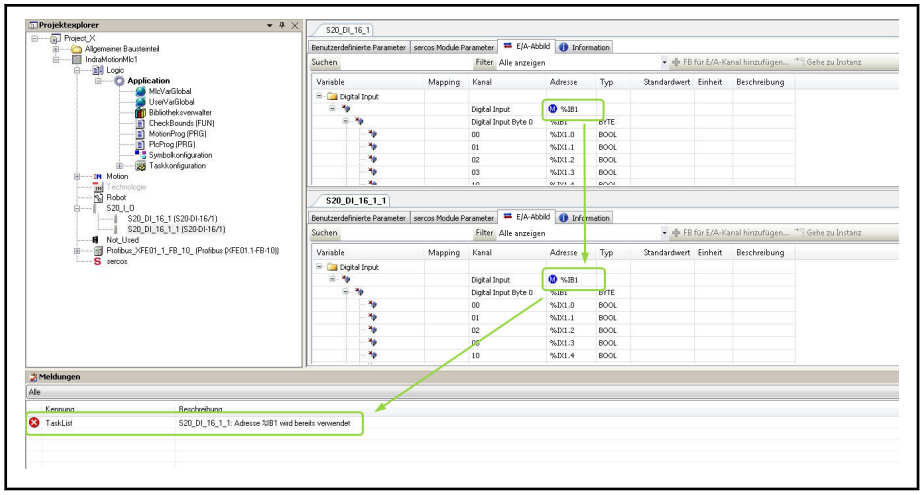


Abb. 4-8: Fehlerhaftes Mapping, Adressen mehrfach verwendet

Typgleiche E/A-Module auf dem jeweiligen Steckplatz, in der gleichen Reihenfolge, sind die Voraussetzung dafür, dass die automatische Adressgenerierung zur gleichen Belegung und damit zum Anschluss der richtigen Ein- und Ausgänge führt.

Kontrollieren Sie dies bitte gewissenhaft bei einem eventuellen Steuerungstausch.

Lokale Variable mappen:

- **Variable** :

  Diese Spalte zeigt das Eingabe- bzw. das Ausgabe-Modul an. Mit dem Plus- bzw. Minus-Symbol lässt sich zwischen Bit- und Byte-Darstellung umschalten.

  Für jede absolute Adresse ist die Zuweisung einer symbolischen Adresse möglich (Doppelklick auf das jeweilige Feld).

  Beim Abbilden auf eine bestehende Variable muss der komplette Pfad eingegeben werden, d.h.

  `<Applikationsname>.<Programmname>.<Variablenname>.`

  Beispiel (Abbildung unten):

  `Application.MotionProg.x_in_1,`

  `Application.MotionProg.x_in_2`

  bzw.

  `Application.MotionProg.x_out_1,`

  `Application.MotionProg.x_out_2.`

- **Mapping**:

  Das Symbol wird in der Mapping-Spalte angezeigt.

  Daraufhin wird der Adresswert durchgestrichen dargestellt, d.h. die %Qx.x-/ oder %Ix.x-Adresse.

- **Typ:**

  Byte-Adressen sind mit "BYTE", Bit-Adressen mit "BOOL" gekennzeichnet.

- **Aktueller Wert:**

  Physikalischer Zustand des Ein- und Ausgangs. Der Zustand wird nur im Diagnosemodus bei Kommunikation zwischen IndraWorks und der Steuerung angezeigt.
- **Einheit:**

  Einheit für den Parameterwert, z.B. "ms" für Millisekunden.
- **Beschreibung:**

  Geben Sie hier einen beliebigen Kommentar zu einer Adresse ein.

Globale Variable mappen:

- **Variable**:

  Diese Spalte zeigt das Eingabe- bzw. das Ausgabe-Modul an. Mit dem Plus- bzw. Minus-Symbol lässt sich zwischen Bit- und Byte-Darstellung umschalten.

  Für jede absolute Adresse ist die Zuweisung einer symbolischen Adresse möglich (Doppelklick auf das jeweilige Feld).

  Beim Abbilden darf **nur der Name der Variablen** eingegeben werden, d.h.

  `<Variablenname>`.

  Beispiel (Abbildung unten):

  `x_in_3` bzw.

  `x_out_3`, `x_out_4`.
- **Mapping**:

  Das Symbol  wird in der Mapping-Spalte angezeigt.

  Die Variable wird als VAR_GLOBAL in die "IoConfig_Globals_Mapping"-Liste eingetragen.
- **Typ:**

  Byte-Adressen sind mit "BYTE", Bit-Adressen mit "BOOL" gekennzeichnet.
- **Aktueller Wert:**

  Physikalischer Zustand des Ein- und Ausgangs. Der Zustand wird nur im Diagnosemodus bei Kommunikation zwischen IndraWorks und der Steuerung angezeigt.
- **Einheit:**

  Einheit für den Parameterwert, z.B. "ms" für Millisekunden.
- **Beschreibung:**

  Geben Sie hier einen beliebigen Kommentar zu einer Adresse ein.

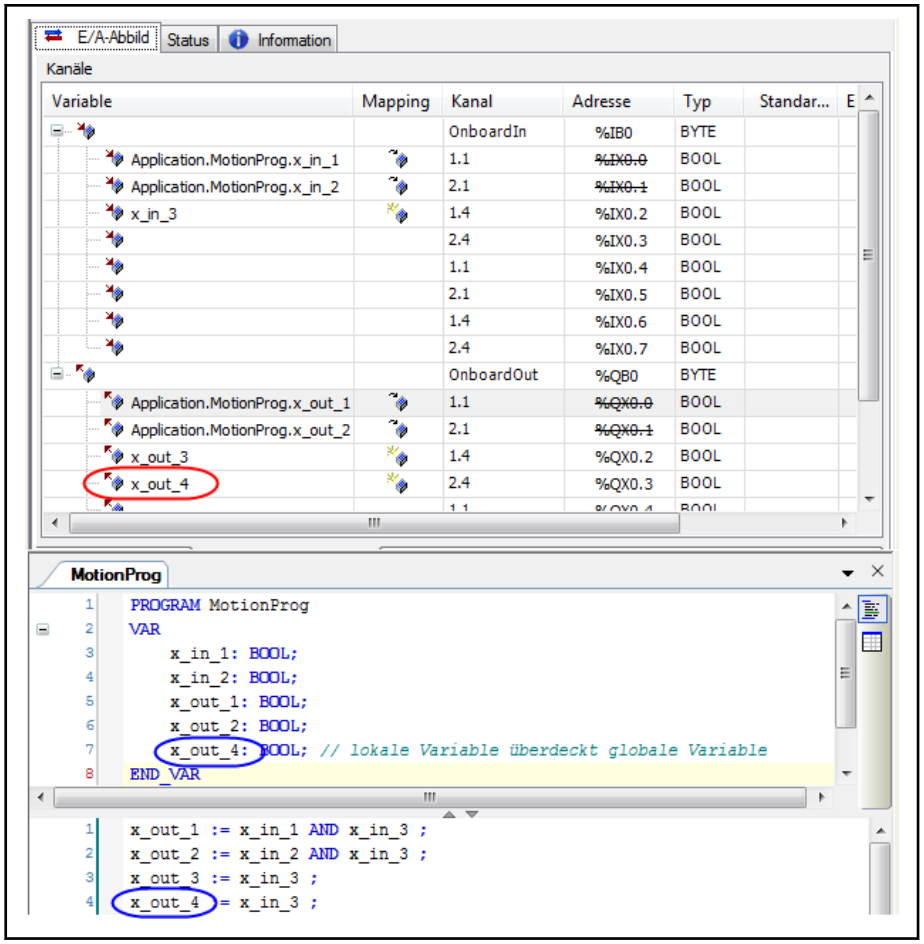


Abb. 4-9: IO-Mapping, Variablendeklaration und Beispielprogramm (Offline)

Der Deklarationsteil der POU "MotionProg" enthält

- die lokalen Variablen der POU,
- die Variablen, die als Ein- und Ausgänge gemappt werden sollen (ohne AT-Konstruktion!).

☞ Deklarieren Sie lokal eine namensgleiche Variable, wie in der globalen Liste bereits vorhanden, wird die lokale Variable verwendet!

Beispiel: `x_out_4`.

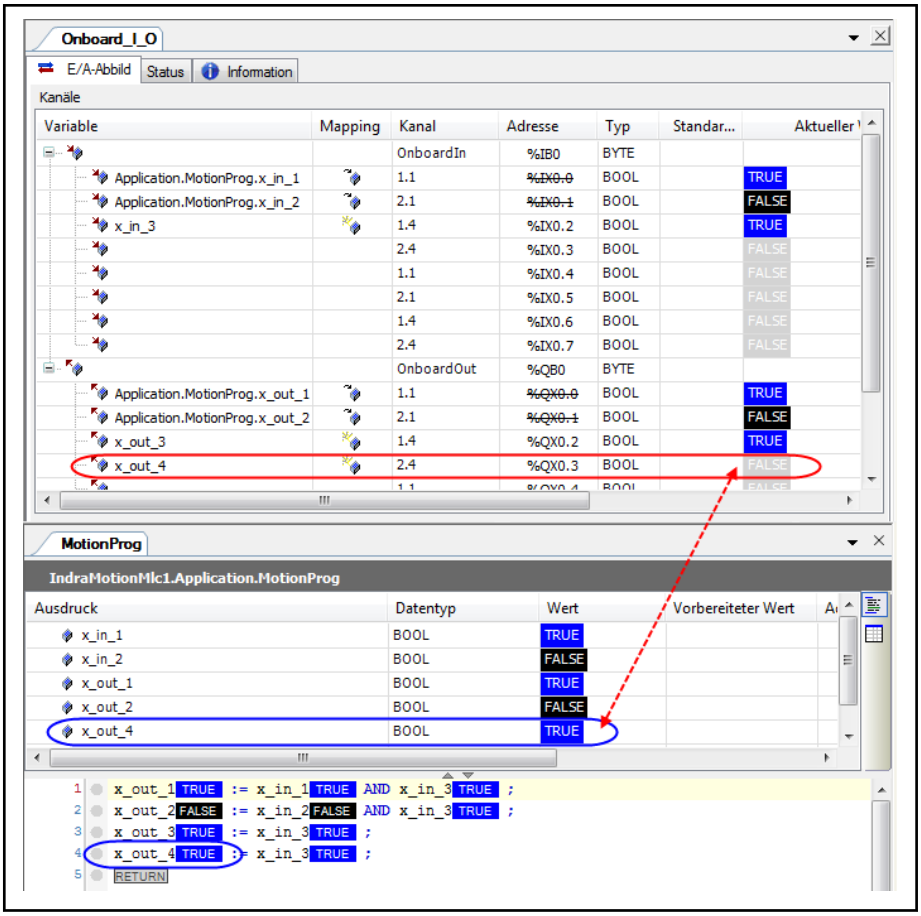


Abb. 4-10: IO-Mapping, Variablendeklaration und Beispielprogramm (Online)

Mapping zurücksetzen: Über diese Schaltfläche können die Mapping-Einstellungen auf die durch die Gerätebeschreibungsdatei definierten Standardwerte zurückgesetzt werden.

Variablen aktualisieren: Festlegung für das Modul , ob die E/A-Variablen in der Buszyklus-Task aktualisiert werden.

Der Default-Wert ist in der Gerätebeschreibung definiert.

- **Einstellungen des übergeordneten Geräts verwenden:** Aktualisierung gemäß den Einstellungen des übergeordneten Geräts.
- **Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet):** Die E/A-Variablen werden in der Buszyklus-Task aktualisiert, falls sie in keiner anderen Task verwendet werden
- **Aktiviert 2 (immer im Buszyklus-Task):** Alle Variablen werden in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

4.3 Feldbusmapping, Zugriff vom Feldbusmaster auf Variablen des Feldbusslaves

Grundlagen

Das Feldbusmapping ermöglicht den Zugriff auf Variablen der Steuerung von einem übergeordneten Feldbusmaster aus.

Folgende Feldbusse werden unterstützt:

- Profibus DP
- Profinet IO

- EtherNet/IP

Wenn man auf eine Variable der Steuerung von einem übergeordneten Master aus zugreifen will, dann muss dieser Variablen zuerst eine Adresse zugeordnet werden (Mapping). Über diese Adresse kann der übergeordnete Master über azyklische Dienste auf die Daten der Steuerung zugreifen (Feldbus-Slaves, Seite 14).

Das Mapping von Variablen kann über das Register "Feldbusmapping" des Feldbus-Slave-Dialogs oder über die Funktionsbausteine der Bibliothek RIL_MappingList.library erfolgen. An dieser Stelle wird der Konfigurator-Dialog "Feldbusmapping" beschrieben.

Um das Feldbusmapping nutzen zu können, muss die Steuerung als Feldbusslave konfiguriert sein.

Die einzelnen Schritte werden anhand des Profinet IO gezeigt. Sie sind für die übrigen Busse analog.

Konfiguration

Im Geräteeditor der jeweiligen Anschaltung gibt es das Registerblatt "Feldbusmapping". Hier können in einer Tabelle einzelnen Variablen Adressen zugeordnet werden.

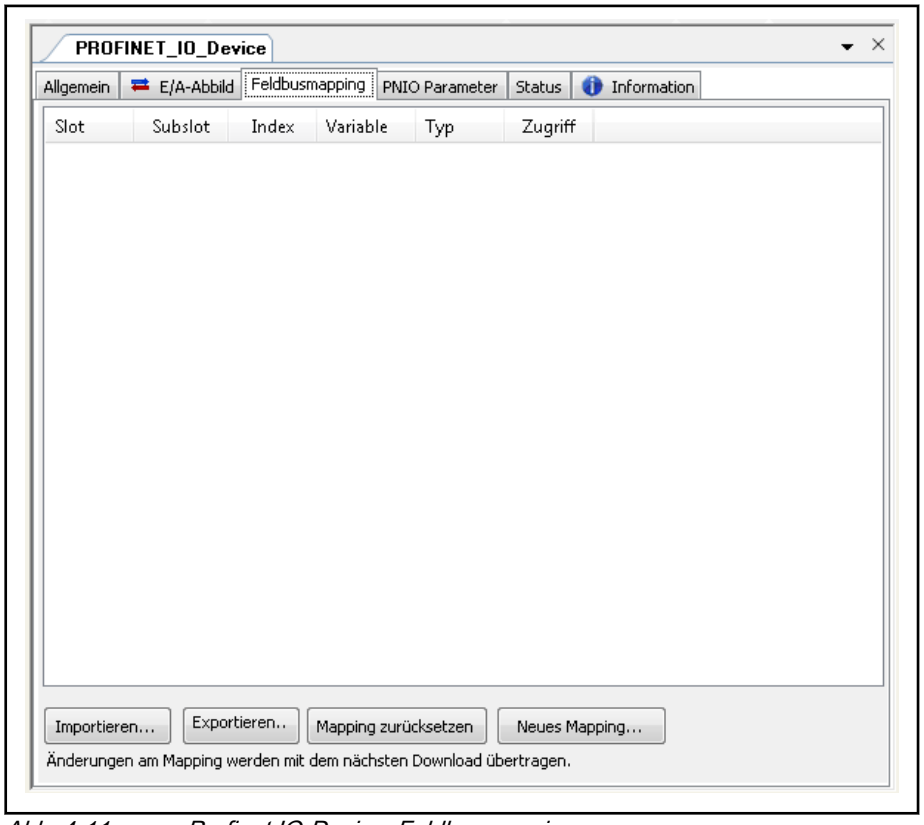


Abb. 4-11: Profinet IO-Device: Feldbusmapping

Neues Mapping erstellen

Über die Schaltfläche "Neues Mapping…" kann man einen neuen Eintrag hinzufügen. Es öffnet sich ein Dialog zur Auswahl einer Variablen.

Gewählt werden können Variablen, die innerhalb der Applikation deklariert sind:

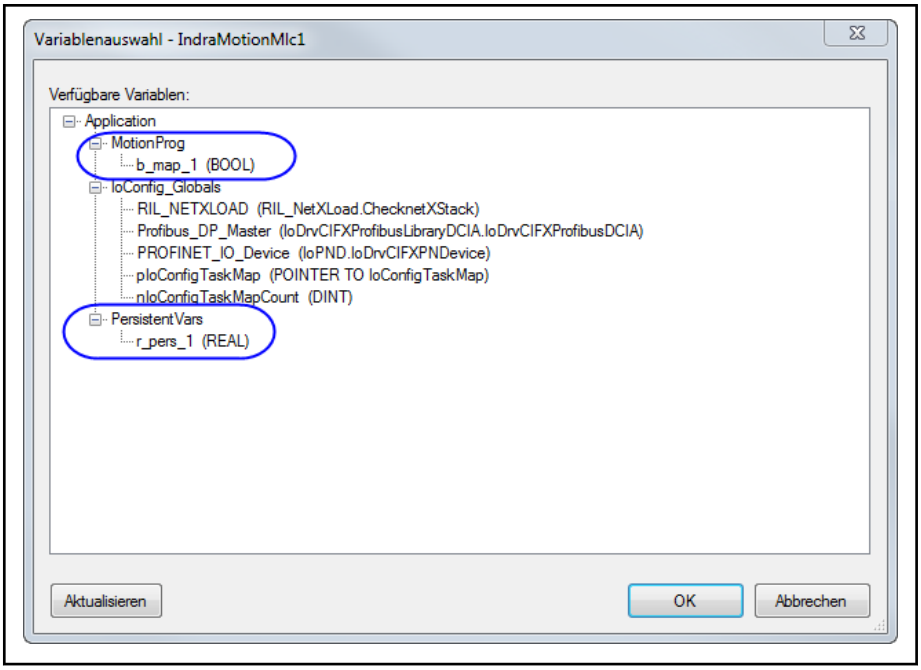


Abb. 4-12: Profinet IO-Device: Feldbusmapping, Auswahl der Variablen

Wenn man die Auswahl mit "OK" bestätigt, wird für die ausgewählte Variable ein neuer Eintrag in der Tabelle erzeugt.

Zusätzlich wird die Variable der Symbolkonfiguration hinzugefügt.

Die Variablenauswahl enthält nur Variablen, die bis zum letzten Kompilieren vorhanden waren. Wurden seitdem neue Variablen dem SPS-Programm hinzugefügt, so sind diese erst nach erneutem Kompilieren sichtbar. Dies kann über die Schaltfläche "Aktualisieren" erreicht werden.

Adresse, Variable und Zugriff können nachträglich durch einen Klick auf die entsprechende Tabellenzelle geändert werden. Nicht editierbare Spalten sind grau.

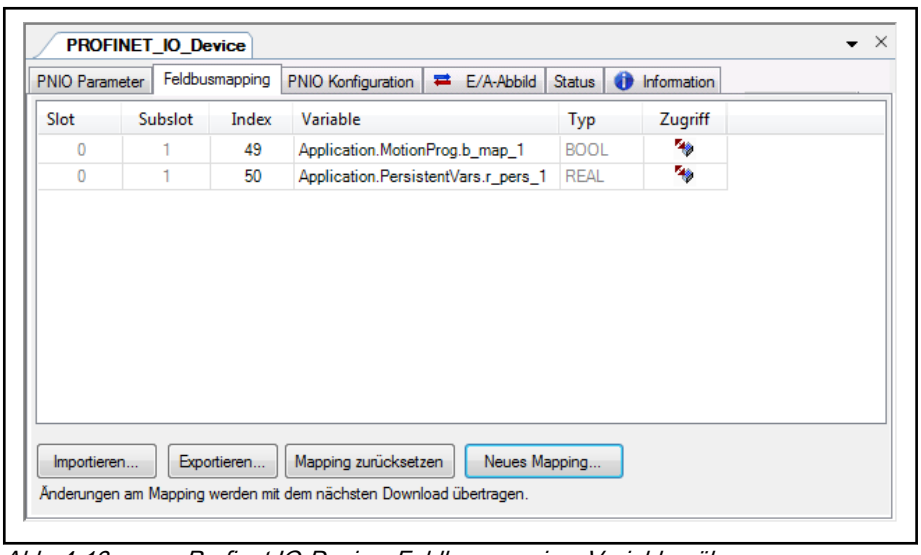


Abb. 4-13: Profinet IO-Device: Feldbusmapping, Variablen übernommen

Über die ersten Spalten der Tabelle kann die Adresse festgelegt werden, wobei die Werte in den grauen Feldern nicht änderbar sind.

- Für **Profibus DP** besteht die Adresse aus **Slot** und **Index**. Hier ist Slot immer 0. Index muss im Bereich von 49 bis 254 liegen.
- Für **Profinet IO** besteht die Adresse aus **Slot**, **Subslot** und **Index**. Hier sind Slot immer 0 und Subslot immer 1. Index muss im Bereich von 49 bis 32767 liegen.
- Für **EtherNet/IP** besteht die Adresse aus **Klasse**, **Instanz** und **Attribut**. Hier sind Klasse immer 199, Instanz muss im Bereich von 2 bis 65535 liegen und Attribut ist immer 1.

Die Spalte **Variable** zeigt den Instanzpfad der Variablen an.

Die Spalte **Typ** gibt den Datentyp der Variablen an.

Die Spalte **Zugriff** zeigt die Zugriffsrechte für die Variable an.

Mapping zurücksetzen Über die Schaltfläche "Mapping zurücksetzen" können alle Tabelleneinträge gelöscht werden.

Import/Export Es besteht die Möglichkeit über die Schaltfläche "Exportieren…" ein vorhandenes Mapping in eine Datei zu speichern und später über "Importieren…" wieder einzulesen.

Bei dieser Datei handelt es sich um eine einfache Textdatei.

So ist es möglich auch ohne vorherigen Export eine solche Datei per Hand zu erstellen.

Die Exportdatei enthält den Namen des Feldbusses „EthernetIP“, „ProfinetIO“, ProfibusDP (Schreibweise entsprechend Exportdatei) und eine Tabelle (siehe unten).

Die Adresse ist in mehrere Spalten unterteilt und setzt sich wie folgt zusammen:

- ProfibusDP: Slot, Index,
- ProfinetIO: Slot, Subslot, Index,
- EthernetIP: Class, Instance, Attribute.

Diese Tabelle beinhaltet ferner den Variablennamen, den Datentyp und die Zugriffsrechte der gemappten Variablen.

```
ProfibusDP
  # Slot    Index Variablename                          Datatyp Access right
     0      49    Application.MotionProg.b_map_01       BOOL    ReadWrite
     0      50    Application.PersistentVars.r_map_01 REAL    ReadWrite
```

Abb. 4-14: Auszug aus der Exportdatei Profibus DP, Trennzeichen TAB:

```
ProfinetIO
# Slot Subslot Index Variablename                          Datatyp Access right
  1    1       49    Application.MotionProg.b_map_01       BOOL    ReadWrite
  1    1       50    Application.PersistentVars.r_map_01 REAL    ReadWrite
```

Abb. 4-15: Auszug aus der Exportdatei Profinet IO, Trennzeichen TAB:

```
EthernetIP
# Class Inst Attr Variablename                          Datatyp  Access rights
  199   2    1    Application.MotionProg.b_map_01       BOOL     ReadWrite
  199   3    1    Application.PersistentVars.r_map_01 REAL     ReadWrite
```

Abb. 4-16: Auszug aus der Exportdatei EtherNet/IP, Trennzeichen TAB:

4.4 Feldbuskonfiguration "Maximalprojekt"

Bei Maschinen gibt es oft verschiedene Varianten und Ausprägungen. Für ein effizientes Software-Engineering sind hierfür nicht mehrere Projektkonfigurationen zu pflegen, sondern es kann **ein** IndraWorks-Projekt erstellt werden, und für eine spezifische Maschinen-Variante werden dann die nicht benötigten Komponenten deaktiviert. Diese Vorgehensweise wird auch als "Maximalprojekt" bezeichnet.

Das "Deaktivieren" von Feldbus-Komponenten ist dahingehend sinnvoll, dass bei konfigurierten, aber physikalisch nicht vorhandenen Komponenten keine Fehler- bzw. Diagnosemeldungen generiert werden.

Durch die im Dateisystem der Steuerung abliegenden Datei "UserDefs.cfg" werden Feldbus-Geräte und unterlagerte Module, die im IndraWorks-Projekt konfiguriert sind, während **des Hochlaufs** der Steuerung deaktiviert. Das IndraWorks-Projekt, inkl. SPS-Kompilat, wird hierbei nicht verändert.

Deaktivieren von Remote-Geräten via "UserDefs.cfg", Übersicht

	Deaktivieren von Remote-Slaves	Deaktivieren von Modulen eines Remote-Slaves
Sercos IO	✓	✓
Profinet IO	✓	✓
Profibus DP	✓	–
EtherNet/IP	– [1)]	–
EtherCAT	–	–
Lokalbus S20 (XM)	(–)	✓ (ab 14V22)

1) in Vorbereitung

Werden mehrere identische Steuerungen unterhalb einer übergeordneten Steuerung betrieben, müssen diese sich bei einer Feldbus-Kommunikation hinsichtlich der Slave-Schnittstellen-Konfiguration unterscheiden (mindestens bezüglich der Adressierung, gegebenenfalls auch bei Konfigurationsmodulen). Hier kann die Slave-Schnittstellen-Konfiguration mittels "UserDefs.cfg" überschrieben werden, so dass das IndraWorks-Projekt für diese untergeordneten Steuerungen identisch (= einmalig) erstellt werden kann:

Überschreiben einer lokalen Feldbus-Slave-Schnittstelle via "UserDefs.cfg", Übersicht

	Überschreiben der lokalen Feldbus-Slave-Adressierung	Deaktivieren von Modulen der lokalen Feldbus-Slave-Schnittstelle
Sercos IO	–	–
Profinet IO	✓	✓
Profibus DP	–	–
EtherNet/IP	–	–
EtherCAT	–	–

Siehe Details zur "UserDefs.cfg"-Erstellung

- Sercos-Slaves / Sercos-Module deaktivieren mittels UserDefs.cfg auf Seite 262

- Profinet IO-Devices und Profinet IO-Module deaktivieren mittels UserDefs.cfg auf Seite 99
- Profibus DP-Slaves deaktivieren mittels UserDefs.cfg auf Seite 61
- S20-Module deaktivieren mittels 'UserDefs.cfgauf Seite 313

4.5 Buszyklus-Task: Kopieren der E/A-Daten zwischen SPS und Feldbus-Stack

4.5.1 Beschreibung der Buszyklus-Task

Die Buszyklus-Task kopiert die E/A-Daten zwischen den Feldbus-Stacks und den SPS I/O-Buffern.

Die I/O-Buffer sind ein Zwischenspeicher zwischen Feldbus-Stacks und SPS E/A-Abbild. Die SPS E/A-Daten werden in ② und ④ aktualisiert:

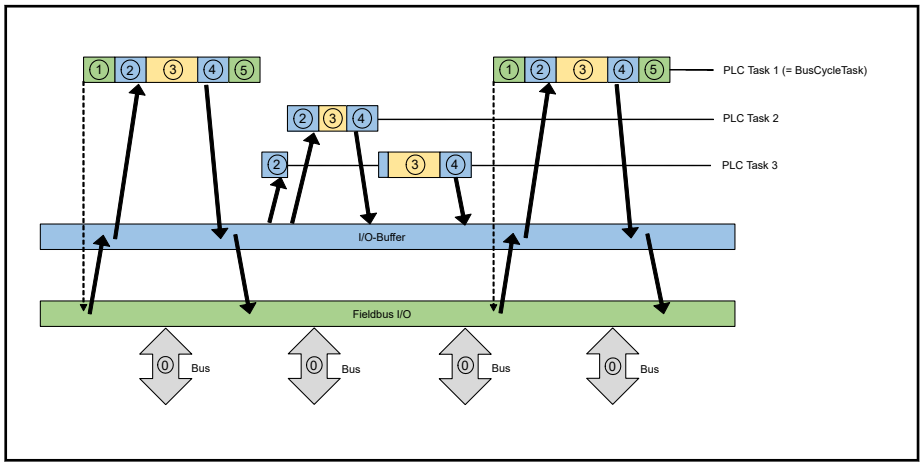


Abb. 4-17: Prinzipdarstellung Buszyklus-Task

⓪ Bus-Kommunikationszyklus

① Kopieren der Feldbus-Eingangsdaten in die SPS I/O-Buffer (Buszyklus-Task)

② Aktualisieren der SPS-Eingangsdaten, die in der jeweiligen SPS-Task verwendet werden (Lesen der I/O-Buffer)

③ Ausführung der POUs, die dieser SPS-Task zugewiesen sind

④ Schreiben der SPS-Ausgangsdaten, die in der jeweiligen SPS-Task verwendet werden (Schreiben in die I/O-Buffer)

⑤ Kopieren der SPS I/O-Buffer zu den Feldbus-Ausgangsdaten (Buszyklus-Task)



Bei Sercos und EtherCAT findet das Kopieren zwischen Feldbus-Stack und SPS I/O-Buffer innerhalb des Kommunikationszyklus statt (und nicht durch die Buszyklus-Task).

Datenkonsistenz

Der Kopiervorgang zwischen Feldbus-Stack und SPS I/O-Buffer kann von einer höherprioren SPS-Task nicht unterbrochen werden. Zudem kann die Buszyklus-Task auch nicht das Lesen u. Schreiben der SPS E/A-Daten einer anderen SPS-Task unterbrechen, d.h. die Daten im SPS I/O-Buffer sind konsistent.

Bei Verwendung von gleichen Eingängen in verschiedenen Tasks kann es passieren, dass sich die Eingangsdaten während der Abarbeitung einer Task verändern. Dies geschieht, wenn diese Task durch eine Task mit höherer Priorität unterbrochen und damit das Prozessabbild erneut eingelesen wird.

Beachte: Es gibt kein lokales Task E/A-Abbild, sondern in ② werden die SPS-Eingangsdaten global aktualisiert.

Empfehlung: Die gleichen Ein- und Ausgänge sollten immer nur in einer Task verwendet werden.

Siehe auch Kap. 4.5.4 "SPS-Einstellung Variablen aktualisieren" auf Seite 32.

Die Datenkonsistenz innerhalb des Feldbus E/A-Abbild ist Feldbus-spezifisch. Typischerweise sind mindestens die Daten einer Slave bzw. Kommunikationsverbindung konsistent.

4.5.2 Konfiguration der Buszyklus-Task

Es kann global eine Buszyklus-Task eingestellt werden, oder es können individuell je Feldbus-Stack verschiedene Buszyklus-Tasks konfiguriert werden.

Im Geräteeditor der Steuerung kann in der Registerkarte "SPS-Einstellungen" eine globale Buszyklus-Task ausgewählt werden (Aufruf mittels Doppelklick auf Steuerungsknoten im IndraWorks-Projektexplorer).

Der Defaultwert "*<Unspezifiziert>*" bedeutet, dass die schnellste zyklische SPS-Task (= Task mit kleinster Zykluszeit) verwendet wird.

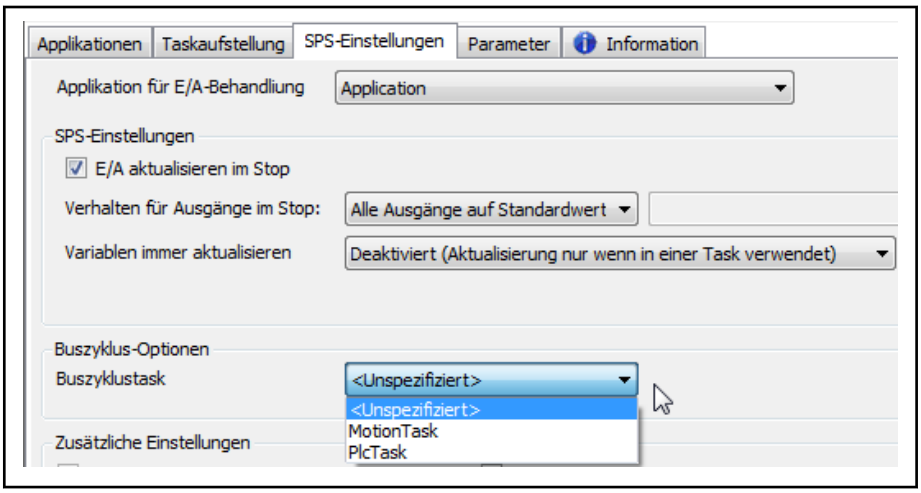


Abb. 4-18: Konfiguration der Buszyklus-Task im Dialog "SPS-Einstellungen"

Hinweise zur Motion-Task:

- Die Motion-Task ist keine zyklische SPS-Task, da diese extern getriggert wird.
- Es wird empfohlen die Motion-Task nicht als Buszyklus-Task zu verwenden. Bei manchen Feldbussen (z.B. PROFINET IO Controller) werden innerhalb der Buszyklus-Task neben dem Kopieren der zyklischen Daten auch weitere Funktionen ausgeführt, die die Motion-Tasklaufzeit negativ beeinflussen können.

Bei einem Feldbus-Master oder Feldbus-Slave (= Steuerung ist Slave) kann im Dialog "E/A-Abbild" eine Buszyklus-Task für diesen Feldbus-Stack ausgewählt werden.

Die Defaulteinstellung "*Zykluseinstellungen des übergeordneten Busses verwenden*" bedeutet, dass die globale Buszyklus-Task aus den "SPS-Einstellungen" übernommen wird.

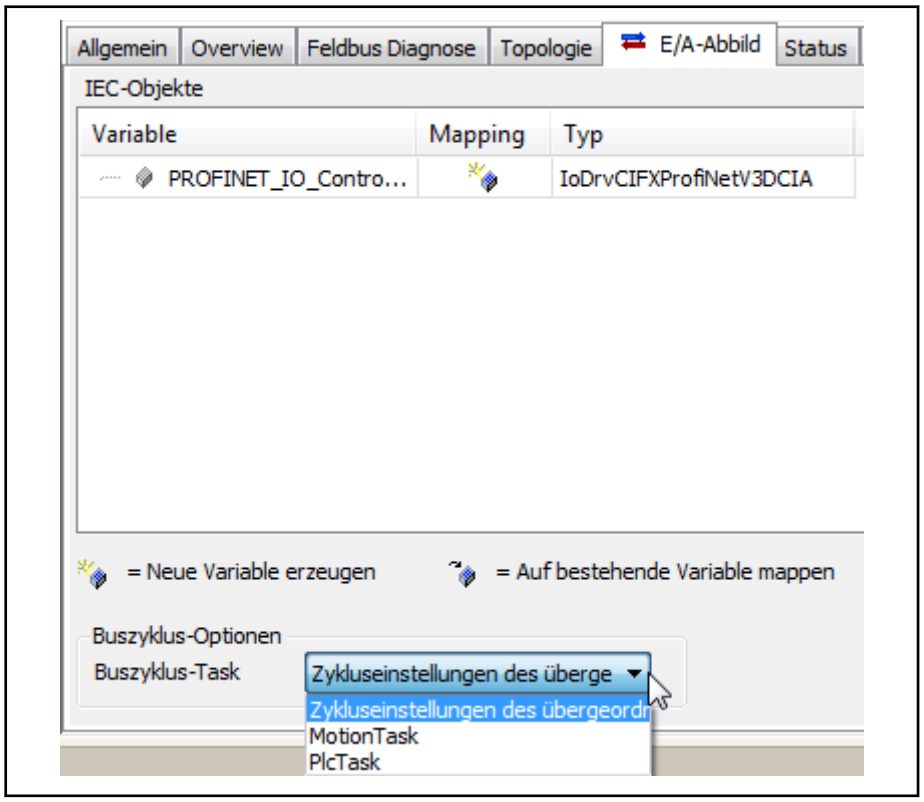


Abb. 4-19: Konfiguration der Buszyklus-Task im Dialog eines Master oder Slave-Knotens

Tipp: Der Dialog "Taskaufstellung" (Doppelklick auf Steuerungsknoten) zeigt an, in welcher Task die E/A-Daten verarbeitet werden.

4.5.3 Bus-synchrone SPS-Verarbeitung

Eine zyklische SPS-Task ist prinzipiell nicht synchron zu einer Feldbus-Kommunikation.

Bei manchen Bussystemen (Sercos, EtherCAT, S20-Lokalbus) kann eine extern getriggerte SPS-Task konfiguriert werden, so dass innerhalb dieser Task eine Bus-synchrone SPS-Verarbeitung möglich ist.

Beispiel **S20-Lokalbus**:

- Für eine Bus-synchrone SPS-Verarbeitung ist eine SPS-Task anzulegen vom Typ "Extern" mit dem Ereignis EVENT_OP_MODE_S20_CYCLIC.
- Diese getriggerte Task ist zudem als Buszyklus-Task des S20 IO Master einzustellen, z.B. auf dem Lokalbus S20 IO Master im Dialog "E/A-Abbild".

- Optional kann zusätzlich beim S20 Lokalbus die Betriebsart taktsynchron zu Sercos eingestellt werden (siehe S20 Lokalbus Master Dialog "Einstellungen").

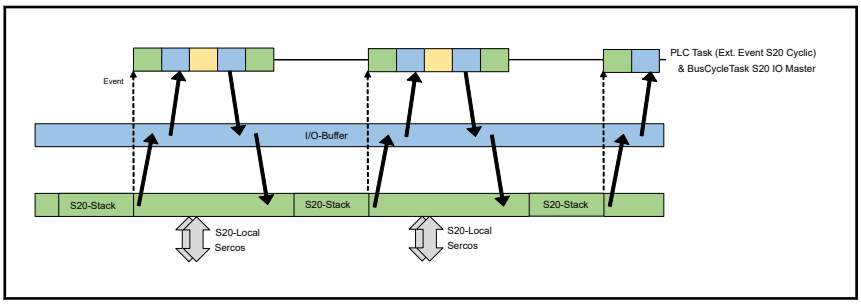


Abb. 4-20: Beispiel für EVENT_OP_MODE_S20_CYCLIC

Beispiel **Sercos**:

- Für eine Bus-synchrone SPS-Verarbeitung ist eine SPS-Task anzulegen vom Typ "Extern" mit dem Ereignis EVENT_OP_MODE_SERCOS_CYCLIC.
- Diese getriggerte Task wird dann vom Sercos-Stack synchron zur Bus-Verarbeitung aufgerufen. Abhängig von der Applikation sollte bei dieser Sercos-Event Task dann bei Bedarf eine höhere Priorität (z.B. Prio 2) als bei der Motion-Task (z.B. Prio 3) eingestellt werden; beispielsweise wenn die Motion-Taskausführung nicht innerhalb eines Sercos-Zyklus abgeschlossen werden kann.

  Exemplarische Konfiguration mit 1 ms Sercos Zykluszeit und 2 ms Motion-Zykluszeit:

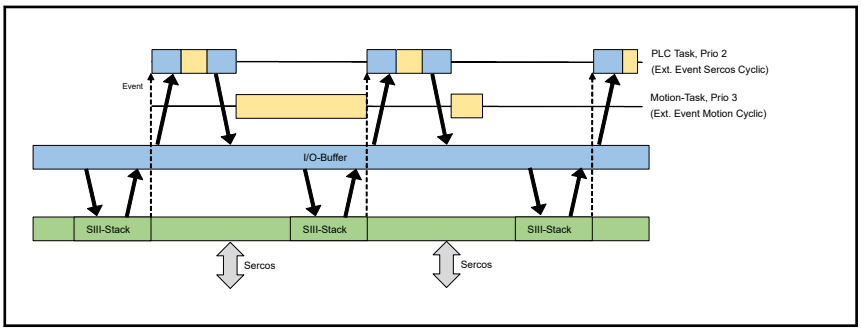


Abb. 4-21: Beispiel für EVENT_OP_MODE_SERCOS_CYCLIC

Beispiel **EtherCAT**:

- Siehe Kap. 8.7 "EtherCAT Master, Bus synchrone SPS-Verarbeitung" auf Seite 239

4.5.4 SPS-Einstellung "Variablen aktualisieren"

Die Einstellung "Variablen aktualisieren" gibt es

- sowohl global unter "SPS-Einstellungen" (Doppelklick auf Steuerungsknoten),
- als auch individuell bei einem Feldbus-Slave (oder unterlagerten E/A-Modul) im Dialog "E/A-Abbild".

Optionen:

- **Deaktiviert**: In der Defaulteinstellung (global) *"Deaktiviert"* werden nur die E/A-Daten aktualisiert, die in einer SPS-Task verwendet werden.
- **Aktiviert 1**: Wenn man die Feldbus E/A-Daten im eingeloggten Zustand beobachten oder forcen möchte, z.B. während einer Inbetriebnahme einer Anlage, ohne dass die E/A-Daten bereits in einer SPS-Task verwen-

det werden, dann muss diese Einstellung auf *"Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet)"* gesetzt werden.

☞ Die IndraWorks Dialoge "E/A-Abbild" zeigen allgemein immer das SPS E/A-Abbild an (und nicht die I/O-Buffer). Mit der Einstellung "Aktiviert 1" werden alle E/A-Daten, die in keiner SPS-Task verwendet werden, mit der Buszyklus-Task zwischen I/O-Buffer und SPS E/A-Abbild kopiert.

Diese Einstellung betrifft alle Bussysteme (auch Sercos und EtherCAT).

- **Aktiviert 2**: In der Einstellung *"Aktiviert 2 (immer in Buszyklus-Task)"* werden alle Variablen bzw. SPS E/A-Daten immer in der Buszyklus-Task aktualisiert und nicht in der Task, in der sie verwendet werden[1)] . Es wird empfohlen, z.B. auch bei einer Inbetriebnahme, nicht die Einstellung "Aktiviert 2" sondern die Einstellung "Aktiviert 1" zu verwenden.

4.6 IP-Kanal bei PROFINET IO & EtherNet/IP (netX)

Allgemeines

Unter dem IP-Kanal versteht man die IP-Kommunikationsanbindung der Ethernet basierten Feldbus-Stacks zum Betriebssystem der Steuerung. Hierbei findet auch ein IP-Routing ("Forwarding") von der Feldbus-Schnittstelle zum Engineering-Port der Steuerung statt.

Mit IndraWorks Version 14V22 wurde der IP-Kanal durch die Umstellung auf netX V3 grundlegend überarbeitet.

Der IP-Kanal wird wie folgt unterstützt:

Steuerung als ...	bis 14V20	ab 14V22
PROFINET IO Controller	IP-Kanal automatisch aktiv	IP-Kanal manuell aktivierbar
PROFINET IO Device	IP-Kanal automatisch aktiv	IP-Kanal manuell aktivierbar
EtherNet/IP Scanner	Kein IP-Kanal	IP-Kanal manuell aktivierbar
EtherNet/IP Adapter	IP-Kanal automatisch aktiv	IP-Kanal manuell aktivierbar

Steuerungssystem / Unterstützung

- Die IP-Kanal Anbindung wird vom System MLC unterstützt
- Aktuell keine Unterstützung bei MTX

☞ Der IP-Kanal ist nur dann funktionsfähig, wenn die IndraWorks Version und die Steuerungs-Firmware zusammenpassen.

Registerkarte 'IP-Kanal-Einstellungen'

Der IP-Kanal kann in der Registerkarte "IP-Kanal-Einstellungen" aktiviert werden (ab 14V22). Es kann ein anderes Subnetz verwendet werden als die Feldbus-Kommunikation.

1) *D.h. bei "Aktiviert 2" wird das Kopieren zwischen SPS E/A-Daten und I/O-Buffer, siehe ② und ④ in Abb. 4-17 "Prinzipdarstellung Buszyklus-Task" auf Seite 29, immer in der Buszyklus-Task ausgeführt und nicht in der Task, in denen die E/A-Daten verwendet werden.*

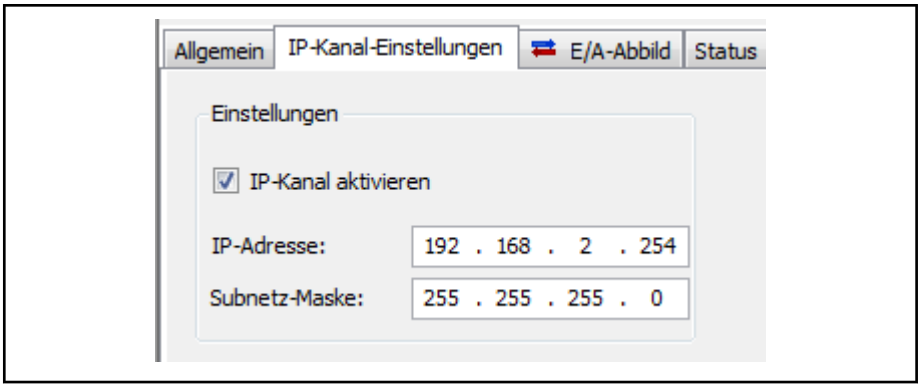


Abb. 4-22: Registerkarte 'IP-Kanal-Einstellungen'

Folgende Randbedingungen sind zu beachten

- Die IP-Adresse des IP-Kanals muss sich von den IP-Adressen der Feldbus-Teilnehmer unterscheiden!

  D. h. auch die IP-Adresse der lokalen Feldbus-Schnittstelle der Steuerung (PNIO Controller / Device, ENIP Scanner / Adapter) darf nicht identisch sein!
- Des Weiteren ist zu beachten, dass der IP-Kanal nur dann aktiviert werden kann, wenn keine weitere Schnittstelle der Steuerung (z.B. Engineering-Port oder IP-Kanal einer weiteren Feldbus-Schnittstelle) das gleiche Subnetz hat! → Im Fehlerfall entsteht ein Eintrag im SPS-Logbuch.
- Bezüglich **IP-Routing** über den Eng.-Port der Steuerung muss zudem die Gateway-Adresse in den Feldbus-Teilnehmern auf die IP-Kanal Adresse gesetzt werden (siehe Anwendungsbeispiel im nachfolgenden Kapitel).

  Hierbei ist zu beachten, dass die Gateway-Adresse eines Feldbusteilnehmers im gleichen Subnetz sein muss wie die IP-Adresse des Gerätes. Für den Anwendungsfall IP-Routing über den Engineering-Port der Steuerung muss die IP-Kanal Adresse also nicht nur als Gateway in den Feldbus-Slaves eingetragen werden, sondern muss auch im gleichen Subnetz wie die Feldbus-Geräte sein.

Anwendungsbeispiele:

IP-Routing über den Engineering Port der Steuerung

In diesem Konfigurationsbeispiel gibt es folgende IP-Einstellungen:

- Engineering Port der Steuerung: 10.110.245.40 (⇔ Netzwerk zum Eng.-PC)
- EtherNet/IP Scanner: 192.168.2.1
- IP-Kanal: 192.168.2.254 (⇔ muss unterschiedlich sein zum EtherNet/IP Scanner oder Adaptern bzw. anderen Geräten im Netzwerk)
- EtherNet/IP Adapter: 192.168.2.2/../21
- Alternativ zum Routing über den Engineering Port könnte auch eine Direktverbindung vom Engineering-PC ins Feldbus-Netzwerk hergestellt werden (im Beispiel mittels zweiter Netzwerkschnittstelle am PC, 192.168.2.250)

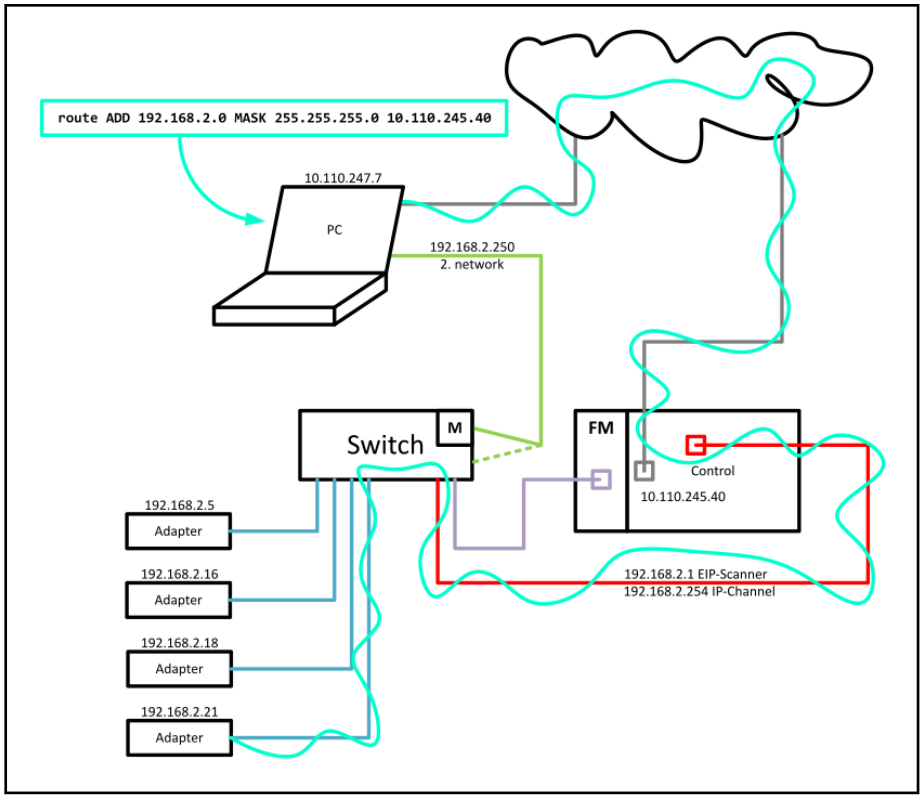


Abb. 4-23: IP-Kanal - Routing-Beispiel

Hinweise zum IP-Routing:

- Das IP-Routing ("Forwarding") in der Steuerung findet automatisch statt (Voraussetzung IP-Kanal aktiviert)
- Um vom Engineering-PC mittels IP-Kommunikation über den Engineering-Port der Steuerung auf das Feldbus-Netzwerk zugreifen zu können, muss zudem ein IP-Routing Eintrag in der Routing-Tabelle des Engineering-PC manuell eingerichtet werden.

  Beispiel Kommandozeilen-Befehl (als "Admin" ausführen):

  ```
  route ADD 192.168.2.0 MASK 255.255.255.0 10.110.245.40
  ```

  (weitere Infos z.B. mittels Kommandozeilen-Befehl "route help")
- Rückweg: des Weiteren benötigt der angesprochene Feldbus-Teilnehmer eine Gateway-Adresse mit der IP-Adresse des IP-Kanals der Steuerung. In obigen Beispiel haben die EtherNet/IP Adapter als Gateway 192.168.2.254 (= IP-Kanal der Steuerung).

  Konfiguration der Gateway-Adresse der Feldbus-Teilnehmer:

  – **EtherNet/IP Scanner**: erhält ein ENIP Adapter seine IP-Adresseinstellungen mittels BootP vom ENIP Scanner (aus der IndraWorks Konfiguration), dann wird hierbei automatisch als Gateway die IP-Kanal Adresse der Steuerung gesetzt. Alternativ muss das Gateway wie alle IP-Einstellungen des Adapters manuell im Gerät gesetzt werden.

  – **PROFINET IO Controller**: die Gateway-Adressen der PN-Devices müssen in der IndraWorks-Konfiguration entsprechend auf die IP-Kanal Adresse gesetzt werden.

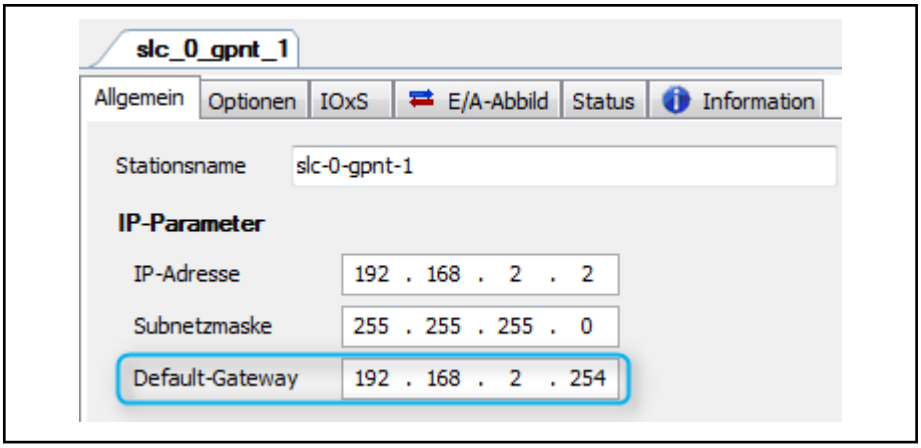


Abb. 4-24: PN-Device - Default-Gateway Konfiguration (= IP-Kanal Adresse)

UDP oder TCP Kommunikation mit der SPS

Für eine UDP oder TCP Kommunikation mit der SPS in das Feldbus Netzwerk muss entsprechend auch der IP-Kanal aktiviert werden. Es gelten die in den vorangegangen Kapiteln beschriebenen Einschränkungen, dass die IP-Kanal Adresse unterschiedlich sein muss zu den Feldbus Master und Slave IP-Einstellungen.

Beispielsweise können die Funktionsbausteine der Bibliothek RIL_SocketComm.library verwendet werden. Als IP-Adresse der Steuerung (für Kommunikation aus dem Feldbus-Netzwerk) ist entsprechend die IP-Kanal Adresse zu verwenden. Als Portnummer wird ein Wert > 1024 empfohlen.

OPC-UA Kommunikation

OPC-UA Kommunikation über die Feldbus-Schnittstelle ist ab IndraWorks 14V22 möglich. Als IP-Adresse der Steuerung (für Kommunikation aus dem Feldbus-Netzwerk) ist entsprechend die IP-Kanal Adresse zu verwenden. Es gelten die in den vorangegangen Kapiteln beschriebenen Einschränkungen, dass die IP-Kanal Adresse unterschiedlich sein muss zu den IP-Einstellungen von Feldbus Master und Slave.

4.7 SPS steuert Antriebe über eine Feldbusverbindung (Feldbusachsen)

4.7.1 Allgemeines

Antriebe (IndraDrive ab MPx2) können in Abhängigkeit von ihrer Schnittstelle über den entsprechenden Feldbus von einer MLC Steuerung gesteuert werden.

☞ An dieser Stelle wird die Kopplung der Antriebe exemplarisch über Profibus DP beschrieben.

Die Kopplung über Profinet IO ist in analoger Weise möglich.

Die Kopplung über Sercos III-Bus ist nicht Gegenstand dieser Beschreibung.

Die Ansteuerung erfolgt aus dem SPS-Programm unter Nutzung von MotionLogic-Bausteinen nach dem PLCopen-Konzept.

Bevor eine solche Feldbusachse am Profibus betrieben werden kann, sind

- ein IndraWorks-Projekt anzulegen,
- die gewünschte SPS aus der Gerätebibliothek in das Projekt zu übernehmen,
- eine Profibus-Schnittstelle (onboard oder Funktionsmodul) als Profibus DP Master zu aktivieren,

- Feldbusachsen der SPS bekannt zu machen, Seite 37, und
- kommunikationsnotwendige Antriebseinstellungen, Seite 39, vorzunehmen.

Die SPS-Programme nutzen MotionLogic-Bausteine, Funktionen und Datentypen, die in mehreren Bibliotheken zusammengefasst werden (Einbinden der notwendigen Bibliotheken, Seite 40).

Diese Bibliotheken werden gemeinsam mit dem/den SPS-Programm(en) in das IndraWorks-Projekt eingebunden.

Zusätzlich zur Steuerung des Feldbusantriebes über das SPS-Programm ist es notwendig, mit dem Antrieb Werte auszutauschen, die in seinen Parametern gespeichert sind.

Für **IndraDrive** kann dies über die PLCopen-Funktionsbausteine (MB_ReadParameter, MB_ReadRealParameter, MB_WriteParameter, MB_WriteRealParameter...) der **RMB_PLCopenFieldBus.library**, erfolgen.

4.7.2 Feldbusachsen im Projekt anlegen

Damit die Antriebe korrekt mit der SPS kommunizieren können, sind System- und Antriebseinstellungen im IndraWorks-Projekt an der SPS notwendig.

Die Feldbusachsen sind am Profibus normale Feldbusteilnehmer.

Die Beschreibungsdateien für mögliche IndraDrives sind in der IndraWorks-Gerätebibliothek enthalten:

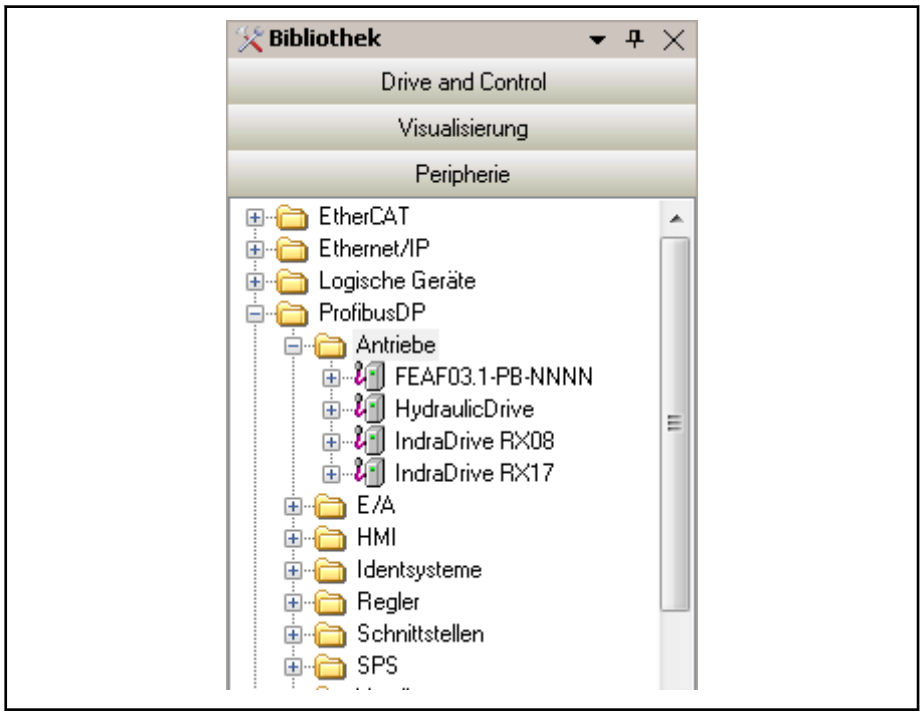


Abb. 4-25: Beschreibungsdateien der Antriebe in der IndraWorks Bibliothek

Ziehen Sie aus der Bibliothek die benötigten Antriebe auf das Profibus DP Master-Objekt.

Antriebe lassen sich im Projektexplorer so auch zwischen bereits vorhandene Antriebe einfügen.

Alternativ können Sie Antriebe im Kontextmenü über **Hinzufügen ▸ Slave ▸ Antriebe** des Profibus DP Master-Objekts hinzufügen.

Hierbei wird der neue Antrieb als letzter Antrieb unter das Profibus DP Master-Objekt eingefügt.

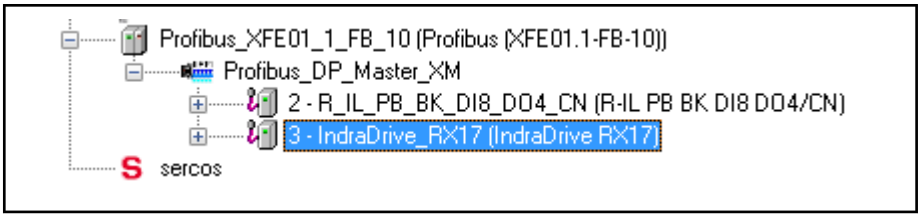


Abb. 4-26: IndraDrive als Profibus-Teilnehmer eintragen

Der Antrieb kann dabei umbenannt werden:

Kontextmenü, **Umbenennen F2**, z.B. "Drive_X" statt "IndraDrive_RX05".

Der Antrieb kann eine neue Stationsadresse erhalten,:

Doppelklick auf das Objekt, ändern der Stationsadresse z.B. auf "11" statt "3".

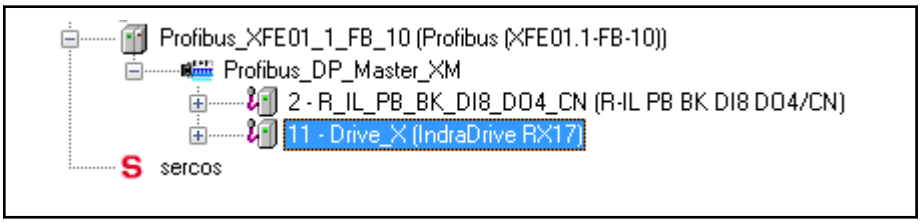


Abb. 4-27: IndraDrive als Profibus-Teilnehmer umbenannt

Der Profibus-Teilnehmer besitzt mehrere Subteilnehmer, deren Inhalt durch die Gerätebeschreibungsdatei vorgegeben ist.

☞ *Beim IndraDrive sind folgende Einstellungen erforderlich:*

- ParamCh_5_Words,
- F-Modul_not_used, **(wenn das Modul nicht benutzt wird)**
- Input_8_Words,
- Output_10_Words.

Die Antriebe (Gerätedateien) im Urzustand haben nicht die notwendigen Einstellungen, um über den Profibus in Zusammenhang mit den PLCopen-Bausteinen aus der RMB_PLCopenFieldBus.Library zu kommunizieren.

Zunächst müssen die im Hinweis enthaltenen Einstellungen durch "Gerät setzen" der Subteilnehmer gemacht werden.

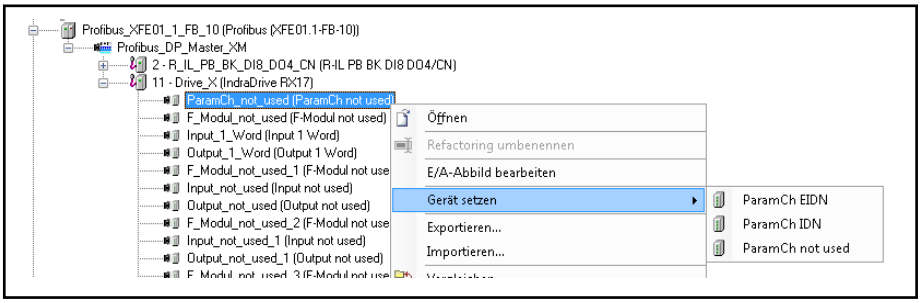


Abb. 4-28: IndraDrive, Einstellen der Subteilnehmer

Auch die Antriebe selbst sind in ihrem Urzustand nicht für eine solche Kommunikation mit einer SPS über einen Profibus konfiguriert.

Aus diesem Grund müssen Antriebseinstellungen für die SPS-Kommunikation vorgenommen werden.

4.7.3 Antriebseinstellungen für die SPS-Kommunikation

Allgemeines

Um Feldbusantriebe mit den PLCopen-Bausteinen ansteuern zu können, werden einige grundlegende Einstellungen benötigt, die die Kommunikation der PLCopen-Bausteine mit den Feldbusantrieben betreffen.

☞ Die hier beschriebenen Einstellungen beziehen sich nur auf die Kommunikation des Antriebes mit der SPS und beschreiben NICHT die allgemeine Antriebsinbetriebnahme.

Die dazu notwendigen Einstellungen können mit dem "IndraWorks Engineering"-Desktop oder mit "IndraWorks DS" vorgenommen werden.

☞ Die Parametrierung kann zur Zeit **nicht** über die Profibus-Anschaltung der SPS erfolgen.

Vom PC, auf dem der IndraWorks Engineering Desktop gestartet ist, ist die Kommunikation zur Parametrierung über RS232 seriell oder einen eigens dafür vorhanden Profibus DP Master herzustellen.

Im folgenden wird die Parametrierung über eine serielle Kommunikation beschrieben.

Herstellen der Kommunikation zur Antriebsparametrierung

Mittels der Funktion **Projekt ▸ Nach Geräten suchen** im IndraWorks Engineering-Desktop wird die Parametrierkommunikation zum Antrieb hergestellt.

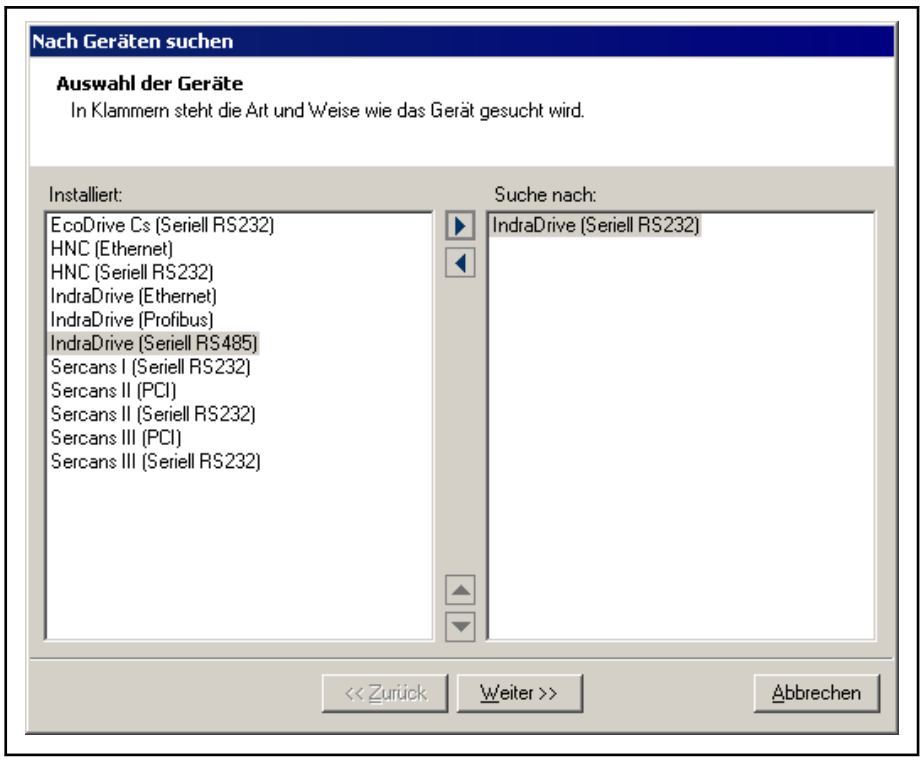


Abb. 4-29: IndraWorks Parametrierkommunikation zum Antrieb herstellen

Ist die Verbindung zum Antrieb hergestellt ist, können die Einstellungen vorgenommen werden.

Kommunikationseinstellungen IndraDrive

Allgemeines

Im folgenden werden die zur Kommunikation notwendigen Einstellungen eines IndraDrives beschrieben.

Einstellung der Busteilnehmeradresse

In dem Knoten "Führungskommunikation" muss die Busteilnehmeradresse des Antriebs eingestellt werden in Abhängigkeit der gesetzten Stationsadresse.

Länge des Parameterkanals

Mittels der direkten Eingabe des Parameters P-0-4083 mit dem Parameter-Editor, wird die Länge des Parameterkanals auf 10 Bytes eingestellt.

Eine neuere IndraDrive-Firmware (ab MPH-03VRS) stellt sich automatisch auf 5 Worte (ParamCH_5_Worte) ein.

Einstellungen Prozessdaten und Profiltype

Hier werden der Profiltyp "Frei konfigurierter Modus" und die Parameter des zyklischen Datenaustauschs mit der SPS eingestellt:

Prozessdaten-Eingänge (AT)

- P-0-4078 Feldbus-Statuswort
- S-0-0051 Lage-Istwert Geber 1
- S-0-0040 Geschwindigkeits-Istwert
- S-0-0390 Diagnose-Nummer
- S-0-0144 Signal-Statuswort

Prozessdaten-Ausgänge (MDT)

- P-0-4077 Feldbus Steuerwort
- S-0-0282 Positioniersollwert
- S-0-0259 Positioniergeschwindigkeit
- S-0-0260 Positionierbeschleunigung
- S-0-0359 Positionierverzögerung
- S-0-0145 Signal-Steuerwort

Signalsteuerwort

Bit 0: S-0-0346: Positioniersteuerwort, Bitnummer 4

Signalstatuswort

Bit 0: S-0-0343: Meldung Interpolator angehalten, Bitnummer 0

Betriebsarteneinstellungen

Als Hauptbetriebsart muss der Betriebsmodus "Antriebsgeführtes Positionieren" eingestellt werden.

Als erste Nebenbetriebsart muss "Geschwindigkeitsregelung" zugewiesen werden.

Die zweite und folgende Nebenbetriebsarten sind nicht relevant; diese werden von den PLCopen-Bausteinen nicht angewählt.

☞ Der Funktionsbaustein MB_Home führt das Referenzieren nur aus, wenn der Antrieb für das Referenzieren parametriert ist.

4.7.4 Einbinden der notwendigen Bibliotheken

Das PLCopen-Konzept wurde für IndraWorks Engineering für zwei Feldbustypen realisiert und durch nachstehende Bibliotheken unterstützt:

Sercos II und Sercos III

- ML_PLCopen.library, Funktionsbausteine der PLCopen für sercos Antriebe.
- RIL_CommonTypes.library, Datentypen systemübergreifend, Adressierung , Fehlermanagement.
- ML_Base.library, ergänzende Datentypen zum System ML*,
- ergänzende Bibliotheken z.B. RIL_Utilities.library.

Profibus DP

- RMB_PLCopenFieldBus.library, Funktionsbausteine und Datentypen der PLCopen für Profibus-Antriebe.
- RIL_CommonTypes.library, Fehlermanagement.
- ergänzende Bibliotheken z.B., RIL_Utilities.library.

Es wurde versucht, das Konzept für beide Feldbustypen weitgehend einheitlich zu halten. Dies betrifft gleiche Namen für funktionell gleiche Funktionsbausteine und Datentypen.

Mit Ausnahme der RMB_PLCopenFieldBus.library gehören die übrigen Bibliotheken zur Standard-Installation des IndraWorks Engineering. Sie werden bei der Installation mit installiert und benötigen **keine** Angabe zum Namenraum.

RMB_PLCopenFieldBus.Library

Die Bibliothek "RMB_PLCopenFieldBus.library" ist nicht im Standardumfang von IndraWorks Projekten enthalten.

Wenn Funktionen dieser Bibliothek benötigt werden, muss die Bibliothek im Projekt über den Bibliotheksverwalter nachinstalliert werden.

Zur Installation gehen Sie wie folgt vor:

1. Öffnen Sie den Bibliotheksverwalter ① im Projektexplorer und wählen Sie den Befehl **Bibliothek hinzufügen...** ②.

   Der Befehl öffnet den Dialog Bibliothek hinzufügen.

2. Öffnen Sie das Verzeichnis **IndraLogic** ▸ **Base** und selektieren Sie die Bibliothek "RMB_PLCopenFieldBus" ③.
3. Installieren Sie die Bibliothek mit "OK".

   Die Bibliothek ist jetzt im Projekt installiert.

4. Selektieren Sie die Bibliothek im Bibliotheksverwalter ④.

   Der Inhalt der Bibliothek wird im unteren Fenster des Bibliotheksverwalters angezeigt.

5. Öffnen Sie den Ordner "Datatypes" und wählen Sie "AXIS_REF" ⑤

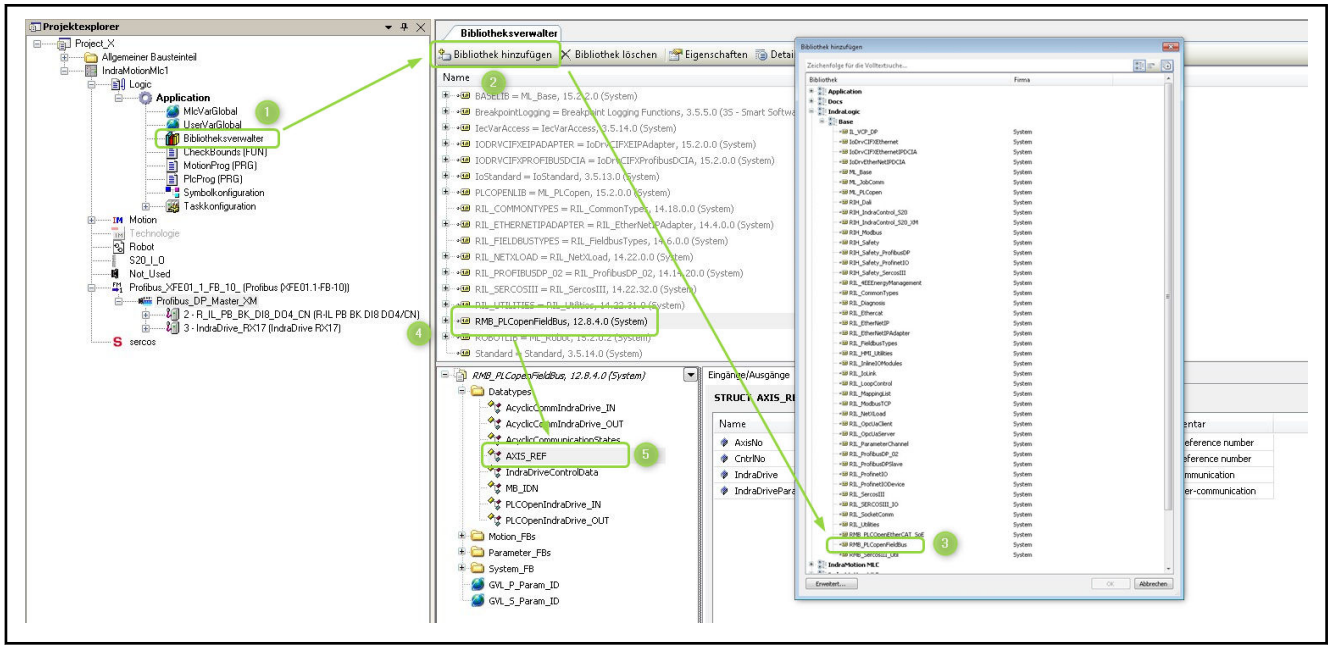


Abb. 4-30: Beispiel Bibliotheksverwalter

Die Bibliothek "RMB_PLCopenFieldBus.library" ist auf allen IndraWorks-Zielsystemen lauffähig.

Die Elemente der Bibliothek sind grundsätzlich mit einer Angabe des Namensraumes "FieldBusDrives." zu verwenden.

Sollen Feldbusachsen und Sercos-Achsen gemeinsam betrieben werden, müssen die beiden Bibliotheken (RMB_PLCopenFieldBus.library und ML_PLCopen.library) im Projekt vorhanden sein. Mit Angabe des Namensraums werden dann die Funktionsbausteine und die Achs-Referenz (AXIS_REF) unterschieden.

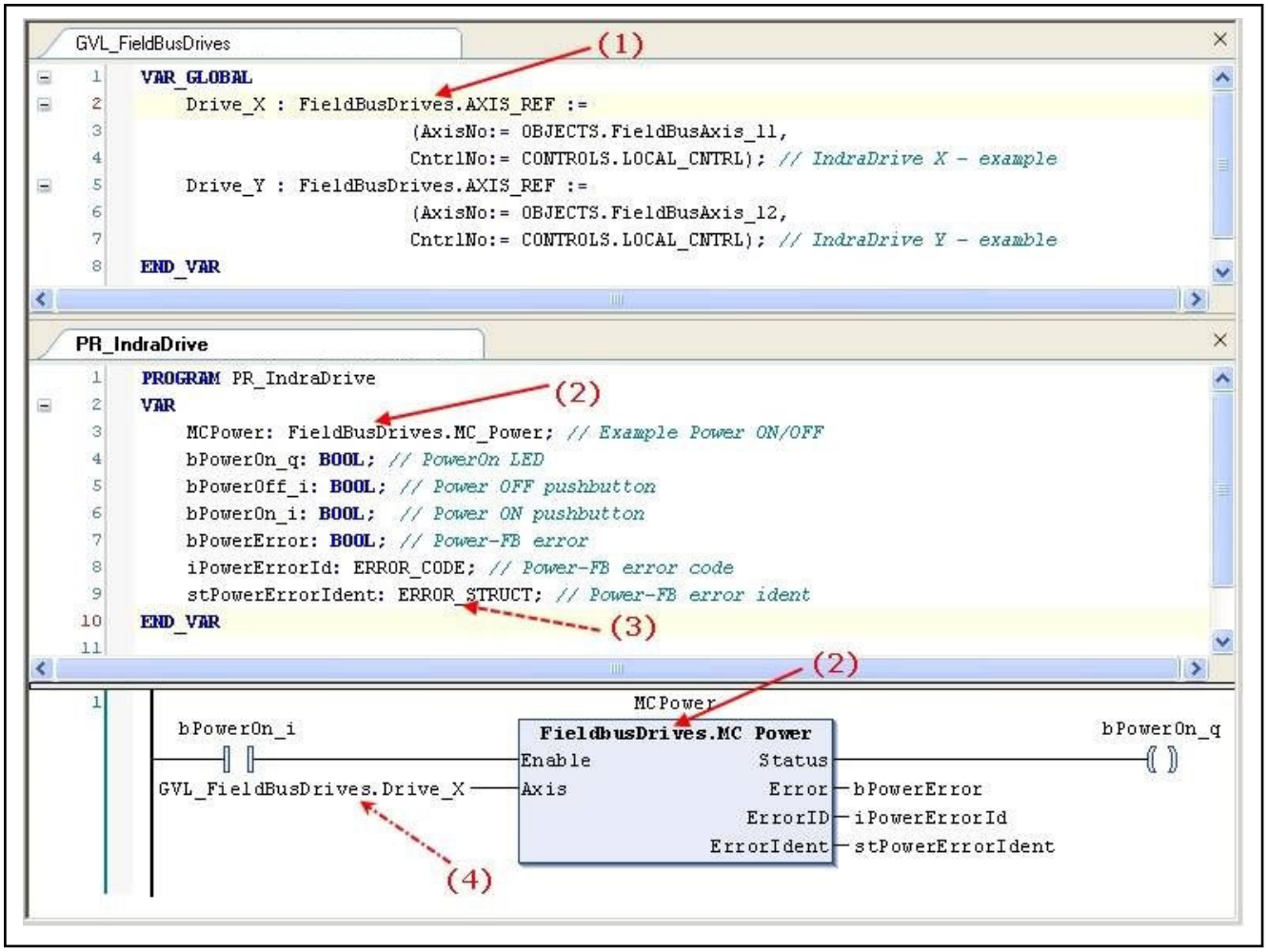
Namensraumangabe "FieldBusDrives."

Die Namensraumangabe "FieldBusDrives." muss bei einem Feldbusantrieb zwingend vorangestellt werden.

Die nachfolgende Abbildung zeigt die Verwendung des Namenraums.

Die Deklaration einer Achsreferenz (AXIS_REF) unter Angabe des Namenraums ist immer notwendig.

Die Angabe des Namenraums bei der Deklaration eines PLCopen-Funktionsbausteins ist nur notwendig, wenn die Bibliothek (ML_PLCopen) mit PLCopen-Funktionsbausteinen im Projekt vorhanden ist.



(1) AXIS_REF entstammt der RMB_PLCopenFieldBus.library, nicht der RIL_CommonTypes.library

(2) MC_Power entstammt der RMB_PLCopenFieldBus.library, nicht der ML_PLCopen.library (Deklaration und Implementation)

(3) ERROR_STRUCT entstammt der RIL_CommonTypes.library

(4) Drive_X entstammt der globalen Variablenliste "GVL_FieldBusDrives"

Abb. 4-31: Beispiele für Namenraumangaben

4.7.5 Adressierung von Feldbusantrieben im SPS-Projekt

AXIS_REF, Achsadressen der Feldbusantriebe

Die Struktur AXIS_REF (RIL_CommonTypes) enthält Adress-Informationen zur gewünschten Steuerung und dem von dieser Steuerung gesteuerten Objekt (Achsnummer).

Im Unterschied zur AXIS_REF-Struktur (RIL_CommonTypes), die die Adressierung von Sercos-basierten IndraDrives unterstützt, wird die AXIS_REF-Struktur (RMB_PLCopenFieldbus) durch zwei zusätzliche Elemente (MP_Drive und MP_DriveParameter) erweitert. Diese beiden Elemente werden nur durch das System genutzt.

Die Achsnummer kann sich von der Stationsnummer, die am Antrieb eingestellt ist, unterscheiden.

Für Sercos-basierte Achsen wird die globale Variablenliste "MLCVarGlobal" mit allen Sercos-basierten Achsen automatisch angelegt und gepflegt.

Für die Feldbusantriebe ist eine globale Variablenliste hier "GVL_FieldBusDrives" mit allen Feldbusantrieben manuell anzulegen und zu pflegen.

Die ersten beiden Elemente jedes Antriebs "CntrlNo" und "AxisNo" werden in ihr vorbelegt. Die Elemente "MP_Drive" und "MP_DriveParameter" bleiben unverändert. Sie müssen deshalb nicht mit in die Liste aufgenommen werden.

☞ Die Informationsübergabe der Adressinformationen wird wie nachstehend beschrieben vorgenommen:

Sercos-basierte Achsen

- Ab IndraWorks Version 14VRS erfolgt die Informationsübergabe per **VAR_IN_OUT CONSTANT**.

Feldbus-Achsen

- Die Informationsübergabe erfolgt stets per **VAR_IN_OUT**.

Globale Variablenliste "GVL_FieldBusDrives", Beispiel

```
VAR GLOBAL
Drive_X: FieldBusDrives.AXIS_REF :=
                (CntrlNo:=LOCAL_CNTRL,
                AxisNo:=FieldBusAxis_11); // IndraDrive X - example
Drive_Y: FieldBusDrives.AXIS_REF :=
                (CntrlNo:=LOCAL_CNTRL,
                AxisNo:=FieldBusAxis_12); // IndraDrive Y - example
Drive_Z: FieldBusDrives.AXIS_REF :=
                (CntrlNo:=LOCAL_CNTRL,
                AxisNo:=FieldBusAxis_13); // IndraDrive Z - example
Drive_n: FieldBusDrives.AXIS_REF :=
                (CntrlNo:=LOCAL_CNTRL,
                AxisNo:=FieldBusAxis_14); // IndraDrive n - example
END_VAR
```

Namensraumangabe FieldBusDrives

Die Namensraumangabe "FieldBusDrives." muss bei einem Feldbusantrieb zwingend vorangestellt werden.

Deklaration AXIS_REF

```
Drive_A: FieldBusDrives.AXIS_REF: =
```

Die Datentypen hierfür sind in der RMB_PLCopenFieldBus.library, Ordner "Datatypes", Element "AXIS_REF" enthalten.

Zuweisung CntrlNo

```
(CntrlNo: = LOCAL_CNTRL
```

Der Enumerationswert LOCAL_CNTRL ist in der RIL_CommonTypes.lib, Ordner" Datatypes of logical addresses", CONTROLS enthalten.

Die Feldbusbausteine beziehen sich immer auf die lokale SPS, also **LOCAL_CNTRL**.

Zuweisung AxisNo

```
AxisNo: = FieldBusAxis_1);
```

Der Enumerationswert FieldBusAxis_1 ist in der RIL_CommonTypes.lib, Ordner "Datatypes of logical addresses", OBJECTS enthalten.

Um Doppeldeutigkeiten zu vermeiden, ist hier für die Feldbusachsen ein eigener Nummernbereich

`FieldBusAxis_1` (6001) bis `FieldBusAxis_99` (6099)

hinterlegt.

Bei Feldbusantrieben werden die Angaben zur Steuerung und die Achsnummer zur Fehlerausgabe (ErrorIdent) verwendet.

Eine Funktionalität ist mit der Vergabe der Achsnummer nicht verbunden.



Es ist sinnvoll, PROFIBUS-Stationsnummer und Achsnummer abzugleichen...

AxisData[], Achsdaten

Das AxisData- Array, bekannt von sercos Achsen, wird bei Feldbusachsen nicht unterstützt.

4.7.6 E/A-Adressierung der Antriebe im SPS-Projekt

Allgemeines

Auf dem Profibus erscheinen die Feldbusantriebe wie normale E/A-Module, eine E/A-Adresszuweisung muss durchgeführt werden.

Um die E/A-Adresszuweisung so einfach wie möglich zu gestalten, wird sie als Variablenmapping direkt auf dem Profibus-Modul des IndraDrives durchgeführt.

E/A-Adresszuweisung IndraDrive

Die Parameterkommunikation, das F-Modul (Sicherheitstechnik) und die zyklische Kommunikation, sind bei IndraDrives jeweils eigene Submodule des Teilnehmers.

Die Kommunikation "Antrieb zur SPS" setzt sich so aus folgenden Teilen zusammen:

- Subteilnehmer (ParamCh_IDN), zur azyklischen Parameterkommunikation,
- Subteilnehmer (F-Modul), Verwendung optional, hier nicht genutzt
- Subteilnehmern des zyklischen Datenaustauschs (8 Worte Input, bzw. 10 Worte Output)

Das Mapping der Adressen wird am jeweiligen Subteilnehmer-Kanäle des Antriebes durchgeführt.

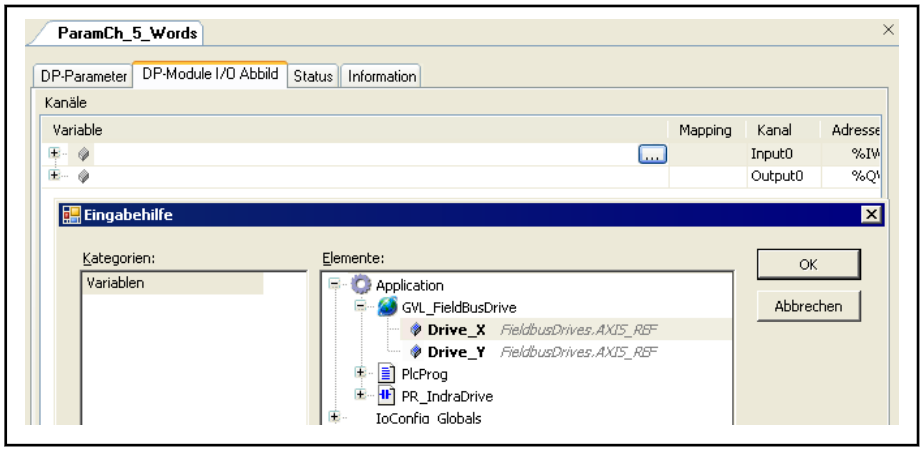


Abb. 4-32: Beispiel: Eingabehilfe für das Mapping zur Auswahl des Antriebs

Im AXIS_REF der Feldbusantriebe wird unterschieden nach"IndraDrive" für den zyklischen Parameteraustausch und "IndraDriveParameter" für den azyklischen Parameteraustausch

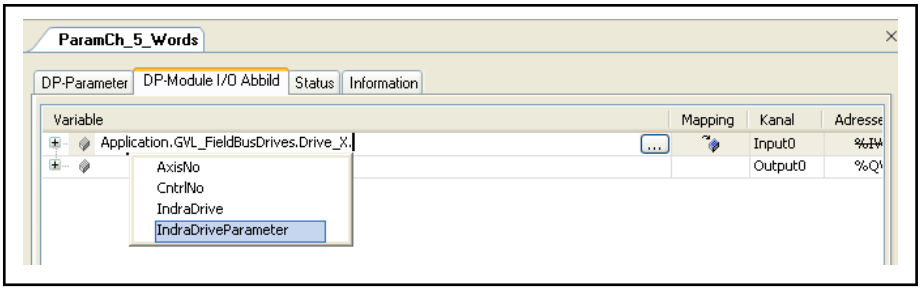


Abb. 4-33: Beispiel: Mapping für den Parameteraustausch

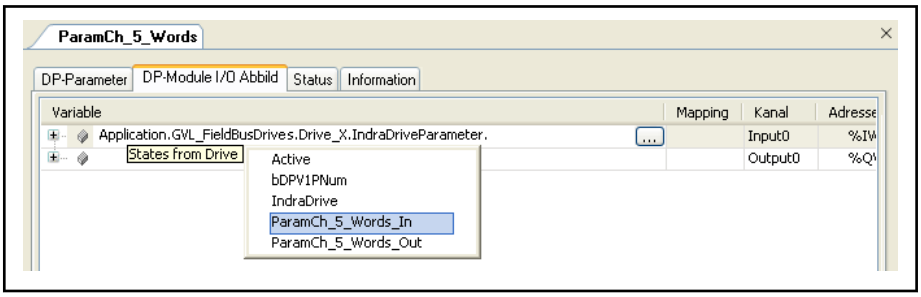


Abb. 4-34: Beispiel: Mapping für die Eingänge des Parameteraustauschs

Durch das direkte Mapping brauchen die Anfangsadressen für die Parameterkommunikation, des zyklischen Datenaustausches und eines ggf. vorhandenen F-Moduls in Bezug auf den Antrieb nicht berücksichtigt zu werden.

5 Profibus DP

5.1 Features des Profibus DP Master

Die aktuelle Implementation des Profibus DP Master beinhaltet folgende Funktionalitäten:

- Busmaster-Funktionalität nach DIN EN 50170, Teil 2
- zyklischer Datenverkehr (DP/V0)
- azyklischer Datenverkehr (DP/V1 Klasse 1) über FBs
- GSD-Dateiimport in die Gerätedatenbank
- Profibus Slave-bezogener Verbindungsstatus in der Oberfläche
- Diagnose des Profibus Master über FB
- Diagnose der Profibus Slaves über FB
- Sync/Freeze
- Deaktivierung von Slaves (bei Bus-Anlauf)

Folgende Funktionalitäten sind für zukünftige Erweiterungen vorgesehen:

- Alarme
- Deaktivierung von Slaves im SPS-Programm

Funktion/Eigenschaft	Wert
Max. Anzahl an Slaves	125
Baudrate	9,6 kBit/s bis 12MBit/s
Max. Anzahl zyklischer Eingangsdaten	3584 Byte
Max. Anzahl zyklischer Ausgangsdaten	3584 Byte
Max. Anzahl zyklischer Eingangsdaten pro Slave	244 Byte
Max. Anzahl zyklischer Ausgangsdaten pro Slave	244 Byte
Max. Länge von konsistenten Datenblöcken	244 Byte
Max. Telegrammdaten azyklisch pro Slave / Telegram	240 Byte
Anzahl gleichzeitiger DP/V1 Klasse 1 Dienste pro Slave	1

Tab. 5-1: Technische Daten

5.2 Features des Profibus DP Slave (Steuerung als Slave)

Die aktuelle Implementation des Profibus DP Slave beinhaltet folgende Funktionalitäten:

- Slave-Funktionalität nach DIN EN 50170, Teil 2
- zyklischer Datenverkehr (DP/V0)
- azyklischer Datenverkehr (DP/V1 Klasse 1)
- Deaktivierung von Slaves bei Anlauf des SPS-Programms

Folgende Funktionalitäten sind für zukünftige Erweiterungen vorgesehen:

- Diagnose des Profibus DP Slave über FB
- Alarme
- Sync/Freeze

Funktion / Eigenschaft	Wert
Max. Anzahl an I/O-Modulen	16 (8 je Richtung)
Baudrate	9,6 kBit/s bis 12MBit/s
Max. Anzahl zyklischer Eingangsdaten	244 Byte
Max. Anzahl zyklischer Ausgangsdaten	244 Byte
Max. Länge von konsistenten Datenblöcken	128 Byte
Max. Telegrammdaten pro azyklischem Telegramm	240 Byte

Tab. 5-2: Technische Daten

5.3 Profibus DP, Grundlagen

5.3.1 Begriffe und Abkürzungen

Master Der PROFIBUS Feldbus-Master wird **Master** genannt.

Slave Die PROFIBUS Feldbus-Slaves nennt man **Slaves**.

Geräteadresse Das Kriterium zur Adressierung eines Slaves ist die Geräteadresse (sog. PROFIBUS-Adresse).

Diese liegt zwischen 1 und 125.

GSD-Datei Dies ist die Gerätebeschreibung, mittels derer IndraWorks die PROFIBUS-Slaves bekannt gemacht werden.

5.4 Profibus DP, Konfiguration Übersicht

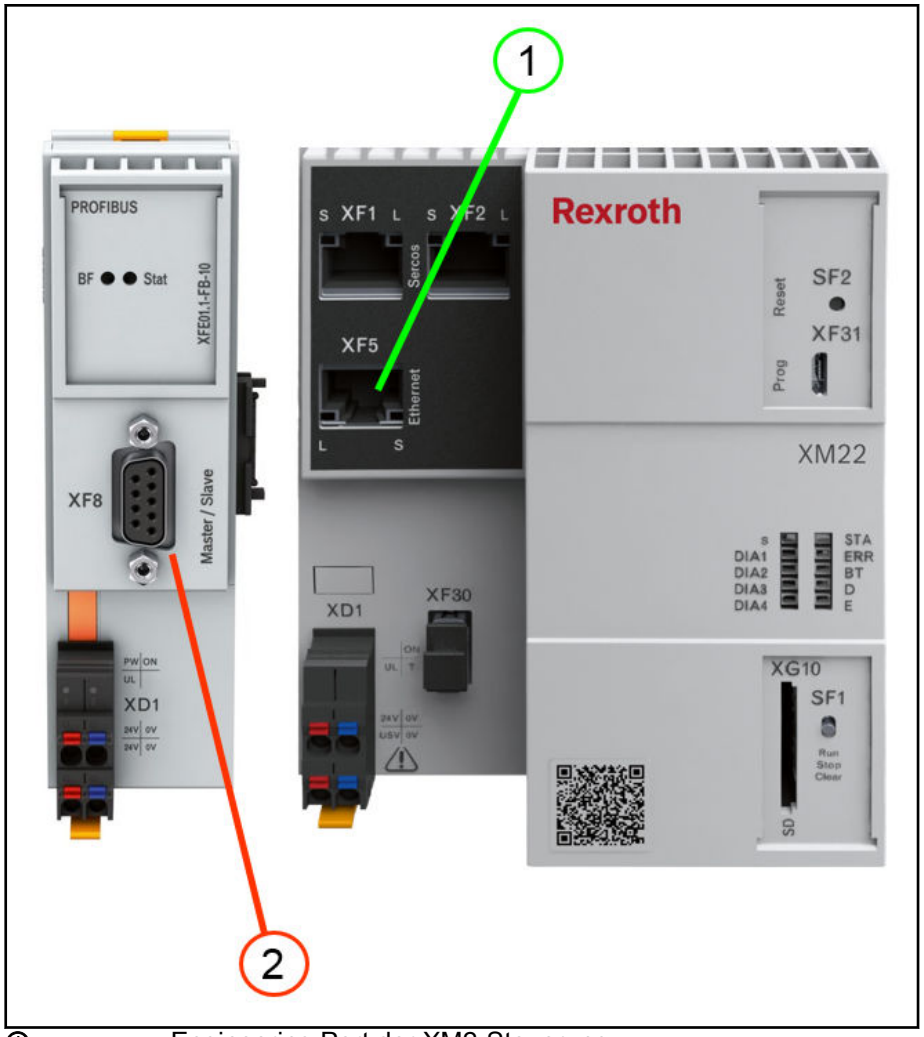
Folgende Steuerungen unterstützen Profibus DP:

- **XM2, XM4**
  Anschluss über das Erweiterungsmodul **XFE01.1-FB-10**
  Konfigurierbar als Profibus Master / Profibus Slave.
- **VPx**
  Anschluss über das Erweiterungsmodul **PFC01.1-FB-13**
  Konfigurierbar als Profibus Master / Profibus Slave.

Installieren und Einfügen von Profibus DP-Geräten

Die Profibus-Anbindung an Steuerungen der Typen XM und VPx erfolgt über Erweiterungsmodule:

- Bei XM Steuerungen werden die Erweiterungsmodule links an die Steuerung angeschlossen.
- Bei VPx Steuerungen wird das Erweiterungsmodul als Einsteckkarte in einem freien Slot im PC installiert.



① Engineering Port der XM2-Steuerung
② Funktionsmodul Real Time Ethernet XFE01.1-FB-10, die Profibus-Schnittstelle kann entweder als Master, oder als Slave konfiguriert werden.

Abb. 5-1: Profibus-Anschaltung an der XM2-Steuerung

Im IndraWorks Projekt muss das Erweiterungsmodul zur Steuerung hinzugefügt und konfiguriert werden.

Das Hinzufügen kann entweder beim erstmaligen Anlegen der Steuerung erfolgen, oder nachträglich in den Eigenschaften der Steuerung.

Nachdem das Profibus-Erweiterungsmodul hinzugefügt wurde, muss konfiguriert werden, ob das Modul als Master oder als Slave betrieben werden soll.

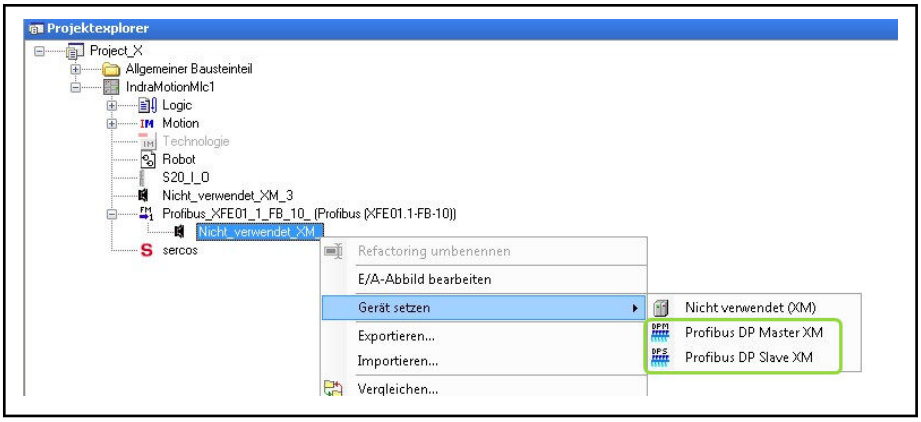


Abb. 5-2: Konfiguration des Profibus-Erweiterungsmoduls im Projektexplorer

5.5 Profibus DP, Master

5.5.1 Profibus DP Master, konfigurieren

Allgemeines

Klicken Sie zum Öffnen des Editorfensters im Projektexplorer doppelt auf das Profibus DP Master-Objekt.

In den Dialogen werden Sie über die Konfiguration des gesamten Profibus informiert und können sie gegebenenfalls modifizieren.

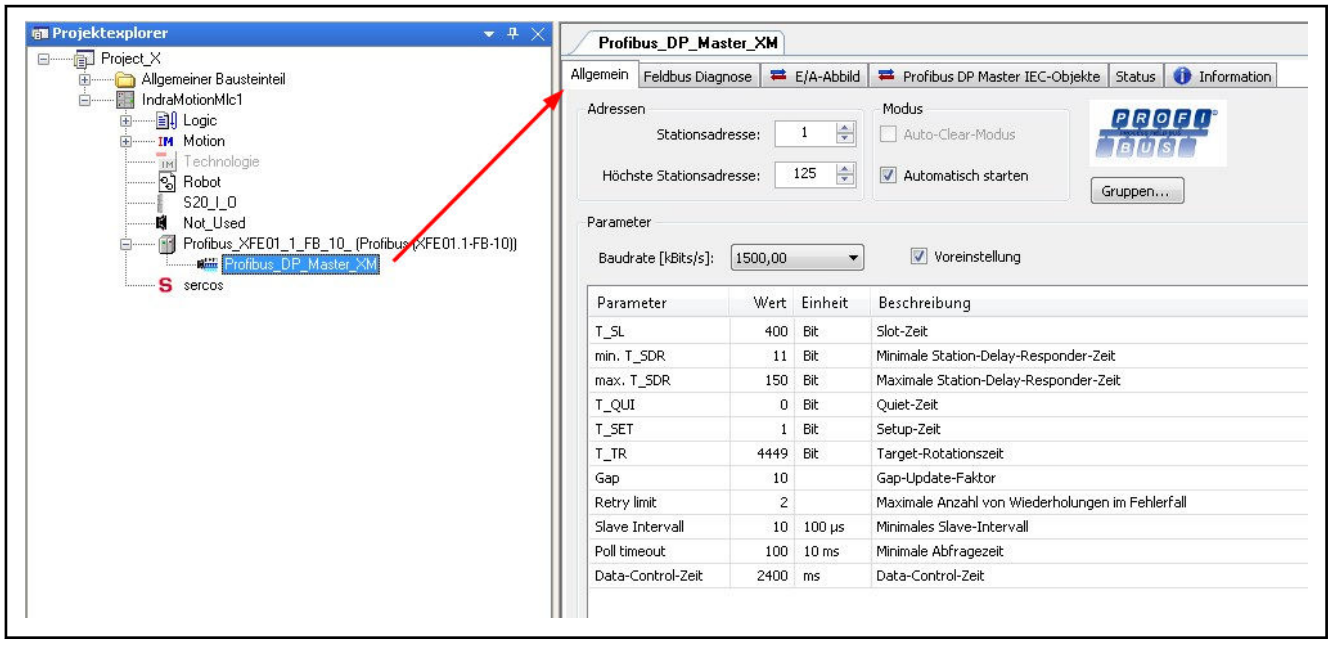


Abb. 5-3: Profibus DP Master-Objekt

Konfigurationsdialoge für Profibus DP Master

- Registerkarte 'Allgemein' auf Seite 51
- Registerkarte 'Profibus DP Master Parameter' auf Seite 53, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'Feldbus-Diagnose' auf Seite 54
- Registerkarte 'E/A-Abbild' auf Seite 57

- Registerkarte 'Status' auf Seite 58,
- Registerkarte 'Information' auf Seite 58

Registerkarte 'Allgemein'

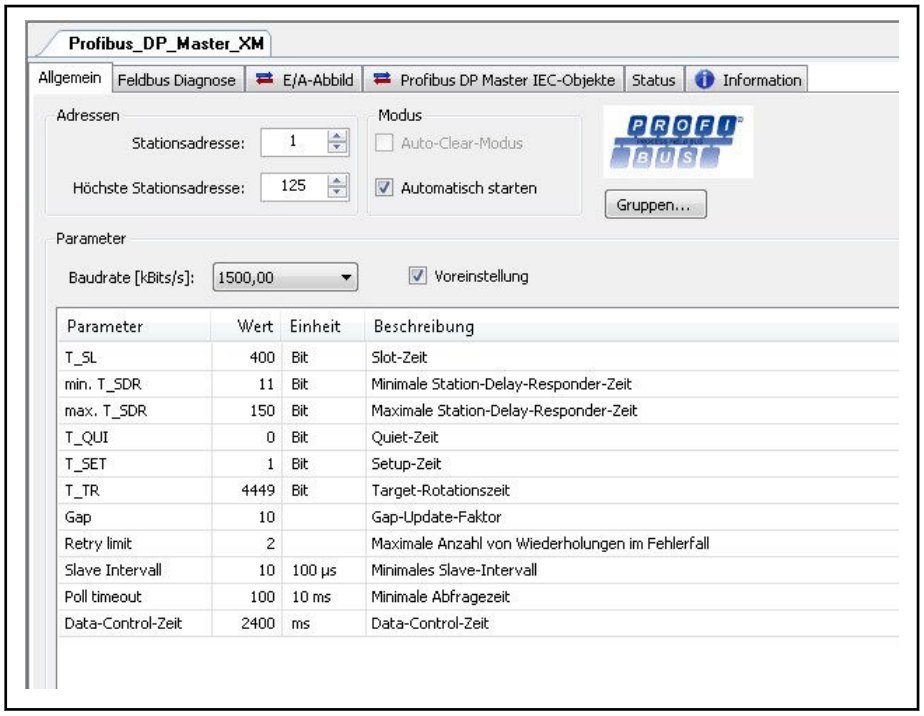


Abb. 5-4: Profibus DP Master - Registerkarte "Allgemein"

Parameter, die für jeden Profibus DP Master konfiguriert werden müssen. Sie sind grundsätzlich durch die Gerätebeschreibungsdatei definiert, können jedoch in Registerkarte "Allgemein" durch den Anwender noch modifiziert werden.

Adressen:

- **Stationsadresse**: Der erlaubte Bereich ist 0 bis125.

  Jedes neu auf dem Bus eingefügte Gerät erhält automatisch die nächst höhere Adresse

  Manuelles Eintragen ist möglich; Adressen werden auf doppeltes Vorkommen geprüft.
- **Höchste Stationsadresse**: Die höchste Stationsadresse auf dem Bus wird angezeigt. Um den Adressbereich, in dem nach neuen aktiven Geräten gesucht wird, klein zu halten, kann hier eine niedrigere Adresse angegeben werden.



Verwenden Sie möglichst kleine Adressen. Hohe Adressen verschlechtern die Busperformance!

Modus:

- **Auto-Clear-Modus:** wird aktuell nicht unterstützt.
- **Automatisch starten:** Wenn Sie diese Option aktivieren, startet der Master selbständig, ansonsten muss er manuell gestartet werden.



Bei der Option "Automatisch starten" ist es vom Treiber abhängig, ob die Einstellung ausgewertet wird.

Gruppeneigenschaften:

- Die Schaltfläche "Gruppen..." führt zum Dialog 'Gruppeneigenschaften'. Die Gruppeneigenschaften beziehen sich auf die dem Master zugeordneten Slaves.

  Bis zu acht Gruppen können eingerichtet werden. Stellen Sie für jede Gruppe ein, ob sie im Freeze-Mode und/oder Sync-Mode betrieben werden soll.

  Durch die Zuordnung der Slaves (siehe "Eigenschaften des DP-Slaves", "Gruppenzuordnung") zu verschiedenen Gruppen kann der Datenaustausch vom Master über ein Global-Control-Kommando synchronisiert werden.

  Mit einem **Freeze-Kommando** veranlasst ein Master einen Slave oder eine Gruppe, die Eingänge im momentanen Zustand ‚einzufrieren' und diese Daten beim darauffolgenden Datenaustausch zu übertragen.

  Mit einem **Sync-Kommando** werden die Slaves veranlasst, die im folgenden Datenaustausch vom Master empfangenen Daten mit dem nächsten Sync-Kommando zeitlich synchron an die Ausgänge durchzuschalten.

  Zum Ein-/Ausschalten der Freeze- und Sync-Option für eine Gruppe, klicken Sie bitte mit der linken Maustaste an entsprechender Stelle in der Tabelle, um bei der gewünschten Option ein 'v' zu platzieren/entfernen. Außerdem können Sie hier die Gruppennamen editieren.

  Die **Gruppennamen** können verändert werden. Zu diesem Zweck muss der Name in der Spalte Gruppe selektiert und die <Leertaste> gedrückt werden, um die Zeichenfolge editierbar zu bekommen. Um die Freeze und Sync-Option für eine Gruppe zu aktivieren, muss ein Mausklick auf die Checkbox in der jeweiligen Spalte ausgeführt werden. Ein "x" in dieser Box bedeutet, dass die Eigenschaft aktiviert ist.

Parameter: Diese Parameter beschreiben das Zeitverhalten der Kommunikation auf dem Profibus.

- **Baudrate [kBits/s]**:

  Die von der Gerätebeschreibungsdatei (GSD-Datei) definierte Auswahl steht hier zur Verfügung.

☞ Im Auswahlfeld Baudrate [kBits/s], darf nur eine Baudrate eingestellt werden, die von allen Slaves unterstützt wird.

- **Voreinstellung**: Ist diese Option aktiviert, werden die Werte in der Parameter-Tabelle, entsprechend der zur Zeit eingestellten Baudrate, auf ihre Standardwerte gesetzt und können **nicht** modifiziert werden.

Die Parameter-Tabelle zeigt für jeden Parameter den Wert, die Einheit und eine kurze Beschreibung.

Die Werte der Parameter können nach einem Mausdoppelklick auf das jeweilige Wert-Feld bearbeitet werden (Option Voreinstellung: deaktiviert).

Parameter [<Einheit>]	Beschreibung
T_SL [Bit[1)]]	Slot-Zeit: Maximale Zeitspanne, die der Master nach Senden einer Anforderung auf das erste Zeichen der Antwort des Slaves wartet.
min. T_SDR [Bit]	Minimale Station-Delay-Responder-Zeit: Minimale Reaktionszeit, nach der ein Teilnehmer am Bus antworten darf.
max. T_SDR [Bit]	Maximale Station-Delay-Responder-Zeit: Maximale Zeitspanne, innerhalb der ein Slave antworten muss.
T_QUI [Bit]	Quiet-Zeit: Ruhezeit, die beim Umsetzen von NRZ-Signalen (Non Return to Zero) auf andere Kodierungen zu berücksichtigen ist (Umschaltzeit für Repeater).
T_SET [Bit]	Setup-Zeit: Zeit für Setup
T_TR [Bit]	Target-Rotationszeit: Token-Soll-Umlaufzeit; projektiertes Zeitintervall, in dem ein Master den Token erhalten soll. Ergibt sich aus der Summe der Token-Halte-Zeiten aller Master am Bus.
Gap	GAP-Update-Faktor: Anzahl der Bus-Umläufe, nach denen im GAP des Masters (Adressbereich von der eigenen Bus-Adresse bis zur Adresse des nächsten aktiven Teilnehmers) nach einer weiteren, neu hinzugekommenen, aktiven Station gesucht wird.
Retry limit	Maximale Anzahl der erneuten Aufrufversuche des Masters, wenn er vom Slave keine gültige Antwort empfangen hat.
Slave Interval [µs]	Zeit zwischen zwei Bus-Zyklen, in denen ein Slave eine Anforderung des Masters bearbeiten kann (Zeitbasis 100µs). Der hier eingetragene Wert muss mit den jeweiligen Vorgaben in den GSD-Dateien der Slaves abgestimmt sein.
Poll timeout [ms]	Maximale Zeit, nach der die Antwort des Masters bei einer Master-Master-Kommunikation vom Requester (DP_Master Klasse 2) abgeholt sein muss (Zeitbasis 1ms).
Data-Control-Zeit [ms]	Zeit, in welcher der Master den zugeordneten Slaves seinen Betriebszustand mitteilt. Gleichzeitig überwacht der Master, ob mit den Slaves innerhalb dieser Zeit jeweils mindestens ein Nutzdatenaustausch stattgefunden hat und aktualisiert die Data_Transfer_List.

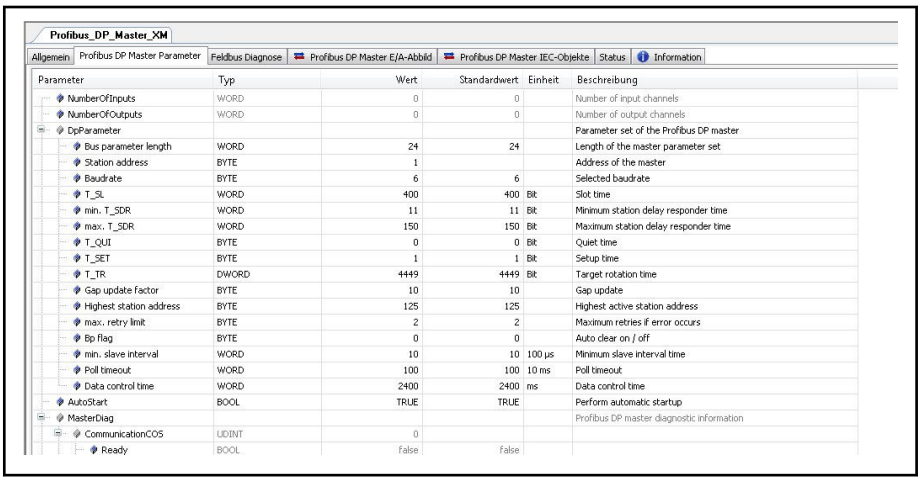
Tab. 5-3: Parameter-Tabelle

Registerkarte 'Profibus DP Master Parameter'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor**.

1) Bit: Zeiteinheit für die Übertragung eines Bits auf dem Profibus

Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.



Parameter	Typ	Wert	Standardwert	Einheit	Beschreibung
NumberOfInputs	WORD	0	0		Number of input channels
NumberOfOutputs	WORD	0	0		Number of output channels
DpParameter					Parameter set of the Profibus DP master
Bus parameter length	WORD	24	24		Length of the master parameter set
Station address	BYTE	1			Address of the master
Baudrate	BYTE	6	6		Selected baudrate
T_SL	WORD	400	400	Bit	Slot time
min. T_SDR	WORD	11	11	Bit	Minimum station delay responder time
max. T_SDR	WORD	150	150	Bit	Maximum station delay responder time
T_QUI	BYTE	0	0	Bit	Quiet time
T_SET	BYTE	1	1	Bit	Setup time
T_TR	DWORD	4449	4449	Bit	Target rotation time
Gap update factor	BYTE	10	10		Gap update
Highest station address	BYTE	125	125		Highest active station address
max. retry limit	BYTE	2	2		Maximum retries if error occurs
Bp flag	BYTE	0	0		Auto clear on / off
min. slave interval	WORD	10	10	100 µs	Minimum slave interval time
Poll timeout	WORD	100	100	10 ms	Poll timeout
Data control time	WORD	2400	2400	ms	Data control time
AutoStart	BOOL	TRUE	TRUE		Perform automatic startup
MasterDiag					Profibus DP master diagnostic information
CommunicationCOS	UDINT	0			
Ready	BOOL	false	false		

Abb. 5-5: Profibus DP Master- Registerkarte "Profibus DP Master Parameter"

Registerkarte 'Feldbus-Diagnose'

Für die Feldbusse der Systeme MLC und MTX wurde ausgehend vom Objektknoten des Masters, hier Profibus DP Master ein einheitliches Diagnosekonzept entwickelt.

Die Registerkarte "Feldbus Diagnose" zeigt Online die am Feldbus-Knoten anliegenden Module.

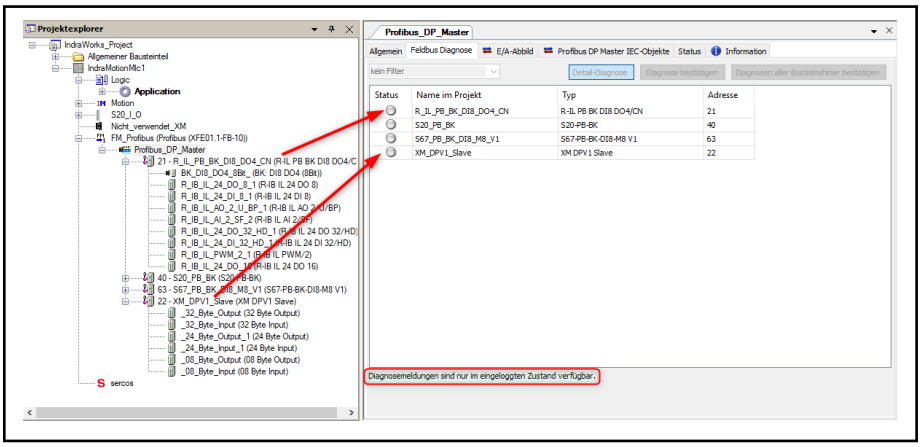


Abb. 5-6: Profibus DP Master - Registerkarte "Feldbus Diagnose" (offline)

Nach dem Einloggen durchläuft der am Knoten befindliche Bus einen Diagnosezyklus in dessen Ergebnis sich die Status-Leuchtdioden verfärben, hier grün, d.h. beide Module melden fehlerfreien Lauf.

Nach Auswahl eines Teilnehmers **(1)** ist für dieses eine Detaildiagnose verfügbar.

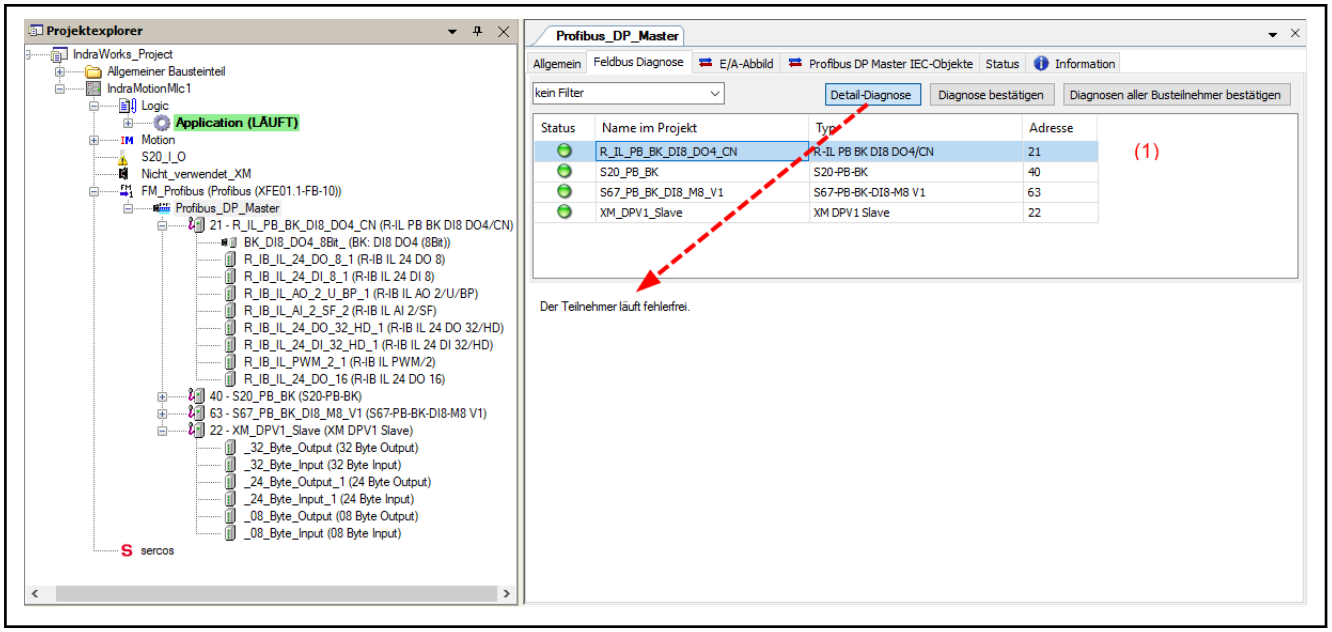


Abb. 5-7: Feldbus-Diagnose, Steuerung Online, eingeloggt, fehlerfrei

Diagnose im Fehlerfall

Das Einloggen wird mit einem zusätzlichen, nicht an der realen Steuerung vorhandenen Klemme, wiederholt.

Am betroffenen Modul erscheint ein gelbes Warndreieck, der Hinweis auf eine vorhandene Diagnose. Die Status-Leuchtdiode des betroffenen Moduls färbt sich rot. Über "Detail-Diagnose" können eine "Standarddiagnose" mit Kurzinformation und eine "Erweiterte Diagnose" mit der detaillierten Angabe der Fehlerursache erhalten werden.

Projektnavigator

- Profibus DP Master: Gelbes Warndreieck → eine Diagnose steht an
- Buskoppler (Adresse 21): Gelbes Warndreieck → eine Diagnose steht an
- Modul (Adresse 10): Gelbes Warndreieck → die Detail-Diagnose zeigt, dass das Modul im Projekt konfiguriert ist, aber real nicht vorhanden ist.

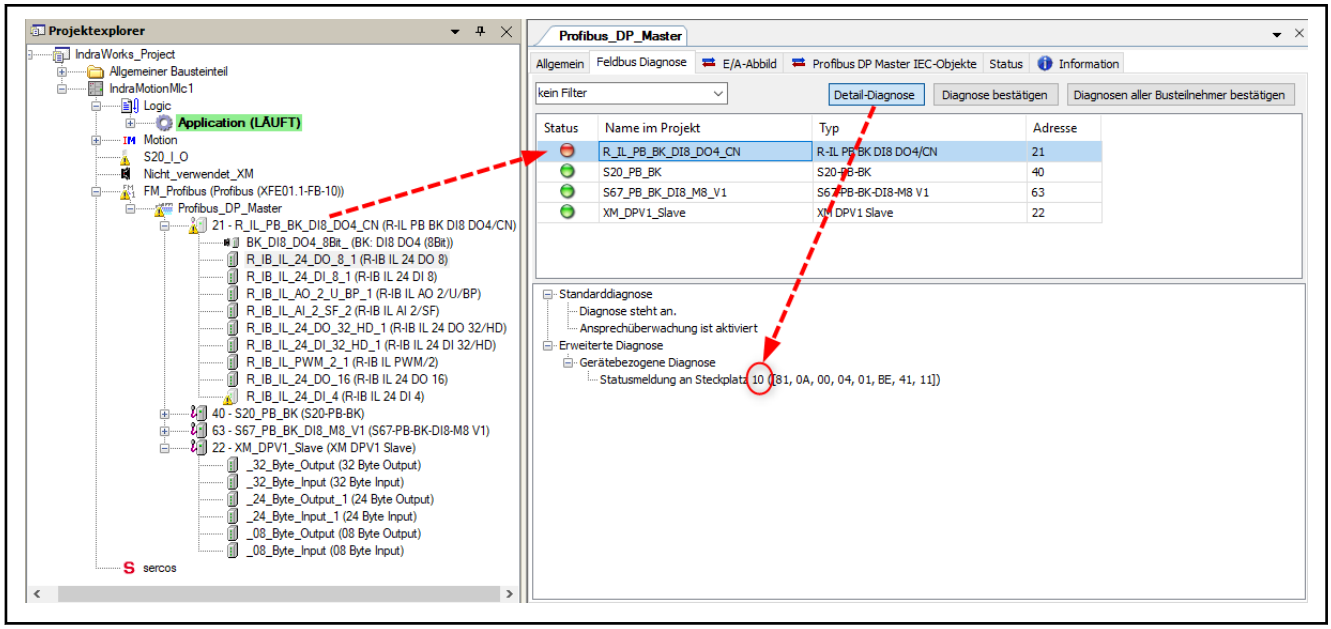


Abb. 5-8: Feldbus-Diagnose, Steuerung Online, eingeloggt, Fehler

Siehe auch Registerkarte 'Status', Seite 58.

Diagnose betätigen: Für den gewählten Busteilnehmer werden, trotz vorhandenen Fehlers, sowohl das Warndreieck als auch die rote Status-Leuchtdiode abgeschaltet. Die "Erweiterte Diagnose" der "Detaildiagnose" bleibt im Text erhalten. (Die nächste Diagnose kann bearbeitet werden....)

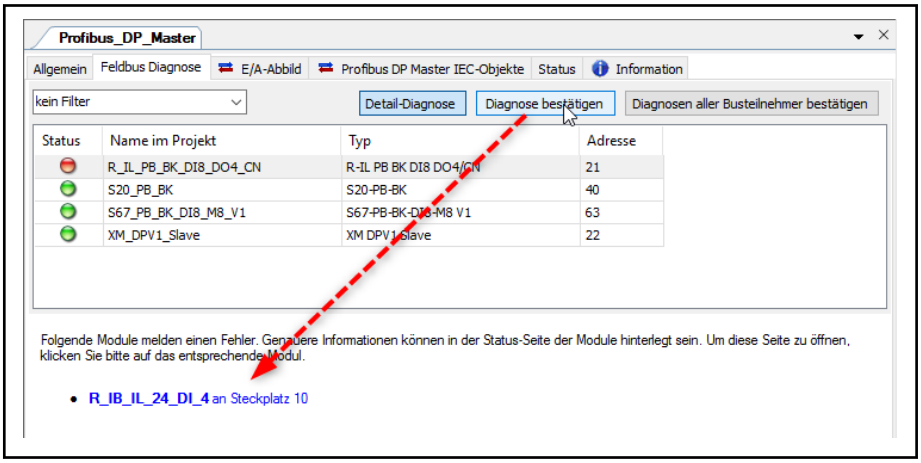


Abb. 5-9: Detailinformationen nach "Diagnose bestätigen"

Diagnose aller Busteilnehmer betätigen: Für alle Busteilnehmer werden, trotz vorhandener Fehler, sowohl das Warndreieck als auch die rote Status-Leuchtdiode abgeschaltet.

Registerkarte 'E/A-Abbild'

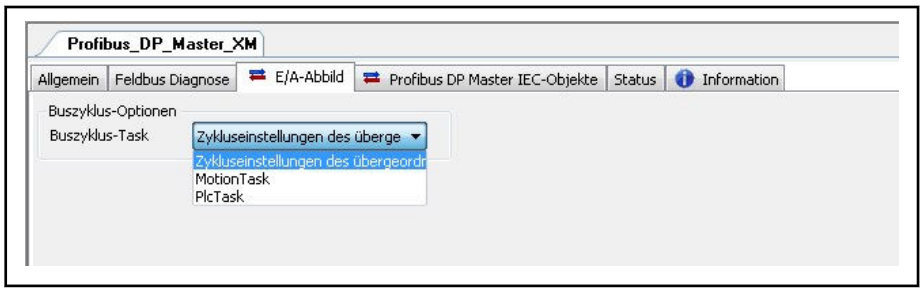


Abb. 5-10: Profibus DP Master - Registerkarte 'E/A-Abbild'

IEC-Objekte: Der zum konkreten Bus-Objekt gehörende Datenbestand ist als (projektglobale) Variable `Profibus_DP_Master_4` (Profibus DP Master-Objekt mit der Nummer 4) vom Typ `IoDrvCIFXProfibusWrapper` adressierbar.

Buszyklus-Task: Durch die Auswahl einer Buszyklus-Task kann der Zyklus des Abbildtausches für den Profibus DP Master an eine spezielle Task gebunden werden. Es ist sinnvoll in dieser Task auch die IO-Daten des Masters zu verarbeiten.

Standardeinstellung: "Zykluseinstellung des übergeordneten Busses verwenden"

Diese Einstellung bewirkt, dass die in "Buszyklus-Optionen" der Steuerung (Doppelklick auf die konkrete Steuerung im Projektnavigator, SPS-Einstellungen) vorgenommene Taskeinstellung für den konkreten Bus übernommen wird.

Alternativ kann der Buszyklus an eine PLC- oder Motion-Task gekoppelt werden.

Registerkarte 'Profibus DP Master IEC-Objekte'

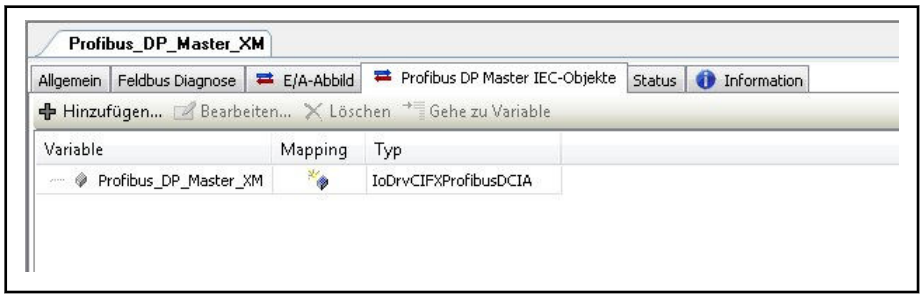


Abb. 5-11: Profibus DP Master - Registerkarte 'Profibus DP Master IEC-Objekte' (offline)

IEC-Objekte: Der zum konkreten Bus-Objekt gehörende Datenbestand ist als (projektglobale) Variable `Profibus_DP_Master_4` (Profibus DP Master-Objekt mit der Nummer 4) vom Typ `IoDrvCIFXProfibusWrapper` adressierbar.

Registerkarte 'Status'

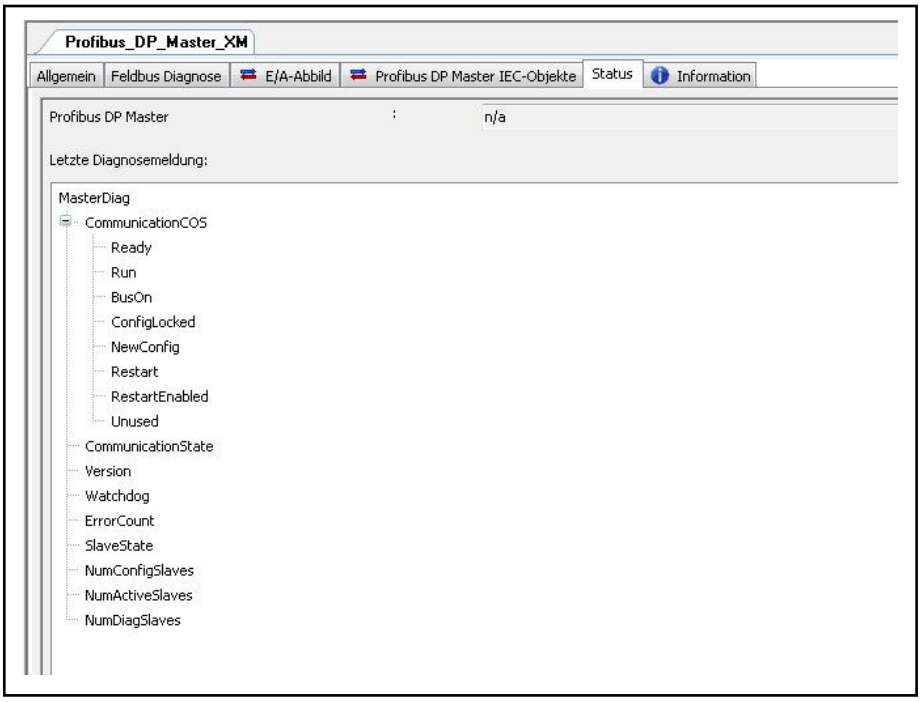


Abb. 5-12: Profibus DP Master - Registerkarte 'Status' (offline)

Die Registerkarte "Status" zeigt Status-Informationen (z.B. "Läuft" (Bus aktiv), und "n/a" (keine Information vorhanden)) und

spezifische Diagnosemeldungen aus dem jeweiligen Gerät, auch zur verwendeten Karte und dem internen Bussystem.

Angezeigt wird ferner die "Letzte Diagnosemeldung", die über "Bestätigen" quittiert werden kann.

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

5.5.2 Profibus DP Master, Slaves konfigurieren

Slaves hinzufügen

Die Slaves befinden sich in der Bibliothek "Peripherie" im Ordner "Profibus DP".

Ziehen Sie aus der Bibliothek die benötigten Slaves auf das Profibus DP Master-Objekt.

Slaves lassen sich im Projektexplorer so auch zwischen bereits vorhandene Slaves einfügen.

Alternativ können Sie Slaves im Kontextmenü über **Hinzufügen ▸ Slave ▸ ...** des Profibus DP Master-Objekts hinzufügen.

Hierbei wird der neue Slave als letzter Slave unter das Profibus DP Master-Objekt eingefügt.

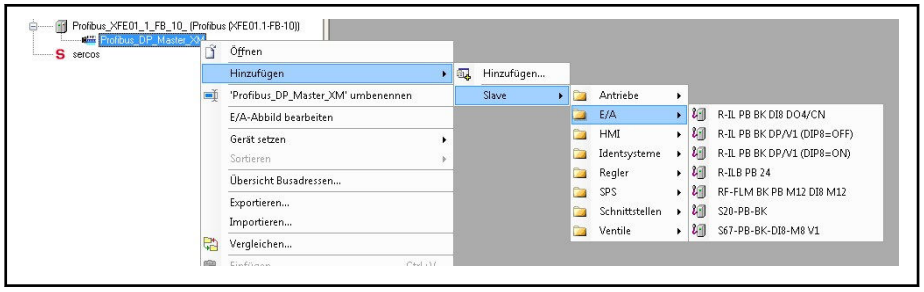


Abb. 5-13: Profibus DP Master, Slave hinzufügen

Ist ein benötigter Slave nicht standardmäßig in der Bibliothek vorhanden, kann er mittels Import seiner GSD-Datei über das Hauptmenü **Extras** ▸ **Gerätedatenbank...** in die Bibliothek integriert werden.

Siehe auch Kap. 4.1.2 "Geräte verwalten" auf Seite 15.

Slaves zum Anschluss von E/A-Modulen

Profibus DP unterscheidet zwei Arten von Slaves zum Anschluss von E/A-Modulen:

1. **Kompakt**: Bei einem kompakten Slave ist der Modulaufbau fest vorgegeben.

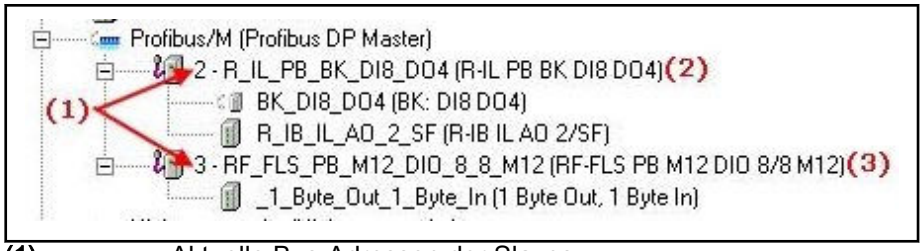
   Nach dem Einfügen eines Slaves in den Projektexplorer sind bei der kompakten Bauweise die Module unterhalb des Slave-Objektknotens bereits vollständig vorhanden. In der Bibliothek sind keine weiteren Module vorhanden.

2. **Modular**: Der Modulaufbau des Slaves ist variabel.

   Die Module (Klemmen) können individuell – aber nach Bestückungsvorschrift des Gerätes – angeordnet werden.

   Direkt nach dem Einfügen des Slaves in den Projektexplorer sind noch keine untergeordneten (E/A)-Geräteebenen des Slaves vorhanden.

   Beim modularen Slave müssen die Module manuell zugeordnet werden. Zum Einfügen von Modulen siehe Kap. 5.5.3 "Profibus DP Master, Module zum Slave hinzufügen" auf Seite 67.

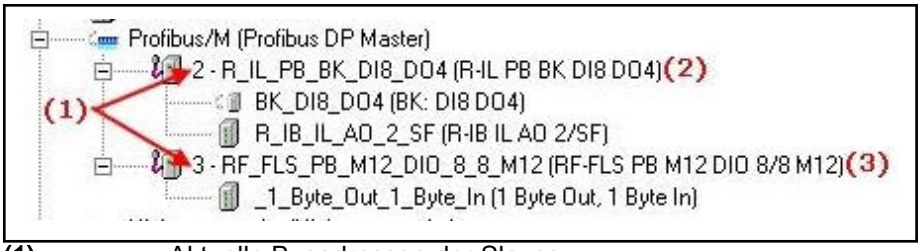


(1) Aktuelle Bus-Adressen der Slaves
(2) Modularer Slave, mit zugeordneten Modulen
(3) Kompakter Slave

Abb. 5-14: Slaves mit E/A-Modulen, am Profibus DP Master-Objekt

Übersicht Busadressen...

Um die komplette Adressbelegung aller Busteilnehmer anzuzeigen, öffnen Sie den Kontext-Menüpunkt **Übersicht Busadressen...** des Profibus DP Master-Objekts.



(1) Aktuelle Busadressen der Slaves
(2) Modularer Slave, mit zugeordneten Modulen
(3) Kompakter Slave

Abb. 5-15: Slaves mit E/A-Modulen, am Profibus DP Master-Objekt

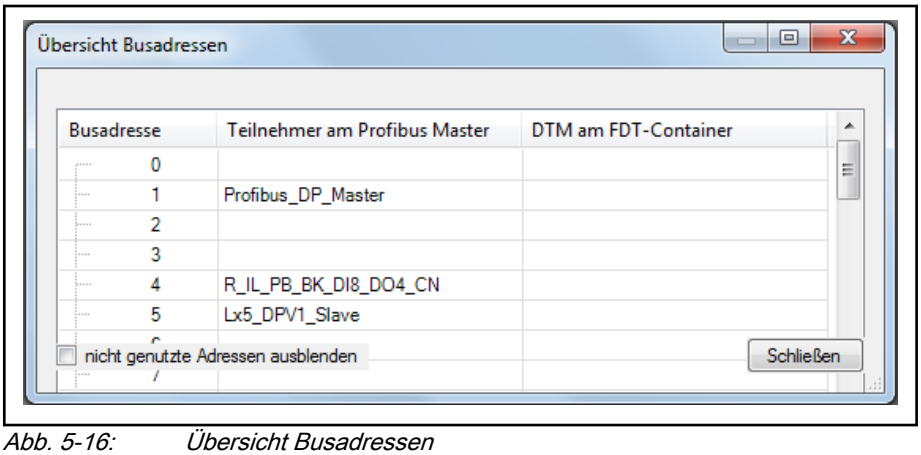


Abb. 5-16: Übersicht Busadressen

Sortieren von Feldbusteilnehmern

Sind zwei oder mehr Feldbusteilnehmer in einem Projekt unter einem Feldbus vorhanden, können diese in einer gewünschten Reihenfolge angezeigt werden.

Über das Kontextmenü des Feldbusses können die Teilnehmer nach Name, Typ und Adresse, jeweils auf- oder absteigend sortiert werden.

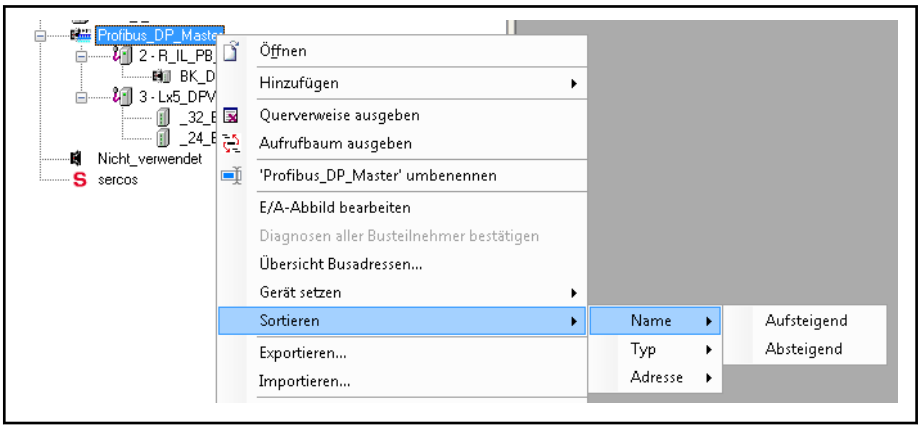


Abb. 5-17: Kontextmenü Sortieren von Feldbusteilnehmern

Profibus DP-Slaves aktivieren / deaktivieren

Die an einer Steuerung konfigurierten Profibus DP-Slaves können im Projektexplorer aktiviert bzw. deaktiviert werden.

Dies erfolgt durch Anwählen / Abwählen der Schaltfläche, die am Icon des Profibus DP-Slaves erscheint.

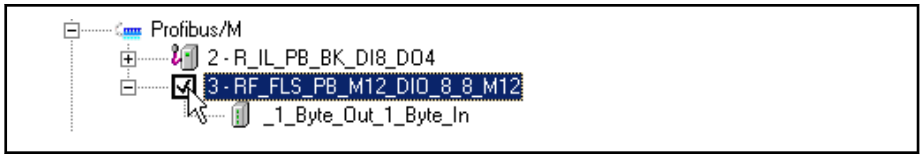


Abb. 5-18: Aktivieren / Deaktivieren von Profibus DP-Slaves

Ist der Haken gesetzt, ist der Busteilnehmer aktiviert. Er wird in der Diagnose des Busses berücksichtigt.

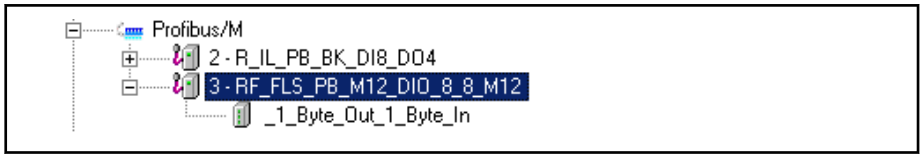


Abb. 5-19: Profibus DP-Slave als aktivierter Teilnehmern

Ist der Haken nicht gesetzt, ist der Busteilnehmer deaktiviert. Er wird in der Diagnose des Busses nicht berücksichtigt.

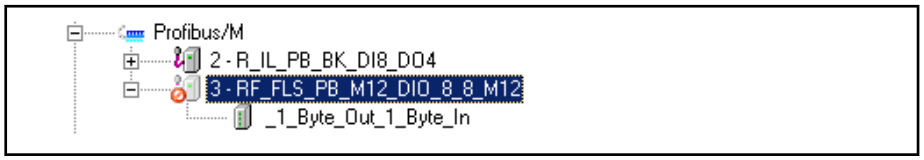


Abb. 5-20: Profibus DP-Slave als deaktivierter Teilnehmer

Wird ein neuer Profibus DP-Slave angelegt, ist er per Voreinstellung aktiviert.

Beim Laden der Konfiguration werden alle aktivierten Profibus DP-Slaves berücksichtigt. Deaktivierte Profibus DP-Slaves werden ignoriert.

☞ Ein aktivierter jedoch nicht vorhandener Busteilnehmer erzeugt in der Diagnose eine Fehlermeldung.

Ein deaktivierter nicht vorhandener Busteilnehmer erzeugt in der Diagnose keine Fehlermeldung.

Siehe auch: Profibus DP-Slaves deaktivieren mittels UserDefs.cfg auf Seite 61

Profibus DP-Slaves deaktivieren mittels UserDefs.cfg

Die Funktion ermöglicht es Profibus DP-Slaves, die in IndraWorks konfiguriert sind, durch Definition in einer auf der FlashCard der Steuerung abliegenden Datei zu deaktivieren.

Dadurch werden Diagnose-Meldungen für Slaves, die zwar konfiguriert, aber nicht vorhanden sind, verhindert.

Aktivierung der Funktion

Die Funktion wird durch das Vorhandensein einer gültigen Datei "UserDefs.cfg" auf der Steuerungs-FlashCard in folgenden Partitionen aktiviert:

Bei XM- und VPx-Steuerungen: **\USER**

Optional können durch Einträge in der "cfg-Datei" eines Systems ein anderer Pfad und Dateiname definiert werden.

Beispiel:

```
[IOSpecialDefs]
IODeactFilePath=MyFilePath
IODeactFileName=MyFileName
```



Die Datei "UserDefs.cfg" kann Einträge für weitere Busse enthalten:

Inline-Module, Sercos, Profinet IO, Profibus DP.

Zeitpunkt des Deaktivierens

Das Auswerten der Datei "UserDefs.cfg" und das Deaktivieren der dort eingetragenen Slaves erfolgt zum Zeitpunkt und im Kontext des internen Events "PrepareUpdateConfiguration".

Das Event tritt bei folgenden Aktionen ein:

- Anlauf nach Power-ON
- Reset (warm)
- Reset (kalt)
- Programm laden nach Reset (Ursprung)

Struktur der Datei "UserDefs.cfg"

Die Datei ist strukturiert in ein- oder mehrfach vorhandene Sektion(en) und den darin enthaltenen Einträgen.

- **Sektionen**: Die Bezeichnung einer Sektion ist in ´**[**´ und ´**]**´ geklammert. Eine Sektion ist begrenzt durch den Beginn der nächsten Sektion oder durch das Ende der Datei.

  Gültige Bezeichner für eine Sektion sind derzeit:

ProfibusDP.Master.0	Erster projektierter Profibus DP-Master (in der Regel Onboard-Master)
ProfibusDP.Master.1	Zweiter projektierter Profibus DP-Master (wenn Onboard-Master vorhanden, dann Profibus DP-Master auf Funktionsmodul 1)
ProfibusDP.Master.2	...
ProfibusDP.Master.3	...
ProfibusDP.Master.4	Fünfter projektierter Profibus DP-Master
ProfibusDP.Master.X	Sektion wird nicht ausgewertet

  Die Ziffern 0..4 definieren den zu verwendenden Profibus DP-Master.

  0 entspricht dabei dem ersten projektierten Profibus DP-Master,

  1 entspricht dem zweiten projektierten Profibus DP-Master, usw..

Beispiel:

- Der Onboard-Profibus DP-Master ist aktiviert und ein Profibus DP-Master Funktionsmodul ist aufgesteckt ⇒
  - "0" entspricht dem Onboard-Master,
  - "1" entspricht dem Master auf dem Funktionsmodul
- Der Onboard-Profibus DP-Master ist deaktiviert und es ist ein Profibus DP-Master-Funktionsmodul aufgesteckt ⇒
  - "0" entspricht dem Master auf dem Funktionsmodul
- Der Onboard-Profibus DP-Master ist aktiviert und es sind zwei Profibus DP-Master-Funktionsmodule aufgesteckt ⇒
  - "0" entspricht dem Onboard-Master,
  - "1" entspricht dem Master auf dem Funktionsmodul, das am nächsten zur Steuerung aufgesteckt ist,

– "2" entspricht dem Master auf dem Profibus DP-Master-Funktionsmodul, das bei Draufsicht auf die Steuerung am weitesten links von der Steuerung aufgesteckt ist

- **Sonderfunktion:** Die Definition des Zeichens ´**X**´ für den Index anstelle einer Ziffer führt dazu, dass diese Sektion in der Auswertung ignoriert wird.

  Es ist möglich, für den Master-Index **0** mehrere Sektionen zu definieren. Die darin enthaltenen Einträge werden alle dem Profibus DP-Master zugeordnet.

  In Kombination mit der oben beschriebenen "X-Funktion" lassen sich damit unterschiedliche Gruppen von Slaves in einfacher Weise behandeln.
- **Slave-Eintrag**: Ein Eintrag zum Deaktivieren eines Profibus DP-Slaves besteht aus dem Schlüsselwort mit Zuweisungszeichen `SlaveDeactAddr=` und der 1- bis 3-stelligen Angabe der Slave-Adresse in dezimaler Form.

  Beispiel:

  ```
  SlaveDeactAddr= 28
  ```

  Der zulässige Wertebereich ist **1…125**.

  Der Wert **0** ist ebenfalls zulässig, aber unwirksam.
- **Kommentare**: Als Kommentarzeichen ist das Semikolon (´**;**´) definiert. Ab der Position des Semikolons bis zum Ende der Zeile wird der Text nicht ausgewertet.

Größe der Datei "UserDefs.cfg"

Entsprechend der gegebenen Größe der Datei wird zu deren Auswertung Speicher allokiert. Die zulässige Größe ist daher praktisch nicht begrenzt.

Beispiel der Datei "UserDefs.cfg"

```
; **************
; Onboard Master
; **************

[ProfibusDP.Master.0]
SlaveDeactAddr = 5
SlaveDeactAddr = 8
SlaveDeactAddr = 25


; ************************
; Master on function module
; ************************

[ProfibusDP.Master.1]
SlaveDeactAddr = 64
SlaveDeactAddr = 30
SlaveDeactAddr = 112
SlaveDeactAddr = 3


; ****************************
; This whole section is ignored
; ****************************

[ProfibusDP.Master.X]
SlaveDeactAddr = 36
SlaveDeactAddr = 7
SlaveDeactAddr = 15
```

Slaves konfigurieren

Klicken Sie zum Öffnen des Editorfensters im Projektexplorer doppelt auf den gewünschten Slave.

In den Dialogen werden Sie über die Konfiguration des gesamten Slaves informiert und können sie gegebenenfalls modifizieren.

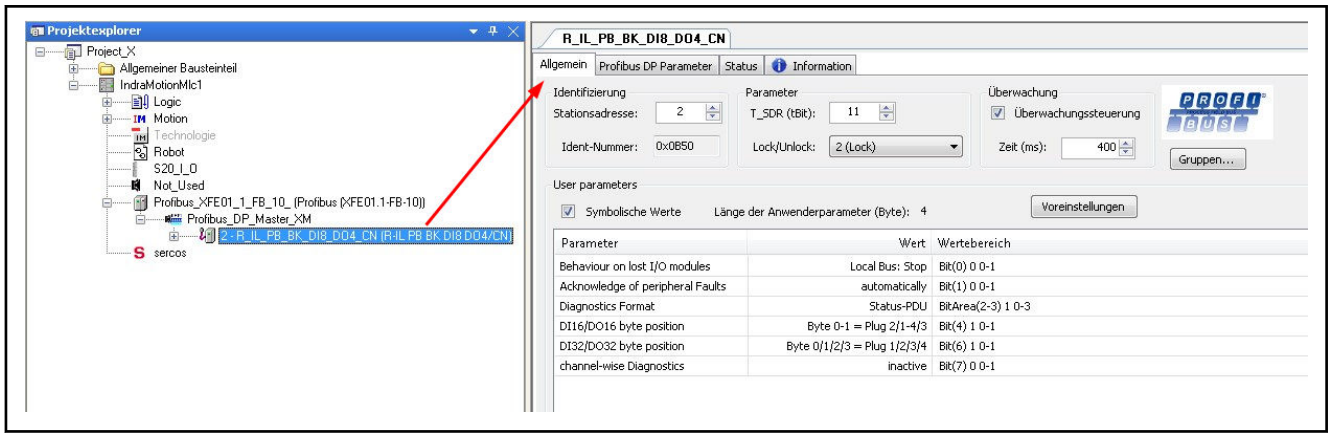


Abb. 5-21: Profibus DP Master, Slaves konfigurieren

Konfigurationsdialoge für Profibus DP Master, Slaves konfigurieren

- Registerkarte 'Allgemein' auf Seite 64
- Registerkarte 'Profibus DP Parameter' auf Seite 66, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'Status' auf Seite 66
- Registerkarte 'Information' auf Seite 66

Registerkarte 'Allgemein'

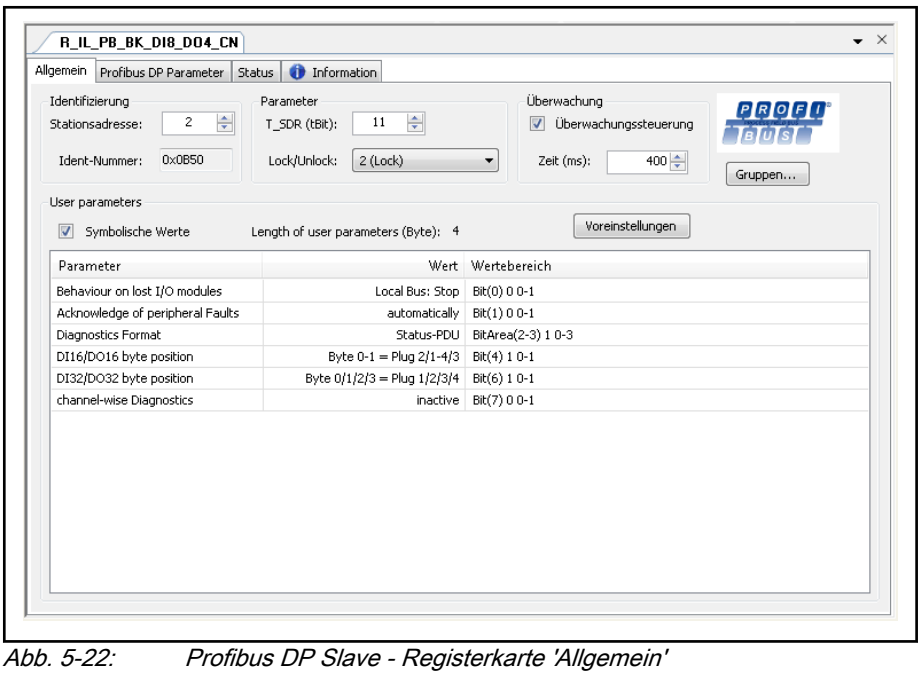


Abb. 5-22: Profibus DP Slave - Registerkarte 'Allgemein'

Die DP-Parameter eines DP-Slaves sind im Unterschied zu denen des Masters nicht ein Standard-Set, sondern werden in der Gerätebeschreibung individuell für jedes Gerät definiert. Sie können im DP-Parameter-Dialog durch den Anwender modifiziert werden.

Identifizierung

- **Stationsadresse**: Die Adresse wird automatisch eingetragen. Sie ist entsprechend der tatsächlichen Buskonfiguration anzupassen. Die Adresse "126" ist für Service und Inbetriebnahme reserviert und kann nicht verwenden werden.
- **Ident-Nummer**: Eindeutige Nummer, die vom PNO (Profibus Nutzer Organisation) diesem Gerätetyp zugewiesen wurde. Dies erlaubt eine eindeutige Referenz zwischen DP-Slave und zugehöriger GSD-Datei.

Parameter

- **T_SDR (tBit)**: Time Station Delay Responder. Reaktionszeit, nach welcher der Slave frühestens dem Master antworten darf.

  "tBit": Zeiteinheit für die Übertragung eines Bits über Profibus;

  Kehrwert der Übertragungsrate; z.B. 1 Bit bei 12 MBaud=1/12.000.000 Bit/sek=83 ns.
- **Lock/Unlock**: Slave wird für andere Master gesperrt oder freigegeben. Wählen Sie aus der Auswahlliste eine der folgenden Optionen:
  - **0 (T_SDR unlock)**: min. T_SDR und slave-spezifische Parameter dürfen überschrieben werden.
  - **1 (Will be unlocked)**: Slave für andere Master freigegeben.
  - **2 (Lock)**: Slave für andere Master gesperrt, alle Parameter werden übernommen.
  - **3 (Unlock)**: Slave für andere Master erneut freigegeben.

Überwachung

- **Überwachungssteuerung** Wenn Sie diese Option aktivieren, gilt die eingetragene Überwachungszeit. Wird der Slave innerhalb dieser Zeit nicht vom Master angesprochen, geht er in den Initialisierungszustand zurück.
- **Zeit (ms)**: Überwachungszeit, relevant für den Fall, dass die Option "Überwachungssteuerung" aktiv ist.

Gruppen

- Die Schaltfläche "Gruppen..." führt zum Dialog 'Gruppeneigenschaften'.

  Dieser Dialog dient der Zuordnung des Slaves zu einer oder mehrerer der möglichen acht Gruppen.

  Die allgemeingültigen Gruppeneigenschaften (Sync-Mode und/oder Freeze-Mode) hingegen werden bei der Konfiguration der Master-Eigenschaften definiert, siehe Kap. "Registerkarte 'Allgemein'" auf Seite 51.

  Über die Schaltfläche "Globale Gruppeneigenschaften" gelangt man ebenfalls zu diesem Dialog.

  Die Gruppe(n), denen der Slave zugeordnet wurde, werden mit einem Haken markiert.

  Ein Slave-Gerät kann nur solchen Gruppen zugeordnet werden, deren Eigenschaften es unterstützt. Die diesbezüglichen Eigenschaften des jeweiligen Slaves (Sync-Mode / Freeze-Mode) werden oberhalb der Tabelle angezeigt. Vom dem Gerät unterstützte Modi sind mit einem Haken versehen

Anwenderparameter:

- **Anwenderparameter** sind, zusätzlich zu den Basis-DP-Parametern (siehe oben), individuelle Parameter eines DP-Slaves, die – wenn in der Gerätebeschreibungsdatei (GSD-Datei) definiert – hier angezeigt werden.

Die Parametertabelle zeigt für jeden Parameter den Parameternamen, den echten oder den symbolischen Wert (siehe unten, Option "Symbolische Werte") und die zugelassenen Werte, die auch in der Gerätebeschreibungsdatei definiert sind. Sie können die Parameterwerte nach einem Mausklick auf das entsprechende Feld in Spalte "Wert" editieren.

- **Symbolische Werte**: Sind in der Gerätebeschreibungsdatei (GSD-Datei) auch symbolische Namen für die Parameter angegeben, können Sie diese Option aktivieren, um die symbolischen anstelle der numerischen Werte in der Spalte "Wert" angezeigt zu bekommen.
- **Länge der Anwenderparameter (Byte)**: Aufsummierte Länge der Anwenderparameter, die über die Gerätebeschreibungsdatei definiert sind.
- **Voreinstellungen**:

  Über diese Schaltfläche werden die in der Tabelle angezeigten Werte auf die Standard-Einstellung zurückgesetzt.

Registerkarte 'Profibus DP Parameter'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor**.



Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

Der Dialog zeigt detailliert die Parameter des Moduls an.

Registerkarte 'Status'

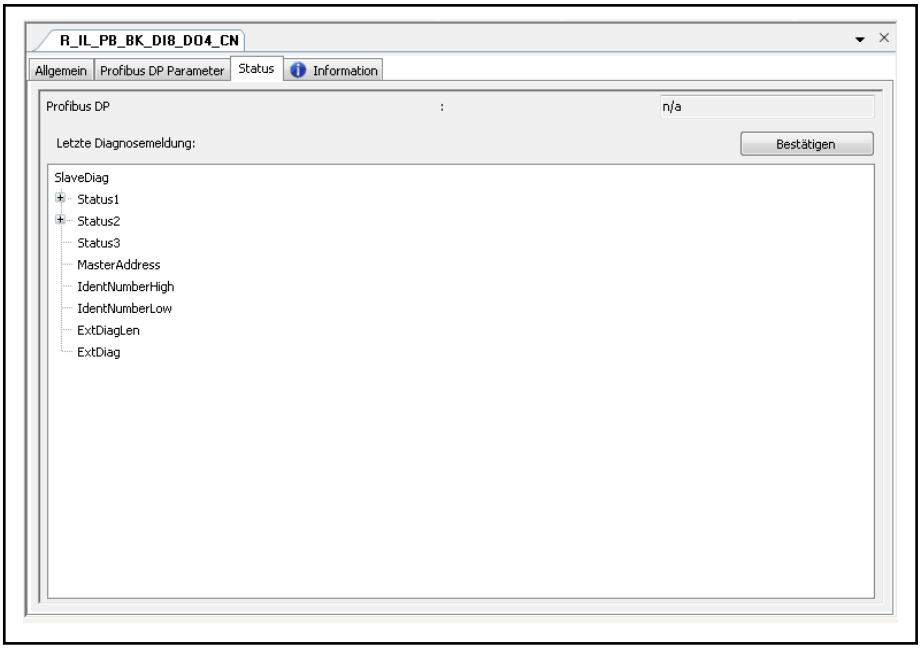


Abb. 5-23: Profibus DP Slave - Registerkarte 'Status'

Die Registerkarte "Status" zeigt Status-Informationen, z.B. "Läuft" (Bus aktiv), und "n/a" (keine Information vorhanden).

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

5.5.3 Profibus DP Master, Module zum Slave hinzufügen

Allgemeines

Die zum jeweiligen Slave passenden Module befinden sich in der Bibliothek "Peripheriegeräte", im Ordner "ProfibusDP" unterhalb des jeweiligen Slaves:

- E/A: E/A-Module für modulare Slaves.
- SPS: Vereinbarung gemeinsam genutzter Speicherräume zum Datenaustausch zwischen den beteiligten Steuerungen.

E/A-Module lassen sich nur in modular aufgebaute Slaves einfügen, siehe Kap. 5.5.2 "Profibus DP Master, Slaves konfigurieren" auf Seite 58.

Ziehen Sie aus der Bibliothek die benötigten Module in das Slave-Objekt.

Neue Module lassen sich im Projektexplorer auch zwischen bereits vorhandene Module einfügen.

Alternativ können Sie Module im Kontextmenü des Slaves über **Hinzufügen ▸ Modul ▸ ...** hinzufügen.

Hierbei wird das neue Modul als letztes Modul unter den Slave eingefügt.

Der Dialog zur Konfiguration und Parametrierung des jeweiligen IO-Moduls (Klemme) erscheint nach Doppelklick auf den entsprechenden Geräteknoten im Projektnavigator.

Konfigurationsdialog für Module am modularen Profibus-Slave

- Registerkarte 'Allgemein' auf Seite 67
- Registerkarte 'E/A-Abbild' auf Seite 68
- Registerkarte 'Status' auf Seite 69
- Registerkarte 'Information' auf Seite 69.

Registerkarte 'Allgemein'

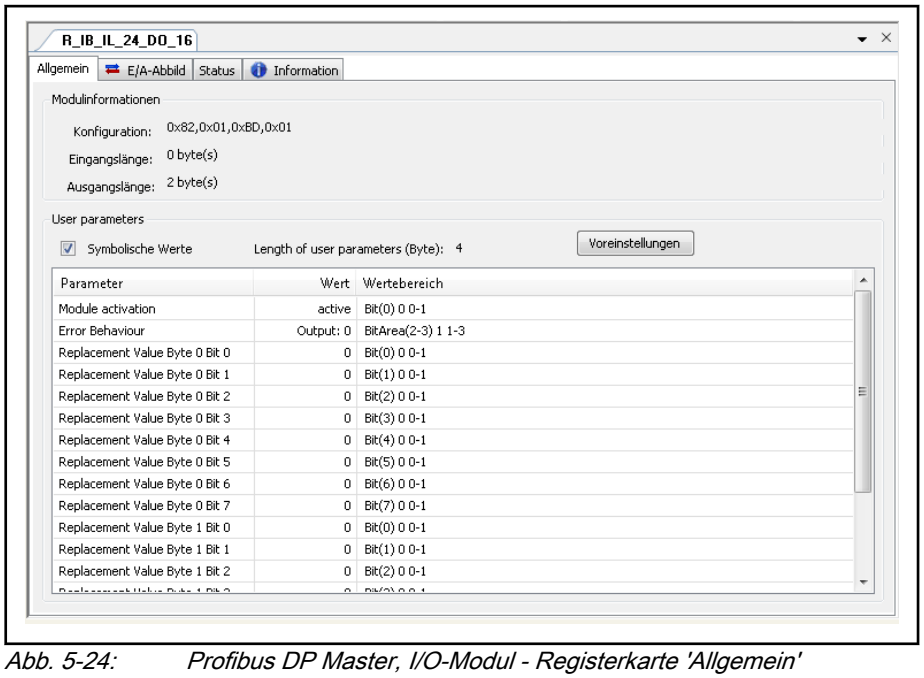


Abb. 5-24: Profibus DP Master, I/O-Modul - Registerkarte 'Allgemein'

Modulinformationen

- Der DP-Parameter-Dialog eines Ein- und Ausgangsmoduls in einer Profibus DP-Konfiguration beschreibt in Sektion Modulinformationen die Einstellungen für die Konfiguration (Modulbeschreibung gemäß Profibus DP-Standard) und die Länge der Eingangs- und Ausgangsdaten (Eingangslänge, Ausgangslänge) in Bytes, wie sie in der Gerätebeschreibungsdatei (GSD-Datei) definiert sind.

User parameters

- **User parameters (Anwenderparameter)** sind, zusätzlich zu den Basis-DP-Parametern (siehe oben), individuelle Parameter eines E/A-Moduls, die - wenn in der Gerätebeschreibungsdatei (GSD-Datei) des Buskopplers definiert - hier angezeigt werden.

  Die Parametertabelle zeigt für jeden Parameter den Parameternamen, den echten oder den symbolischen Wert (siehe unten, Option "Symbolische Werte") und die zugelassenen Werte, die auch in der Gerätebeschreibungsdatei definiert sind. Die Parameterwerte können nach einem Mausklick auf das entsprechende Feld in Spalte "Wert" editiert werden.
- **Symbolische Werte**: Sind in der Gerätebeschreibungsdatei (GSD-Datei) auch symbolische Namen für die Parameter angegeben, können Sie diese Option hier aktivieren, um diese symbolischen, anstelle der realen Werte in der Spalte "Wert" angezeigt zu bekommen.
- **Länge der Anwenderparameter (Byte)**: Die Länge der Anwenderparameter (Length of user parameters) gibt die Gesamtlänge aller Anwenderparameter aus der Gerätebeschreibung an. Nach einem Mausklick auf das jeweilige Feld in der Spalte "Wert" kann ein Parameterwert editiert werden.
- **Voreinstellungen**: Über diese Schaltfläche werden die in der Tabelle angezeigten Werte auf die Standard-Einstellung zurückgesetzt.



Detailliere Angaben zu den einzelnen Parametern finden Sie in der jeweiligen Modulbeschreibung.

Registerkarte 'E/A-Abbild'

Die Registerkarte dient der Zuordnung der Ein- und Ausgänge zu Variablen, die als lokale oder globale Variablen in den einzelnen POUs verwendet werden können.

Im Online-Modus wird der aktuelle Wert der Variablen angezeigt.

Die Zuordnung ist unter Mapping der Feldbus-Ein- und Ausgänge beschrieben (Mapping der Feldbus-Ein- und Ausgänge auf Seite 19).

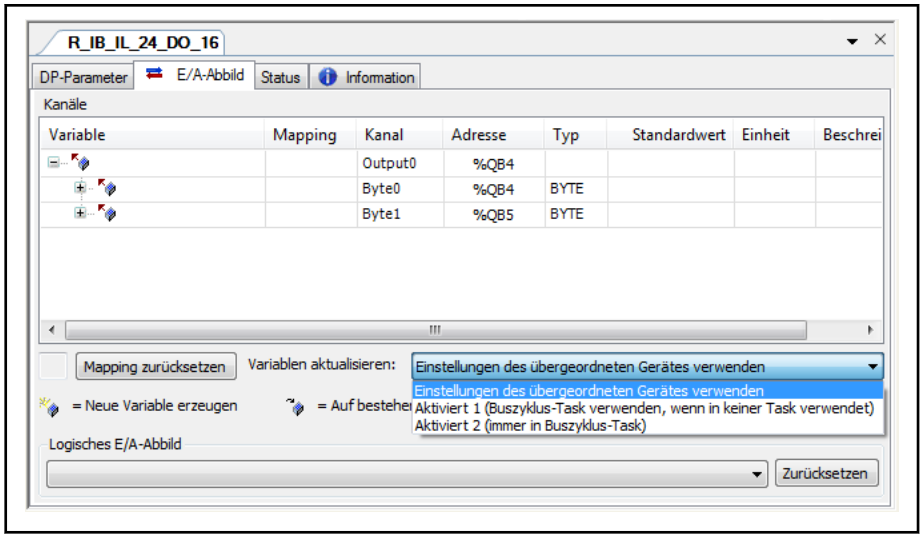


Abb. 5-25: Profibus DP Master, Registerkarte E/A-Abbild

Mapping zurücksetzen: löscht die im Editor vorgenommenen Zuordnung.

Variablen immer aktualisieren: ist diese Option aktiviert, werden alle Variablen in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

Registerkarte 'Status'

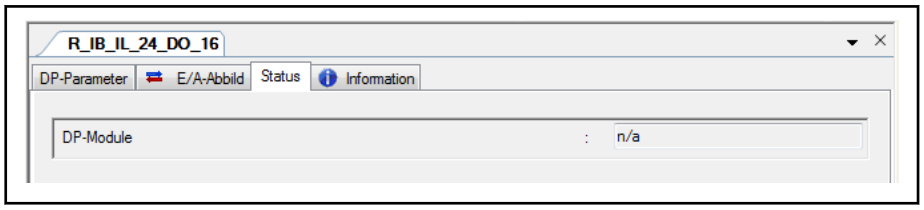


Abb. 5-26: Profibus DP Master, Slave-Module: Registerkarte Status

- "DP-Module":

  Der Bereich DP-Module zeigt im Onlinebetrieb Status-Informationen (z.B. "Läuft", "Läuft nicht (n/a)") aus der Steuerung.

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

5.6 Profibus DP, Slave (Steuerung als Slave)

5.6.1 Profibus DP Slave, konfigurieren

Allgemeines

Eine Steuerung kann entweder beim Anlegen der Steuerung oder über den Kontext-Menüpunkt **Gerät setzen** über ein Erweiterungsmodul als Profibus DP Slave konfiguriert werden.

Die Kopplung zwischen der Mastersteuerung und dem Slave erfolgt über Abbildspeicher, deren Inhalt ausgetauscht wird, siehe Kap. 5.6.3 "Profibus DP Slave, Master-Slave-Kopplung" auf Seite 76.

Klicken Sie zum Öffnen des Editorfensters im Projektexplorer doppelt auf das Profibus DP Slave-Objekt.

In den Dialogen werden Sie über die Konfiguration des gesamten Slaves, d.h. der Steuerung selbst als Slave, informiert und können sie gegebenenfalls modifizieren.

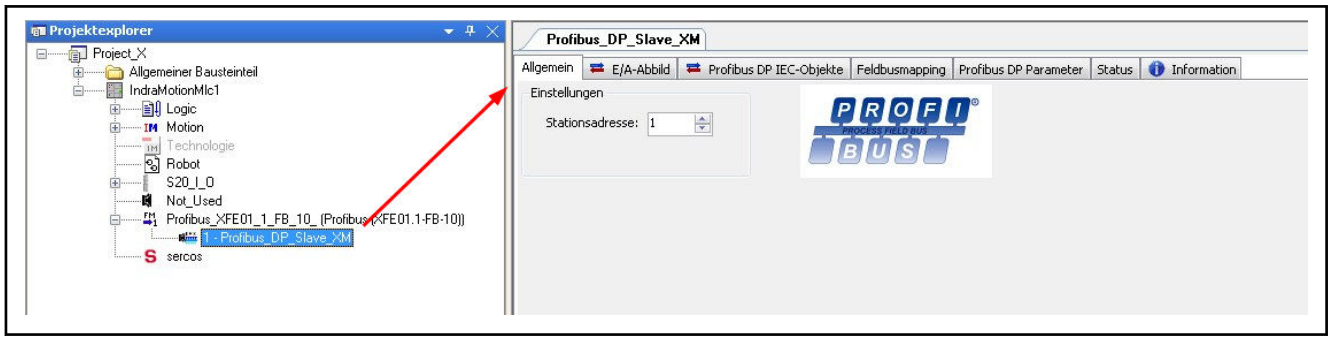


Abb. 5-27: Profibus DP-Slave-Objekt konfigurieren

Konfigurationsdialog für einen Profibus DP Slave

- Registerkarte 'Profibus DP-Parameter' auf Seite 72
- Registerkarte 'E/A-Abbild' auf Seite 71
- Registerkarte 'Feldbusmapping' auf Seite 71
- Registerkarte 'Profibus DP Konfiguration' auf Seite 73, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde
- Registerkarte 'Status' auf Seite 73
- Registerkarte 'Information' auf Seite 74

Registerkarte 'Allgemein'

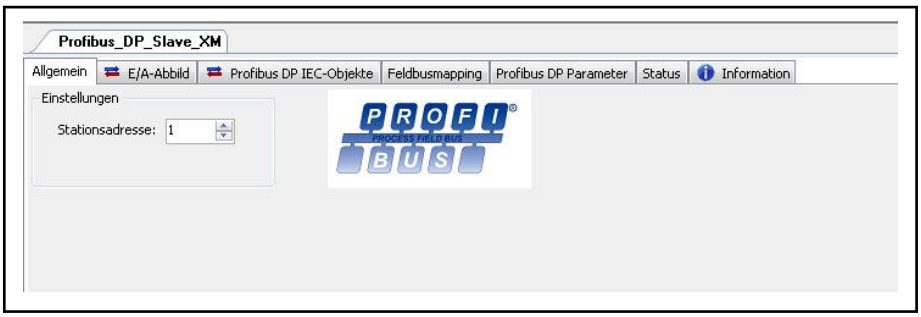


Abb. 5-28: Profibus DP-Slave-Objekt: DP-Parameter

Einstellungen

- **Stationsadresse**: Die Adresse wird automatisch eingetragen. Sie ist entsprechend der tatsächlichen Buskonfiguration anzupassen.

  Die Nummer ist identisch zu wählen mit der Nummer des für die Kopplung an den Profibus DP Masters vorgesehenen Slaves-Objektes.

Registerkarte 'E/A-Abbild'

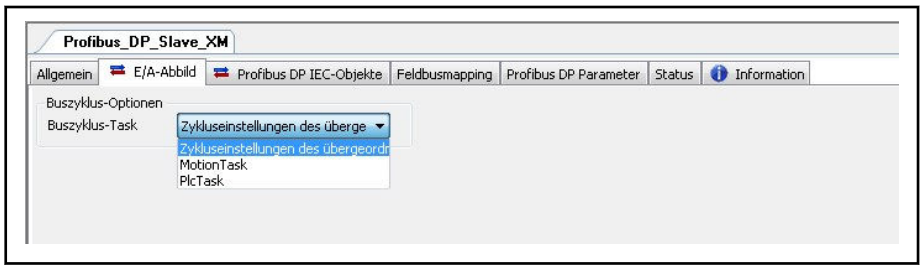


Abb. 5-29: Profibus DP-Slave-Objekt: E/A-Abbild

Buszyklus-Task: Durch die Auswahl einer Buszyklus-Task kann der Zyklus des Abbildtausches für den Profibus DP Slave an eine spezielle Task gebunden werden. Es ist sinnvoll in dieser Task auch die IO-Daten des Slaves zu verarbeiten.

Standardeinstellung: "Zykluseinstellung des übergeordneten Busses verwenden"

Diese Einstellung bewirkt, dass die in "Buszyklus-Optionen" der Steuerung (Doppelklick auf die konkrete Steuerung im Projektnavigator, SPS-Einstellungen) vorgenommene Taskeinstellung für den konkreten Bus übernommen wird.

Registerkarte 'Profibus DP IEC-Objekte'

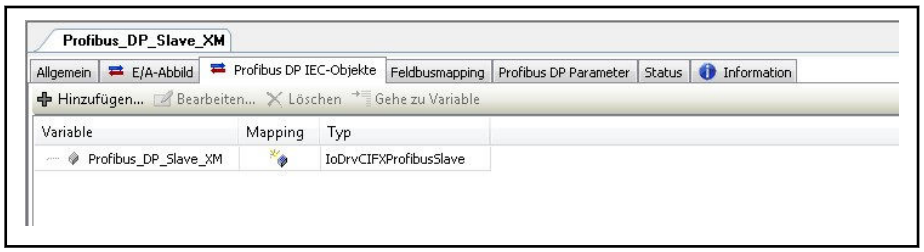


Abb. 5-30: Profibus DP-Slave-Objekt: Profibus DP IEC-Objekte

Die Registerkarte "Profibus DP IEC-Objekte" zeigt Informationen über den Profibus-Master Treiber an, den Sie im Anweisungsteil aufrufen können.

IEC-Objekte: Der zum konkreten Bus-Objekt gehörende Datenbestand ist als (projektglobale) Variable `Profibus_DP_Slave_1` (Profibus DP Slave-Objekt mit der Nummer 1) vom Typ `IoDrvCIFXProfibusSlave` adressierbar.

Registerkarte 'Feldbusmapping'

Das Feldbusmapping ermöglicht den Zugriff auf Variablen der Steuerung von einem übergeordneten Feldbusmaster aus.

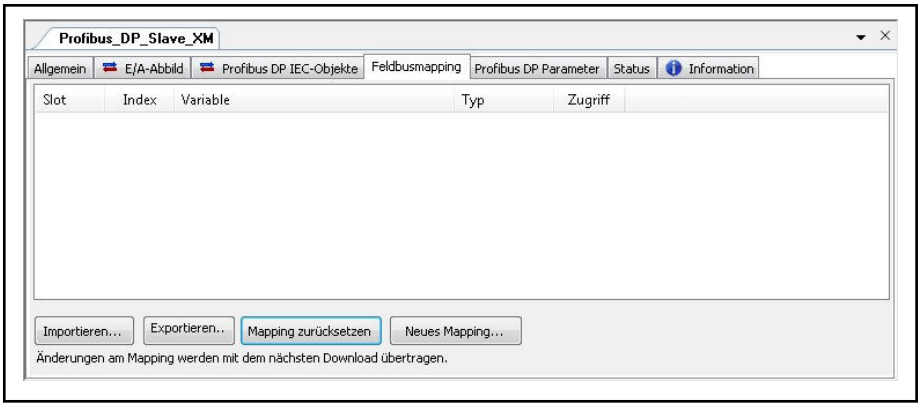


Abb. 5-31: Profibus DP-Slave-Objekt: Feldbusmapping

Wenn man auf eine Variable der Steuerung von einem übergeordneten Master aus zugreifen will, dann muss dieser Variablen zuerst eine Adresse zugeordnet werden (Mapping). Über diese Adresse kann der übergeordnete Master über azyklische Dienste auf die Daten der Steuerung zugreifen.

Um das Feldbusmapping nutzen zu können, muss die Steuerung als entsprechender Feldbusslave konfiguriert sein.

Details siehe Feldbusmapping, Zugriff vom Feldbusmaster auf Variablen des Feldbusslaves auf Seite 24.

Registerkarte 'Profibus DP-Parameter'

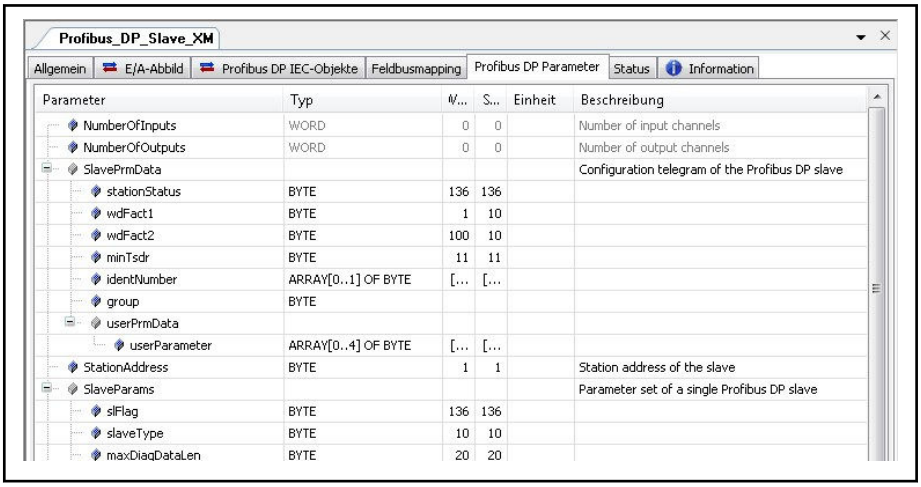


Abb. 5-32: Profibus DP-Slave-Objekt: DP-Parameter

Die DP-Parameter eines DP-Slaves sind im Unterschied zu denen des Masters nicht ein Standard-Set, sondern werden in der Gerätebeschreibung individuell für jedes Gerät definiert. Sie können im DP-Parameter-Dialog durch den Anwender modifiziert werden.

Parameter

- **T_SDR (Bit)**: Time Station Delay Responder. Reaktionszeit, nach welcher der Slave frühestens an den Master antworten darf.

  Bit: Zeiteinheit für die Übertragung eines Bits über Profibus;

  Kehrwert der Übertragungsrate; z.B. 1 Bit bei 12 MBaud=1/12.000.000 Bit/sek=83 ns.

- **Lock/Unlock**: Slave wird für andere Master gesperrt oder freigegeben. Wählen Sie aus der Auswahlliste eine der folgenden Optionen:

- **0 (T_SDR unlock)**: min. T_SDR und slave-spezifische Parameter dürfen überschrieben werden.
- **1 (Will be unlocked)**: Slave für andere Master freigegeben.
- **2 (Lock)**: Slave für andere Master gesperrt, alle Parameter werden übernommen.
- **3 (Unlock)**: Slave für andere Master erneut freigegeben.

Überwachung

- **Überwachungssteuerung:** Wenn Sie diese Option aktivieren, gilt die eingetragene Überwachungszeit. Wird der Slave innerhalb dieser Zeit nicht vom Master angesprochen, geht er in den Initialisierungszustand zurück.
- **Zeit (ms)**: Überwachungszeit, relevant für den Fall, dass die Option "Überwachungssteuerung" aktiv ist.

Gruppen:

In Vorbereitung

Anwenderparameter:

- **Anwenderparameter** sind, zusätzlich zu den Basis-DP-Parametern (siehe oben), individuelle Parameter eines DP-Slaves, die – wenn in der Gerätebeschreibungsdatei (GSD-Datei) definiert – hier angezeigt werden.

  Die Parametertabelle zeigt für jeden Parameter den Parameternamen, den echten oder den symbolischen Wert (siehe unten, Option "Symbolische Werte") und die zugelassenen Werte, die auch in der Gerätebeschreibungsdatei definiert sind. Sie können die Parameterwerte nach einem Mausklick auf das entsprechende Feld in Spalte "Wert" editieren.
- **Symbolische Werte**: Sind in der Gerätebeschreibungsdatei (GSD-Datei) auch symbolische Namen für die Parameter angegeben, können Sie diese Option aktivieren, um die symbolischen anstelle der numerischen Werte in der Spalte "Wert" angezeigt zu bekommen.
- **Länge der Anwenderparameter (Byte)**: Aufsummierte Länge der Anwenderparameter, die über die Gerätebeschreibungsdatei definiert sind.
- **Voreinstellungen**: Über diese Schaltfläche werden die in der Tabelle angezeigten Werte auf die Standard-Einstellung zurückgesetzt.

Registerkarte 'Profibus DP Konfiguration'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor**.

☞ Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

☞ Bei laufendem Bus ist eine Übertragung geänderter Parameter durch Betätigung der Schaltfläche <Parameter schreiben> möglich.

Registerkarte 'Status'

Das Registerblatt "Status" zeigt Status-Informationen (z.B. "Läuft" (Bus aktiv), und "n/a" (keine Information vorhanden)) und spezifische Diagnosemeldun-

gen aus dem jeweiligen Gerät, auch zur verwendeten Karte und dem internen Bussystem.

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

5.6.2 Profibus DP Slave, Koppelmodule hinzufügen

Allgemeines

Die Schnittstelle eines Profibus DP Slave zum Profibus DP Master ist als gemeinsamer Speicherraum ausgeführt. Dieser kann sich in seiner Größe bedarfsabhängig unterscheiden.

Der Koppelbereich zum Datenaustausch mit dem Profibus DP Master wird in Form von Ein- und Ausgangsmodulen am Profibus DP Slave-Objekt festgelegt.

Hierzu können Sie im Kontextmenü des Profibus DP Slave Objekts über **Hinzufügen ▸ Gerät ▸ ...** entsprechende Ein- und Ausgangsmodule hinzufügen. Es stehen die in der Abbildung dargestellten Module zur Verfügung.

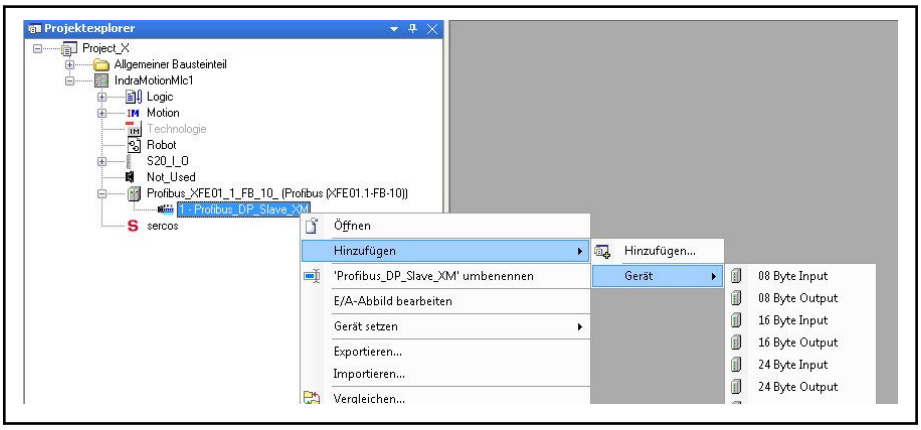


Abb. 5-33: Profibus DP-Slave-Objekt: Module hinzufügen

Eingänge auf der Profibus DP Master-Seite werden zu Ausgängen auf der Profibus DP Slave-Seite und umgekehrt.

Die Stationsnummern an der Koppelstelle sind gleich.

Klicken Sie zum Öffnen der Dialoge zur Konfiguration der unterhalb des Profibus DP Slave Objekts eingefügten Module doppelt auf das jeweilige Modul-Objekt.

Alternativ können Sie die Dialoge über das Kontextmenü des Moduls öffnen **Öffnen**. In dem Konfigurationsdialog der Profibus DP Slave Koppelmodule werden folgende Register angeboten:

Register

- Registerkarte 'Allgemein' auf Seite 75
- Registerkarte 'DP-Module Parameter' auf Seite 75, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde
- Registerkarte 'DP-Module E/A-Abbild' auf Seite 75

- Registerkarte 'Status' auf Seite 76
- Registerkarte 'Information' auf Seite 76

Registerkarte 'Allgemein'

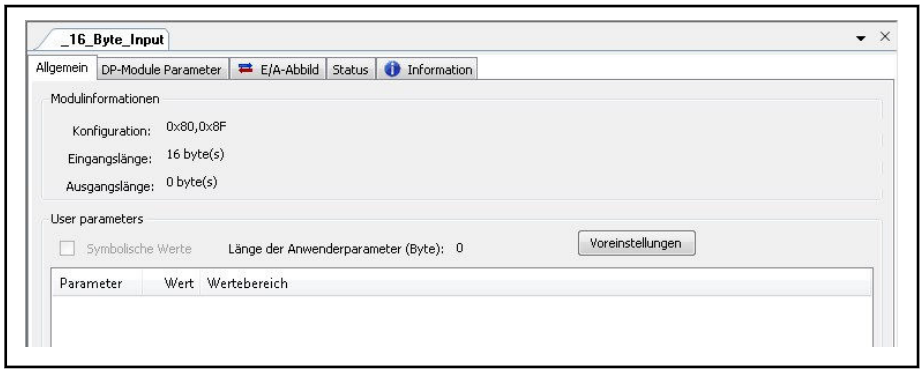


Abb. 5-34: Profibus DP Slave, Slave-Module: Allgemein

Modulinformationen

- Die Registerkarte "Allgemein" eines Ein- und Ausgangsmoduls in einer Profibus DP-Konfiguration beschreibt in Sektion Modulinformationen die Einstellungen für die Konfiguration (Modulbeschreibung gemäß Profibus DP-Standard) und die Länge der Eingangs- und Ausgangsdaten (Eingangslänge, Ausgangslänge) in Bytes, wie sie in der Gerätebeschreibungsdatei (GSD-Datei) definiert sind.

Anwenderparameter:

→ Keine

Registerkarte 'DP-Module Parameter'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor**.

Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

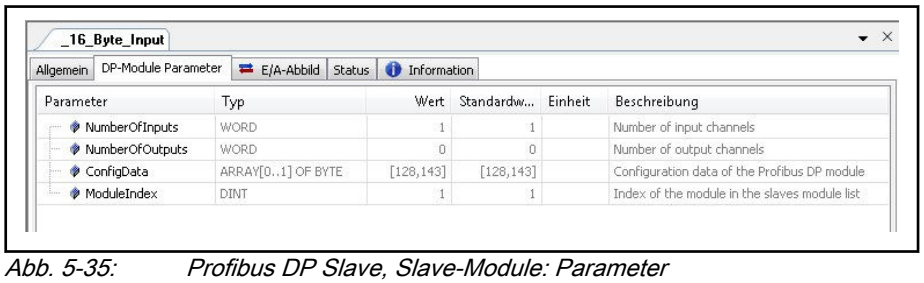


Abb. 5-35: Profibus DP Slave, Slave-Module: Parameter

Der Dialog gibt Auskunft über Lage und Größe der Parameter.

Registerkarte 'DP-Module E/A-Abbild'

Die Registerkarte dient der Zuordnung der Modul-Ein- und Ausgänge zu Variablen, die als lokale oder globale Variablen in den einzelnen POUs verwendet werden können.

Im Online-Modus wird der aktuelle Wert der Variablen angezeigt.

Die Zuordnung ist unter Mapping der Onboard-, Inline- und Feldbus-Ein- und Ausgänge beschrieben (Mapping der Feldbus-Ein- und Ausgänge auf Seite 19).

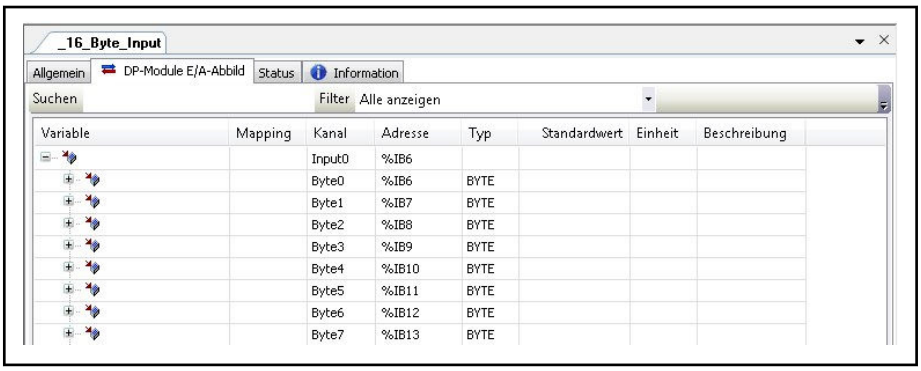


Abb. 5-36: Profibus DP Slave, DP-Module E/A-Abbild

Mapping zurücksetzen: Löscht die im Editor vorgenommene Zuordnung.

Variablen immer aktualisieren: Ist diese Option aktiviert, werden alle Variablen in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

Registerkarte 'Status'

"DP-Module":

Der Bereich DP-Module zeigt im Onlinebetrieb Status-Informationen (z.B. "Läuft", "Läuft nicht (n/a)") aus der Steuerung.

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

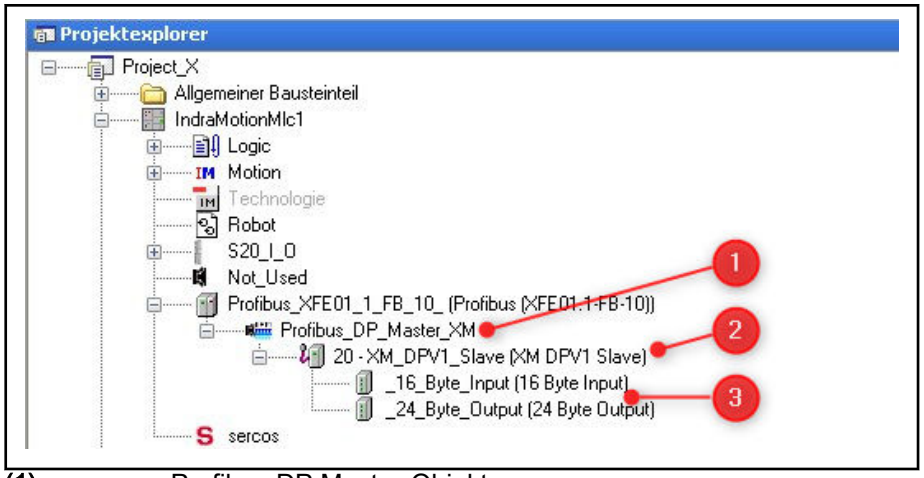
5.6.3 Profibus DP Slave, Master-Slave-Kopplung

Zur Kopplung einer als Profibus DP Slave zu betreibenden Steuerung stehen in der **Bibliothek** ▸ **Peripherie** ▸ **ProfibusDP** ▸ **SPS** fünf SPS-Koppel-Module zur Verfügung, die die Ankopplung nachstehender Steuerungen erlauben:

- **L20 DPV1 Slave** für die Steuerung IndraLogic L20 DP,
- **L40 DPV1 Slave** für die Steuerung IndraLogic L40 DP und
- **Lx5 DPV1 Slave** für die Steuerungen MLC L25/L45/L65/L75.
- **XM DPV1 Slave** für die Steuerungen MLC XM2/XM4.
- **VPx DPV1 Slave** für die Steuerung MLC VPx.

Durch das Hinzufügen eines SPS-Koppelmoduls im Projektnavigator erfolgt die Aktivierung einer Firmwarekomponente im Profibus DP Master zur Verbindung mit einer Profibus DP Slave-Steuerung über den Profibus.

Jedes dieser SPS-Koppelmodule besitzt in der Gerätebibliothek eine Auswahl von Modulen, die im Abbildspeicher des Masters Eingangs- und Ausgangsvariablen zum Datenaustausch zwischen der Profibus DP Master-Steuerung und der Profibus DP Slave-Steuerung frei schalten.



(1) Profibus DP Master-Objekt
(2) Slave zur Ankopplung einer XM-Steuerung
(3) Speichermodule für den Datenaustausch zwischen den Steuerungen.

Abb. 5-37: Slave mit Speichermodulen zur Kopplung einer Profibus DP Slave-Steuerung

Das nachstehende Bild zeigt die Kopplung zwischen zwei Steuerungen auf Profibus DP-Basis.

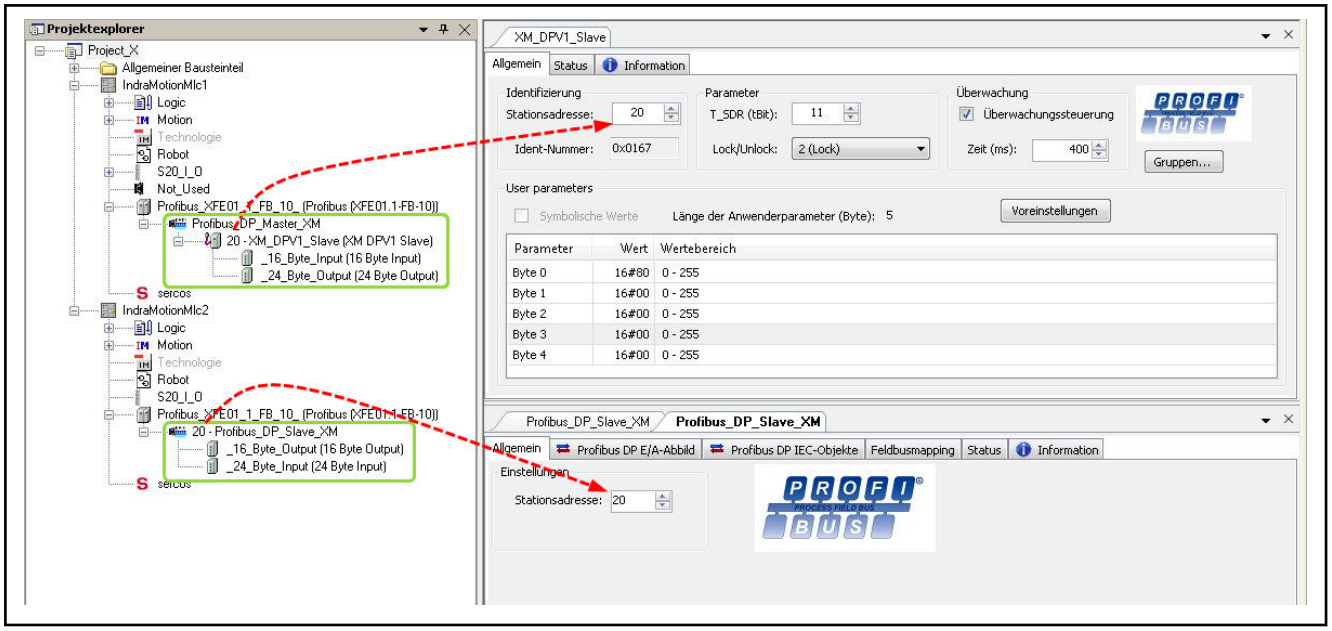


Abb. 5-38: Steuerungskopplung über Profibus DP

Eingänge auf der Profibus DP Master-Seite werden zu Ausgängen auf der Profibus DP Slave-Seite und umgekehrt. Die Stationsnummern an der Koppelstelle sind gleich.

6 Profinet IO

6.1 Features des Profinet IO Controller

Die aktuelle Implementation des Profinet IO Controller beinhaltet folgende Funktionalitäten:

- zyklischer Datenverkehr
- azyklischer Datenverkehr (FBs in RIL_ProfinetIO.library)
- GSDML-Dateiimport in die Gerätedatenbank
- Busscan von Profinet IO Devices
- Profinet IO-Device-bezogener Verbindungsstatus in der Oberfläche
- Diagnose des Profinet IO Controller über FB
- Profinet IO Device-bezogener Status über FB
- Shared Device
- Unterstützte Protokolle:
  - RTC (real-time cyclic) Class 1
  - RTA (real-time acyclic)
  - DCP (Discovery and Configuration)
  - CL-RPC (Connectionless Remote Procedure Call)
  - LLDP (Link Layer Discovery Protocol)
  - SNMP v1 (Simply Network Management Protocol)
- IP-Kanal
  - Manuell aktivierbar mit separater IP-Adresse, siehe Kap. 4.6 "IP-Kanal bei PROFINET IO & EtherNet/IP (netX)" auf Seite 33

Funktion / Eigenschaft	Wert
Max. Anzahl an Devices, ein Device mit Send Clock = 1 ms	64 [2) 3)]
Max. Anzahl an Devices, alle Devices mit Send Clock > 1 ms	128 [2) 3)]
Max. Anzahl an Modulen pro Device	255
Max. Anzahl an Submodulen (Summe über alle Devices)	2048
Baudrate	100 MBit/s
Auto negotiation/ autocrossing	Ja
Min. Zykluszeit	1 ms [2) 3)]
Max. Anzahl zyklischer Eingangsdaten	5652 Byte [1)]
Max. Anzahl zyklischer Ausgangsdaten	5700 Byte [1)]
Max. Anzahl zyklischer Eingangsdaten pro Device	1440 Byte [1)]
Max. Anzahl zyklischer Ausgangsdaten pro Device	1440 Byte [1)]
Max. Telegrammdaten azyklisch pro Device / Telegramm	1392 Byte
Max. Telegrammdaten azyklisch pro Device / Request	4096 Byte

Tab. 6-1: Technische Daten

[1)] Inklusive IOxS Statusbytes (= Modulanzahl "E+A" + 4)

[2] Wenn bei einem einzigen Device eine Send Clock = 1 ms eingestellt ist, können max. 64 Devices betrieben werden.

☞ Manche Devices unterstützen nur eine Send Clock von 1 ms und ein höherer Wert kann nicht eingestellt werden!

☞ Die Buszyklus-Task muss folgende Bedingung erfüllen:

Zykluszeit der Buszyklus-Task ≥ (2 * kleinste konfigurierte Send Clock).

Die Buszyklus-Task ist in der Defaulteinstellung "Unspezifiziert" die schnellste zyklische SPS-Task, siehe auch Kap. 4.5 "Buszyklus-Task: Kopieren der E/A-Daten zwischen SPS und Feldbus-Stack" auf Seite 29. (Wenn bei einem PNIO Controller keine Devices konfiguriert sind, gilt eine interne Default Send Clock von 4 ms, die nicht geändert werden kann).

Ansonsten kann gegebenenfalls eine Watchdog-Exception beim PROFINET Datenaustausch entstehen. Hierzu wird auch eine Compiler-Warnung "Kritische Buszykluszeit ..." in IndraWorks angezeigt.

Ein externes CIFX Event für eine Bus-synchrone SPS-Verarbeitung wird derzeit nicht unterstützt.

[3] **Empfehlung zur Einstellung der Zykluszeiten**

Unter den Voraussetzungen

- max. 30 % Auslastung einer 100 Mbit Vollduplexverbindung
- Nutzdaten 50 Byte E/A

werden folgende minimale Zykluszeiten empfohlen:

Anzahl Devices	Min. Zykluszeit bei allen Devices ("Send Clock x Reduction Ratio")
bis 32	1 ms
bis 64	2 ms
bis 128	4 ms

☞ Bei sporadischen Verbindungsabbrüchen in der zyklischen Kommunikation ist eine höhere Zykluszeit ("Send Clock x Reduction Ratio") einzustellen. Ursache hierfür kann u.a. ein begrenzter Datendurchsatz in Switches oder in den PN-Devices (z.B. bei Linientopologien) sein.

6.2 Features des Profinet IO Device (Steuerung als Slave)

Allgemeines

Die aktuelle Implementation des Profinet IO Device beinhaltet folgende Funktionalitäten:

- zyklischer Datenverkehr
  - Real Time
  - Isochrone Real Time, keine synchrone Applikation
- azyklischer Datenverkehr
  - Feldbusmapping

- Profinet IO Device-bezogener Verbindungsstatus in der Oberfläche
- Diagnose des Profinet IO Device über FB
- Shared Device
- Unterstützte Protokolle:
  - RTC (real-time cyclic) Class 1
  - RTC (real-time cyclic) Class 3
  - RTA (real-time acyclic)
  - DCP (Discovery and Configuration)
  - CL-RPC (Connectionless Remote Procedure Call)
  - LLDP (Link Layer Discovery Protocol)
  - SNMP (Simple Network Management Protocol)
  - MRP-Client (Media Redundancy Protocol - Client)

Folgende Funktionalitäten sind für zukünftige Erweiterungen vorgesehen:

- Alarme



Jedes konfigurierte Profinet IO Device benötigt 8,6 kB der persistenten Retain-Daten des IEC-Programms.

Hinweise zur Konfiguration siehe Kap. "Profinet IO Device - Geräteeditor" auf Seite 126.

Funktion / Eigenschaft	Wert
Max. Anzahl an IO-Modulen	128 Slots (richtungsunabhängig)
Max. Anzahl an Submodulen	132 (4 Kopfstation + 1 pro IO Modul)
Max. IO Verbindungen	4
Baudrate	100 MBit/s
Auto negotiation/ autocrossing	Ja
Min. Zykluszeit (RT / IRT ab 14V22)	RT: 1 ms IRT: 250 µs
Max. Anzahl zyklischer Eingangsdaten	1440 Byte
Max. Anzahl zyklischer Ausgangsdaten	1440 Byte
Max. Telegrammdaten pro azyklischem Telegramm	8192 Byte
Identification & Maintenance	I&M 0-4

Tab. 6-2: Technische Daten

6.3 Profinet IO, Grundlagen

6.3.1 Begriffe und Abkürzungen

Controller Der Profinet IO Feldbus-Master wird **Controller** genannt.

Device Die Profinet IO Feldbus-Slaves nennt man **Devices**.

Stationsname Das Kriterium zur Adressierung eines Devices ist der Devicename (=Stationsname).

GSDML-Datei Dies ist die Gerätebeschreibung, mittels derer IndraWorks die Profinet IO Devices bekannt gemacht werden.

RT / IRT – RT Class1 / RT Class3 Real Time (RT), oder RT Class1 ist die unsynchronisierte Echtzeit Kommunikation, jedes Device fährt seinen eigenen Zyklus.

Isochrone Real Time (IRT), oder RT Class3 ist die synchronisierte Echtzeit Kommunikation. Die Zykluszeit beginnt für alle Devices zum gleichen Zeitpunkt. Jedem Device wird ein Zeitschlitz zum Senden der zyklischen Daten zugewiesen.

Es ist nicht zwingend, dass dieser synchrone Takt bis zur Applikation weitergereicht wird. Die Applikation kann auch unsynchron laufen, bei synchroner Buskommunikation.

6.3.2 Bedeutung von Gerätenamen bei Profinet IO

Namensbasierte Geräteadressierung

Bei Profinet wird jedem Gerät (IO-Device) ein logischer, individueller und eindeutiger Gerätename (Devicename, Stationsname, IdentStr) zugeordnet.

Anhand dieses Gerätenamens teilt der Profinet IO-Controller den IO-Devices die konfigurierte IP-Adresse zu.

Der eigentliche Profinet Datenverkehr wird im Folgenden über diese IP-Adresse und die damit aufgelöste MAC-Adresse abgewickelt.

Bei der Konfiguration des Profinet IO-Controllers muss darauf geachtet werden, dass die Stationsnamen so eingestellt werden, wie sie auch auf den IO-Devices verwendet werden (sollen).

Benennung von IO-Devices - meistgenutzte Verfahren

1. Der Stationsname kann mit einem Konfigurations-Tool auf dem Gerät editiert, oder konfiguriert werden.

   Beispiel:

   Bei einer Steuerung als IO-Device erfolgt die Konfiguration mittels IndraWorks.

   Bei der Konfiguration wird ein Stationsname festgelegt und zusammen mit der Applikation auf die Steuerung geladen.

2. Der Gerätename ist auf dem Gerät hardwareseitig einstellbar, beispielsweise mittels Dip-Switch, oder Jumper.

   Beispiel: S67-PN-BK

3. Der Gerätename wird mittels IndraWorks in das Gerät geschrieben (und im Gerät gespeichert).

   Hierzu im Projektexplorer über das Kontextmenü des Profinet IO Controllers den Dialog "Geräte Suchen" aufrufen und das Gerät benennen.

☞ Der Stationsname wird zur Adressierung des Profinet IO-Devices benutzt.

Fehlende oder abweichende IP-Adress-Einstellungen auf Seiten des Devices werden ignoriert.

Der Stationsname darf aus folgenden Zeichen bestehen:

- Kleinbuchstaben ohne Umlaute "a" bis "z".

  Großbuchstaben ohne Umlaute "A" bis "Z" sind zulässig, werden aber intern in Kleinbuchstaben gewandelt.

  Die Stationsnamen sind nicht case-sensitiv.
- Ziffern "0" bis "9" (dürfen nicht am Anfang des Namens stehen).
- Sonderzeichen Bindestrich "-" und Punkt "." (dürfen weder am Anfang noch am Ende des Namens stehen).
- Der Name darf max. 240 Zeichen lang sein. Bei manchen Geräten / Herstellern gilt die Beschränkung 127 Zeichen.

  Der Name kann mittels Punkten "." in Teilstrings gegliedert werden. Ein solcher Teilstring darf max. 63 Zeichen lang sein.

  Der Name darf nicht mit der Zeichenfolge "port-xyz" (mit x,y,z = 0...9) beginnen.

  Der Name darf nicht die Form "a.b.c.d" (mit a,b,c,d = 0...999) haben.

6.3.3 Shared Device

Bei einem Shared Device teilen sich mehrere Profinet IO Controller die (Sub-)Module eines Profinet IO Devices. Zu jedem (Sub-)Modul kann nur eine Verbindung gleichzeitig aufgebaut werden. Es gibt keine Priorisierung der Verbindung. Es gilt das Prinzip „First come, first served“.

Beispiel in nachfolgender Abbildung:

Zwei XM-Steuerungen als Profinet IO Controller im Datenaustausch mit S20-PN-BK+ (unten).

Dargestellt ist mit Farben (blau, rot) jeweils die E/A-Kommunikationsbeziehung zwischen Profinet IO Controller und Profinet IO Device.

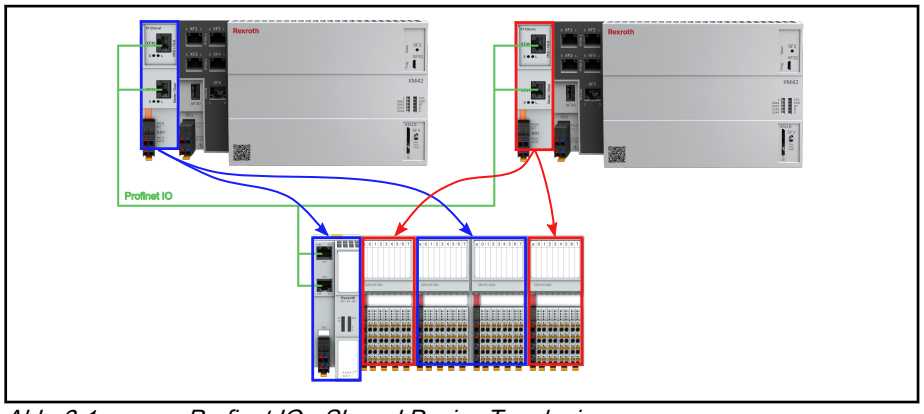


Abb. 6-1: Profinet IO - Shared Device Topologie

6.3.4 Media Redundancy Protokoll (MRP)

Das "Media Redundancy Protokoll" ermöglicht es eine Ethernet-Ring-Redundanz aufzubauen. Hierfür muss jedes Gerät, dass Teil des Ringes ist, das MRP unterstützen. Es gibt verschiedene Rollen in einem MRP-Ring:

- Den MRP-Redundancy-Manager (RM), welcher den Ringzustand überwacht, logisch auftrennt und bei Ringbruch die Umschaltung von Ring zu Linie vornimmt.

- Den MRP-Client, welcher als Ringteilnehmer von einem übergeordneten Profinet IO Controller konfiguriert wird und die Umschaltmechanismen unterstützt.

Umschaltzeiten bei Ringbruch entsprechen typischerweise ~200 ms, abhängig von der Ringgröße, Teilnehmerzahl und Ethernetleitungslänge. Die Zykluszeitüberwachung muss entsprechend größer gewählt werden, um bei Ringbruch den Verbindungsabbruch zu verhindern.

6.3.5 Identifizieren

Um im Profinet IO Netzwerk die Gerätenamen einer Hardware zuzuordnen gibt es einen Profinet Dienst, der den Teilnehmer Aufblinken lässt.

Siehe Profinet IO Controller Dialog "Geräte suchen", Kap. "Geräte suchen " auf Seite 113, Befehl "LED blinken".

VPx	Bei einer Identifizierung wird der Text "Identifizierung PROFINET Device" in dem Dialog "Diagnosis and Settings" angezeigt.
XMx	Bei einer Identifizierung blinkt die DIAG3 LED dreimal innerhalb von drei Sekunden.
XFE01.1-FB-03/04	Bei einer Identifizierung blinkt die SF-LED dreimal innerhalb von drei Sekunden.

Tab. 6-3: Identifizierung der Rexroth Steuerungen mit Profinet IO Schnittstelle

6.4 Profinet IO, Konfiguration Übersicht

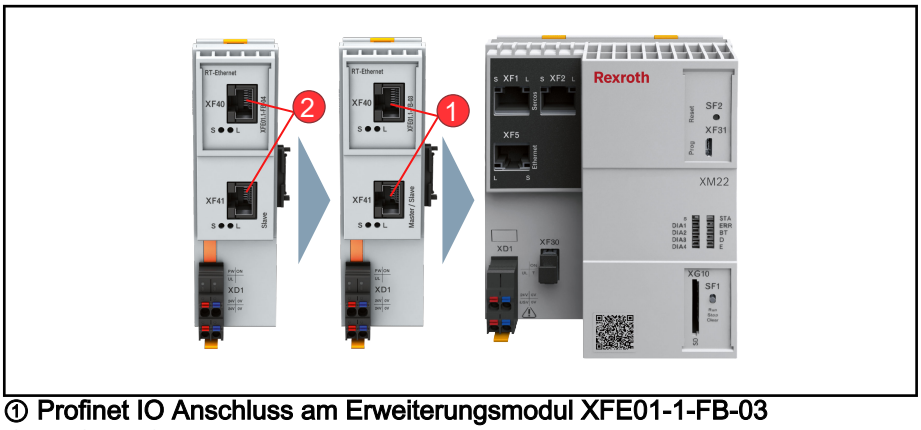
Profinet IO wird von XM und VPx Steuerungen unterstützt.

Das folgende Kapitel gibt eine Übersicht über die Anschlussmöglichkeiten an den jeweiligen Steuerungstyp.

Anbindung an XM2-Steuerungen

Bei XM2 Steuerungen erfolgt die Profinet IO Anbindung über Erweiterungsmodule:

- Maximal 3 Erweiterungsmodule **XFE01-1-FB-03**, jeweils konfigurierbar als Profinet IO Controller, oder als Profinet IO Device
- Maximal 3 Erweiterungsmodule **XFE01-1-FB-04**, konfigurierbar als Profinet IO Device



① Profinet IO Anschluss am Erweiterungsmodul XFE01-1-FB-03
② Profinet IO Anschluss am Erweiterungsmodul XFE01-1-FB-04

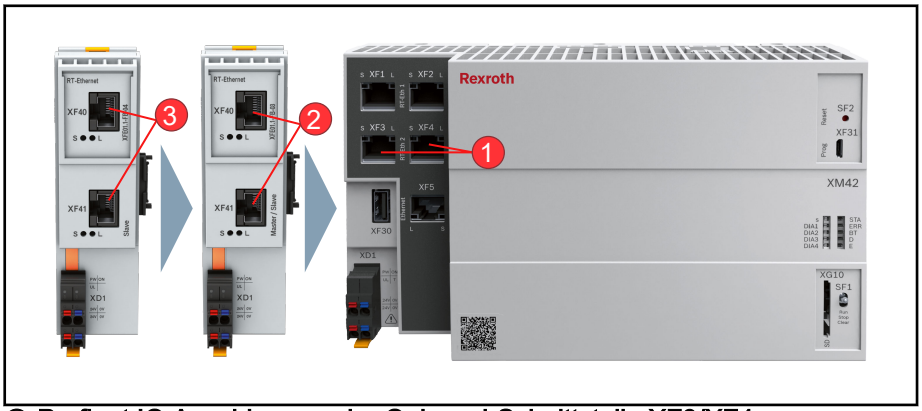
Abb. 6-2: Profinet IO – Anbindung an XM2 Steuerung

Beide Anschlüsse am Erweiterungsmodul sind gleichwertig und können zur Verbindung mit Profinet IO Geräten verwendet werden (es gibt keinen dedizierten Ein- oder Ausgang).

Anbindung an XM4-Steuerungen

Bei XM4 Steuerungen erfolgt die Profinet IO Anbindung entweder über die Onboard-Schnittstelle der Steuerung, oder über Erweiterungsmodule:

- Onboard-Schnittstelle **XF3/XF4** konfigurierbar als Profinet IO Controller, oder Device
- Maximal 3 Erweiterungsmodule **XFE01-1-FB-03**, jeweils konfigurierbar als Profinet IO Controller, oder als Profinet IO Device
- Maximal 3 Erweiterungsmodule **XFE01-1-FB-04**, konfigurierbar als Profinet IO Device



① Profinet IO Anschluss an der Onboard-Schnittstelle XF3/XF4
② Profinet IO Anschluss am Erweiterungsmodul XFE01-1-FB-03
③ Profinet IO Anschluss am Erweiterungsmodul XFE01-1-FB-04

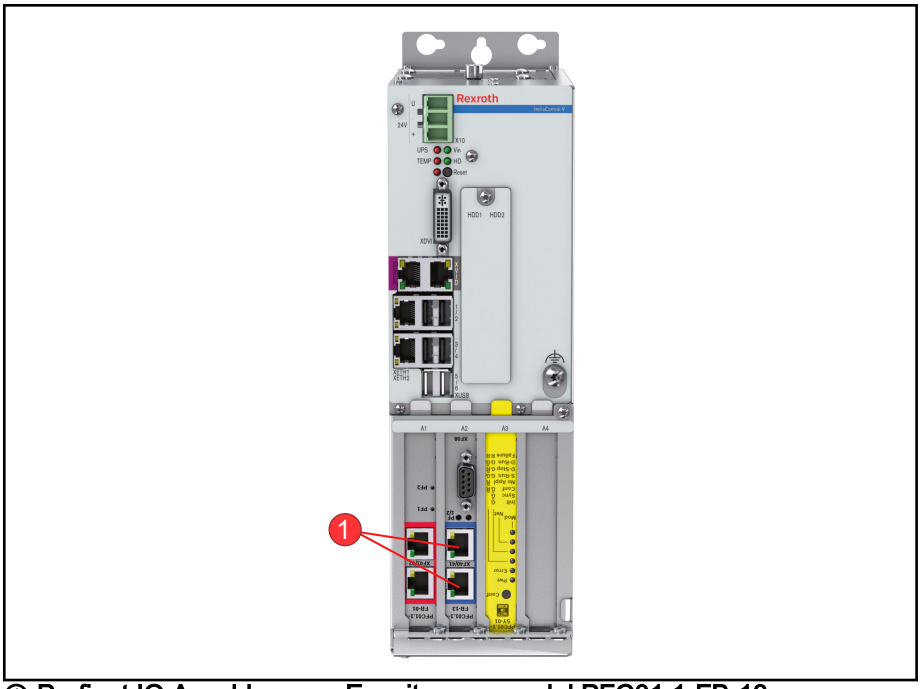
Abb. 6-3: Profinet IO – Anbindung an XM4 Steuerung

Beide Anschlüsse am Erweiterungsmodul und an der Onboard-Schnittstelle sind gleichwertig und können zur Verbindung mit Profinet IO Geräten verwendet werden (es gibt keinen dedizierten Ein- oder Ausgang).

Anbindung an VPx-Steuerungen

Bei VPx Steuerungen erfolgt die Profinet IO Anbindung über Erweiterungsmodule:

- Maximal 2 Erweiterungsmodule **PFC01.1-FB-13**, jeweils konfigurierbar als Profinet IO Controller, oder Device



① Profinet IO Anschluss am Erweiterungsmodul PFC01.1-FB-13

Abb. 6-4: Profinet IO – Anbindung an VPx Steuerung

Beide Anschlüsse am Erweiterungsmodul sind gleichwertig und können zur Verbindung mit Profinet IO Geräten verwendet werden (es gibt keinen dedizierten Ein- oder Ausgang).

Profinet IO – Bus Topologie

Eine einfache Topologie besteht aus der Steuerung als Profinet IO Controller und den Profinet Devices, wie z. B. E/A-Blöcke und Profinet IO-Antriebe, die über Profinet mit dem Profinet IO Controller angeschlossen sind.

Die XM- und VPx-Steuerungshardware unterstützt keinen Profinet IO-Ring (MRP / Redundanz Klasse 1)!

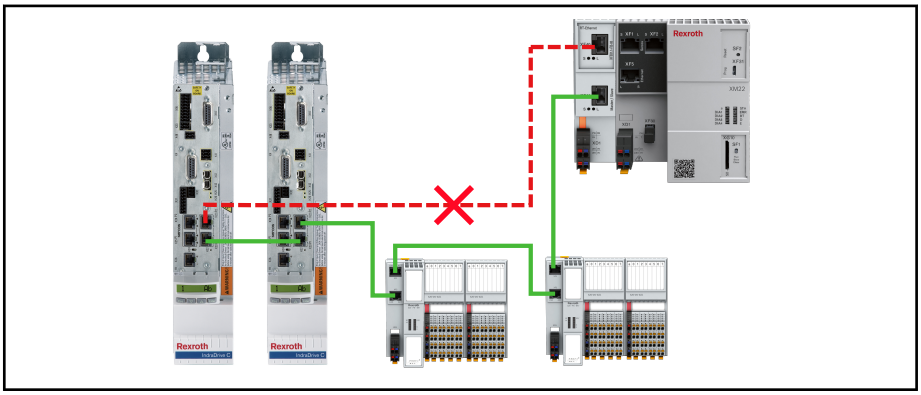


Abb. 6-5: XM2-Steuerung mit Erweiterungsmodul XFE01-1-FB-03 als Profinet IO Controller mit E/A-Blöcken und Antrieben

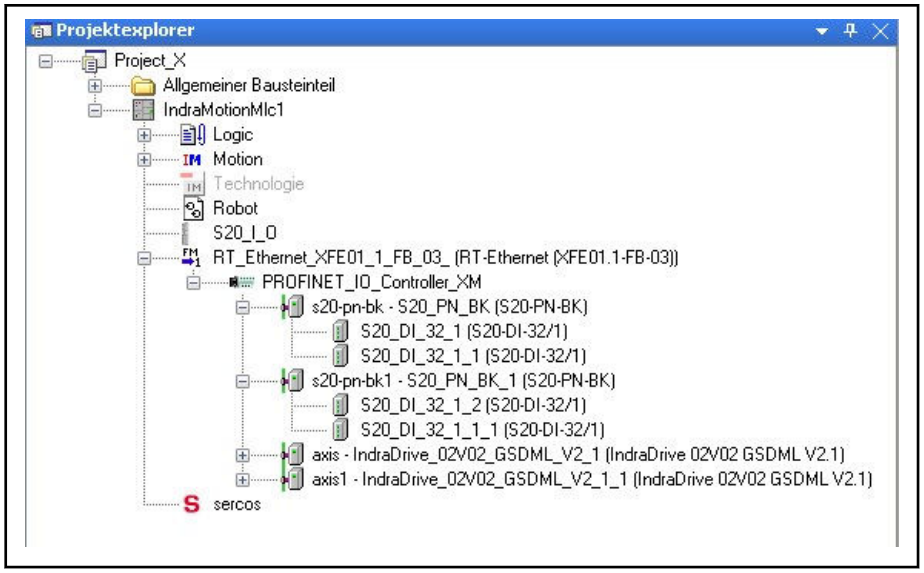


Abb. 6-6: Beispiel: Profinet IO-Konfiguration im Projektexplorer

Eine Beispiel für eine komplexere Topologie veranschaulicht folgende Grafik:

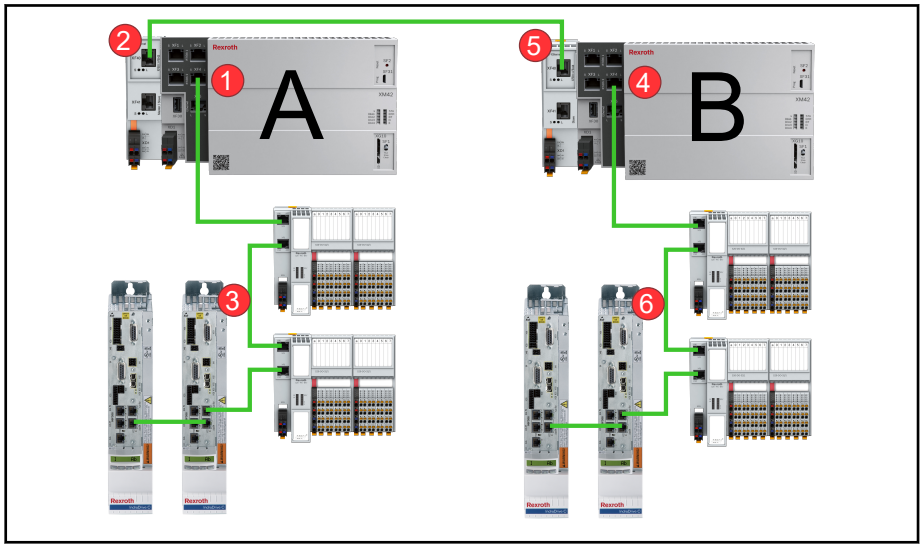


Abb. 6-7: Profinet IO – System aus zwei XM Steuerungen (Controller / Device)

Die Topologie besteht aus zwei XM Steuerungen, die über Profinet miteinander verbunden sind und die jeweils über ein eigenes, unterlagertes Profinet System mit EA-Baugruppen und Antrieben verfügen.

Die Hierarchie im Gesamtsystem (Controller / Device) baut sich folgendermaßen auf:

- Steuerung **"A"** ist die Primärsteuerung (Controller).

  Über die Onboard-Schnittstelle ① kommandiert die Primärsteuerung die unterlagerten Profinet Devices ③.

  Über das angeschlossene Erweiterungsmodul ② kommandiert die Primärsteuerung zusätzlich die Sekundärsteuerung **"B"** (Device).

  Die Schnittstellen sind wie folgt konfiguriert:

- Die Onboard-Schnittstelle XF3/XF4 ① ist als Controller konfiguriert und kommandiert die angeschlossenen Profinet IO Devices (E/A's & Antriebe).
- Das Erweiterungsmodul XFE01-1-FB-03 ② ist ebenfalls als Controller konfiguriert und kommandiert Steuerung **"B"** (Device).

- Steuerung **"B"** ist die Sekundärsteuerung (Device).

  Die Sekundärsteuerung wird von der Primärsteuerung **"A"** über das angeschlossene Erweiterungsmodul ⑤ kommandiert.

  Die Sekundärsteuerung wiederum kommandiert die unterlagerten Profinet Devices ⑥ über die Onboard-Schnittstelle ④.

  Die Schnittstellen sind wie folgt konfiguriert:

  - Das Erweiterungsmodul XFE01-1-FB-04 ⑤ arbeitet ausschließlich als Profinet IO Device und nimmt die Kommandos der Primärsteuerung **"A"** entgegen.
  - Die Onboard-Schnittstelle XF3/XF4 ④ ist als Controller konfiguriert und kommandiert die angeschlossenen Profinet IO Devices (E/A's & Antriebe).

6.5 Profinet IO, Controller

6.5.1 Profinet IO Controller, konfigurieren

Geräteeditor

Den Geräteeditor können Sie mittels Doppelklick auf das Profinet Objekt im Projektexplorer öffnen.

Über die Registerkarten können die Einstellungen des Profinet Objekts konfiguriert werden.

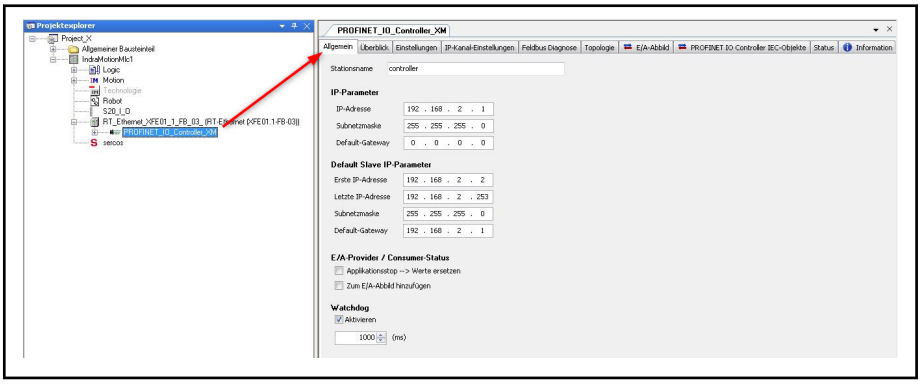


Abb. 6-8: Profinet IO Controller – generischer Geräteeditor

Registerkarte 'Allgemein'

Der PROFINET IO-Controller wird durch den Stationsnamen identifiziert (wie auch die Slaves).

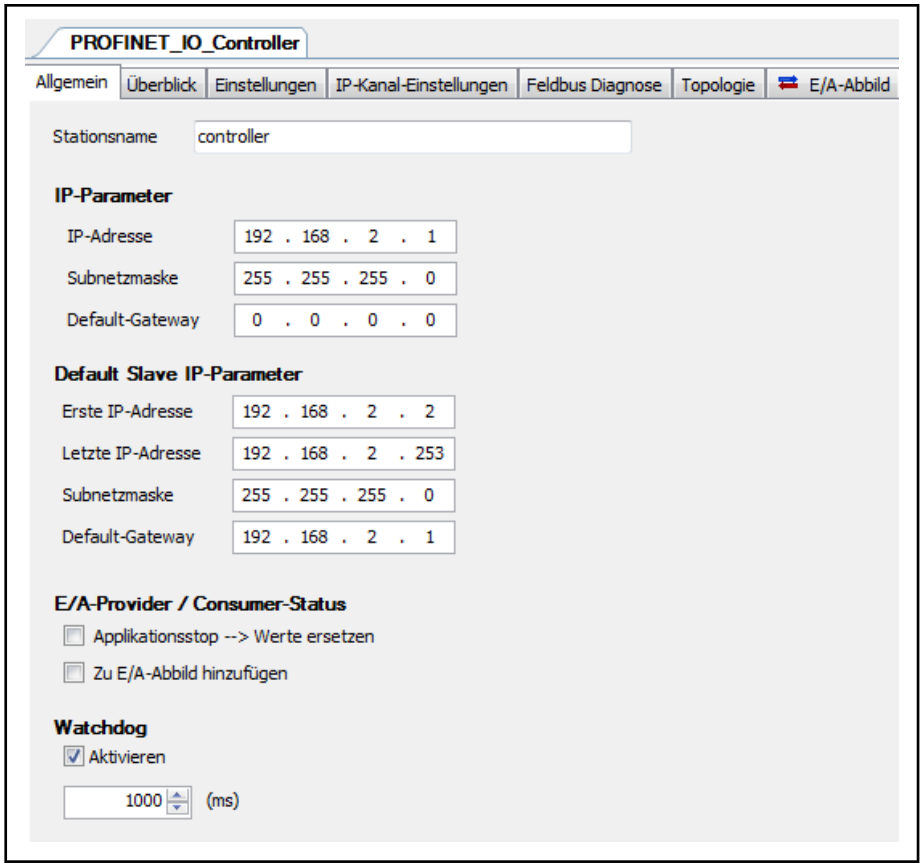


Abb. 6-9: Profinet IO Controller – Registerkarte "Allgemein"

Stationsname

- Stationsname: Voreinstellung "controller"

☞ *Der Stationsname darf aus folgenden Zeichen bestehen:*

- Kleinbuchstaben ohne Umlaute "a" bis "z".

  Großbuchstaben ohne Umlaute "A" bis "Z" sind zulässig, werden aber intern in Kleinbuchstaben gewandelt.

  Die Stationsnamen sind nicht case-sensitiv.
- Ziffern "0" bis "9" (dürfen nicht am Anfang des Namens stehen).
- Sonderzeichen Bindestrich "-" und Punkt "." (dürfen weder am Anfang noch am Ende des Namens stehen).
- Der Name darf max. 240 Zeichen lang sein. Bei manchen Geräten / Herstellern gilt die Beschränkung 127 Zeichen.

  Der Name kann mittels Punkten "." in Teilstrings gegliedert werden. Ein solcher Teilstring darf max. 63 Zeichen lang sein.

  Der Name darf nicht mit der Zeichenfolge "port-xyz" (mit x,y,z = 0...9) beginnen.

  Der Name darf nicht die Form "a.b.c.d" (mit a,b,c,d = 0...999) haben.

☞ Jedes Gerät muss im Subnetz einen eindeutigen Stationsnamen erhalten, da dieser für bestimmte Funktionalitäten im Netzwerkbetrieb benötigt wird!

IP-Parameter

Die folgenden Adressen müssen zur Identifizierung des Controllers der aktuellen Umgebung entsprechend konfiguriert werden:

- IP Adresse: Voreinstellung "192.168.2.1"



Beachten Sie bei der Anpassung der IP-Adresse auch die aktuellen Adresseinstellungen für Devices, die über einen Scan-Vorgang in die Konfiguration eingefügt werden, um Doppelbelegungen zu vermeiden (Default Slave IP-Parameter auf Seite 90)!

- Subnetzmaske: Voreinstellung "255.255.255.0"
- Default-Gateway: Standard-Gateway, Voreinstellung "0.0.0.0" (die Gateway-Adresse ist optional)

Beispiel: IP Adressen nach Scannen

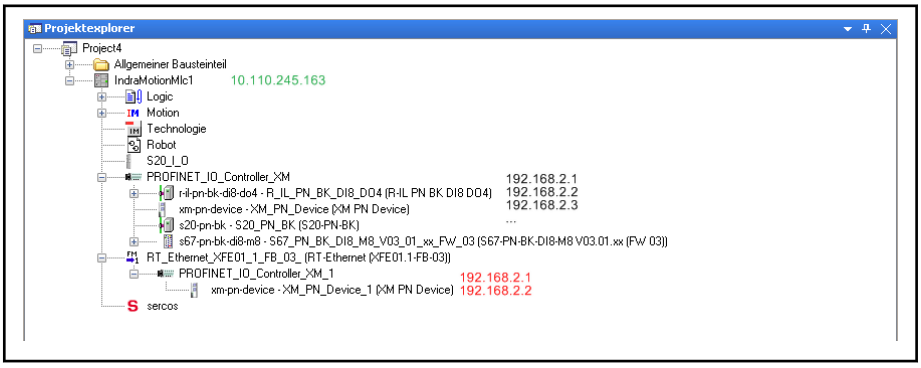


Abb. 6-10: IP-Adressen nach Scannen der Controller

Die Steuerung befindet sich in drei von einander unabhängigen Netzen:

- grün (oben): Betriebsnetz (Engineering Schnittstelle)
- schwarz (Mitte): PNIO Controller auf der Onboard-Schnittstelle
- rot (unten): PNIO Controller über das Erweiterungsmodul XFE01-1-FB-03



Obwohl die Netze der beiden Controller von einander unabhängig sind, sollte auf dem zweiten Controller ein anders Subnetz z.B. 192.168.**3**.1.. benutzt werden, um eventuelle zukünftige Probleme zu vermeiden.

Default Slave IP-Parameter

Die Adresseinstellungen für Devices werden nur verwendet, wenn Devices, die beim Scannen der Hardware (**Projekt ▸ Nach Geräten suchen..**) gefunden wurden, im Projektexplorer abgebildet werden und dabei nicht bereits im Scan-Dialog Adressdefinitionen vorgenommen wurden.

Die in diesem Dialog definierten Werte werden automatisch auf die eingefügten Geräte übertragen. Bei der IP-Adresse wird jeweils die nächste freie Adresse aus dem definierten Bereich eingestellt.

Die Voreinstellungen stammen aus der Gerätebeschreibungsdatei. Sie können editiert werden, wenn der Cursor ins jeweilige Eingabefeld gesetzt wird:



Beachten Sie bei der Anpassung dieser Adressen die aktuelle IP-Adresse des Controllers, um Doppelbelegungen zu vermeiden!

- Erste IP Adresse: Voreinstellung "192.168.2.2"
- Letzte IP Adresse: Voreinstellung "192.168.2.254"
- Subnetzmaske: Voreinstellung "255.255.255.0"
- Default Gateway: Standard Gateway; Voreinstellung "0.0.0.0"

E/A-Provider / Consumer-Status

- **Applikationsstop → Werte ersetzen**

  ☑: Wenn der Anwender die Applikation stoppt, wird der Provider Status auf "BAD" gesetzt.

  Die Slaves setzten daraufhin die Ein-/Ausgänge auf vordefinierte Ersatzwerte.

- **Zu E/A-Abbild hinzufügen**

  ☑: Die eingehenden Statusinformationen werden für alle Module zum E/A-Abbild hinzugefügt (Provider State zu den Eingangsdaten und Consumer State zu den Ausgangsdaten).

Watchdog

- Ist der Watchdog "Aktiviert", gilt die eingetragene Überwachungszeit. Das heißt, wenn diese Zeit abläuft, ohne dass der Controller aktiv war, erfolgt eine gerätespezifische Reaktion, z. B. eine Fehlermeldung.
- Zeit (ms): Überwachungszeit, Voreinstellung 1000ms. Mögliche Werte: 0 bis 65535.

Die Voreinstellungen für die Überwachungssteuerung stammen aus der Gerätebeschreibung.

Registerkarte 'Überblick'

Die Registerkarte zeigt eine Übersicht aller Devices mit ihren Kommunikationseinstellungen.

Der Stationsname, die IP-Adresse und die Subnetzmaske können direkt in der Liste geändert werden.

Bei den anderen Parametern öffnet sich der Dialog "Kommunikationseinstellungen", der auch als Kontextmenü der selektierten Zeile aufgerufen werden kann.



Es können mehrere Devices gleichzeitig selektiert werden und die Eintellungen in den ausgewählten Devices gleichzeitig gesetzt werden.

- Befehl **Default-IP**:

  Setzt die IP-Adresse der ausgewählten Devices auf einen Wert innerhalb des Standardadressenbereichs.

  Der Adressbereich ist in den Einstellungen des Controllers definiert.

- Befehl **EA-Leistung**:

  Öffnet den Dialog "Kommunikationseinstellungen".

  – Dialog "**Kommunikationseinstellungen**":

    In den Kommunikationseinstellungen kann für alle ausgewählten Devices die Übertragungsgeschwindigkeit für EA-Daten konfiguriert werden.

    Für die einzelnen Parameter (RT-Class, Send Clock, Reduction Ratio) steht ein Wertebereich zur Auswahl, der für alle ausgewählten Geräte gültig ist.

    Siehe Kap. "Registerkarte 'Allgemein'" auf Seite 103.

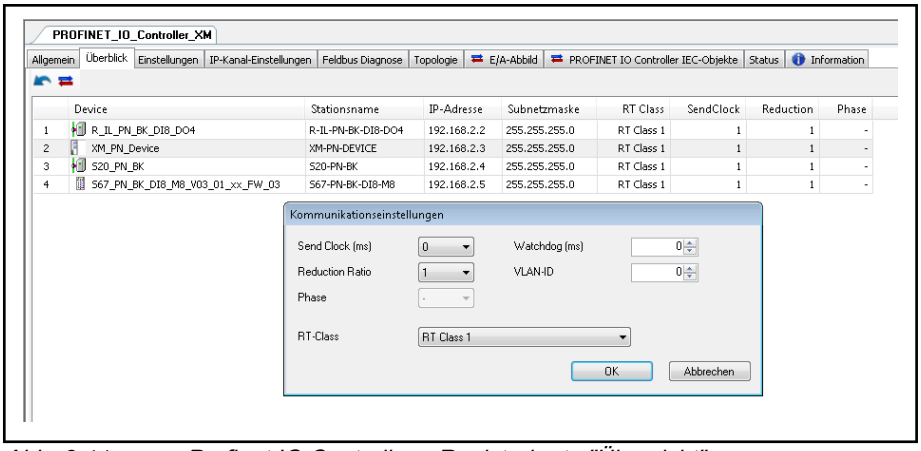


Abb. 6-11: Profinet IO Controller – Registerkarte "Übersicht"

Registerkarte 'Einstellungen'

Die Registerkarte stellt zusätzliche Einstellungen für den PNIO Controller zur Verfügung.

Deaktivierte E/A-Module

E/A-Module von PNIO Devices können deaktiviert werden:

- Im Projektexplorer, siehe Kap. "Profinet IO-Module aktivieren / deaktivieren" auf Seite 110
- Alternativ über eine **UserDefs.cfg** Datei auf der Steuerung, siehe Kap. "Profinet IO-Devices und Profinet IO-Module deaktivieren mittels UserDefs.cfg" auf Seite 99

Für die Handhabung der Slot-Konfiguration der Devices können folgende Optionen ausgewählt werden (je PNIO Controller Instanz):

- **Slot-Komprimierung** (Default):

  Die Slot-Konfiguration der aktivierten Module wird "komprimiert", so dass eine fortlaufende Slot-Nummerierung ohne Lücken der aktivierten Module entsteht.

  ⇒ *Anwendungsfall:* es wird eine "Maximalprojekt"-Konfiguration erstellt und für eine spezifische Anlage ist nur eine Teilmenge der konfigurierten E/A-Module vorhanden, d.h. die physikalisch nicht vorhandenen Module werden mit den oben genannten Methoden in der Konfiguration deaktiviert.

  Für PNIO Devices, bei denen die Slot-Nummern der Steckplatznummer am Lokalbus entspricht (z.B. IL PN BK DI8 DO4; S20-PN-BK/+; Siemens ET200S) und somit bei den physikalisch vorhandenen Modulen eine fortlaufende Nummerierung ohne Lücken entsteht, muss dann entsprechend auch in der Konfiguration bei den aktivierten Modulen eine Slot-Konfiguration ohne Lücken erstellt werden (= "Slot-Komprimierung").

- **Keine Slot-Komprimierung bei Shared Devices**:

  Bei PNIO Devices, die als "Shared Device" betrieben werden können, findet keine Slot-Komprimierung statt (= wenn im Slave Dialog "Optionen" das Häkchen "Shared Device" gesetzt werden kann, unabhängig davon, ob die Option gesetzt ist oder nicht).

  ⇒ *Anwendungsfall:* bei einer Shared Device Anwendung bedient ein PNIO Controller typischerweise nur eine Teilmenge der physikalisch vorhandenen Module eines PNIO Devices. Die Module, die vom PNIO Controller nicht bedient werden, müssen zwar konfiguriert sein, müssen

aber dann deaktiviert werden ohne dass eine Slot-Komprimierung stattfindet.

- **Keine Slot-Komprimierung**:

  Es findet bei allen PNIO Devices dieses PNIO Controllers keine Slot-Komprimierung statt.

  ⇒ *Anwendungsfall:* diese Option kann für Shared Device Anwendungen (alternativ zur Einstellung oberhalb) angewendet werden, oder bei PNIO Devices mit festen oder konfigurierbaren Slot-Nummern, so dass keine fortlaufende Slot-Nummerierung ohne Lücken besteht. (Beachte: diese Einstellung gilt dann für alle PNIO Devices des PNIO Controllers).

Hinweis: Für Diagnosezwecke kann die "aktive" Slotnummer in einer Konfiguration im generischen Parameterdialog eines PNIO E/A-Moduls angeschaut werden, siehe im eingeloggten Zustand Dialog *"PNIO Module Parameter"* → *"Module parameters"* → "*Slot number*". (Der Dialog muss freigeschaltet werden über IndraWorks → Extras → Optionen → IndraLogic 2G → Geräteeditor → Generische Konfigurationseditoren anzeigen).

Registerkarte 'IP-Kanal-Einstellungen'

Siehe "Registerkarte 'IP-Kanal-Einstellungen'" auf Seite 33

Registerkarte 'Profinet IO Controller Parameter'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor**.

Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

Registerkarte 'Feldbus Diagnose'

Für die Feldbusse der Systeme MLC und MTX wurde ausgehend vom Objektknoten des Masters, hier Profinet IO Controller, ein einheitliches Diagnosekonzept entwickelt.

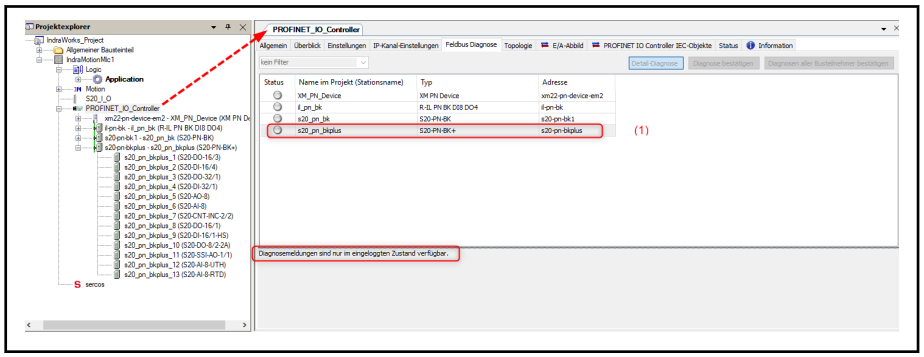


Abb. 6-12: Feldbus-Diagnose, Steuerung Online, nicht eingeloggt

Der Buskoppler (1) ist am Bussystem nicht vorhanden. Er würde einen Fehler erzeugen. Aus diesem Grund ist der Buskoppler durch **Eigenschaften ▸ Übersetzen ▸ Vom Übersetzen ausschließen** deaktiviert und von der Diagnose ausgenommen

Nach dem Einloggen durchläuft der am Knoten befindliche Bus einen Diagnosezyklus in dessen Ergebnis sich die Status-Leuchtdioden verfärben, hier grün, d.h. das Modul meldet fehlerfreien Lauf.

Nach Auswahl eines Moduls (Anklicken der Zeile) ist für dieses eine Detaildiagnose verfügbar.

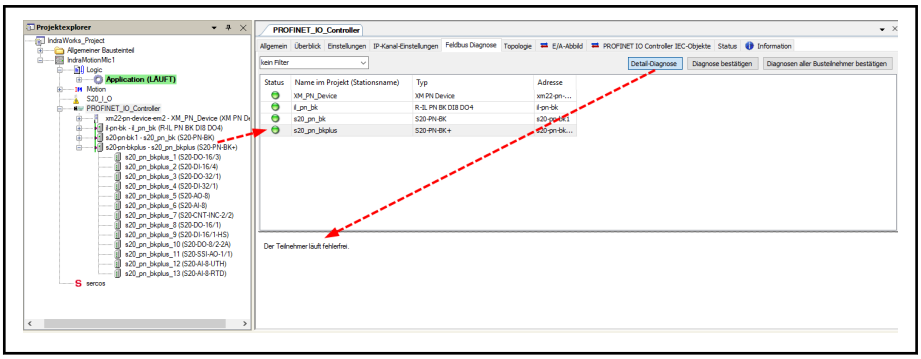


Abb. 6-13: Feldbus-Diagnose, Steuerung Online, eingeloggt, fehlerfrei

Das Einloggen wird mit dem zusätzlichen, nicht an der realen Steuerung vorhandenen Koppler wiederholt, diese Mal ohne Ausschluss vom Übersetzen.

Am betroffenen Modul (und am Controller) erscheint ein gelbes Warndreieck, der Hinweis auf eine vorhandene Diagnose. Die Status-Leuchtdiode des betroffenen Moduls färbt sich rot. Über <Detail-Diagnose> können eine "Standarddiagnose" mit Kurzinformation und eine "Detail-Diagnose" mit der detaillierten Angabe der Fehlerursache erhalten werden.

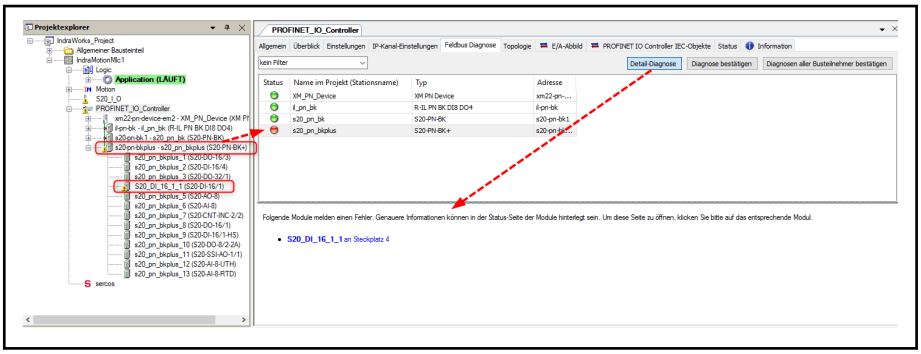


Abb. 6-14: Feldbus-Diagnose, Steuerung Online, eingeloggt, Fehler

Bei kurzzeitigen Fehlern bleibt die Anzeige nur so lange bestehen, wie der Fehler vorhanden ist.

Diagnose betätigen:

Für das gewählte Module werden, trotz vorhandenen Fehlers, sowohl das Warndreieck als auch die rote Status\-Leuchtdiode abgeschaltet.

Die "Erweiterte Diagnose" der "Detaildiagnose" bleibt im Text erhalten (Die nächste Diagnose kann bearbeitet werden....).

Diagnose aller Busteilnehmer betätigen:

Für alle Module werden, trotz vorhandener Fehler, sowohl das Warndreieck als auch die rote Status\-Leuchtdiode abgeschaltet.

Registerkarte 'Topologie'

Diese Registerkarte bietet eine Übersicht über die Topologie des PROFINET IO-Netzwerks und über die Nachbarschaft/Peer-Information für jedes einzelne PROFINET IO-Gerät.

☞ Nur verfügbar für Geräte mit LLDP-Unterstützung (z.B. Conformance Class B) und alle neueren Geräte nach Norm PROFINET V2.3.

Zum Auslesen der Topologieinformation ist das vorherige Setzen einer gültigen IP-Adresse in den PNIO Devices notwendig. Sehen Sie dazu auch den Befehl Auto IP.

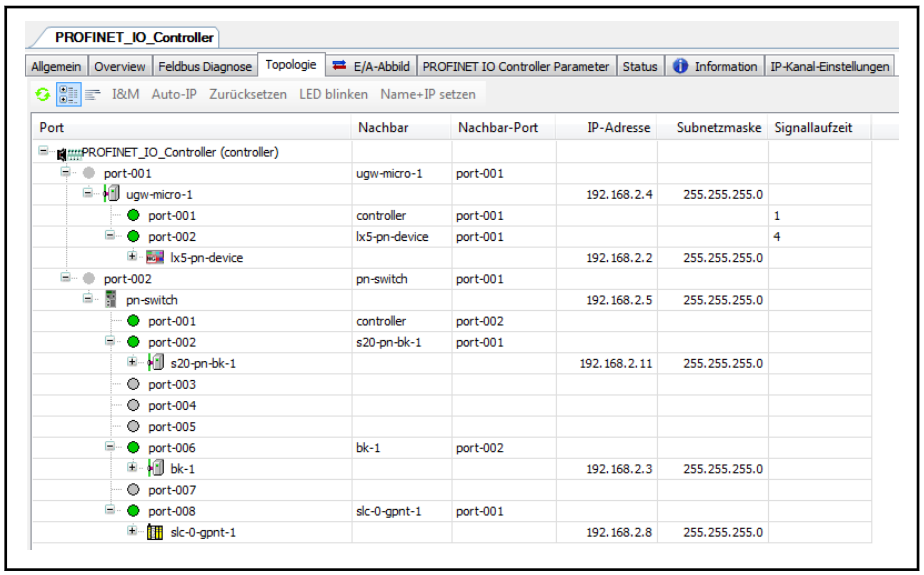


Abb. 6-15: PROFINET IO-Controller - Registerkarte 'Topologie'

Port	Nummer des lokalen Ports, beginnend mit 1. Mögliche Status sind: : unbekannt oder nicht verbunden : OK : Fehler
Nachbar	Gerätename und Port des Nachbargeräts.
Nachbar-Port	
IP-Adresse	Eingestellte IP-Adresse des Slaves
Subnetzmaske	Eingestellte Subnetzmaske des Slaves
Signallaufzeit	Signallaufzeit in ns (Richtwert 6 ns für 1 Meter Kabellänge)

Befehle:

Aktualisieren	Aktualisieren der angezeigten Daten (Bus Scan)
Baum-/Listenansicht	Umschaltung zwischen Baum- und Listenstruktur Darstellung
I&M	Abrufen der I&M-Daten (Identification and Maintenance) des Geräts und Anzeige im Dialog
Auto IP	Automatisches Setzen einer gültigen IP-Adresse innerhalb des Netzwerks (Netzwerkeinstellungen des Masters), falls die vom Gerät gescannte Adresse ungültig ist
Reset	Zurücksetzen der Geräteeinstellungen des selektierten Geräts auf Werkseinstellungen

LED blinken	Um ein aufgelistetes Gerät an der Hardware zu identifizieren, selektieren Sie das Gerät und betätigen die Schaltfläche "Blinken LED". Das Gerät sollte darauf mit einem Blinksignal reagieren.
Name+IP setzen	Es wird versucht, den Stationsnamen und die IP-Adresse, die für den ausgewählten Slave angegeben wurden, zu setzen.

Registerkarte 'E/A-Abbild'

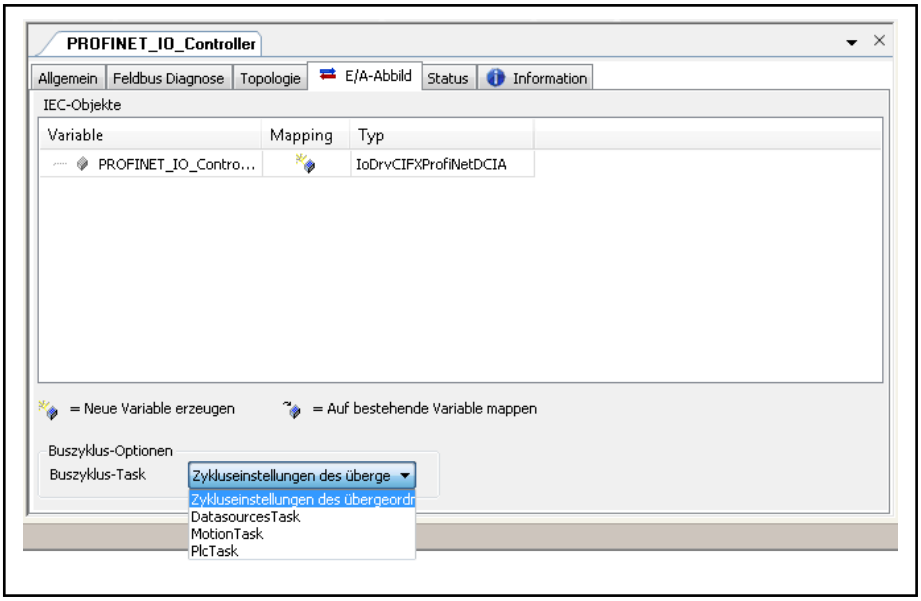


Abb. 6-16: Profinet IO Controller - Registerkarte "E/A-Abbild"

IEC-Objekte: Mit der Definition des Profinet IO Controllers werden die Bibliotheken "IoDrvCIFXProfinetDCIA.library" bzw. RIL_ProfinetIO.library automatisch angezogen. Der zur Realisierung des Profinet IO Controllers erforderliche Speicherraum kann somit definiert werden.

Buszyklus-Optionen

Buszyklus-Task: Durch die Auswahl einer Buszyklus-Task kann der Zyklus des Abbildtausches für den Profinet IO Controller an eine spezielle Task gebunden werden. Es ist sinnvoll in dieser Task auch die IO-Daten des Masters zu verarbeiten.

Standardeinstellung: Zykluseinstellung des übergeordneten Busses verwenden

Diese Einstellung bewirkt, dass die in "Buszyklus-Optionen" der Steuerung (Doppelklick auf die konkrete Steuerung im Projektnavigator, SPS-Einstellungen) vorgenommene Taskeinstellung für den konkreten Bus übernommen wird.

Registerkarte 'Status'

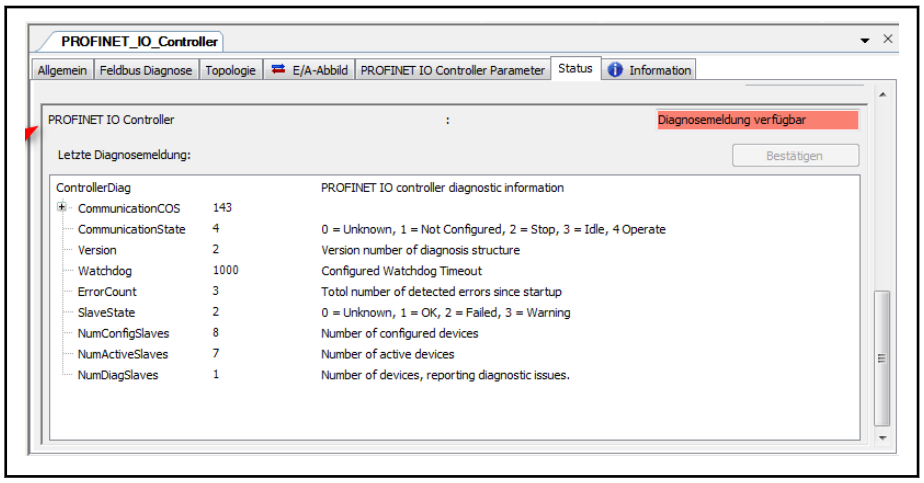


Abb. 6-17: Profinet IO Controller – Registerkarte "Status"

Die Registerkarte "Status" im "Profinet IO Geräteeditor" zeigt Statusinformationen (z.B. "Running", "Stopped") und spezifische Diagnosemeldungen des Profinet-Geräts.

Zusätzlich werden Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt.

Die Ausgabe erfolgt als hexadezimaler Wert, der sich aus den gesetzten Status-Flags ergibt (siehe Dokument "Protocol Interface Manual" für Profinet IO).

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

6.5.2 Profinet IO Controller, Devices hinzufügen

Allgemeines

Die verfügbaren Profinet IO Devices befinden sich in der Bibliothek "Peripherie" im Ordner "Profinet IO".

Ziehen Sie aus der Bibliothek die benötigten Devices auf das Profinet-Objekt.

Devices lassen sich im Projektexplorer so auch zwischen bereits vorhandenen Devices einfügen.

Alternativ können Sie Devices im Kontextmenü über **Hinzufügen** ▸ **...** des Profinet-Objekts hinzufügen.

Hierbei wird das neue Device als letztes Modul unter das Profinet-Objekt eingefügt.

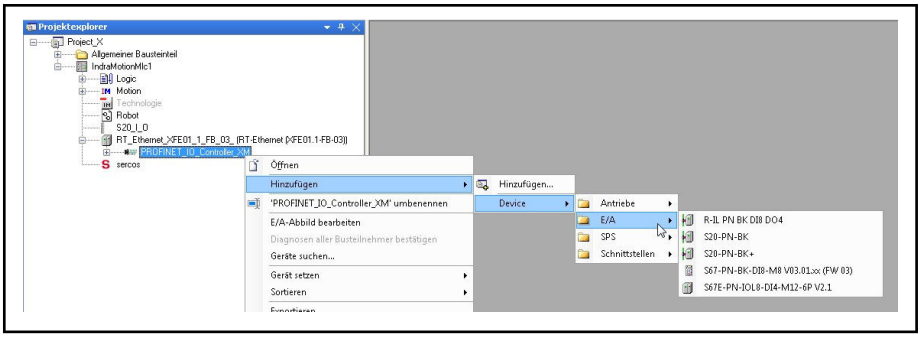


Abb. 6-18: Profinet IO Controller, Devices hinzufügen

Ist ein benötigtes Profinet IO Device nicht standardmäßig in der Bibliothek vorhanden, kann es mittels Import seiner GSDML-Datei über das Hauptmenü **Extras** ▸ **Gerätedatenbank** ▸ **Geräte hinzufügen** in die Bibliothek hinzugefügt werden. Siehe auch Gerätedatenbank auf Seite 15.

Profinet IO-Devices aktivieren / deaktivieren

Die an einer Steuerung konfigurierten Profinet IO-Devices können im Projektexplorer aktiviert bzw. deaktiviert werden.

Dies erfolgt durch Anwählen / Abwählen der Schaltfläche, die am Icon des Profinet IO-Devices erscheint.

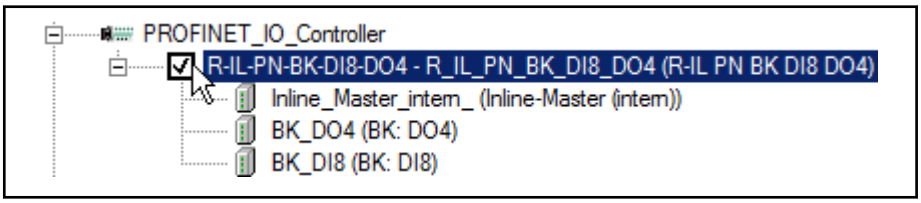


Abb. 6-19: Aktivieren / Deaktivieren von Profinet IO-Devices

Ist der Haken gesetzt, ist der Busteilnehmer aktiviert. Er wird in der Diagnose des Busses berücksichtigt.

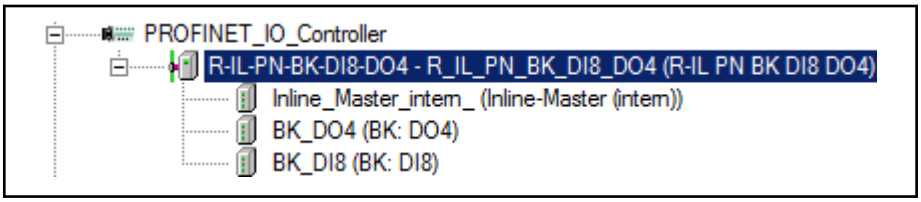


Abb. 6-20: Profinet IO-Device als aktivierter Teilnehmer

Ist der Haken nicht gesetzt, ist der Busteilnehmer deaktiviert. Er wird in der Diagnose des Busses nicht berücksichtigt.

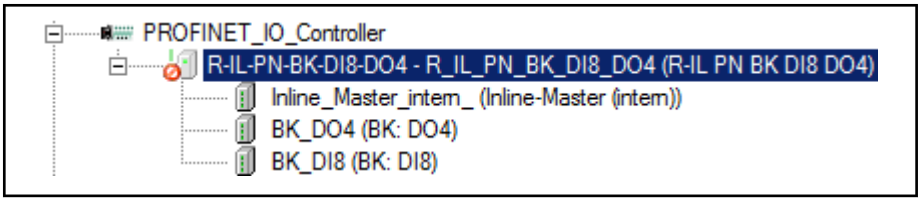


Abb. 6-21: Profinet IO-Device als deaktivierter Teilnehmer

Wird ein neues Profinet IO-Device angelegt, ist es per Voreinstellung aktiviert.

Beim Laden der Konfiguration werden alle aktivierten Profinet IO-Devices berücksichtigt. Deaktivierte Profinet IO-Devices werden ignoriert.

Ein aktivierter jedoch nicht vorhandener Busteilnehmer erzeugt in der Diagnose eine Fehlermeldung.

Ein deaktivierter nicht vorhandener Busteilnehmer erzeugt in der Diagnose keine Fehlermeldung.

Siehe auch Profinet IO-Devices und Profinet IO-Module deaktivieren mittels UserDefs.cfg auf Seite 99

Profinet IO-Devices und Profinet IO-Module deaktivieren mittels UserDefs.cfg

Die Funktion ermöglicht es Profinet IO-Devices / Profinet IO-Module, die in IndraWorks konfiguriert sind, durch Definition in einer auf der FlashCard der Steuerung abliegenden Datei zu deaktivieren.

Dadurch werden Diagnose-Meldungen für Profinet IO-Devices / Profinet IO-Module, die zwar konfiguriert, aber nicht vorhanden sind, verhindert.

Aktivierung der Funktion

Die Funktion wird durch das Vorhandensein einer gültigen Datei "UserDefs.cfg" auf der Steuerungs-FlashCard in folgenden Partitionen aktiviert:

Bei XM- und VPx-Steuerungen: **\USER**

Optional können durch Einträge in der "cfg-Datei" eines Systems ein anderer Pfad und Dateiname definiert werden.

Beispiel:

```
[IOSpecialDefs]
IODeactFilePath=MyFilePath
IODeactFileName=MyFileName
```

Die Datei "UserDefs.cfg" kann Einträge für weitere Busse enthalten:

Inline-Module, Sercos, Profinet IO, Profibus DP.

Zeitpunkt des Deaktivierens

Die Auswertung der Datei "UserDefs.cfg" und das Deaktivieren der dort eingetragenen Profinet IO-Devices / Profinet IO-Module erfolgt zum Zeitpunkt und im Kontext des internen Events "PrepareUpdateConfiguration".

Das Event tritt bei folgenden Aktionen ein:

- Anlauf nach Power-ON
- Reset (warm)
- Reset (kalt)
- Programm laden nach Reset (Ursprung)

Struktur der Datei "UserDefs.cfg"

Die Datei ist strukturiert in ein- oder mehrfach vorhandene Sektion(en) und den darin enthaltenen Einträgen.

- **Sektionen**: Die Bezeichnung einer Sektion ist in '**[**' und '**]**' geklammert. Eine Sektion ist begrenzt durch den Beginn der nächsten Sektion oder durch das Ende der Datei.

  Jeder Sektionsbezeichner beinhaltet noch eine Instanznummer `[x, 0..4]` des jeweiligen Objektes. Sie kennzeichnet in aufsteigender Reihenfolge die möglichen Profinet-Feldbusinstanzen des gleichen Typs.

- **Sonderfunktion**: Mit der Instanznummer **x** wird die Sektion bei der Auswertung ignoriert.

  Es ist möglich, für **eine** Instanznummer **mehrere** Sektionen zu definieren. Die darin enthaltenen Einträge werden alle dem angegebenen Instanznummer zugeordnet.

In Kombination mit der oben beschriebenen "X-Funktion" lassen sich damit unterschiedliche Gruppen von Devices in einfacher Weise behandeln.

- **Kommentare**: Als Kommentarzeichen ist das Semikolon (';') definiert. Ab der Position des Semikolons bis zum Ende der Zeile wird der Text nicht ausgewertet.

Profinet

Bei Profinet werden die Devices (=Slaves) über textuelle Device-Namen adressiert, der in der Konfiguration auch als "Stationsname" bezeichnet wird.

☞ *Der Stationsname darf aus folgenden Zeichen bestehen:*

- Kleinbuchstaben ohne Umlaute "a" bis "z".

  Großbuchstaben ohne Umlaute "A" bis "Z" sind zulässig, werden aber intern in Kleinbuchstaben gewandelt.

  Die Stationsnamen sind nicht case-sensitiv.
- Ziffern "0" bis "9" (dürfen nicht am Anfang des Namens stehen).
- Sonderzeichen Bindestrich "-" und Punkt "." (dürfen weder am Anfang noch am Ende des Namens stehen).
- Der Name darf max. 240 Zeichen lang sein. Bei manchen Geräten / Herstellern gilt die Beschränkung 127 Zeichen.

  Der Name kann mittels Punkten "." in Teilstrings gegliedert werden. Ein solcher Teilstring darf max. 63 Zeichen lang sein.

  Der Name darf nicht mit der Zeichenfolge "port-xyz" (mit x,y,z = 0...9) beginnen.

  Der Name darf nicht die Form "a.b.c.d" (mit a,b,c,d = 0...999) haben.

Profinet IO-Controller

- Die Bezeichnung `PROFINETIO.Controller.` und eine Instanznummer beschreibt eine Steuerung als IO-Controller.

  Beispiel:

  ```
  [PROFINETIO.Controller.2]
  ```
- **Komplettes Profinet-Device**: Ein Eintrag zum Deaktivieren eines kompletten Profinet-Devices unterhalb eines Controllers besteht aus dem Schlüsselwort `DeviceDeactName=` und der Angabe des Device-Namens als Text.

  Beispiel:

  ```
  DeviceDeactName = myDeviceName1
  ```
- **Modul-Eintrag**: Zum Deaktivieren von wahlfreien Modulen unterhalb von einem Device werden

  das Schlüsselwort `DeviceName=` und der DeviceName, sowie

  das Schlüsselwort `DeactModule=` und durch Komma getrennt, die zu deaktivierenden Modulnummern von **1** bis **n**.

  Modulnummer **0** ist die Busanschaltung des Devices selbst und darf hier nicht angegeben werden.

  Beispiel:

  ```
  DeviceName = myDeviceName3, DeactModule=3,4,7
  ```
- **Letzte Modul-Einträge**: Zum Deaktivieren der **n** letzten konfigurierten Module unterhalb eines Device werden

das Schlüsselwort `DeviceName=` und der DeviceName, sowie

das Schlüsselwort `DeactLastModules=` und die zu deaktivierenden Anzahl von Modulen angegeben.

Damit werden letzten (=untersten) Module im Konfigurationsbaum deaktiviert.

Beispiel:

```
DeviceName = myDeviceName2, DeactLastModules=2
```

Profinet IO-Device

- Die Bezeichnung `PROFINETIO.Device.` und eine Instanznummer beschreibt eine Steuerung, die als IO-Device betrieben wird.

  Beispiel:

  ```
  [PROFINETIO.Device.0]
  ```

- **Device-Namen überschreiben**: Mit `StationNameSet=` und einem Device-Namen wird der projektierte Station-Name des Devices überschrieben.

  Beispiel:

  ```
  StationNameSet= overloadDeviceName
  ```

- **Modul-Eintrag**: Zum Deaktivieren von wahlfreien Modulen unterhalb von einem Device werden

  das Schlüsselwort `DeactModule=` und durch Komma getrennt, die zu deaktivierenden Modulnummern von **1** bis **n**.

  Modulnummer **0** ist die Busanschaltung des Devices selbst und darf hier nicht angegeben werden.

  Beispiel:

  ```
  DeactModule=3,4,7
  ```

- **Letzte Modul-Einträge**: Zum Deaktivieren der **n** letzten konfigurierten Module unterhalb eines Device werden

  das Schlüsselwort `DeactLastModules=` und die zu deaktivierenden Anzahl von Modulen angegeben.

  Damit werden letzten (=untersten) Module im Konfigurationsbaum deaktiviert.

  Beispiel:

  ```
  DeactLastModules=2
  ```

Größe der Datei "UserDefs.cfg"

Entsprechend der gegebenen Größe der Datei wird zu deren Auswertung Speicher allokiert. Die zulässige Größe ist daher praktisch nicht begrenzt.

Beispiel der Datei "UserDefs.cfg"

Profinet IO Controller.0

```
;************************************************
;---------- PROFINET IO Controller.0 ----------
;************************************************
[PROFINETIO.Controller.0]

DeviceDeactName = myDeviceName1
DeviceName = myDeviceName2, DeactLastModules=2
DeviceDeactName = myDeviceName3
```

Profinet IO Controller.2

```
;************************************************
;---------- PROFINET IO Controller.2 ----------
;************************************************
[PROFINETIO.Controller.2]
```

```
DeviceDeactName = myDeviceName12
DeviceName = myDeviceName14, DeactModule=2,3
```

Profinet IO Device.0

```
;************************************************
;---------- PROFINET IO Device.0 ----------
;************************************************
[PROFINETIO.Device.0]
StationNameSet= newDeviceName
```

Profinet IO Device.1

```
;************************************************
;---------- PROFINET IO Device.1 ----------
;************************************************
[PROFINETIO.Device.1]
DeactLastModules= 3
```

Devices zum Anschluss von I/O-Modulen

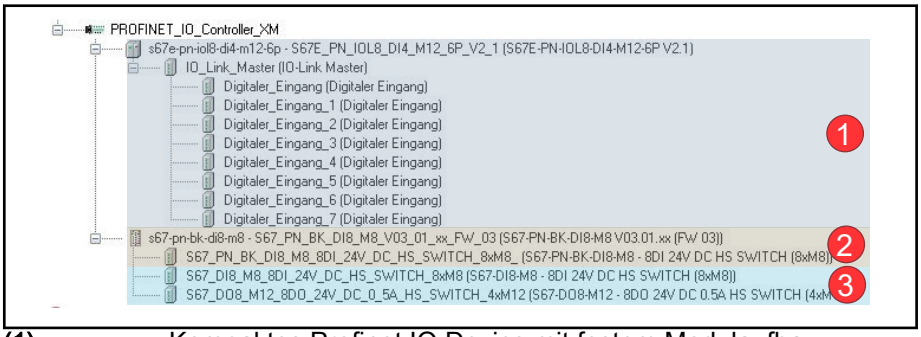
Profinet IO unterscheidet zwei Arten von I/O-Devices:

1. **Kompakt**: Bei einem kompakten Device ist der Modulaufbau fest vorgegeben.

   Nach dem Einfügen eines kompakten Profinet IO Devices in den Projektexplorer sind die Module unterhalb des Buskopplers bereits vollständig vorhanden. In der Gerätebibliothek sind die Module nicht sichtbar.

2. **Modular**: Der Modulaufbau ist variabel.

   Ausgehend von einem Buskoppler (mit ggf. vorhandenen eigenen I/Os) können Module individuell – aber nach Bestückungsvorschrift des Gerätes – angeordnet werden.



(1) Kompaktes Profinet IO Device mit festem Modulaufbau
(2) Modulares Profinet IO Device mit zur Busklemme gehörenden Ein-Ausgängen
(3) Klemme, die individuell zugeordnet wurden

Abb. 6-22: I/O-Module am Profinet IO Device

Profinet IO Device - Geräteeditor

Zur Konfiguration und Parametrierung des Profinet IO Device-Objekts öffnen Sie den Geräteeditor durch Doppelklick auf das entsprechende Profinet Device im Projektexplorer. Alternativ können Sie den Geräteeditor über das Kontextmenü des Profinet IO Device öffnen.

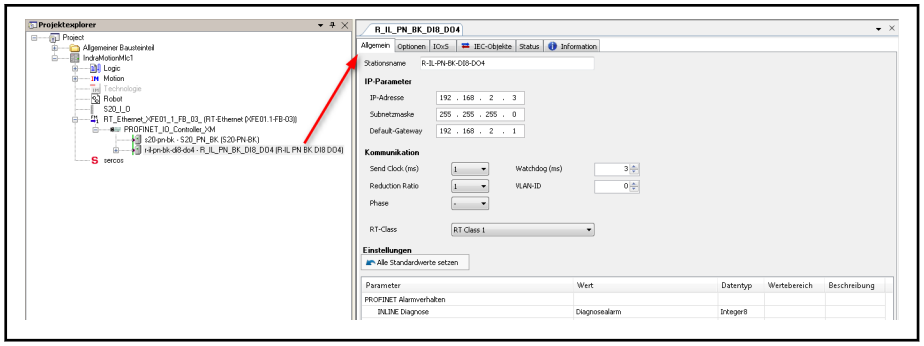


Abb. 6-23: Profinet IO Device - Registerkarte 'Allgemein'

Konfigurationsdialoge für Profinet IO Devices

- Registerkarte 'Allgemein' auf Seite 103
- Registerkarte 'IOxS' auf Seite 107,
- Registerkarte 'PNIO Parameter' auf Seite 108

  dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras ▶ Optionen ▶ IndraLogic 2G ▶ Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'Status' auf Seite 109,
- Registerkarte 'Information' auf Seite 109,

Registerkarte 'Allgemein'

Für alle Einstellungen in dieser Registerkarte gilt, dass es von der Gerätebeschreibung abhängt, ob die Einstellungen editierbar sind und welche Werte vorgegeben und möglich sind.

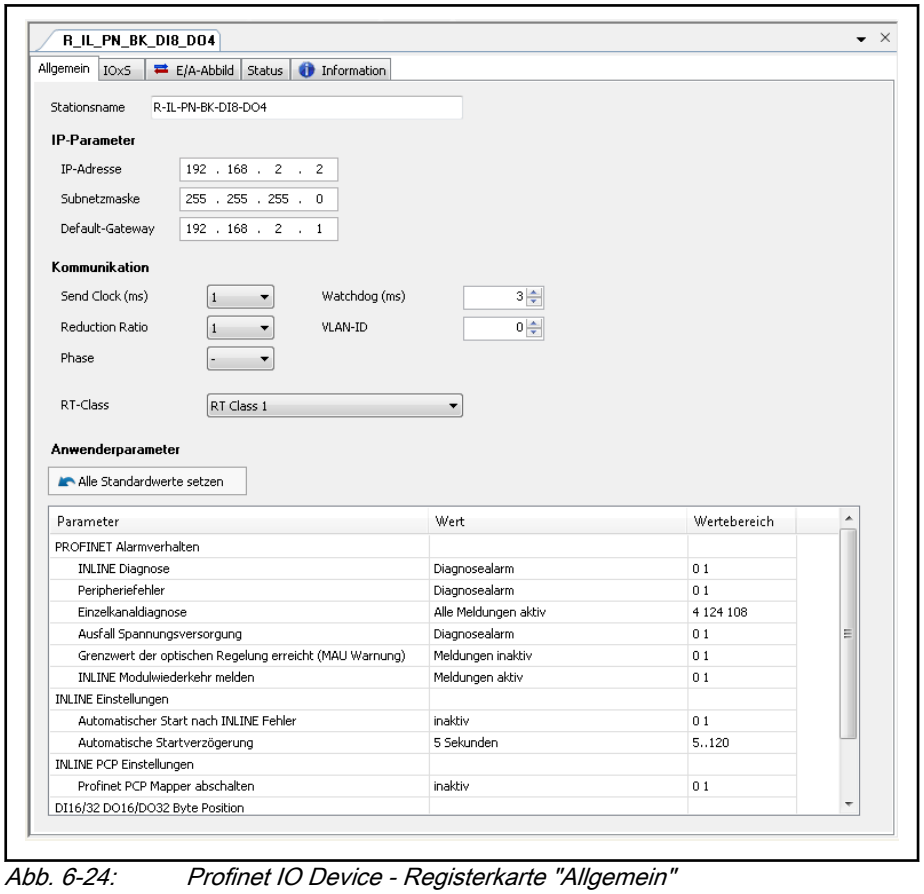


Abb. 6-24: Profinet IO Device - Registerkarte "Allgemein"

Stationsname

Der Stationsname wird innerhalb von Profinet zur Adressierung der IO Devices verwendet und muss innerhalb des Profinet-Subnetzes eindeutig sein.

Das Feld "Stationsname" muss also in den meisten Fällen angepasst werden.

Beim Anlegen des IO Devices wird hier der Defaultname aus der Gerätebeschreibungsdatei vorgegeben.

☞ *Der Stationsname darf aus folgenden Zeichen bestehen:*

- Kleinbuchstaben ohne Umlaute "a" bis "z".

  Großbuchstaben ohne Umlaute "A" bis "Z" sind zulässig, werden aber intern in Kleinbuchstaben gewandelt.

  Die Stationsnamen sind nicht case-sensitiv.
- Ziffern "0" bis "9" (dürfen nicht am Anfang des Namens stehen).
- Sonderzeichen Bindestrich "-" und Punkt "." (dürfen weder am Anfang noch am Ende des Namens stehen).
- Der Name darf max. 240 Zeichen lang sein. Bei manchen Geräten / Herstellern gilt die Beschränkung 127 Zeichen.

  Der Name kann mittels Punkten "." in Teilstrings gegliedert werden. Ein solcher Teilstring darf max. 63 Zeichen lang sein.

  Der Name darf nicht mit der Zeichenfolge "port-xyz" (mit x,y,z = 0...9) beginnen.

  Der Name darf nicht die Form "a.b.c.d" (mit a,b,c,d = 0...999) haben.

☞ Der Stationsname wird zur Adressierung des Profinet IO Devices benutzt. Fehlende oder abweichende IP-Adress-Einstellungen auf Seiten des Devices werden ignoriert.

IP-Parameter

Die Felder zu "IP-Adresse", "Subnetzmaske" und "Default-Gateway" enthalten Standardeinträge, die von den Einstellungen des IO Controllers abgeleitet sind.

Diese Werte können mit den entsprechenden Netzwerk-Kenntnissen angepasst werden. Es wird jedoch empfohlen, diese Werte nicht zu verändern.

Kommunikation Parameter

Dieser Bereich beschreibt das Zeitverhalten der Kommunikation auf dem Profinet. Sie werden durch den Profinet IO Controller beim Verbindungsaufbau eingestellt.

Aus den beiden Parametern

`Send clock` und

`Reduction Ratio`

ergibt sich die tatsächliche Zykluszeit

`(t = Reduction Ratio * Send clock ),`

gemäß der das Device Daten sendet:

- **Send clock (ms):** Sendezeit in Millisekunden.
- **Reduction ratio:** Faktor zur Berechnung der Zykluszeit aus der Sendezeit.

- **Watchdog**: Überwachungszeit (ms), mögliche Werte: 0 bis 65535. Wenn das IO Device innerhalb dieser Zeit kein Signal mehr vom IO Controller erhalten hat, erfolgt eine gerätespezifische Reaktion, z.B. eine Fehlermeldung.
- **Phase**: Die Phase wird intern im Controller eingestellt.
- **VLAN ID:** VLAN Kennzeichner: Hier ist eine Nummer einzugeben, zwischen 0–4095 für VLAN Typ 802.1Q oder zwischen 0–32767 für VLAN Type ISL (Verfügbarkeit ist geräteabhängig).
- **RT Class:** Der IO Controller unterstützt derzeit nur Realtime (RT) Class 1. Dieses Feld ist für künftige Erweiterungen vorgesehen.

Anwenderparameter

Schaltflächen "Anwenderparameter"

- **Alle Standardwerte setzen:** Durch Betätigen dieser Schaltfläche werden alle geänderten Einstellungen durch die Voreinstellungen aus GSDML-Datei ersetzt.

Parameter-Tabelle

In der Parameter-Tabelle kann durch einen Doppelklick auf den jeweiligen Wert dieser editiert werden. Abhängig vom Parameter geschieht dies über eine Auswahlliste oder eine direkte Werteingabe.

- **Parameter:** Name des Parameters bzw. Parameter-Kategorie (dann ohne Wertezuordnung)
- **Wert:** Aktueller Parameterwert
- **Wertebereich:** Angabe, was in Spalte 'Wert' eingetragen sein darf.

  Syntax:

  ```
  <Parameter_Typ>(<verwendete Bits>)
                  <Grundwert> <Wertebereich>.
  ```

  Beispiel:

  "BitArea (4-5) 0 0-2" bedeutet:

  Es handelt sich um eine Bit-Kombination, die in den Bits 4 und 5 des Konfigurations- Bytes abgelegt wird.

  Der Grundwert ist 0, der Wert kann zwischen 0 und 2 liegen.

Registerkarte 'Optionen'

Diese Registerkarte bietet zusätzliche Konfigurations-Optionen (ab IndraWorks 14V22).



Abb. 6-25: Profinet IO Device - Registerkarte 'Optionen'

Shared Device

Bei einem Shared Device teilen sich mehrere Profinet IO Controller die (Sub-)Module eines Profinet IO Devices. Zu jedem (Sub-)Modul kann nur eine Verbindung gleichzeitig aufgebaut werden (Ausnahme "Shared Input"). Es gibt keine Priorisierung der Verbindung. Es gilt das Prinzip „First come, first served“.

Die Option "Shared Device" wird nur angeboten, wenn das Profinet IO Device dieses Feature unterstützt.

Wird die Option ausgewählt, so wird keine Verbindung zur Kopfstation des IO Devices aufgebaut, stattdessen nur eine Verbindung zu den ausgewählten

Modulen. Es kann nur ein Profinet IO Controller eine Verbindung zur Kopfstation haben, d.h. bei allen anderen IO Controllern, die auch auf dieses Device zugreifen wollen, muss diese Option gesetzt werden.

Um das Feature "Shared Device" zu verwenden, muss das gleiche IO Device bei mehreren IO Controllern konfiguriert werden. Hierbei ist darauf zu achten:

- die Controller müssen unterschiedliche Namen haben, sowie unterschiedliche IP-Adressen im gleichen Subnetz.
- das Device muss in beiden Konfigurationen den gleichen Stationsname, IP-Adresse, Subnetzmaske und Default-Gateway besitzen.
- nur in einem Controller darf bei den Device-Optionen der Haken bei "Shared Device" nicht gesetzt sein.
- die Module des Devices müssen in allen Controllern konfiguriert werden, müssen aber wechselseitig deaktiviert werden, so dass die Module exklusiv nur bei einem Controller aktiv sind (Ausnahme "Shared Input").

Bei Verwendung eines "Shared Device" darf bei den deaktivierten Modulen **keine Slot-Komprimierung** stattfinden.

In der Defaulteinstellung ist für deaktivierte E/A-Module die Slot-Komprimierung aktiv, d.h. diese muss für eine Shared Device Anwendung ausgeschaltet werden, siehe Kap. "Registerkarte 'Einstellungen'" auf Seite 92.

Beispiel: Shared Device Konfiguration

In nachfolgender Abbildung gilt: XM4 (links oben) und XM4 (rechts oben) sind Profinet IO Controller im Datenaustausch mit S20-PN-BK+ (unten). Dargestellt ist mit Farben (blau, orange) jeweils die E/A-Kommunikationsbeziehung zwischen Profinet IO Controller und Profinet IO Device.

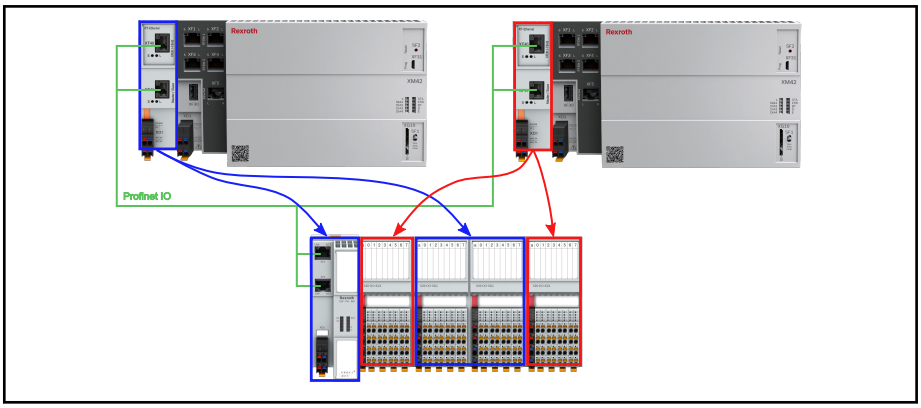


Abb. 6-26: "Shared Device" Beispiel Topologie

Es wird folgende Beispiel-Konfiguration verwendet:

- XM42 (blaue Markierung)
  - Stationsname: **extension.xm42.1**
  - IP: **192.168.3.1**
  - Subnetz: **255.255.255.0**
- XM42 (rote Markierung)
  - Stationsname: **extension.xm42.2**
  - IP: **192.168.3.2**
  - Subnetz: **255.255.255.0**

- S20-PN-BK+
  - Stationsname: **s20-pn-bk**
  - IP: **192.168.3.3**
  - Subnetz: **255.255.255.0**
  - die komplette Modulkonfiguration ist auf beiden IO Controller vorhanden, die Module sind aber wechselseitig deaktiviert.

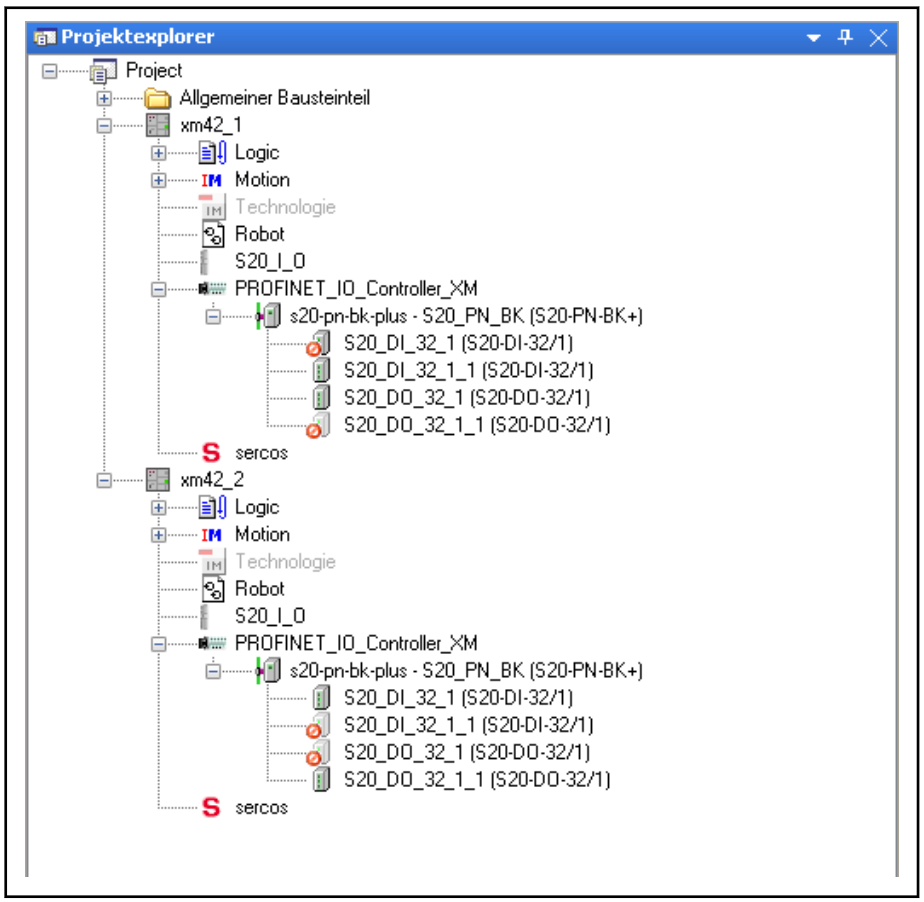


Abb. 6-27: "Shared Device" Beispiel Konfiguration

Registerkarte 'IOxS'

Durch einen entsprechenden Eintrag in der Gerätebeschreibung ermöglich diese Registerkarte den Zugriff auf IO-Provider- / IO-Consumer-Daten.



Untermodule ohne eigene E/As stellen ebenfalls ein "Provider State" für Eingänge zur Verfügung.

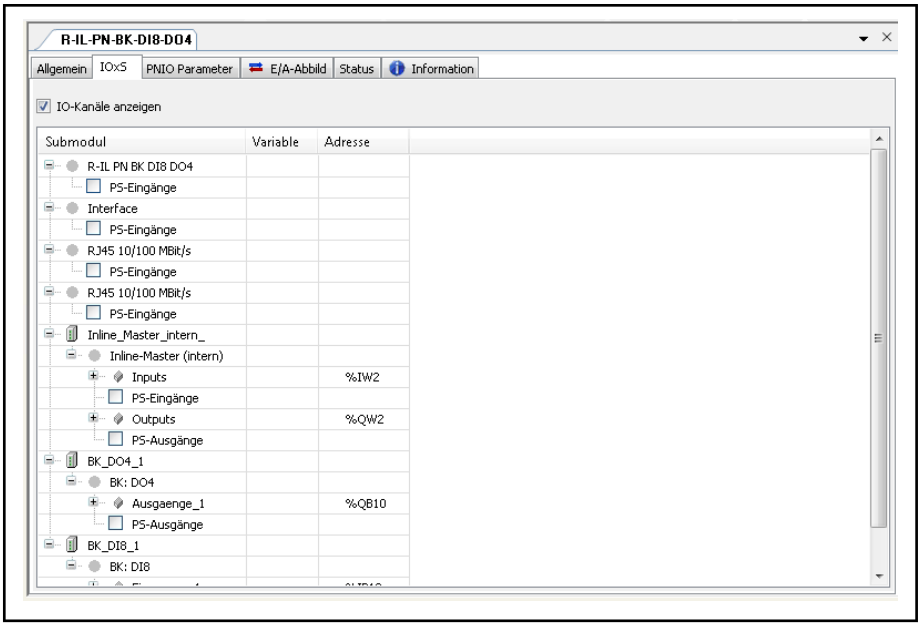


Abb. 6-28: Profinet IO Device - Registerkarte "IOxS"

Wenn die Checkbox aktiviert ist, stellt das zugehörige Modul zusätzliche EAs mit dem Provider State in dem IO-Mapping-Dialog zur Verfügung.

Das entsprechende Modul kann durch einen Doppelklick auf das Adressfeld geöffnet werden.

Der Wert hex 80 (Bit 7) des Provider State-Bytes zeigt den Status "OK". Detaillierte Informationen über die Definition dieses Bytes kann in der technischen Spezifikation zum Profinet IO gefunden werden.

Im Falle dass ein IO Provider State für Ausgangsdaten gemappt ist, muss der Ausgangs PS explizit auf "GOOD" (=0x80) gesetzt werden, da ansonsten die Daten vom Gerät nicht übernommen werden und evtl. die Ersatzwerte verwendet werden.

Im Falle dass die Applikation gestoppt wird, werden die nicht gemappten IO Provider-States für Ausgänge auf "bad" gesetzt; gemappte IO Provider-States werden auf ihre konfigurierten Default-Werte gesetzt.

Registerkarte 'PNIO Parameter'

Diese Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor**.

Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

Bei laufendem Bus ist eine Übertragung geänderter Parameter durch Betätigung der Taste <Parameter schreiben> möglich (offline unsichtbar).

Die "MAC Adresse" dient nur zu Informationszwecken und wird nur gesetzt, wenn

- das IO Device aus dem "Geräte suchen"-Menü übernommen wurde
- die Adresse aus dem "Geräte suchen"-Menü übertragen wurde

Registerkarte 'Status'

Die Registerkarte "Status" im "Profinet IO-Geräteeditor" zeigt Status-Informationen (z. B. "Running", "Stopped") und spezifische Diagnosemeldungen aus dem Gerät.

Zusätzlich werden die Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt. Die Ausgabe erfolgt als hexadezimaler Wert, der sich aus den gesetzten Status- Flags ergibt (entsprechend dem Dokument "Protocol Interface Manual" für Profinet IO).

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

6.5.3 Profinet IO Controller, Module zum Device hinzufügen

Allgemeines

Dieser Abschnitt enthält grundlegende Informationen zu Konfiguration und Parametrierung von I/O-Modulen (E/A Klemmen) am modularen Profinet IO Devices.

Neu anzulegende Profinet IO Module können entweder über die Gerätebibliothek, oder über das Kontextmenü des Profinet Objekts im Projektexplorer hinzugefügt werden.

In der Gerätebibliothek finden Sie die Module, die zu dem jeweiligen Profinet IO Device passen, in der Bibliothek "Peripherie" im Ordner "Profinet IO" unterhalb des Profinet IO Device. Ziehen Sie das gewünschte Modul aus der Bibliothek auf das Profinet IO Device im Projektexplorer. Auf diese Weise können I/O-Module auch zwischen bereits vorhandene Module im Projektexplorer eingefügt werden.

Alternativ können Sie das neu anzulegende Profinet Modul auch über das Kontextmenü des Profinet IO Objekts hinzufügen. Öffne Sie hierzu das Kontextmenü des Profinet IO Objekts im Projektexplorer und wählen Sie den Kontextbefehl: **Hinzufügen ▸ Modul ▸ ...**. Das neue Modul wird als letztes Modul unter das Profinet IO Device eingefügt.

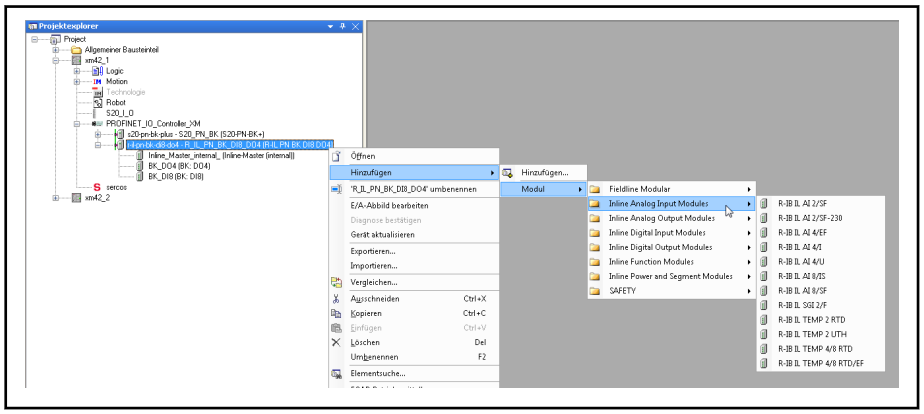


Abb. 6-29: Profinet IO Controller, Module zum IO Device hinzufügen

Zur Konfiguration der neuen Module öffnen Sie den Geräteeditor durch Doppelklick auf das jeweilige Modul im Projektexplorer.

Dialoge für Module am Profinet IO Device

- Profinet IO-Module aktivieren / deaktivieren auf Seite 110
- Profinet IO-Devices und Profinet IO-Module deaktivieren mittels UserDefs.cfg auf Seite 99
- Registerkarte 'Allgemein' auf Seite 111
- Registerkarte 'PNIO Parameter' auf Seite 112, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'E/A-Abbild' auf Seite 112
- Registerkarte 'Statusauf Seite 113
- Registerkarte 'Information' auf Seite 113

Profinet IO-Module aktivieren / deaktivieren

Die an einem Profinet IO-Device konfigurierten Profinet IO-Module können im Projektexplorer aktiviert bzw. deaktiviert werden.

Dies erfolgt durch Anwählen / Abwählen der Schaltfläche, die am Icon des Profinet IO-Moduls erscheint.

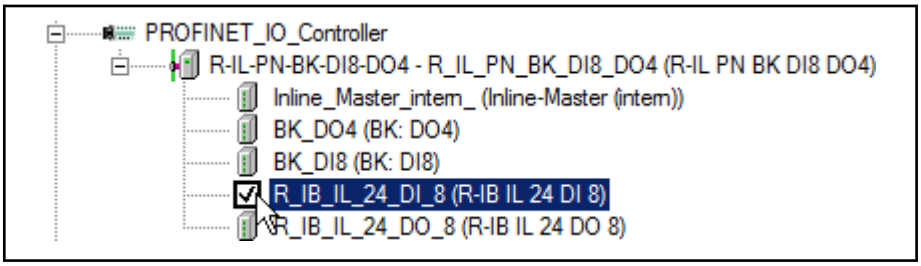


Abb. 6-30: Aktivieren / Deaktivieren von Profinet IO-Modulen

Ist der Haken gesetzt, ist der Busteilnehmer aktiviert. Er wird in der Diagnose des Busses berücksichtigt.

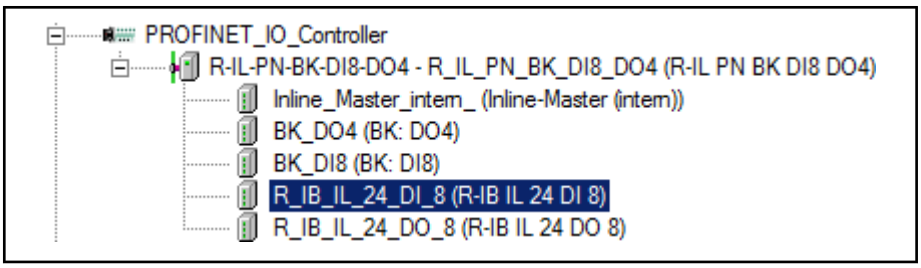


Abb. 6-31: Profinet IO-Modul als aktivierter Teilnehmer

Ist der Haken nicht gesetzt, ist der Busteilnehmer deaktiviert. Er wird in der Diagnose des Busses nicht berücksichtigt.

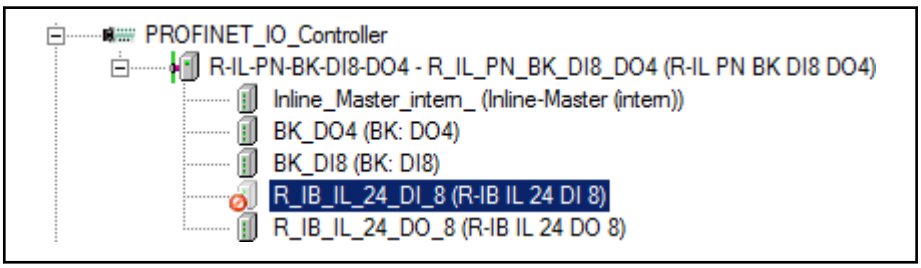


Abb. 6-32: Profinet IO-Modul als deaktivierter Teilnehmer

Wird ein neues Profinet IO-Modul angelegt, ist es per Voreinstellung aktiviert.

Beim Laden der Konfiguration werden alle aktivierten Profinet IO-Module berücksichtigt. Deaktivierte Profinet IO-Module werden ignoriert.



Ein aktivierter nicht vorhandener Busteilnehmer erzeugt in der Diagnose eine Fehlermeldung.

Ein deaktivierter nicht vorhandener Busteilnehmer erzeugt in der Diagnose keine Fehlermeldung.

Siehe auch Profinet IO-Devices und Profinet IO-Module deaktivieren mittels UserDefs.cfg auf Seite 99

Registerkarte 'Allgemein'

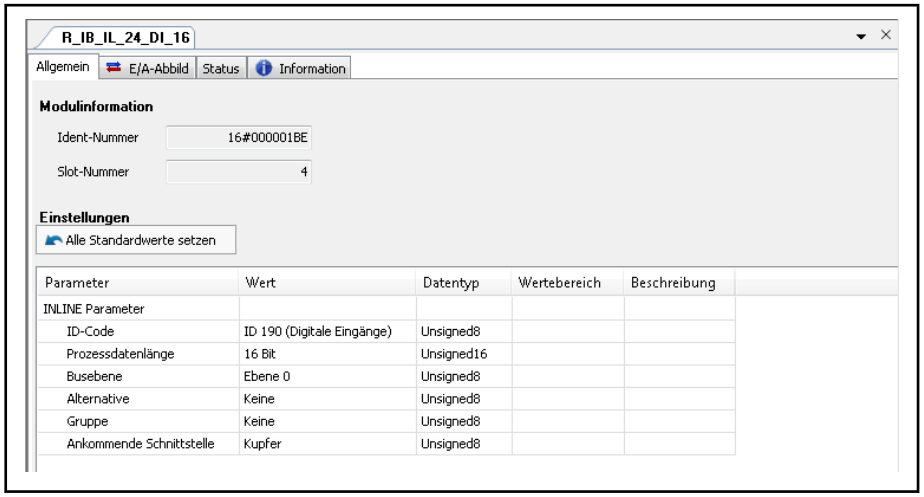


Abb. 6-33: Profinet IO Modul - Registerkarte 'Allgemein'

Modulinformation

- **Id-Nummer**: Kennung des Moduls, aus der Gerätebeschreibung.
- **Slot-Nummer**: Position des Moduls unterhalb des Devices, startet mit "1" für das erste Modul und wird für jedes weitere hochgezählt; ergibt sich also automatisch aus der aktuellen Struktur im Gerätebaum.

Anwenderparameter

Geräteabhängig gibt es keine oder mehrere zusätzliche Einstellmöglichkeiten für die anwenderspezifischen Parameter:

- **Länge der Anwenderparameter**: Die Gesamtlänge aller Anwender-Parameter wird in Bytes angezeigt.

In der Parameter-Tabelle können Sie die Werte editieren, indem Sie auf den Wert doppelklicken. Abhängig vom Parameter geschieht dies über eine Auswahlliste oder eine direkte Werteingabe.

- **Parameter**: Name des Parameters oder der Parameter-Kategorie (ohne Wertezuordnung).
- **Wert**: Aktueller Parameterwert
- **Wertebereich:**
  - **Grundwert**: Standardwert des Parameters.
  - **Erlaubte Werte**: Angabe, was in Spalte "Wert" eingetragen sein darf.

    Syntax:

    `<Parameter Typ> (<verwendete Bits>) <Grundwert> <erlaubter Wertebereich>.`

Beispiel:

"BitArea (4-5) 0 0-2" bedeutet: Es handelt sich um eine Bit-Kombination, die in den Bits 4 und 5 des Konfigurations-Bytes abgelegt wird. Der Grundwert ist 0, der Wert kann zwischen 0 und 2 liegen.

- *Schaltflächen "Anwenderparameter" (nur Online aktiv)*
  - **Alle Standardwerte setzen**: Durch Betätigen dieser Schaltfläche werden alle geänderten Einstellungen durch die Voreinstellungen aus GSDML-Datei ersetzt.

Registerkarte 'PNIO Parameter'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor**.

Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

"Parameter schreiben" gibt es nur Online.

Der Dialog gibt Auskunft über Lage und Größe der Parameter.

Submodule: Einige neuere Profinet Module haben Submodule, welche in der GSDML Datei definiert sein müssen. Mit den Import der GSDML-Datei werden diese Submodule erzeugt.

Registerkarte 'E/A-Abbild'

Die Registerkarte dient der Zuordnung der Modul-Ein- und Ausgänge zu Variablen, die als lokale oder globale Variablen in den einzelnen POUs verwendet werden können.

Im Online-Modus wird der aktuelle Wert der Variablen angezeigt.

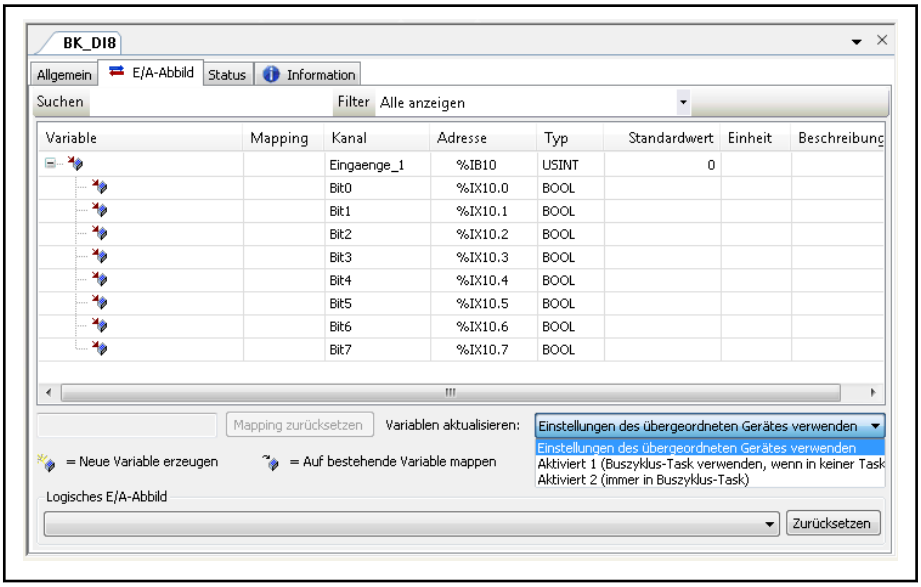


Abb. 6-34: Profinet IO Modul - Registerkarte 'E/A-Abbild', hier Digitale-Ausgangs-Busklemme

Kanäle: Der Bereich "Kanäle" beim diesem PNIO Device die IOPS-Eingänge enthalten.

Die Zuordnung ist unter Mapping der Feldbus-Ein- und Ausgänge auf Seite 19 beschrieben.

Mapping zurücksetzen: Über diese Schaltfläche können die Mapping-Einstellungen auf die durch die Gerätebeschreibungsdatei definierten Standardwerte zurückgesetzt werden.

Variablen aktualisieren: Festlegung für das Modul , ob die E/A-Variablen in der Buszyklus-Task aktualisiert werden.

Der Default-Wert ist in der Gerätebeschreibung definiert.

- **Einstellungen des übergeordneten Geräts verwenden:** Aktualisierung gemäß den Einstellungen des übergeordneten Geräts.
- **Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet):** Die E/A-Variablen werden in der Buszyklus-Task aktualisiert, falls sie in keiner anderen Task verwendet werden
- **Aktiviert 2 (immer im Buszyklus-Task):** Alle Variablen werden in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

Registerkarte 'Status'

Die Registerkarte "Status" im "Profinet IO-Geräteeditor" zeigt Status-Informationen (z. B. "Running", "Stopped") und spezifische Diagnosemeldungen aus dem Gerät.

Zusätzlich werden die Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt. Die Ausgabe erfolgt als hexadezimaler Wert, der sich aus den gesetzten Status- Flags ergibt (entsprechend dem Dokument "Protocol Interface Manual" für Profinet IO).

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

6.5.4 Profinet IO Controller, Geräte suchen

Geräte suchen

Der Befehl "Geräte suchen" dient zur automatischen Ermittlung von Hardware Komponenten, die mit der Steuerung verbunden sind.

Voraussetzungen

Bevor der Befehl ausgeführt werden kann, muss der Profinet IO-Treiber für die entsprechende Schnittstelle in der Steuerung geladen sein.

Hierzu muss die Steuerung initial mit einem SPS-Projekt eingeloggt werden, bei dem die betreffende Feldbus-Schnittstelle als Profinet IO-Controller konfiguriert ist.

Des Weiteren müssen die Gerätebeschreibungsdateien der jeweils angeschlossenen Geräte in der IndraWorks Gerätedatenbank vorhanden sein.

Wenn die Gerätebeschreibungsdatei für den konkreten Teilnehmer in der Gerätedatenbank fehlt, ist die Suche nur teilweise möglich! In diesem Fall muss die Gerätebeschreibungsdatei in der Gerätedatenbank hinzugefügt werden und die Gerätesuche wiederholt werden (Geräte-Beschreibungsdatei nachladen, Seite 15).

Führen Sie zur Suche folgende Schritte aus:

- Schalten Sie die Steuerung Online
- Die Applikation muss im Betriebszustand "Läuft" befinden
- Selektieren Sie das betreffende Profinet IO Controller-Objekt im Projektexplorer
- Öffnen Sie das Kontextmenü des Objekts und wählen Sie den Befehl "Geräte suchen...".

Kontextbefehl "Geräte suchen"

Der Befehl "Geräte suchen" dient dazu, um die aktuell angeschlossene Hardware zu erfassen, d. h. ihre Struktur zu ermitteln, sie in einem Dialog darzustellen und für die Übernahme in den Projektexplorer bereitzustellen.

Die Suchfunktion ist für verschiedene Netzwerktypen unterschiedlich ausgeführt:

Der Aufruf kann in der Steuerung fest implementiert sein (in einem Dialog des Wizards zum Anlegen der Steuerung), oder aber als Menüpunkt des Feldbusses zu Verfügung gestellt sein (OnBoard-Schnittstelle oder am Erweiterungsmodul).

Beim Ausführen des Befehls "Geräte suchen" wird automatisch eine Verbindung zur Steuerung auf- und danach wieder abgebaut.

Um die Suche zu starten selektieren Sie das betreffende Profinet Objekt im Projektexplorer, öffnen dessen Kontextmenü und wählen den Befehl **Geräte suchen...**.

Die bei der Suche gefundenen Geräte und zugehörigen Module werden im Dialog "Geräte suchen – Gefundene Geräte" angezeigt.

Funktionen für Profinet IO-Geräte:

In einem Profinet IO-Bussystem können Geräte sowie Module gescannt werden.

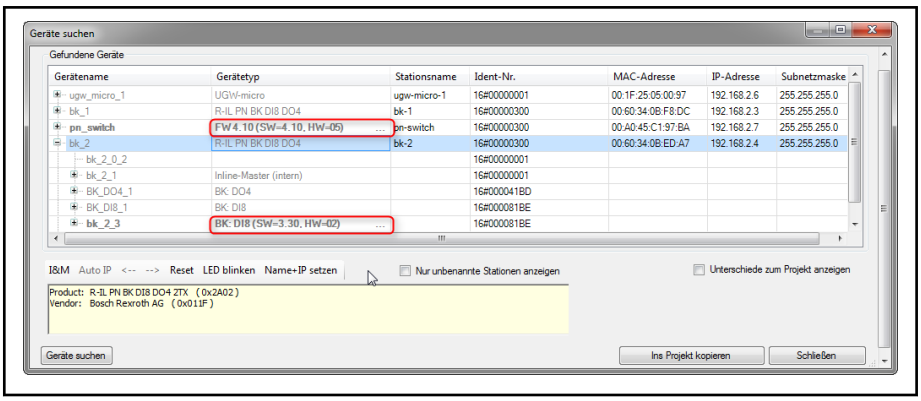


Abb. 6-35: Dialog "Geräte suchen" für Profinet IO, Busklemme mit DI8-Modul gefunden

Für die im Bild oben markierte Busklemme gibt es mehrere unterschiedliche Typen (Auswahlliste).

Der Anwender muss entscheiden, welcher Typ zum Einsatz kommen soll.

Vorgehensweise bei der Konfiguration

- Im Projektexplorer im Kontextmenü des Profinet IO-Controllers den Befehl "Geräte suchen" ausführen.

  Der Dialog zeigt alle gefundenen Geräte an.

- Im Feld "**Stationsname**" (Adressierung bei Profinet IO) den gewünschten Namen für das jeweilige Gerät eingegeben und mittels dem Button "**Name+IP setzen**" in das Gerät übertragen (die IP-Adresse kann hierbei noch auf 0.0.0.0 stehen).

  Weitere Geräte müssen einzeln editiert werden, da die Funktion "Name+IP setzen" nur auf das selektierte Gerät wirkt.

- Mit dem Button "**Auto IP**" wird dem Gerät eine freie IP-Adresse aus dem IP-Bereich der PNIO Controller Konfiguration zugeteilt.

- Hat das Gerät einen Stationsnamen und eine gültige IP-Adresse erhalten, kann die Konfiguration ins IndraWorks-Projekt übernommen werden (Button "ins Projekt kopieren"). Hierbei können auch mehrere Geräte gleichzeitig selektiert werden.

Weitere Funktionen

- Zu jedem Gerät werden in weiteren Spalten im Fenster "Gefundene Geräte" **Stationsname**, **MAC Adresse**, **IP Adresse** und **Subnetzmaske** angezeigt.

Jedes Gerät muss einen **Stationsnamen** erhalten, da dieser für bestimmte Funktionalitäten im Netzwerkbetrieb benötigt wird!

- Wenn beim Scannen der Filter **Nur unbenannte Stationen anzeigen** aktiviert ist, werden nur Geräte aufgelistet, für die noch kein Stationsname vergeben wurde.
- Der **Stationsname** kann hier im Dialog geändert werden.

  Das Feld "Stationsname" geht nach einem Maus-Doppelklick vom Anzeige- in den Editiermode.

  Um den neuen Namen im Bussystem bekannt zu machen, muss bei selektiertem Geräteeintrag die Schaltfläche <Name+IP setzen> betätigt werden.

Wenn Werte in der Liste der gefundenen Geräte geändert wurden (z.B. Änderung der IP-Adresse), wird der entsprechende Wert kursiv dargestellt.

Dieser neue Wert ist dann nur im Editor und noch nicht im Gerät geändert.

Nachdem der Wert in das Geräte geladen wurde, wird er wieder normal dargestellt.

Werte, die Unterschiede zwischen Projekt und gefundenen Geräten zeigen, werden orange dargestellt.

☞ Nach einer Veränderung der zuvor gescannten Daten und nach Durchführung des Befehls <Name+IP setzen> muss der Befehl "Geräte suchen" erneut betätig werden.

In der Anzeige erscheinen dann die aktuellen Daten des Gerätes.

- Auch die Einträge für **Gerätename**, **IP Adresse** und **Subnetzmaske** können auf diese Weise bereits hier im Dialog bearbeitet werden, bevor das Gerät mit diesen Einstellungen ins Projekt übernommen wird.

  Wenn das Gerät ohne definierte Adresse übernommen wird, wird die Voreinstellung, die im "PNIO Controller Parameter"-Dialog des Geräteeditors für solche Fälle festgelegt ist, verwendet.

  Siehe Kap. "Registerkarte 'Allgemein' " auf Seite 88

  Eine endgültige Adress-Definition kann für ein Gerät in dessen Konfigurationsdialog "PNIO-Parameter" vorgenommen werden.

  Siehe Kap. "Registerkarte 'Allgemein'" auf Seite 103.

- Die **Modul-ID-Nummer** wird nur dann in der Liste angezeigt, wenn
  1. sie vom Device bereitgestellt wird,
  2. wenn Device und Controller Teil desselben logischen Netzwerks sind und
  3. wenn IP-Adresse und Netzwerkmaske des Devices korrekt gesetzt sind.

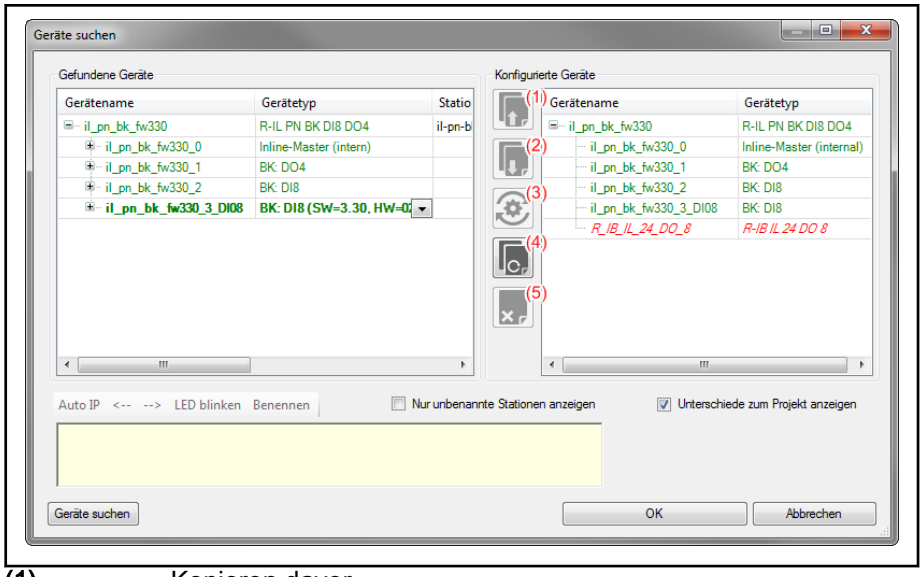
  Wenn die ID-Nummer nicht vorliegt, kann ein Device dennoch ins Projekt "kopiert" werden, wenn es durch die sonst verfügbaren Informationen eindeutig identifiziert werden kann.

Mit **Auto-IP** werden selektierte Geräte, die keine gültige IP-Konfiguration haben, so eingestellt, dass sie zu den Controller-Einstellungen passen und keine Konflikte mit anderen Geräten auftreten.

Mit dem Befehl **I&M** werden die I&M-Daten ("Identification and Maintenance") des Gerätes angezeigt, u.a. Seriennummer, Hardware-Revision und Software-Revision.

Unterschiede zum Projekt anzeigen

Diese Option öffnet einen Dialog, in welchem die gefundenen Geräte und Module mit den konfigurierten Geräten verglichen werden können.



(1) Kopieren davor
(2) Kopieren danach
(3) Wechseln nach
(4) Alles Kopieren
(5) Löschen

Abb. 6-36: Dialog "Unterschiede zum Projekt anzeigen": R_IB_IL_24_DO_8 nur im Projektexplorer

Die Geräte, welche in grün dargestellt werden, sind auf beiden Seiten identisch, wogegen die rot dargestellten Geräte nur in der Ansicht der gefundenen Geräte bzw. der konfigurierten Geräte vorhanden sind.

- Mit den Befehlen "Kopieren (davor)" und "Kopieren (danach)" können die markierten Geräte auf die Seite der " Konfigurierten Geräten" kopiert werden. Das Kopieren erfolgt befehlsabhängig vor oder nach dem ausgewählten Gerät.
- Ein konfiguriertes Gerät kann durch ein gefundenes Gerät ersetzt werden indem der Befehl "Ersetzen durch" ausgeführt wird. Für diese Aktion müssen die betreffenden Geräte auf beiden Seiten markiert werden.

Auto IP: Dieses Kommando setzt automatisch eine gültige IP-Adresse, falls die vom (markierten) Gerät gescannte Adresse ungültig ist.

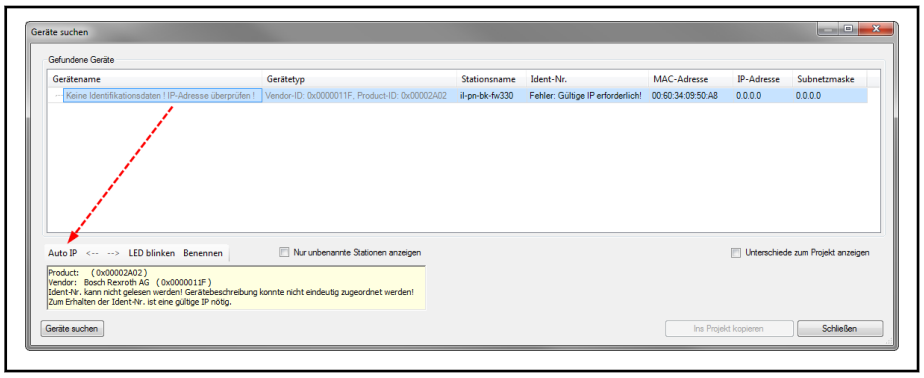


Abb. 6-37: Dialog "Geräte suchen", Gerät ohne Identifikationsdaten gefunden

Nach dem Markieren der Zeile wird <Auto IP> betätigt. Die fehlenden Daten werden entsprechend Voreinstellung ersetzt (Ergebnis siehe Abb. 6-35 "Dialog Geräte suchen für Profinet IO, Busklemme mit DI8-Modul gefunden" auf Seite 114).

<-- Online-Gerät zuweisen: Mit diesem Kommando werden die Einstellungen des konfigurierten Gerätes dem gescannten Gerät zugewiesen.

--> konfigurierten Gerät zuweisen: Mit diesem Kommando werden die Einstellungen des gescannten Gerätes dem konfigurierten Gerät zugewiesen.

LED blinken: Mit diesem Kommando kann ein aufgelistetes Gerät an der Hardware selbst identifiziert werden.

Das Gerät sollte darauf mit einem Blinksignal (häufig "Link ok") reagieren.

Benennen: Übernahme des geänderten **Stationsnamen**.

Konfigurationsabgleich

☞ Beachten Sie, dass diese Funktionalität nur für Profinet IO-Bussysteme verfügbar ist!

Der Befehl bietet einen komfortablen Weg, die Konfigurationseinstellungen eines Profinet IO-Device-Gerätes, die im Projekt definiert sind, mit denen des entsprechenden Gerätes (gleiche Geräte-ID und Hersteller-ID) in der angebundenen Hardware abzugleichen.

Dies bedeutet, dass die lokalen Einstellungen mit den Hardware-Einstellungen überschrieben werden.

Der Dialog "Geräte suchen" macht in diesem Falle eine Vorauswahl und zeigt diese an.

☞ Der Anwender muss also die Vorauswahl mit den physikalisch vorhandenen Geräten vergleichen und die Auswahl im Dialog anpassen!

Folgende Schritte sind dazu auszuführen:

- Die Kommunikationseinstellungen entsprechend setzen.
- Dann im Offline-Modus das betreffende Geräteobjekt im Gerätebaum auswählen und Befehl Konfigurationsabgleich" aus dem Kontextmenü auswählen.
- Der Dialog "Geräte suchen" wird geöffnet und die Suche nach passenden Geräten in der Hardware wird gestartet. Die gefundenen werden aufgelistet. Diejenigen, die bereits im Projekt verwendet werden (identifiziert anhand des Stationsnamens), können mit "Verwendete Geräte ausblenden" unsichtbar gesetzt werden.
- Der Suchprozess kann über die Schaltfläche "Geräte suchen" erneut gestartet werden.

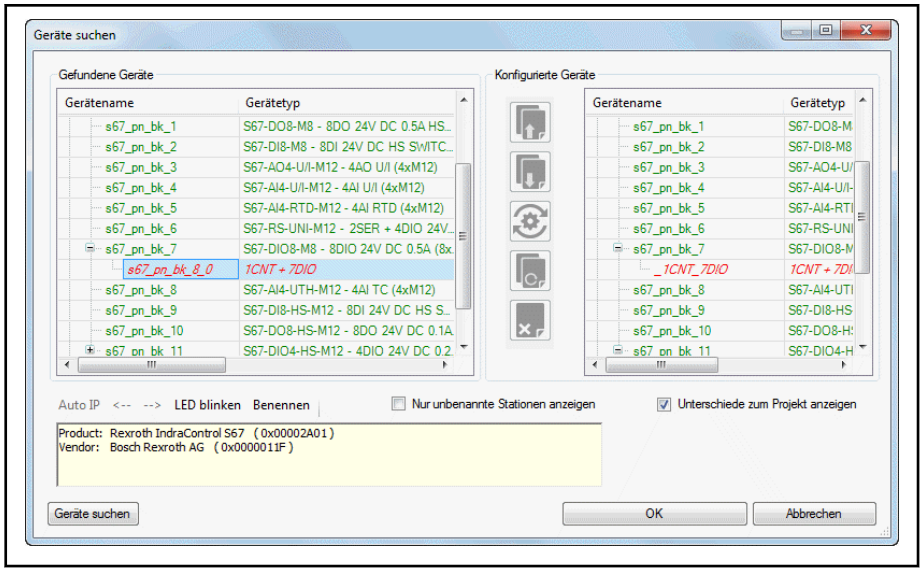


Abb. 6-38: Dialog "Geräte suchen" für Offline-Konfigurationsabgleich, Beispiel

<-- Online-Gerät zuweisen: Mit diesem Kommando werden die Einstellungen des konfigurierten Gerätes dem gescannten Gerät zugewiesen.

--> konfigurierten Gerät zuweisen: Mit diesem Kommando werden die Einstellungen des gescannten Gerätes dem konfigurierten Gerät zugewiesen.

Möglicherweise vorliegende Sub-Geräteobjekte werden dadurch nicht entfernt.

6.5.5 Profinet IO Controller, Topologie und Gerätetausch

Topologie und Gerätetausch, Allgemeines

Überblick: Wie bei anderen Feldbussen besteht auch bei Profinet die Möglichkeit "Slaves" (IO-Geräte) auszutauschen, ohne das dabei auf IndraWorks zurückgegriffen werden muss. Hierzu wird mit der Profinet-Topologieerkennung die Netzwerk- und Gerätestruktur eines Profinet-Geräteaufbaus ermittelt und gespeichert. Auf Basis der Topologieerkennungsdaten können ausgetauschte IO-Geräte am Profinet-Feldbus wieder in Betrieb genommen werden.

Namensbasierte Geräteadressierung: Bei Profinet wird jedem Gerät (IO-Device) ein logischer, individueller und eindeutiger Gerätename zugeordnet (Devicename, Stationsname, IdentStr). Anhand dieses Gerätenamens teilt der Profinet IO-Controller den IO-Devices die konfigurierte IP-Adresse zu. Der eigentliche Profinet-Datenverkehr wird im Folgenden über diese IP-Adresse und die damit aufgelöste MAC-Adresse abgewickelt.

Der Zugriff auf IP-Adressen und/oder Gerätenamen erfolgt mit dem, standardmäßig bei jedem Profinet-Gerät vorhandenen, DCP-Protokoll (DCP=Discovery and Configuration Protokoll).

Nachbarschaftserkennung: Mittels des LLDP (Link Layer Discovery Protokoll) können benachbarte Geräte Informationen austauschen. Dabei sendet jedes Gerät in periodischen Abständen Informationen über sich selbst. Der jeweilige Nachbar speichert diese Informationen für jeden seiner Netzwerkports ab.

Topologieerkennung: Die LLDP-Informationen aller konfigurierten Geräte werden vom Profinet IO-Controller ausgelesen, zu einer Topologieliste zusammengestellt, im XML-Format gesichert und auf der Steuerung abgelegt.

Austausch eines IO-Devices

Wenn ein IO-Device in einer Anlage ersetzt werden soll, muss lediglich der Gerätename des IO-Devices auf das Ersatzgerät übertragen werden.

Anhand dieses Gerätenamens wird das neue IO-Device vom Profinet IO-Controller erkannt, bekommt die passende IP-Adresse zugewiesen und wird damit wieder in den Produktivdatenverkehr aufgenommen.

Möglichkeiten zum Austausch eines IO-Devices:

Zur Übertragung des Gerätenamens auf das Ersatzgerät gibt es verschiedene Vorgehensweisen:

1. der Gerätename ist auf einem Speichermedium abgelegt. Das Speichermedium wird vom Originalgerät auf das Ersatzgerät übertragen. Seitens Profinet ist in diesem Fall keine gesonderte Aktivität notwendig.
2. der Gerätename ist auf dem Gerät selbst einstellbar. Das Ersatzgerät erhält die gleichen Einstellungen. Seitens Profinet ist hierzu keine gesonderte Aktivität notwendig.

   Beispiel: S67-PN-BK
3. Der Gerätename wird aus einem Engineering-System manuell auf das Ersatzgerät übertragen (benannt). Dies ist beispielsweise im IW-Engineering im Dialog **Profinet Geräte suchen ▸ Benennen** möglich.
4. Der Profinet IO-Controller ermittelt und speichert die Topologie seines Netzwerkes.

   Bei Geräteausfall und Auftauchen eines neuen, kompatiblen Gerätes wird der Gerätename automatisch übertragen. Dies wird im Weiteren näher erläutert.
5. Eine HMI-Applikation bildet die Vorgehensweise des Engineering-Systems nach. Dies ist derzeit noch nicht geplant.

Voraussetzungen für den Gerätetausch:

- Damit die vollständige Ermittlung der Gerätetopologie möglich ist, müssen alle Profinet-Geräte LLDP unterstützen. Eventuell verwendete Switches müssen auch gleichzeitig Profinet IO-Device sein und im Engineering-System konfiguriert sein.
- Die Profinet-Konfiguration muss fehlerfrei sein, um dem Profinet IO-Controller ein vollständiges Lernen der Topologie zu ermöglichen.
- *Anforderungen an das Ersatzgerät:*
  - Baugleichheit mit dem Originalgerät,
  - Installation an genau der gleichen Stelle im Netzwerk,
  - Verwendung der gleichen Anschlussports
  - Ein eventuell noch vorhandener Gerätename darf im vorhandenen Netzwerk nicht vorkommen.

    Idealerweise hat das Gerät den Zustand "Werkseinstellungen" (reset to factory default)

Ablauf des Gerätetausches

Erzeugung der Topologiedaten:

Nach jedem Projekt-Download wird automatisch eine neue Topologieinformation erzeugt. Dazu wartet der Profinet-Treiber bis der Feldbus vollständig und fehlerfrei angelaufen ist.

Wenn alle konfigurierten Devices ohne Diagnose sind, wird mit Hilfe der Livelist-Daten und der LLDP-Daten der IO-Devices die Topologieinformation der konfigurierten Geräte erzeugt und in einer XML-Datei abgelegt.

Der Name und der Ablageort der XML-Datei kann über die Konfigurationsdatei festgelegt werden. Der Profinet-Treiber ergänzt den Namen um die Instanznummer des Profinet IO-Controllers.

(Default ist `/ata0b/PNTOPO_n.xml`; n – Instanznummer Profinet)

Nach dem Aus- und Einschalten der Steuerung wird die XML-Datei wieder eingelesen.

Bei einem Download eines SPS-Projektes wird die XML-Datei verworfen.

Die Topologiedaten können auch mit Hilfe des Funktionsbausteins `IL_PNIOStoreTopology` innerhalb der SPS zu einem beliebigen Zeitpunkt neu erzeugt werden.

Tausch eines Gerätes: Während des Betriebs wird stetig überwacht, ob alle konfigurierten IO-Devices erreichbar sind.

Wenn ein IO-Device nicht mehr erreichbar ist, werden die Nachbargeräte des fehlenden Gerätes überprüft (sofern erreichbar), ob diese auf der Verbindung zum fehlenden IO-Device ein anders IO-Device erkennen.

Wenn ein neues IO-Device erkannt wird, erfolgt eine Kompatibilitätsprüfung. Hierbei wird geprüft, ob das neue Gerät dem Originalgerät entspricht (anhand von VendorID und DeviceID).

Wenn die Kompatibilität sichergestellt ist, erhält das neue Gerät den Gerätenamen des Originalgerätes. Danach erkennt der Profinet IO-Controller, dass sich ein neues IO-Device mit passendem Namen am Bus angemeldet hat und nimmt das IO-Device in den Produktivdatenverkehr auf.

Bei weiteren nicht erreichbaren IO-Devices wird ebenso verfahren.

Unterstützung der Topologie- und Gerätetauschfunktionen durch die SPS

Zur Unterstützung der Topologie- und Gerätetauschfunktionen enthält die RIL_ProfinetIO.library mehrere Funktionsbausteine:

IL_PNIOAssignName

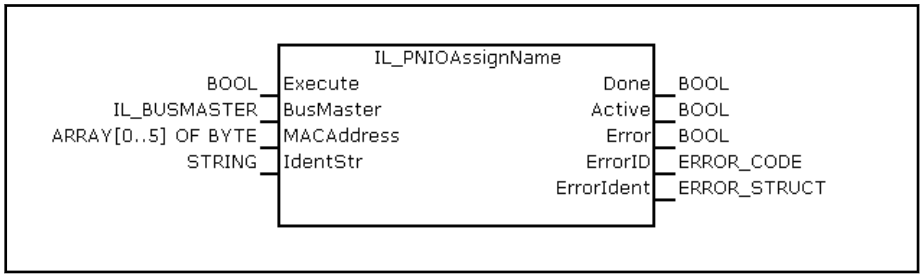


Abb. 6-39: IL_PNIOAssignName

Der Funktionsbaustein weist einem Profinet IO-Device den Gerätenamen (=Identstring) zu.

Dies wird auch als "Gerät benennen" bezeichnet.

Das IO-Device wird durch seine MAC-Adresse adressiert.

Die MAC-Address kann dem Baustein IL_PNIOGetLiveList entnommen werden.

IL_PNIOCheckTopology

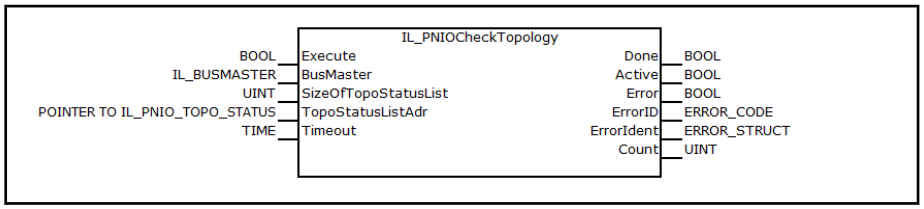


Abb. 6-40: IL_PNIOCheckTopology

Der Funktionsbaustein überprüft die gespeicherte Topologie der Profinet-Gerätetauschfunktion anhand der aktuell vorhandenen Profinet-Topologie. Dies erfolgt durch Auslesen der Port-Informationen aus den Profinet IO-Devices.

IL_PNIODeviceReplacement

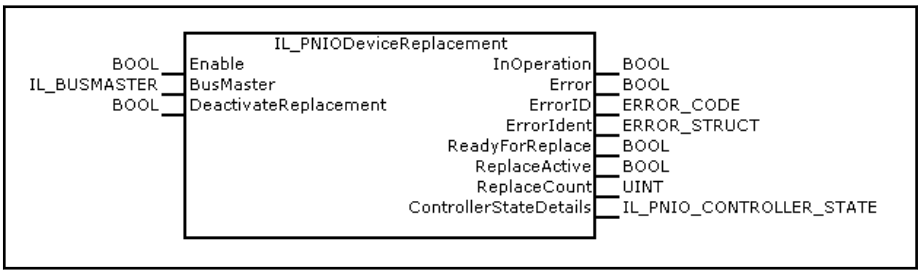


Abb. 6-41: IL_PNIODeviceReplacement

Der Funktionsbaustein liefert eine Diagnose über den Status der automatischen Gerätetauschfunktion. Die Gerätetauschfunktion kann auch deaktiviert werden.

Verwendung des FB IL_PNIODeviceReplacement

- Durch den zyklischen Aufruf des FBs IL_PNIODeviceReplacement kann überwacht werden, wie viele PN-Devices in zyklischer Kommunikation sind und ggf. Diagnose melden (siehe Ausgang "ControllerStateDetails").
- Der Ausgang "ReadyForReplace" zeigt an, dass die Topologie-Daten gültig vorliegen (nachdem diese nach einem Download / Reset ermittelt wurden bzw. nach dem Hochfahren der Steuerung aus der Datei gültig eingelesen wurden).

  Der Zustand "ReadyForReplace" kann nur erreicht werden, wenn die Gerätetauschfunktion nicht deaktiviert ist (Eingang "DeactivateReplacement" bei IL_PNIODeviceReplacement).
- Mittels dem FB IL_PNIORestoreTopology kann dann die Topologie wiederhergestellt werden (z.B. nach einem Kreuztausch von Geräten) oder allgemein die Topologie überprüft werden.

  Im Fehlerfall "Topologie-Fehler" können durch den FB IL_PNIOCheckTopology dann weitere Details ausgelesen werden. Beide Funktionsbausteine habe als Voraussetzung, dass "ReadyForReplace" vorliegt, und es wird empfohlen während der Ausführung von "Restore" und "Check" die Gerätetauschfunkion zu deaktivieren (Eingang "DeactivateReplacement" bei IL_PNIODeviceReplacement), da sich intern genutzte Ressourcen blockieren könnten.

IL_PNIODeviceResetToDefault

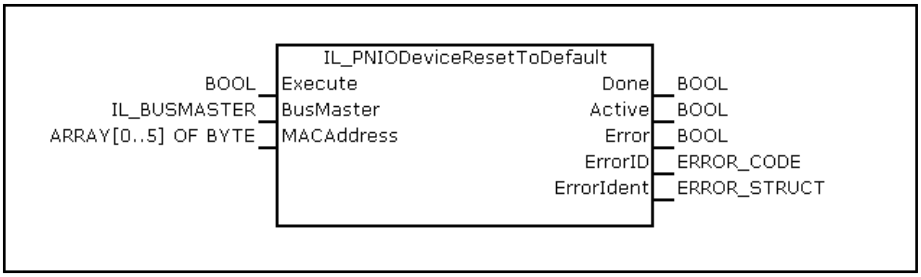


Abb. 6-42: IL_PNIODeviceResetToDefault

Der Funktionsbaustein setzt ein Profinet IO-Device auf den Auslieferungszustand (Factorydefault) zurück.

Hintergrund: Mit diesem Aufruf kann ein bereits verwendetes IO-Device wieder in den Auslieferungszustand gebracht werden. Je nach Hersteller und IO-Device kann dies auch über andere Methoden bewerkstelligt werden.

IL_PNIODeviceSetIP

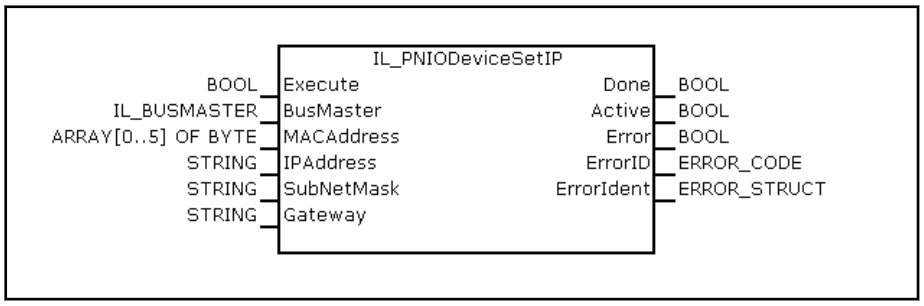


Abb. 6-43: IL_PNIODeviceSetIP

Der Funktionsbaustein setzt die IP-Einstellungen eines Profinet IO-Devices. Das IO-Device wird über seine MAC-Adresse adressiert. Die MAC-Address kann dem Baustein IL_PNIOGetLiveList entnommen werden.

IL_PNIOGetLiveList

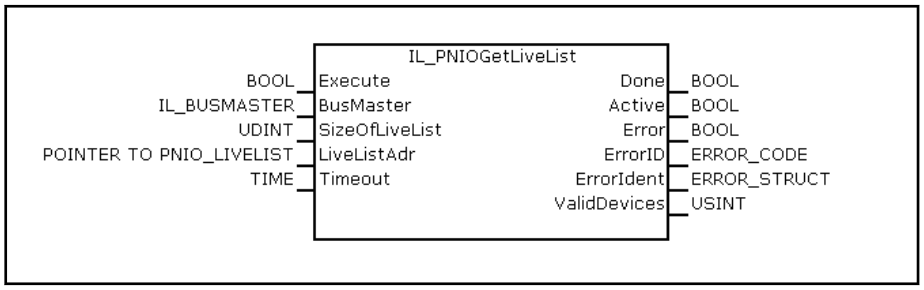


Abb. 6-44: IL_PNIOGetLiveList

Der Funktionsbaustein liefert eine Liste der aktuell erreichbaren Profinet-Teilnehmer.

Die Adressierung des IO-Devices erfolgt anhand der MAC-Adresse. Die MAC-ADDRESS kann dem Baustein IL_PNIOGetLiveList entnommen werden.

IL_PNIOGetLogMsg

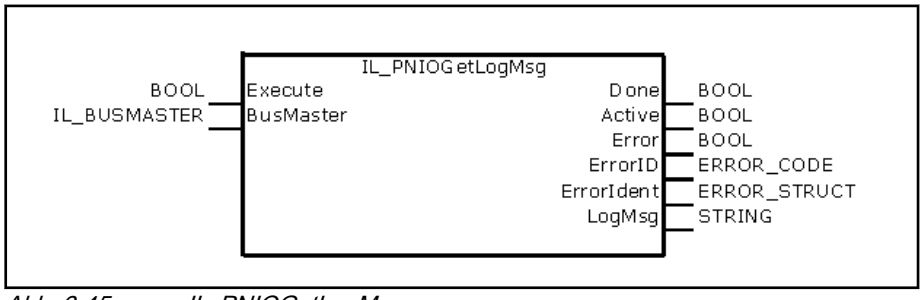


Abb. 6-45: IL_PNIOGetLogMsg

Der Funktionsbaustein holt eine Loggermeldung zum Gerätetausch aus dem Profinet IO-Controller.

Es wird die jeweils älteste Meldung geholt. Jede Meldung kann nur einmal geholt werden.

Aufbau der Loggermeldungen:

Diese Loggermeldungen können sowohl im Engineering-System als auch am Baustein IL_PNIOGetLogMsg auftreten.

Jede Loggermeldung besteht aus 3 Teilen:

1. dem Begriff "PN_Topology"
2. der Instanznummer des Controllers 0..5
3. dem eigentlichen Meldungstext

Meldungstext	Erläuterung
new topology file stored	Eine neue XML-Datei mit Topologie- Informationen wurde angelegt
new topology list generated	Neue Topologie-Informationen berechnet
topology file deleted	Die Topologieinformationen wurden gelöscht. z.B. beim Löschen des Projektes auf der Steuerung
read topology file error	Es konnte keine Datei mit Topologieinformationen gelesen werden.
fragmentary; DEVNAME not configured	Das Device mit dem Namen DEVNAME ist in der Anlagenhierarchie vorhanden, jedoch nicht in der Konfiguration des Engineering-Systems. Das kann, abhängig vom Aufbau des Netzes und davon, welches Gerät getauscht wird, dazu führen, dass das Ersatzgerät nicht "benannt" werden kann.
device DEVNAME not in XML	Das konfigurierte Device DEVNAME fehlt in der Topologie Datei
missing device in config DEVNAME	Das Device DEVNAME aus der Topologie Datei fehlt in der Konfiguration.
check device for DEVNAME	Für das fehlende Device DEVNAME wurde ein anderes Device gefunden. Dieses wird jetzt auf Kompatibilität überprüft.
not fitting device Type : DEVICETYPE	Die Kompatibilitätsprüfung ergab einen unpassenden DeviceType-String
not fitting VendorID : XXX-YYY	Die Kompatibilitätsprüfung ergab eine unpassende Vendor-ID XXX. Erwartet wird die ID YYY
not fitting DeviceID : XXX-YYY	Die Kompatibilitätsprüfung ergab eine unpassende Device-ID XXX. Erwartet wird die ID YYY
Success naming DEVNAME	Ein Device wurde erfolgreich mit dem Namen DEVNAME benannt
Error naming DEVNAME	Das Benennen des Devices DEVNAME war nicht erfolgreich

Tab. 6-4: Loggermeldungen

Beispiel: PN_Topology 0: new topology file stored

IL_PNIOStoreTopology

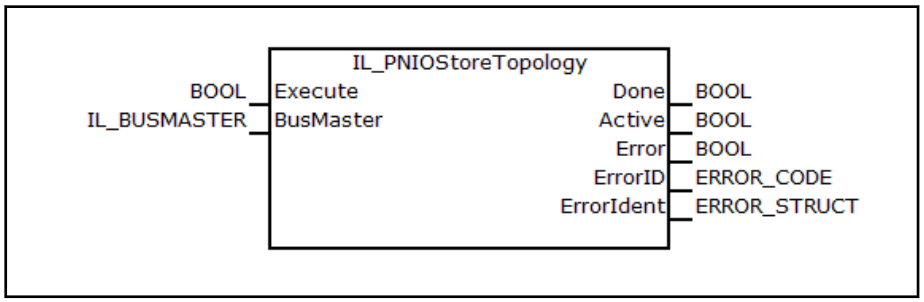


Abb. 6-46: IL_PNIOStoreTopology

Der Funktionsbaustein startet das Generieren der aktuellen Netzwerktopologie-Datenbasis, wenn alle konfigurierten IO-Devices diagnosefrei vorhanden sind.

Die Topologie wird in einer XML-Datei abgelegt.

IL_PNIORestoreTopology

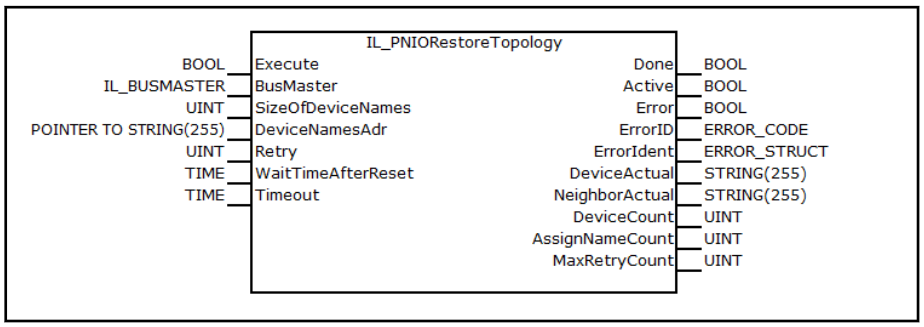


Abb. 6-47: IL_PNIORestoreTopology

Der Funktionsbaustein versucht die gespeicherte Topologie des PNIO Controllers wiederherzustellen (z.B. Anwendungsfall "Kreuztausch" von PN-Devices). Der FB kann auch zur Überprüfung der Topologie-Information verwendet werden, d.h. ob die "Nachbarschafts-Informationen" von allen konfigurierten PN-Devices ein komplett verbundenes Netzwerk bilden.

Einschränkungen:

- Änderungen an der Netzwerk-Verkabelung werden nicht automatisch erkannt. Soll in einem bestehenden Netzwerk umgebaut werden, muss im Anschluss entweder das Projekt neu geladen werden oder mittels RIL_ProfinetIO.library, IL_PNIOStoreTopology eine erneute Ermittlung der Topologie angestoßen werden.
- Die DAP (DeviceAccessPoint) Modul-Ident-Number innerhalb eines Devices (DeviceID) wird derzeit nicht geprüft. Dieser Wert des Ersatzgerätes ist erst nach dem vollständigen Anlauf des Devices am Profinet IO ermittelbar.
- Der IO-Ausbau von modularen Devices ist nicht Teil der Topologiedaten. Auch diese Werte können erst nach dem vollständigen Anlauf des Devices am Profinet ermittelt werden.
- Wenn ein Gerät fehlt wird **ein** erreichbarer Nachbar dieses Gerätes gesucht. Hat dieser Nachbar dann ein passendes Gerät in seinen LLDP-Informationen wird dieses benannt. Das hat den Effekt, dass dieses Gerät an seinem anderen Port auch ‚falsch' verbunden sein kann und trotzdem benannt wird.

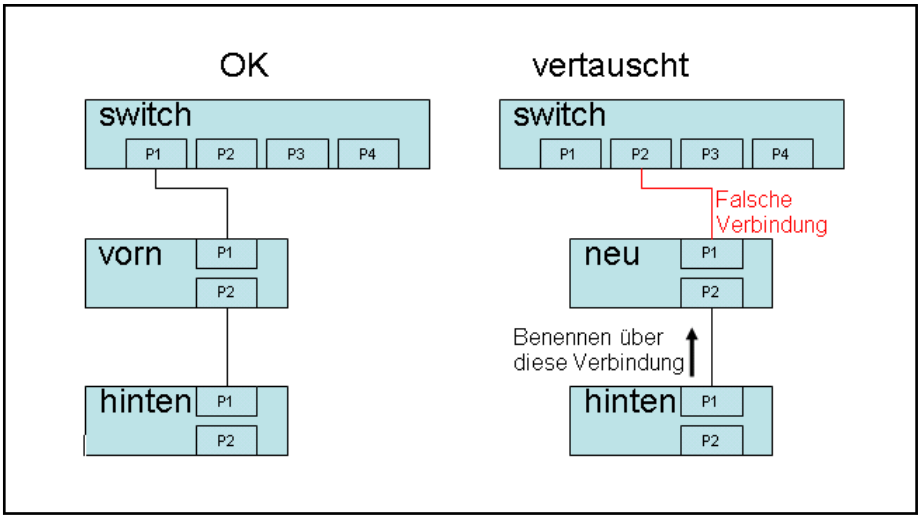


Abb. 6-48: Mögliche Probleme beim Benennen eines Devices

6.6 Profinet IO, Device (Steuerung als Slave)

6.6.1 Profinet IO Device, konfigurieren

Profinet IO Device - Geräteeditor

Ein Profinet IO Device kann bei XM2- und VPx-Steuerungen an einem RT-Ethernet Erweiterungsmodul konfiguriert werden.

Öffnen sie hierzu im Projektexplorer den Knoten des RT-Ethernet-Erweiterungsmoduls und öffnen Sie das Kontextmenü des unterlagerten Knotens.

Wählen Sie im Kontextmenü des Knotens den Befehl "Gerät setzen" und selektieren Sie in der Auswahlbox "Profinet IO Device XM".

Das Erweiterungsmodul ist jetzt konfiguriert.

☞ Bei XM4-Steuerungen kann auch die Onboard-Schnittstelle als Profinet IO Device gesetzt werden.

Selektieren Sie hierzu im Projektexplorer den Knoten der Onboard-Schnittstelle und wählen Sie den Befehl "Gerät setzen".

Selektieren Sie in der Auswahlbox "Profinet IO Device XM", die Onboard-Schnittstelle ist jetzt konfiguriert.

Die Kopplung zwischen der Controllersteuerung und dem Device erfolgt über Abbildspeicher, deren Inhalt ausgetauscht wird (siehe Profinet IO Device, Controller-Device-Kopplung auf Seite 133).

Klicken Sie zum Öffnen des Editorfensters im Projektexplorer doppelt auf das Profinet IO Device-Objekt.

In den Dialogen werden Sie über die Konfiguration der Device-Seite des Profinet IO-Busses informiert und können sie gegebenenfalls modifizieren.

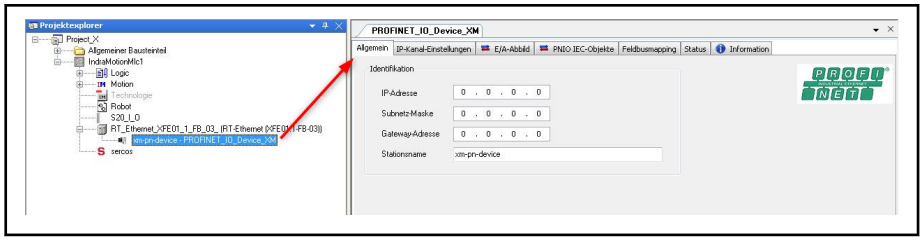


Abb. 6-49: Profinet IO Device - Registerkarte 'Allgemein'

Jedes konfigurierte Profinet IO Device benötigt 8,6 kB der persistenten Retain-Daten des IEC-Programms (ab Kompatibilitätsmodus 14V22).

Um die persistenten Retain-Daten anzulegen gehen Sie wie folgt vor:

1. Fügen Sie der Applikation eine "Globale Persistente Variablenliste ..." hinzu (Defaultname "PersistentVars").
2. Führen Sie den Befehl "Übersetzen" aus (Menüaufruf: **Erstellen** ▸ **Übersetzen**, alternativ **Shift** + **F11**)
3. Selektieren Sie die eben hinzugefügte Variablenliste im Projektexplorer.

   Im IndraWorks Menü wird die Persistente Variablenliste angezeigt.
4. Öffnen Sie das Menü der Variablenliste und wählen Sie den Befehl "Alle Instanzpfade hinzufügen".

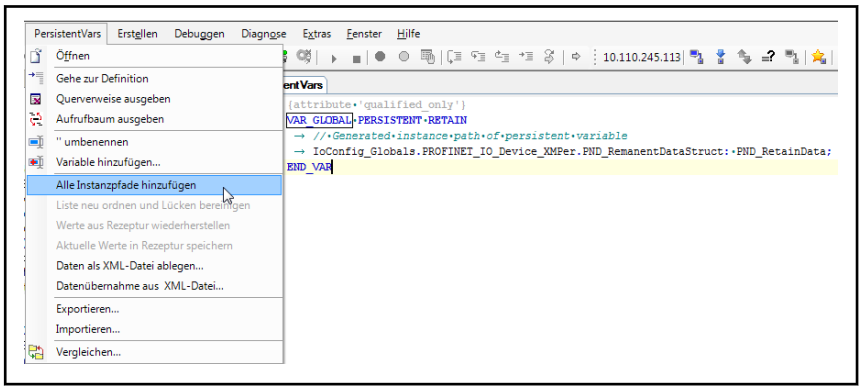


Abb. 6-50: Steuerung als Profinet IO Device - persistente retain Daten

Hiermit wird für jedes konfigurierte Profinet IO Device ein Eintrag vom Typ `PND_RetainData` angelegt.

Siehe:

Konfigurationsdialoge für eine Steuerung als Profinet IO Device

- Registerkarte 'Allgemein' auf Seite 128
- Registerkarte 'Feldbusmappingauf Seite 129
- Registerkarte 'PNIO Parameter' auf Seite 129, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'E/A-Abbild' auf Seite 129
- Registerkarte 'Statusauf Seite 129
- Registerkarte 'Information' auf Seite 130

Registerkarte 'Allgemein'

Für alle Einstellungen im Dialog gilt, dass es von der Gerätebeschreibung abhängt, ob die Einstellungen editierbar sind und welche Werte vorgegeben und möglich sind.

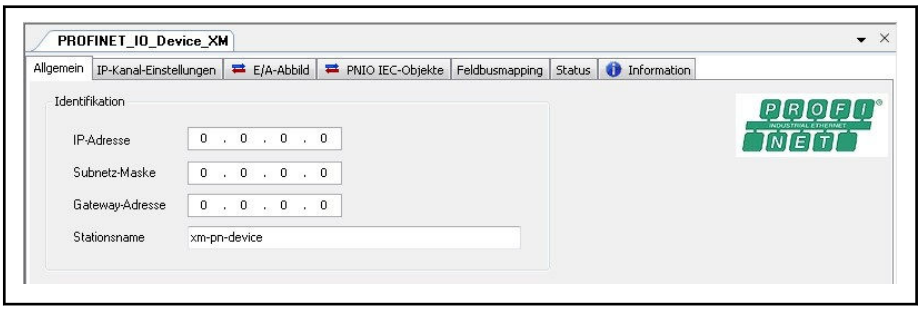


Abb. 6-51: Profinet IO Device - Registerkarte 'Allgemein'

Stationsname

Stationsname: Das Feld enthält den Standardeintrag.

Das Device wird durch einen subnetzweit eindeutigen Device-Namen identifiziert.

IP-Parameter

Die Felder zu "IP Adresse", "Subnetzmaske", "Default Gateway" sind leer zu lassen.

Die Einträge in diese Felder müssen manuell mit dem dazugehörigen Modul zur Ankopplung eines Profinet IO Devices des Profinet IO Controllers, wie im Bild unten gezeigt, synchronisiert werden.

Entscheidend ist dabei der Stationsname.

☞ *Der Stationsname darf aus folgenden Zeichen bestehen:*

- Kleinbuchstaben ohne Umlaute "a" bis "z".

  Großbuchstaben ohne Umlaute "A" bis "Z" sind zulässig, werden aber intern in Kleinbuchstaben gewandelt.

  Die Stationsnamen sind nicht case-sensitiv.
- Ziffern "0" bis "9" (dürfen nicht am Anfang des Namens stehen).
- Sonderzeichen Bindestrich "-" und Punkt "." (dürfen weder am Anfang noch am Ende des Namens stehen).
- Der Name darf max. 240 Zeichen lang sein. Bei manchen Geräten / Herstellern gilt die Beschränkung 127 Zeichen.

  Der Name kann mittels Punkten "." in Teilstrings gegliedert werden. Ein solcher Teilstring darf max. 63 Zeichen lang sein.

  Der Name darf nicht mit der Zeichenfolge "port-xyz" (mit x,y,z = 0...9) beginnen.

  Der Name darf nicht die Form "a.b.c.d" (mit a,b,c,d = 0...999) haben.

☞ Vergeben von IP-Adressparametern kann zu Verbindungsproblemen führen, nur einstellen wenn Vergabe über Profinet IO Controller nicht möglich!

Registerkarte 'IP-Kanal-Einstellungen'

Siehe Kap. 4.6 "IP-Kanal bei PROFINET IO & EtherNet/IP (netX)" auf Seite 33.

Registerkarte 'Feldbusmapping'

Das Feldbusmapping ermöglicht den Zugriff auf Variablen der Steuerung von einem übergeordneten Feldbusmaster aus.

Wenn man auf eine Variable der Steuerung von einem übergeordneten Master aus zugreifen will, dann muss dieser Variablen zuerst eine Adresse zugeordnet werden (Mapping). Über diese Adresse kann der übergeordnete Master über azyklische Dienste auf die Daten der Steuerung zugreifen.

Um das Feldbusmapping nutzen zu können, muss die Steuerung als entsprechender Feldbusslave konfiguriert sein.

Details siehe Feldbusmapping, Zugriff vom Feldbusmaster auf Variablen des Feldbusslaves auf Seite 24.

Registerkarte 'PNIO Parameter'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor**.

Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

Registerkarte 'E/A-Abbild'

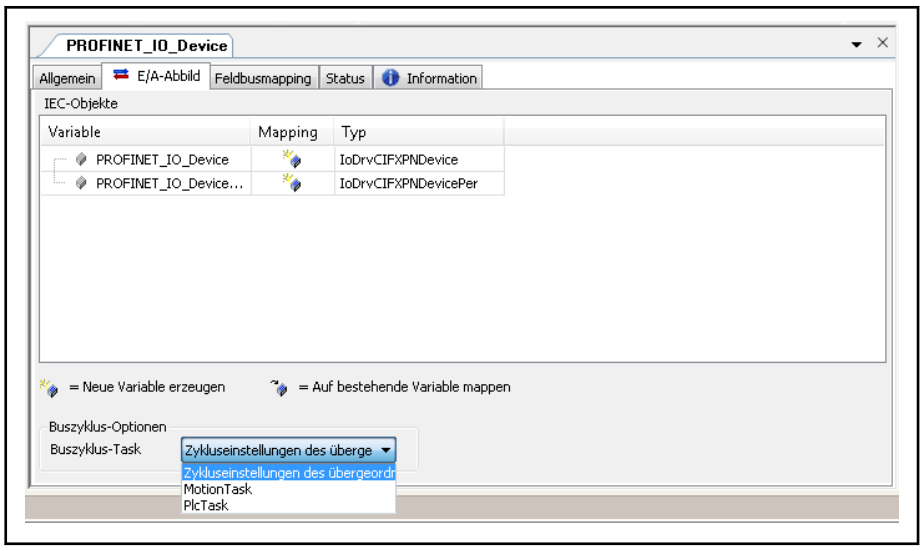


Abb. 6-52: Profinet IO Device - Registerkarte 'E/A-Abbild'

IEC-Objekte: Mit der Definition des Profinet IO Controllers werden die Bibliotheken "IoDrvCIFXPNDevice.library" bzw. RIL_ProfinetIODevice.library automatisch angezogen. Der zur Realisierung des Profinet IO Devices erforderliche Speicherraum kann somit definiert werden.

Buszyklus-Optionen: Durch die Auswahl einer Buszyklus-Task kann der Zyklus des Abbildtausches für den Profinet IO Controller an eine spezielle Task gebunden werden. Es ist sinnvoll in dieser Task auch die IO-Daten des Controllers zu verarbeiten.

Registerkarte 'Status'

Die Registerkarte "Status" im "Profinet IO-Geräteeditor" zeigt Status-Informationen (z. B. "Running", "Stopped") und spezifische Diagnosemeldungen aus dem Gerät. Zusätzlich werden die Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt.

Die Ausgabe erfolgt als hexadezimaler Wert, der sich aus den gesetzten Status-Flags ergibt (siehe Dokumentation "Protocol Interface Manual" für den Profinet IO).

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

6.6.2 Profinet IO Device, Koppelmodule hinzufügen

Allgemeines

Die Schnittstelle eines Profinet IO Device zum Profinet IO Controller ist als gemeinsamer Speicherraum ausgeführt. Dieser kann sich in seiner Größe bedarfsabhängig unterscheiden.

Der Koppelbereich zum Datenaustausch mit dem Profinet IO Controller wird in Form von Ein- und Ausgangsmodulen am Profinet IO Device-Objekt festgelegt.

Hierzu können Sie im Kontextmenü des Profinet IO Device-Objekts über **Hinzufügen ▸ Gerät ▸ ...** entsprechende Ein- und Ausgangsgeräte hinzufügen.

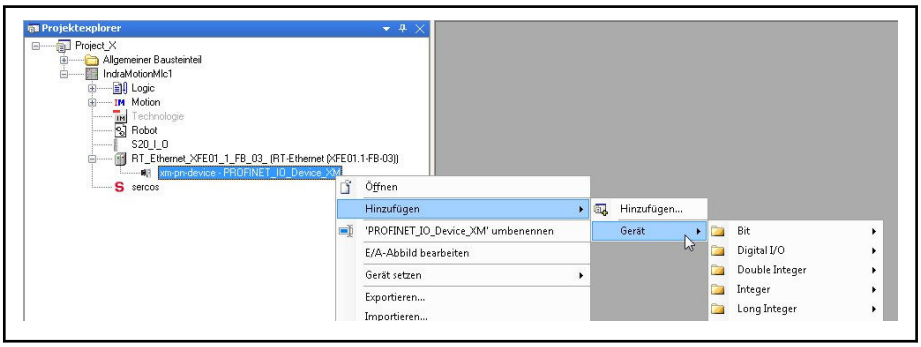
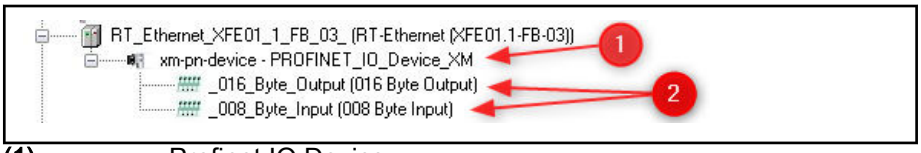


Abb. 6-53: Profinet IO Device: Gerät hinzufügen

Der Datenaustausch zwischen Controller und Device erfolgt über gemeinsame Speicherbereiche, wobei die Ausgänge des einen Eingängen des anderen und umgekehrt sind.

Die Kontrolle, notfalls der manuelle Abgleich der Module des Devices zu den Modulen des Controller, ist erforderlich.



(1) Profinet IO Device
(2) Koppelmodule für den Datenaustausch zwischen den Steuerungen.

Abb. 6-54: Profinet IO Device mit Speichermodulen zur Ankopplung eines Profinet IO Controllers über ein Funktionsmodul

Klicken Sie zum Öffnen der Dialoge zur Konfiguration der unterhalb des Profinet IO Device-Objekts eingefügten Koppelmodule doppelt auf das jeweilige Modul-Objekt.

Alternativ können Sie die Dialoge über das Kontextmenü des Moduls öffnen **Öffnen**.

In dem Konfigurationsdialog der Profinet IO Device-Koppelmodule werden folgende Register angeboten:

Konfigurationsdialoge für Koppelmodule eines Profinet IO Devices zur Ankopplung eines Profinet IO Controllers

- Registerkarte 'Allgemein' auf Seite 131
- Registerkarte 'PNIO Module Parameter' auf Seite 131, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'PNIO Module E/A-Abbild' auf Seite 131
- Registerkarte 'Status' auf Seite 132
- Registerkarte 'Information' auf Seite 132.

Registerkarte 'Allgemein'

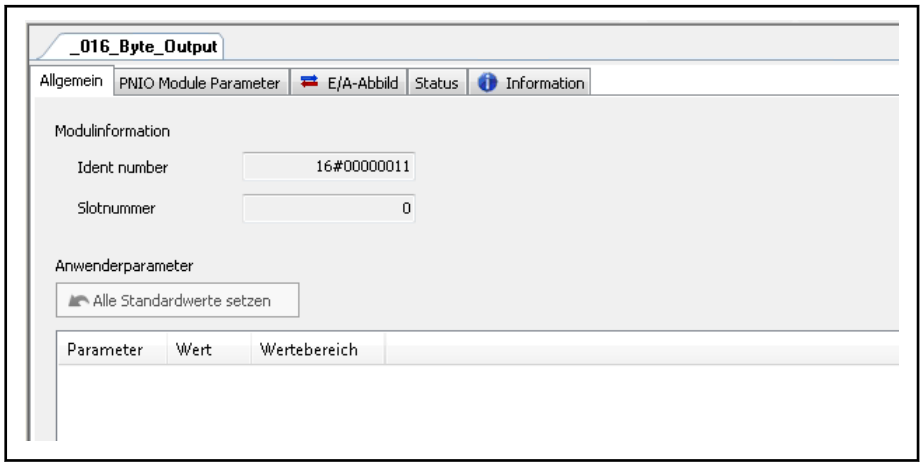


Abb. 6-55: Profinet IO Koppelmodule - Registerkarte 'Allgemein'

Modulinformation

- **Ident-Nummer**: Kennung des Moduls, aus der Gerätebeschreibung
- **Slot-Nummer**: 0 für Austausch-Speicherabbild

Anwenderparameter

keine.

Registerkarte 'PNIO Module Parameter'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor**.

☞ Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

Der Dialog gibt Auskunft über Lage und Größe der Parameter.

Registerkarte 'PNIO Module E/A-Abbild'

Die Registerkarte dient der Zuordnung der Modul-Ein- und Ausgänge zu Variablen, die als lokale oder globale Variablen in den einzelnen POUs verwendet werden können.

Im Online-Modus wird der aktuelle Wert der Variablen angezeigt.

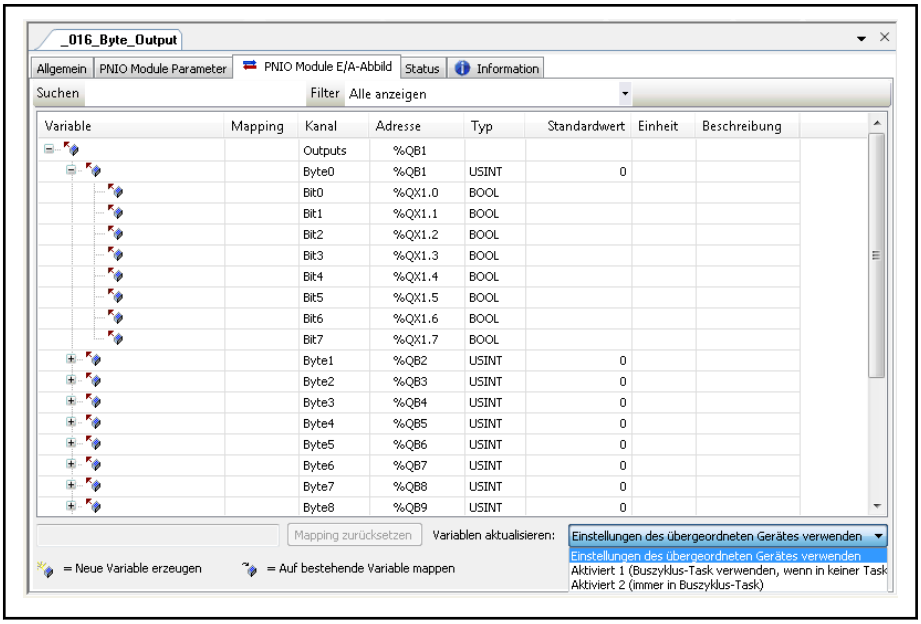


Abb. 6-56: Profinet IO-Koppelmodul - Registerkarte 'PNIO Module E/A-Abbild'

Kanäle: Der Bereich "Kanäle" enthält die Variablen des E/A-Koppelmoduls, hier der Ausgänge.

Die Zuordnung ist unter Mapping der Feldbus-Ein- und Ausgänge auf Seite 19 beschrieben.

Mapping zurücksetzen: Über diese Schaltfläche können die Mapping-Einstellungen auf die durch die Gerätebeschreibungsdatei definierten Standardwerte zurückgesetzt werden.

Variablen aktualisieren: Festlegung für das Modul , ob die E/A-Variablen in der Buszyklus-Task aktualisiert werden.

Der Default-Wert ist in der Gerätebeschreibung definiert.

- **Einstellungen des übergeordneten Geräts verwenden:** Aktualisierung gemäß den Einstellungen des übergeordneten Geräts.
- **Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet):** Die E/A-Variablen werden in der Buszyklus-Task aktualisiert, falls sie in keiner anderen Task verwendet werden
- **Aktiviert 2 (immer im Buszyklus-Task):** Alle Variablen werden in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

Registerkarte 'Status'

Die Registerkarte "Status" im "Profinet IO-Geräteeditor" zeigt Status-Informationen (z. B. "Running", "Stopped") und spezifische Diagnosemeldungen aus dem Gerät. Zusätzlich werden die Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt.

Die Ausgabe erfolgt als hexadezimaler Wert, der sich aus den gesetzten Status-Flags ergibt (siehe das Dokument "Protocol Interface Manual" für den Profinet IO).

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

6.6.3 Profinet IO Device, Controller-Device-Kopplung

Allgemeines

Zur Kopplung einer als Profinet IO Device zu betreibenden Steuerung steht in der **Bibliothek** ▸ **Peripherie** ▸ **ProfinetIO** ▸ **SPS** Devices zur Verfügung, die die Ankopplung nachstehender Steuerungen erlaubt:

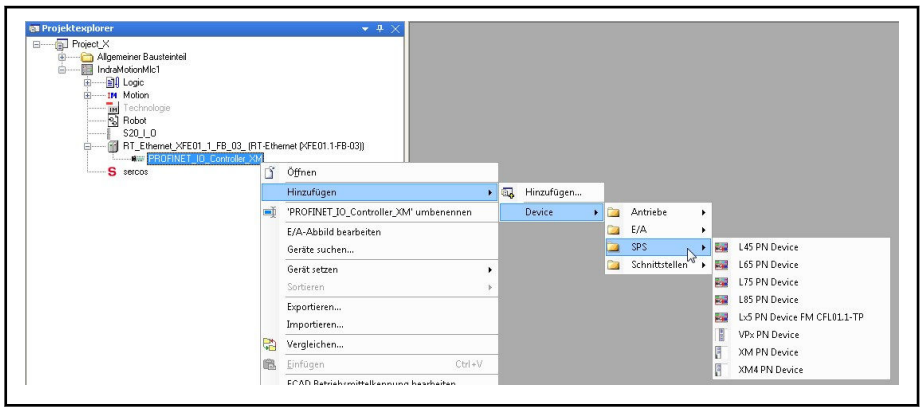
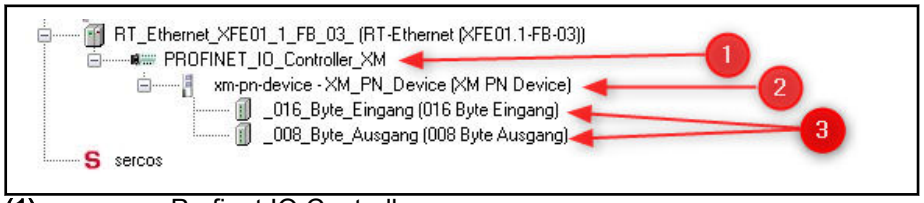


Abb. 6-57: Profinet IO Controller Auswahl des Moduls zur Ankopplung einer Profinet IO Device-Steuerung

- **XM PN Device** für die Steuerung XM2 über deren Erweiterungsmodul "RT_Ethernet_XFE01.1-FB-03"

Durch das Hinzufügen des Moduls zur Ankopplung einer Profinet IO Device-Steuerung im Projektnavigator erfolgt die Aktivierung einer Firmwarekomponente im Controller zur Verbindung mit einem Profinet IO Device über das Profinet.

Jedes dieser Module besitzt in der Gerätebibliothek eine Auswahl von (Unter-)Modulen, die im Abbildspeicher des Controllers Eingangs- und Ausgangsvariablen zum Datenaustausch zwischen der Profinet IO Controller- und der Profinet IO Device-Steuerung frei schalten.



(1) Profinet IO Controller
(2) Modul zur Ankopplung
(3) Abbild-Speichermodule für den Datenaustausch zwischen den Steuerungen

Abb. 6-58: Profinet IO Controller mit Modul zur Ankopplung eines Profinet IO Devices über ein Funktionsmodul

Konfigurationsdialog Profinet IO, Controller-Device-Kopplung

- Registerkarte 'Allgemein' auf Seite 134
- Registerkarte 'IOxS' auf Seite 136
- Registerkarte 'PNIO Parameters' auf Seite 137, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'Status' auf Seite 137

- Registerkarte 'Informationauf Seite 137.



Die Dialoge des Abbildspeichers entsprechen denen auf der Seite des Devices (Profinet IO Device, Koppelmodule hinzufügen auf Seite 130).

Registerkarte 'Allgemein'

Klicken Sie zum Öffnen des Editorfensters im Projektexplorer doppelt auf das Profinet IO Modul zur Ankopplung eines Profinet IO Devices.

Für alle Einstellungen im Dialog gilt, dass es von der Gerätebeschreibung abhängt, ob die Einstellungen editierbar sind und welche Werte vorgegeben und möglich sind.

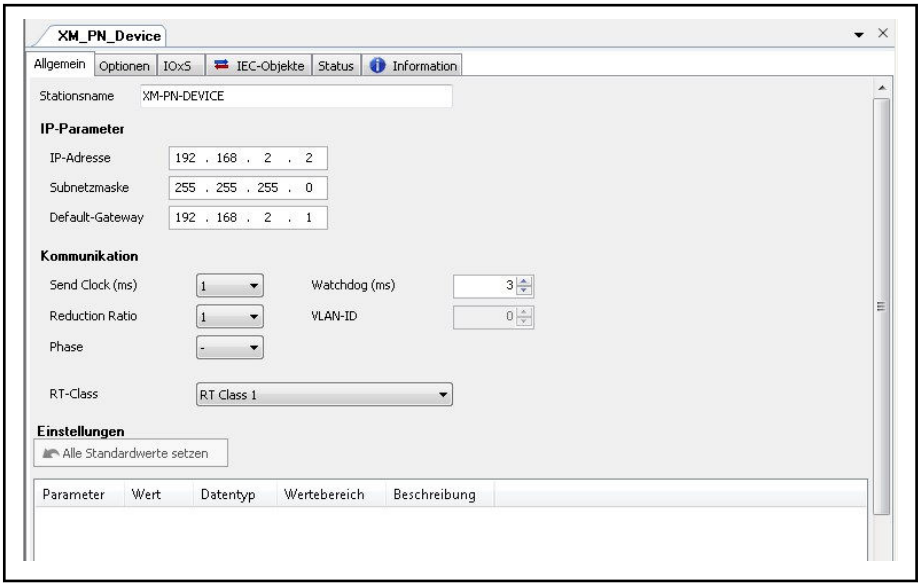


Abb. 6-59: Profinet IO, Controller-Device-Kopplung - Registerkarte 'Allgemein'

Stationsname

- Stationsname: Voreinstellung "XM-PN-DEVICE"

Der Stationsname darf aus folgenden Zeichen bestehen:

- Kleinbuchstaben ohne Umlaute "a" bis "z".

  Großbuchstaben ohne Umlaute "A" bis "Z" sind zulässig, werden aber intern in Kleinbuchstaben gewandelt.

  Die Stationsnamen sind nicht case-sensitiv.
- Ziffern "0" bis "9" (dürfen nicht am Anfang des Namens stehen).
- Sonderzeichen Bindestrich "-" und Punkt "." (dürfen weder am Anfang noch am Ende des Namens stehen).
- Der Name darf max. 240 Zeichen lang sein. Bei manchen Geräten / Herstellern gilt die Beschränkung 127 Zeichen.

  Der Name kann mittels Punkten "." in Teilstrings gegliedert werden. Ein solcher Teilstring darf max. 63 Zeichen lang sein.

  Der Name darf nicht mit der Zeichenfolge "port-xyz" (mit x,y,z = 0...9) beginnen.

  Der Name darf nicht die Form "a.b.c.d" (mit a,b,c,d = 0...999) haben.

Jedes Gerät muss einen im Subnetz eindeutigen Stationsnamen erhalten, da dieser für bestimmte Funktionalitäten im Netzwerkbetrieb benötigt wird!

IP-Parameter

Die folgenden Adressen müssen zur Identifizierung des Controllers der aktuellen Umgebung entsprechend konfiguriert werden.

- IP-Adresse: Voreinstellung "192.168.2.2", leitet sich von der Controller-Adresse und den benachbarten Devices / Busklemmen ab.

Beachten Sie bei der Anpassung der IP-Adresse auch die aktuellen Adresseinstellungen für Devices, die über einen Scan-Vorgang in die Konfiguration eingefügt werden, um Doppelbelegungen zu vermeiden!

Siehe "Default Slave IP-Parameter" auf Seite 90.

- Subnetzmaske: Voreinstellung "255.255.255.0"
- Default-Gateway: Standard-Gateway, Voreinstellung "192.168.2.1" (die Gateway-Adresse ist optional)

Kommunikation

Diese Parameter beschreiben das Zeitverhalten der Kommunikation auf dem Profinet.

Aus dem Produkt der beiden Parametern

`Reduction Ratio` und

`Send clock` ergibt sich die tatsächliche Zykluszeit

(`t = Reduction Ratio * Send clock`),

gemäß der das Device Daten sendet:

- **Send clock (ms)**: Sendezeit in Millisekunden.
- **Reduction ratio**: Faktor zur Berechnung der Zykluszeit aus der Sendezeit.
- **RT Class**: Unterstützt wird aktuell RT_Class_1.

- **VLAN ID**: geben Sie einen Wert zwischen 0 und 4095 für den VLAN typ 802.1Q ein.

  Für den VLAN typ ISL geben Sie einen Wert zwischen 0 und 32767 ein, falls verfügbar.
- **Watchdog**: Zeit (ms): Überwachungszeit, Voreinstellung 1000ms. Mögliche Werte: 0 bis 65535.

  Hat das Profinet IO Device innerhalb dieser Zeit kein Signal mehr vom Controller erhalten, erfolgt eine gerätespezifische Reaktion, z.B. eine Fehlermeldung.

Anwenderparameter

- **Alle Standardwerte setzen:** Dieser Befehl setzt alle gemachten Einstellungen auf die Standardeinstellungen (siehe Spalte "Standardwert") aus der GSDML-Datei zurück.

Aktuell keine Anwenderparameter vorhanden.

Registerkarte 'IOxS'

Die Registerkarte ermöglicht den Zugriff auf IO-Provider / IO-Consumer Daten.



Untermodule ohne eigene E/As stellen ebenfalls ein Provider State für Eingänge zur Verfügung

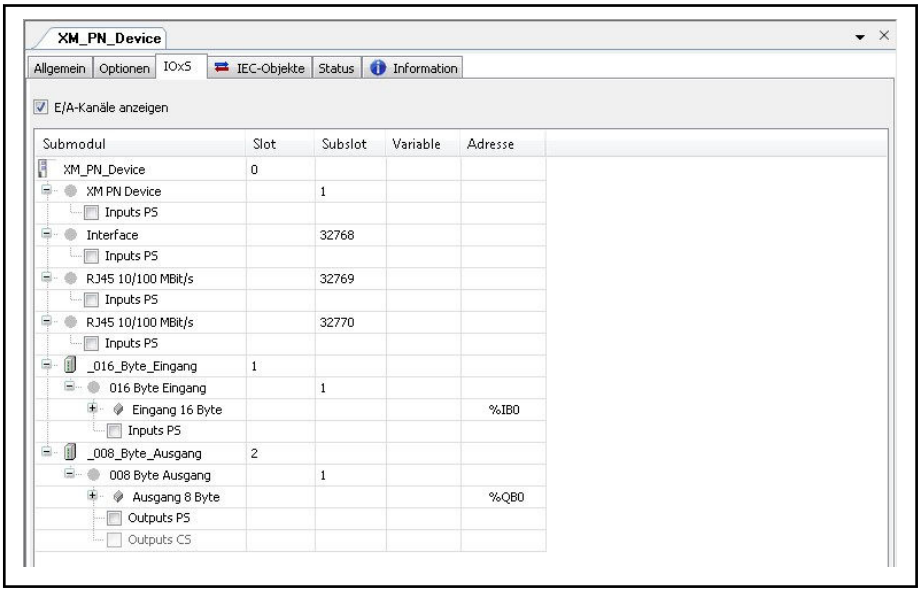


Abb. 6-60: Profinet IO, Controller-Device-Kopplung - Registerkarte 'IOxs'

Wenn die Checkbox aktiviert ist, stellt das zugehörige Modul zusätzliche EAs mit dem Provider State in dem I/O Mapping-Dialog zur Verfügung. Das entsprechende Modul kann durch einen Doppelklick auf das Adressfeld geöffnet werden.

Der Wert hex 80 (Bit 7) des Provider State-Bytes zeigt den Status "OK". Detaillierte Informationen über die Definition dieses Bytes kann in der technischen Spezifikation zum Profinet IO gefunden werden.



Im Falle dass ein IO Provider State für Ausgangsdaten gemappt ist, muss der Ausgangs PS explizit auf "GOOD" (=0x80) gesetzt werden, da ansonsten die Daten vom Gerät nicht übernommen werden und evtl. die Ersatzwerte verwendet werden.

☞ Im Falle dass die Applikation gestoppt wird, werden die nicht gemappten IO Provider-States für Ausgänge auf "bad" gesetzt; gemappte IO Provider-States werden auf ihre konfigurierten Default-Werte gesetzt.

Registerkarte 'PNIO Parameters'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor**.

☞ Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

☞ Bei laufendem Bus ist eine Übertragung geänderter Parameter durch Betätigung der Taste "Parameter schreiben" möglich.

Registerkarte 'Status'

Die Registerkarte "Status" im "Profinet IO-Geräteeditor" zeigt Status-Informationen (z. B. "Running", "Stopped") und spezifische Diagnosemeldungen aus dem Gerät. Zusätzlich werden die Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt.

Die Ausgabe erfolgt als hexadezimaler Wert, der sich aus den gesetzten Status-Flags ergibt (siehe das Dokument "Protocol Interface Manual" für den Profinet IO).

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

6.7 Profinet IO, Tool Calling Interface (TCI)

TCI (Tool Calling Interface) ist eine Schnittstelle zum Aufruf von Device Tools (Windows-Programmen) aus einem Engineering System (z. B. IndraWorks). Die Device Tools dienen zum Parametrieren von Feldbus-Geräten und werden von den Geräteherstellern bereitgestellt. Ein Anwendungsfall für Device Tools ist die proprietäre Berechnung der Prüfsumme (iParCRC) über die individuellen Geräteparameter (iParameter) von PROFIsafe-Geräten, die im Engineering System eingegeben wurden. Die Geräteparameter werden über TCI an das Device Tool übergeben. TCI wurde von "PROFIBUS International / PNO" spezifiziert.

IndraWorks (Engineering System) unterstützt die TCI-Schnittstelle in der Version TCI 1.1 für PROFINET-Geräte in der Conformance Class C2:

- Am Geräte-/Modulknoten im Projektexplorer kann das Device Tool über ein Kontextmenüeintrag aufgerufen werden
- IndraWorks übergibt beim Aufruf Daten (TPF-Datei) an das Device Tool
- TCI wird von den Steuerungsfamilien MLC und MTX unterstützt

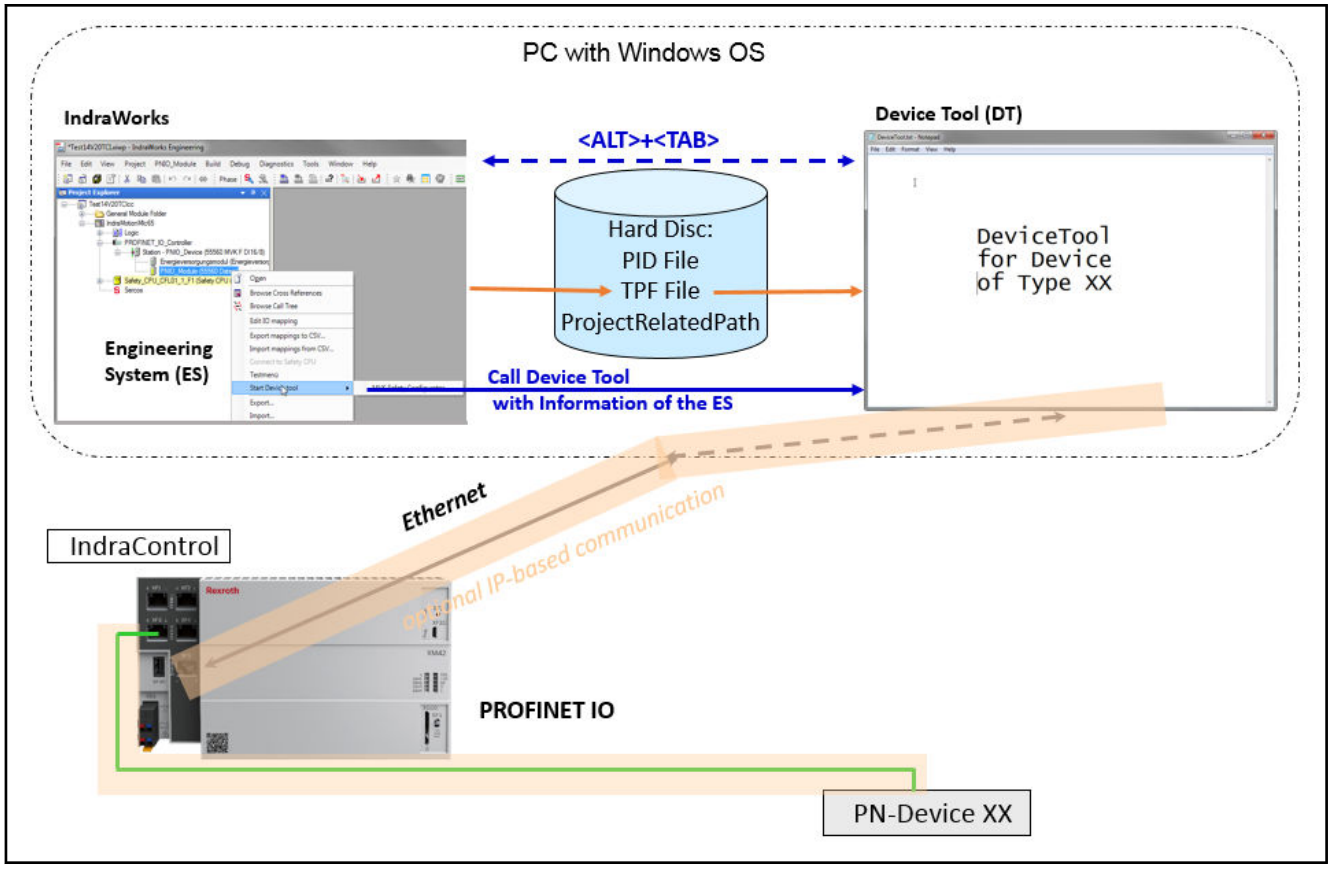


Abb. 6-61: Überblick: TCI in IndraWorks

6.7.1 Handlungsschritte für den Aufruf eines Device Tools aus IndraWorks

1. Installieren Sie das benötigte Device Tool (TCI-fähige Device Tools tragen sich selbständig in der Windows-Registry ein)
2. Starten Sie IndraWorks Engineering
3. Fügen Sie die GSDML-Datei für das PROFINET-Geräte zur Gerätebibliothek von IndraWorks hinzu
4. Legen Sie ein Projekt an. Fügen Sie eine Steuerung hinzu und aktivieren Sie einen PNIO-Controller
5. Fügen Sie das PROFINET-Gerät mit Modulen etc. hinzu und parametrieren Sie diese über den Editor "PNIO-Parameter"
6. Öffnen Sie im Projektexplorer für das PROFINET-Gerät/-Modul das Kontextmenü. Wenn ein Device Tool verfügbar ist, wird der Kontextmenüeintrag "Device Tool starten" angezeigt (IndraWorks durchsucht hierfür automatisch die Windows-Registry, ob für das für das PROFINET-Gerät ein Device Tool installiert ist)
7. Es wird üblicherweise ein weiterer Kontextmenüeintrag mit dem Name des Device Tools angeboten. Durch Betätigung dieses Kontextmenüeintrags wird das Device Tool aufgerufen. Beim Aufruf übergibt IndraWorks dem Device Tool mittels der TPF-Datei Daten

6.7.2 Funktionale Eigenschaften der TCI-Schnittstelle in IndraWorks (Engineering System)

Datenübergabe von IndraWorks (Engineering System) zum Device Tool

IndraWorks übergibt beim Aufruf des Device Tools mittels der TPF-Datei Daten.

Unter dem Pfad "_LinkToCommonAppData" (z. B. "C:\ProgramData\Rexroth \IndraWorks\A") befinden sich im Unterordner "TCI\TPF" die erzeugten TPF-Dateien (TPF[n].xml). Dabei ist "n" eine Nummer, die hochgezählt wird.

Grundlegende Eigenschaften von TCI

☞ TCI erlaubt kein Daten-Rückfluss vom Device Tool zu IndraWorks.

☞ Die Device Tools existieren unabhängig von IndraWorks. Zum Beispiel beim Schließen des IndraWorks-Projekts als auch beim Beenden von IndraWorks werden offene Device Tools nicht geschlossen.

☞ Der Kontextmenüeintrag "Device Tool starten" wird auch für PROFINET-Geräte/-Module angeboten, die in IndraWorks deaktiviert wurden. Die Information, dass das PROFINET-Gerät/-Modul deaktiviert wurde, ist nicht in der TPF-Datei enthalten. Somit ist auch ein (offline) Engineering dieser Geräte / Module mittels Device Tools möglich.

☞ Der Kontextmenüeintrag "Device Tool starten" wird auch für PROFINET-Module angeboten, die mit einem anderen PROFINET-Controller geteilt bzw. von diesem betrieben werden ("Shared Device"). Jedoch ist die Information, dass das PROFINET-Modul ein "Shared Device" ist, nicht in der TPF-Datei enthalten. PROFINET-Module, die von einem anderen PROFINET-Controller betrieben werden, sind in IndraWorks deaktiviert. Verwenden Sie ein entsprechende Namensgebung dieser PROFINET-Module, damit Sie "Shared Devices" im Device Tools erkennen können.

Datenablage

IndraWorks bietet für jedes IndraWorks-Projekt ein Verzeichnis für die Datenablage der Device Tools. Das Verzeichnis wird mit dem Eintrag "ProjectRelatedPath" in der TDF-Datei an das Device Tool übergeben. Jedes Device Tools kann sich dort einen Unterordner anlegen und darin seine Daten ablegen. Das Verzeichnis wird beim Erstellen eines IndraWorks-Archivs mit archiviert.

Der Pfad für ProjectRelatedData ist per Default das Unterverzeichnis "TCI" im Verzeichnis "Public Documents"

Für jedes Projekt wird ein Ordner erstellt, in dem das Device Tool seine Daten ablegen kann. Der Ordner hat den gleichen Namen wie die xiwp-Datei des IndraWorks-Projekts.

Beispiel: Wird ein neues Projekt mit dem Namen "Project189" angelegt, dann trägt die xiwp-Datei den Namen "Project189.xiwp". Der "ProjectRelatedPath" ist dann "C:\Users\Public\Documents\TCI\Project189".

Achten Sie darauf, dass der Name der xiwp-Datei eindeutig ist.

Der Pfad bleibt bestehen und wird NICHT gelöscht, wenn das IndraWorks-Projekt gelöscht wird.

In der TPF-Datei wird als "ProjectRelatedPath" der Pfad "C:\Users\Public\Documents\TCI" angegeben. Im Windows-Explorer kann wie folgt zu diesem Verzeichnis navigiert werden: **Libraries ▸ Documents ▸ Public Documents ▸ TCI**.

Kommunikation vom Device Tool zum PROFINET-Device

IndraWorks (Engineering System) unterstützt die TCI-Schnittstelle in der Version TCI 1.1 für PROFINET-Geräte in der Conformance Class C2, d. h. der TCI-Communication-Server wird nicht unterstützt.

Für die Kommunikation zwischen Device Tool und Gerät kann TCP/IP verwendet werden. Voraussetzung ist, dass das PROFINET-Gerät TCP/IP unterstützt. Gehen Sie in folgenden Schritten vor:

- Tragen Sie die IP-Adresse der Engineering-Schnittstelle der Steuerung in die Routingtabelle des Engineering-PCs ein
- Stellen Sie in den PROFINET-Geräten die IP-Adresse des PNIO-Controllers der Steuerung als Gateway-Adresse ein

6.7.3 Die wichtigsten UseCases im Zusammenhang mit TCI

IndraWorks-Projekt schließen / IndraWorks beenden

Beim Schließen des IndraWorks-Projekts als auch beim Beenden von IndraWorks werden offene Device Tools nicht geschlossen!

Der ProjectRelatedPath bleibt bestehen und wird NICHT gelöscht. Der ProjectRelatedPath existiert eigenständig und die Device Tools können weiterhin auf diesen Pfad Daten ablegen.

IndraWorks-Projekt speichern

Der Inhalt des ProjectRelatedPath wird nicht in die xiwp-Datei von IndraWorks aufgenommen. Der Pfad existiert eigenständig.

IndraWorks-Projekt speichern unter

Es wird eine neuer ProjectRelatedPath gebildet und der entsprechende Pfad angelegt. Der Inhalt (Daten der Device Tools) des bisherigen ProjectRelatedPath wird in den neuen ProjectRelatedPath kopiert. Der Inhalt des ProjectRelatedPath wird nicht in xiwp-Datei von IndraWorks aufgenommen. Der Pfad existiert eigenständig.

Achten Sie darauf, dass der Name der xiwp-Datei eindeutig ist.

Erfolgt das "Speichern unter" auf ein bereits vorhandenes Projekt, dann wird der Inhalt eines evtl. bereits bestehenden ProjectRelatedPath mit den Daten der Device Tools überschrieben.

IndraWorks-Projekt archivieren

Der Inhalt des ProjectRelatedPath (Daten der Device Tools) wird mit in die Archivdatei aufgenommen. Die Projektdaten von IndraWorks und die Daten der Device Tools sind in der IndraWorks-Archivdatei enthalten.

☞ Achten Sie darauf, dass Sie vor dem Archivieren die Daten aus allen offenen Device Tools gespeichert haben. Eine Möglichkeit dies zu erreichen ist, alle Device Tools vor dem Archivieren zu schließen.

IndraWorks-Projekt wiederherstellen

Aus der Archiv-Datei wird der Inhalt des ProjectRelatedPath (Daten der Device Tools) auf dem ProjectRelatedPath entpackt. Die im Archiv enthaltenen Daten der Device Tools stehen jetzt unter dem ProjectRelatedPath zur Verfügung. Die Projektdaten von IndraWorks sind in der xiwp-Datei enthalten.

☞ Es wird empfohlen, alle Device Tools vor dem Wiederherstellen zu schließen, um Zugriffprobleme auf Dateien im ProjectRelatedPath zu vermeiden.

☞ Achten Sie darauf, dass der Name der xiwp-Datei eindeutig ist.

Erfolgt das "Wiederherstellen" auf ein bereits vorhandenes Projekt, dann wird der Inhalt eines evtl. bereits bestehenden ProjectRelatedPath mit den Daten der Device Tools aus der IndraWorks-Archivdatei überschrieben.

6.8 Profinet IO, FAQs

6.8.1 Übersicht

In den nachfolgenden Kapiteln sind Hinweise zu folgenden FAQs beschrieben:

- Wo findet man die Profinet Gerätebeschreibungsdateien (GSDML) für die Steuerungssysteme?, Seite 141
- IP-Routing zwischen Eng.-Port der Steuerung und Profinet Netzwerk funktioniert nicht?, Seite 142
- SPS-Logger Meldung: PN_Topology <station-name> IP x.x.x.x multiple No-of-Peers n, Seite 142
- Profinet IO Controller: Profinet-Device umbenennen funktioniert nicht? (im laufenden Betrieb), Seite 142
- Profinet IO Controller: wird eine Ringtopologie / Redundanz unterstützt?, Seite 143
- Profinet IO Device (Steuerung als Slave): XMx oder VPx als PN-Device wird sporadisch nicht erkannt, Seite 143

6.8.2 Wo findet man die Profinet Gerätebeschreibungsdateien (GSDML) für die Steuerungssysteme?

Siehe www.boschrexroth.com → Produkte → Elektrische Antriebe und Steuerungen → Steuerungssystem auswählen, z.B. Embedded-Steuerungen → XM → XM22 → Downloads.

6.8.3 IP-Routing zwischen Eng.-Port der Steuerung und Profinet Netzwerk funktioniert nicht?

☞ Die IP-Kanal Anbindung wird von dem System MLC unterstützt.
Aktuell besteht keine Unterstützung bei dem System MTX.

Der IP-Kanal ist nur dann funktionsfähig, wenn keine weitere Schnittstelle der Steuerung (z.B. Eng.-Port oder IP-Kanal einer weiteren Feldbus-Schnittstelle) das gleiche Subnetz hat wie die IP-Kanal Konfiguration der PROFINET Schnittstelle.

Im Fehlerfall wird ein Eintrag im SPS-Logger erzeugt, z.B. "netX IP-chan not started - subnet already used" (als ERROR oder als INFO Meldung).

Die Anzeige des SPS-Loggers ist bei Bedarf zu aktivieren:

IndraWorks → Extras → Optionen → IndraLogic 2G → "SPS Logger aktivieren", dann Doppelklick auf die Steuerung im Projektbaum → Registerkarte "Log".

Ab 14V22 muss der IP-Kanal manuell aktiviert werden, weitere Details und Beispielkonfigurationen siehe Kap. 4.6 "IP-Kanal bei PROFINET IO & EtherNet/IP (netX)" auf Seite 33.

6.8.4 SPS-Logger Meldung: "PN_Topology <station-name> IP x.x.x.x multiple No-of-Peers n"

Problem: Die Profinet IO Controller Gerätetauschfunktion hat erkannt, dass die Profinet Topologie nicht eindeutig identifiziert werden kann.

Ursache: Der Auslöser ist typischerweise ein Standard-Switch, der das Protokoll LLDP nicht unterstützt. D.h. eine eindeutige Nachbarschaftserkennung der Profinet-Devices ist nicht möglich und in der Folge ist dann auch die Gerätetauschfunktion des Profinet IO Controllers nicht funktionsfähig.

Lösung:

- Wenn die Gerätetauschfunktion nicht benötigt wird, kann diese Meldung ignoriert werden.

  Alternativ kann die Gerätetauschfunktion über den Funktionsbaustein IL_PNIODeviceReplacement (aus der RIL_ProfinetIO Bibliothek) deaktiviert werden, z.B. direkt nach dem Start vom SPS Programm.

- Wenn die Gerätetauschfunktion genutzt werden soll, dann müssen alle Geräte im Profinet Netzwerk das Protokoll LLDP unterstützten. Switch Geräte sollten idealerweise auch als Profinet-Device eingebunden und konfiguriert werden.

Weitere Details zur Profinet IO Controller Gerätetauschfunktion siehe Kap. 6.5.5 "Profinet IO Controller, Topologie und Gerätetausch" auf Seite 119.

6.8.5 Profinet IO Controller: Profinet-Device umbenennen funktioniert nicht? (im laufenden Betrieb)

Ausgangssituation: Eine Profinet Konfiguration läuft ohne Fehler und ein Profinet-Device soll umbenannt werden (d.h. Stationsname ändern).

Problem: Wenn über den "Geräte suchen" Dialog ein Profinet-Device umbenannt wird, dann funktioniert dieses Umbennen nicht bzw. der Profinet IO Controller benennt das Gerät automatisch wieder mit dem alten Stationsnamen.

Beschreibung: Wenn eine Profinet Konfiguration ohne Fehler läuft, speichert sich der Profinet IO Controller die Topologie automatisch ab (= Gerätetauschfunktion). Das Umbennen eines Profinet-Devices ist dann vergleichbar

wie ein "Gerätetausch", d.h. der Profinet IO Controller benennt das Gerät wieder entsprechend seiner gespeicherten Topologieinformation.

Lösung: Zum Umbenennen eines Profinet-Device gehen Sie wie folgt vor:

1. Den Stationsnamen in der IndraWorks Konfiguration ändern und das Projekt über einen SPS-Download auf die Steuerung laden.

   Hierdurch wird die "alte" Topologieinformation der vorhergehenden Projektkonfiguration automatisch gelöscht.

2. Danach über den Dialog "Geräte suchen" das Umbenennen des Gerätes ausführen.

   Wenn die geänderte Konfiguration wieder ohne Fehler läuft, wird die neue Topologieinformation durch die Gerätetauschfunktion gespeichert.

Der Status der Gerätetauschfunktion kann über den Funktionsbaustein IL_PNIODeviceReplacement (aus der RIL_ProfinetIO Bibliothek) überwacht werden.

Weitere Details zur Profinet IO Controller Gerätetauschfunktion siehe Kap. 6.5.5 "Profinet IO Controller, Topologie und Gerätetausch" auf Seite 119.

6.8.6 Profinet IO Controller: wird eine Ringtopologie / Redundanz unterstützt?

Nein, der Profinet IO Controller unterstützt keine Ringtopologie oder Redundanz. Die Profinet Verkabelung muss immer einer Linie entsprechen. Die zwei Ethernet Ports der Profinet Schnittstelle am Steuerungssystem können bei Bedarf für zwei Linien verwendet werden.

6.8.7 Profinet IO Device (Steuerung als Slave): XMx oder VPx als PN-Device wird sporadisch nicht erkannt

Wenn eine Steuerung, z.B. XMx, als Profinet IO Device konfiguriert wird (Onboard oder Extensionmodul), dann kann in den IndraWorks Dialogen sowohl ein Profinet Stationsname konfiguriert werden als auch die IP-Konfiguration für diese Profinet Schnittstellen eingestellt werden.

⇒ Es wird empfohlen, diese Profinet IP-Konfiguration (IP-Adresse, Subnetz und Gateway) der Profinet IO Device Schnittstelle komplett auf 0.0.0.0 zu setzen.

Ein Profinet IO Device wird nur über den Profinet Stationsnamen adressiert. Die IP-Konfiguration für ein Profinet IO Device wird im Profinet IO Controller konfiguriert und beim Verbindungsaufbau in das Profinet IO Device geschrieben, unabhängig davon, ob das Profinet IO Device evtl. bereits eine IP-Konfiguration hatte.

7 EtherNet/IP

7.1 Features des EtherNet/IP Scanners

Die aktuelle Implementation des Scanners beinhaltet folgende Funktionalitäten:

- Zyklischer Datenverkehr ("implicit messaging")
- Azyklischer Datenverkehr ("explicit messaging")
- EDS-Dateiimport in die Gerätedatenbank (Hinweis: modulare EDS-Dateien werden nicht unterstützt)
- Adapter-bezogener Status über zyklische Daten
- Adapter-bezogene Diagnose über FBs
- Detaildiagnose in der Oberfläche für alle Adapter
- BootP- bzw. DHCP-Mechanismus zur IP-Adressvergabe

Funktion / Eigenschaft	Wert
Maximale Anzahl an Verbindungen ("implicit messaging")	64
Baudrate	100 MBit/s
Auto negotiation/ autocrossing	Ja
Min. Zykluszeit	2 ms
Max. Anzahl zyklischer Eingangsdaten	4096 Byte
Max. Anzahl zyklischer Ausgangsdaten	4096 Byte
Max. Anzahl zyklischer Eingangsdaten pro Adapter-Gerät, abgestimmt mit Adapter-Geräten	480 Byte
Max. Anzahl zyklischer Ausgangsdaten pro Adapter-Gerät, abgestimmt mit Adapter-Geräten	480 Byte

Tab. 7-1: Technische Daten

7.2 Features des EtherNet/IP Adapter

Die aktuelle Implementation des EtherNet/IP Adapters (an den Realtime Ethernet-Schnittstellen) beinhaltet folgende Funktionalitäten:

- Zyklischer Datenverkehr ("implicit messaging")
- Diagnose des Adapters
- Azyklischer Datenverkehr nach Mappingkonzept

Funktion/Eigenschaft	Wert
Max. Anzahl Eingangsdaten, abgestimmt mit Scanner-Geräten	480 Bytes [1])
Max. Anzahl Ausgangsdaten, abgestimmt mit Scanner-Geräten	480 Bytes [1])
IO Verbindung	1 "Exclusive Owner", bis zu 2 "Listener"
IO Verbindungstyp	Zyklisch, min. 2 ms

Funktion/Eigenschaft	Wert
Explicit Messages	*CIP Standard Dienste:* • Set_Attribute_Single • Get_Attribute_Single *Rockwell-spezifische Dienste:* • ReadDataTable • WriteDataTable
UCMM	unterstützt
Max. Anzahl von anwenderspezifische Objekte	20
Max. Anzahl der Verbindungen	8 explicit und implicit Connections
DCHP	unterstützt
BOOTP	unterstützt
Baudrate	10 und 100 MBit/s
Max. Telegrammdaten pro azyklischem Telegramm	450 Byte

[1]) pro Verbindung

Tab. 7-2: Technische Daten

Unterstützte CIP-Klassen

Die CIP Klassen sind in der CIP Spezifikation der ODVA enthalten.

Sie beschreiben, unabhängig von der physikalischen Schnittstelle, z.B. Ethernet, CAN, die Eigenschaften der Objekte (Band 1).

Die physikalische Schnittstelle wird in einer weiteren Spezifikation beschrieben.

Für EtherNet/IP ist das der Band 2, welcher die Adaption von EtherNet/IP an CIP beschreibt.

Die Klassen 1, 2, 4, die in Band 1 ("Common Industrial Protocol") beschrieben sind, werden unterstützt.

Aus dem Band 2 ("EtherNet/IP Adaption of CIP") werden die Klassen 16#F5 und 16#F6 unterstützt.

Für den azyklischen Datenverkehr wird ein „Vendor Specific Object“ Klasse 16#C7 definiert.

Die Klassen aus der unten stehenden Tabelle werden nachfolgend näher beschrieben.

Klasse	Name
16#01	Identity Object
16#02	Message Router Object
16#04	Assembly Object
16#F5	TCP/IP Interface Object
16#F6	Ethernet Link Object
16#C7	Vendor Specific Object

Tab. 7-3: Unterstützte CIP-Klassen

Indentity Object (16#01)

Die Klasse Identity dient dazu, allgemeine Informationen des Adapters bereitzustellen, die diesen eindeutig identifizieren.

Instance	Name	Attribute	Name	Supported Services	
				Get Attribute Single	Get Attribute All
0	Class	1	Revision	Yes	Yes
		2	Max. Instance		
		6	Max. Class Attrib.		
		7	Max. Instance Attrib.		
1	Instance Attributes	1	Vendor ID	Yes	Yes
		2	Device Type		
		3	Product Code		
		4	Major Revision		
			Minor Revision		
		5	Status		
		6	Serial Number		
		7	Product Number		
		8	State		
		9	Conf. Consist. Value		

Abb. 7-1: Klasse „Identity Object“: Klassen-Attribut (Instanz = 0) und Instanz-Attribut (Instanz = 1)

Message Router Object (16#02)

Das „Message Router Object“ stellt Verbindungspunkte in Form von Klassen oder Instanzen bereit, welche einen Client zum Adressieren von Diensten (Lesen, Schreiben) nutzen kann. Diese Nachrichten können sowohl verbindungsorientiert (connected) als auch verbindungslos (unconnected) vom Client an den Adapter gesendet werden.

Für das Message Router Object (Predefined Standard Object 16#02) werden keine Services unterstützt.

Für den azyklischen Datenverkehr wird ein „Vendor Specific Object“ 16#C7 auf dem Message Router definiert. Die Services Get/Set_Attribute_Single werden bei diesem Objekt unterstützt. Und verschiedene „Vendor Specific Services“ können dafür definiert werden (z.B. die Services für Read/Write DataTable, Read/Write Fragmented Data).

Instance	Name	Attribute	Name	Supported Services	
				Get Attribute Single	Set Attribute Single
1	Instance Attributes	1	Einstiegspunkt Tunnelprotokoll	N/A	N/A
2			Variable 1	Yes	Yes
3			Variable 2 (read only)	Yes	No
4			Variable 3	Yes	Yes
...			...	...	...
0xFFFF			Variable 0xFFFF	Yes	Yes

Abb. 7-2: Klasse „Vendor Specific Object (16#C7) “: nur Instanz-Attribut (Instanz = 1, 2, 3, …, 16#FFFF)

Assembly Object (16#04)

Mit Hilfe der Assembly-Klasse lassen sich mehrere auch verschiedenartige Objekte zusammenfassen. Diese können z. B. Ein- und Ausgangsdaten sein. Die herstellerspezifischen Instanzen werden benutzt, um diese Objekte in

verschiedenen Anordnungen bereitzustellen. Hierdurch steht Ihnen ein effizienter Weg zum Austausch von Prozessdaten zur Verfügung.

Instance	Name	Attribute	Name	Supported Services	
				Get Attribute Single	Set Attribute Single
0	Class	1	Revision	Yes	No
		2	Max. Instance		No
1-x	Instance Attributes	3	Data	Yes	Yes
		4	Size		No

Abb. 7-3: Klasse „Assembly Objekt“: Klassen-Attribut (Instanz = 0) und Instanz-Attribut (Instanz = 1)

TCP/IP Interface Object (16#F5)

Das „TCP/IP Interface Object“ stellt die Einrichtung zur Konfiguration der TCP-IP-Netzwerk-Schnittstelle eines Adapters bereit.

Beispiele konfigurierbarer Objekte umfassen die IP-Adresse, Netzwerkmaske und Gateway-Adresse des Adapters.

Instance	Name	Attribute	Name	Supported Services		
				Get Attribute Single	Get Attribute All	Set Attribute Single
0	Class	1	Revision	Yes	Yes	No
		2	Max. Instance			No
1	Instance Attributes	1	Status	Yes	Yes	Yes
		2	Configuration Capability			No
		3	Configuration Control			Yes
		4	Physical Link Object			No
		5	Interface Configuration			No
		6	Host Name			Yes

Abb. 7-4: Klasse „TCP/IP Interface Object“: Klassen-Attribut (Instanz = 0) und Instanz-Attribut (Instanz = 1)

Ethernet Link Object (16#F6)

Das „Ethernet Link Object“ enthält linkspezifische Zähler- und Statusinformationen für eine Kommunikationsschnittstelle vom Typ Ethernet.

Instance	Name	Attribute	Name	Supported Services		
				Get Attribute Single	Get Attribute All	Set Attribute Single
0	Class	1	Revision	Yes	No	No
		2	Max. Instance	Yes	No	No
1	Instance Attributes	1	Interface Speed	Yes	Yes	Yes
		2	Interface Flags	Yes		No
		3	Physical Address	Yes		No
		4	Interface Counters (not yet implemented)	No		No
		5	Media Counters (not yet implemented)	No		No
		6	Interface Control	Yes		Yes

Abb. 7-5: Klasse „Ethernet Link Object“: Klassen-Attribut (Instanz = 0) und Instanz-Attribut (Instanz = 1)

7.3 Features des EtherNet/IP Adapter (Engineering Port)

Die aktuelle Implementation des EtherNet/IP Adapters (Standard Ethernet Schnittstelle / Engineering Port) beinhaltet folgende Funktionalitäten:

- Zyklischer Datenverkehr ("implicit messaging")
- Diagnose des Adapters

Funktion/Eigenschaft	Wert
Max. Anzahl Eingangsdaten	128 Bytes[1)]
Max. Anzahl Ausgangsdaten	128 Bytes
IO Verbindung	1 Exclusive Owner, 1 Listener
IO Verbindungstyp	Zyklisch, min. 5 ms
Explicit Messages	Get_Attribute Set_Attribute
UCMM	unterstützt
Max. Anzahl von anwenderspezifische Objekte	-
Max. Anzahl der Verbindungen	1 explicit und 1 implicit Connection
DCHP	-
BOOTP	-
Baudrate	10 und 100 MBit/s

Tab. 7-4: Technische Daten

7.4 EtherNet/IP, Grundlagen

7.4.1 Allgemeines

Der Feldbus EtherNet/IP basiert auf Ethernet. Unter Beteiligung der ControlNet International (CI), der Open DeviceNet Vendor Association (ODVA), sowie der Industrial Ethernet Association (IEA) wurde das Applikationsprotokoll "Common Industrial Protocol (CIP)" auf Ethernet umgesetzt und als offener Industriestandard zur Verfügung gestellt.

Basierend auf Standard-TCP und -UDP erlaubt EtherNet/IP die Durchgängigkeit zwischen Office Netzwerk und der zu steuernden Anlage.

Die meisten EtherNet/IP-Endgeräte unterstützen DHCP und BootP bei der Vergabe der IP-Adresse. Wie jedes standardkonforme Ethernet eignet sich auch EtherNet/IP nicht für anspruchsvolle Echtzeit-Anwendungen (<1ms), wie z.B. die Steuerung von Servomotoren; die typische Zykluszeit eines EtherNet/IP-Netzwerkes liegt bei etwa 10ms.

1) Die Datenbreite des Koppelbereiches ist auf 8 Byte Eingangs- und 8 Byte Ausgangsdaten plus 4Byte Run Header voreingestellt. Sie lässt sich jeweils über das Kontextmenü "Gerät setzen" verändern.

7.4.2 Begriffe und Abkürzungen

Begriff	Erläuterung
EtherNet/IP	"Ethernet Industrial Protocol": Umsetzung des Applikationsprotokolls "Common Industrial Protocol" (CIP) auf Standard Ethernet (TCP/IP, UDP/IP). EtherNet/IP und CIP werden verwaltet von der ODVA ("Open DeviceNet Vendor Association", www.odva.org)
ENIP, EIP	Abkürzungen für EtherNet/IP
Scanner	Der EtherNet/IP Feldbus-Master wird **Scanner** genannt
Adapter	Die EtherNet/IP Feldbus-Slaves nennt man **Adapter**
Originator	Das Gerät, das die Verbindung aufbaut, wird **Originator** genannt
Target	Das Gerät, zu dem die Verbindung aufgebaut wird, ist ein **Target**
O→T	Datenrichtung vom Originator zum Target
T→O	Datenrichtung vom Target zum Originator
RPI	"Requested Packet Interval": Zykluszeit für die zyklischen I/O-Datentelegramme
Assembly	Daten werden in Form von Assembly Objekten definiert (z.B. zyklische Ein-/Ausgangsdaten oder Konfigurationsdaten)
EDS-Datei	"Electronic Data Sheet": Gerätebeschreibungsdatei für EtherNet/IP Adapter, mittels derer IndraWorks die Adapter-Geräte bekannt gemacht werden
BootP	"Bootstrap Protocol": Vorgänger von DHCP ("Dynamic Host Configuration Protocol"), wird verwendet um Geräten eine IP-Adresse zuzuweisen

Tab. 7-5: Begriffe und Abkürzungen, EtherNet/IP-Feldbus

7.5 EtherNet/IP, Scanner

7.5.1 EtherNet/IP Scanner konfigurieren

Übersicht

Der EtherNet/IP-Konfigurationseditor stellt Dialoge für die Konfiguration von EtherNet/IP Scanner- und Adapter-Geräten bereit, welche im Geräteeditor zusätzlich zu den Standarddialogen angezeigt werden.

Um EtherNet/IP-Geräte als Objekte im Gerätebaum des Projekts einfügen und konfigurieren zu können, müssen die zugehörigen Gerätebeschreibungsdateien (*eds-Dateien der Adapter-Geräte) in der Gerätedatenbank installiert werden.

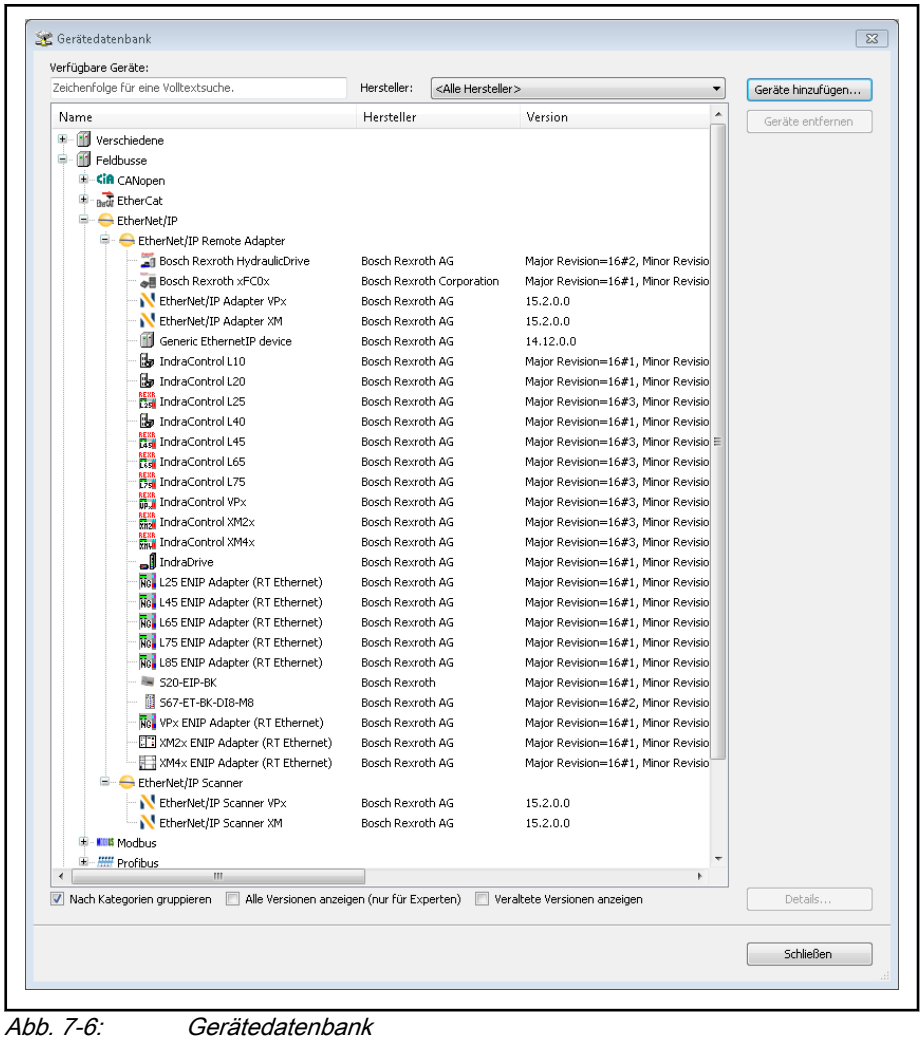


Abb. 7-6: Gerätedatenbank

Scanner hinzufügen

Ein EtherNet/IP Scanner kann im Projektexplorer mit dem Befehl "Gerät setzen" übernommen werden. Mit Einfügen eines EtherNet/IP Scanners im Gerätebaum wird automatisch die zugehörige gerätespezifische Bibliothek im Bibliotheksverwalter eingebunden.

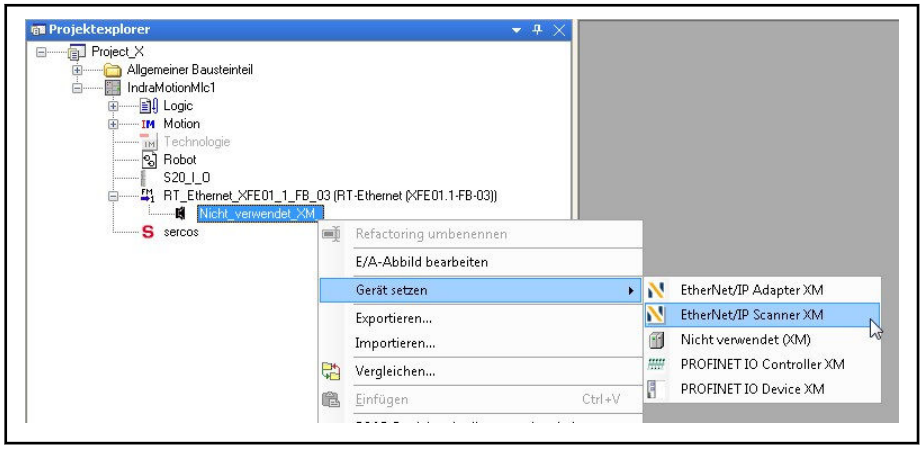


Abb. 7-7: Scanner am Erweiterungsmodul XFE01.1-FB-03 einfügen

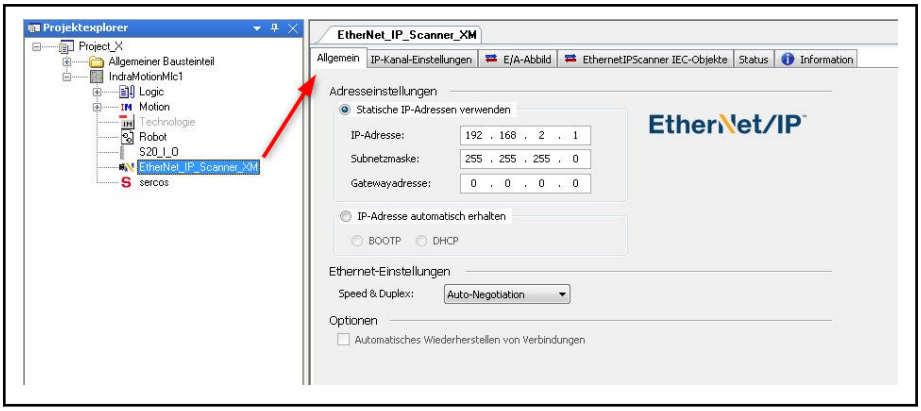


Abb. 7-8: EtherNet/IP Scanner auf der OnBoard Schnittstelle einer XM4-Steuerung

Registerkarte 'Allgemein'

Bei einem EtherNet/IP Scanner können seine IP-Adresse, die Subnet-Maske und die Gateway-Adresse gesetzt werden.

Alternativ kann die IP-Adresse des Scanners auch von einer übergeordneten Steuerung via BootP oder DHCP bezogen werden. Hierzu ist der entsprechende Schalter zu betätigen.

Zusätzlich können noch die Netzwerkeinstellungen für Richtung und Geschwindigkeit vorgegeben werden.

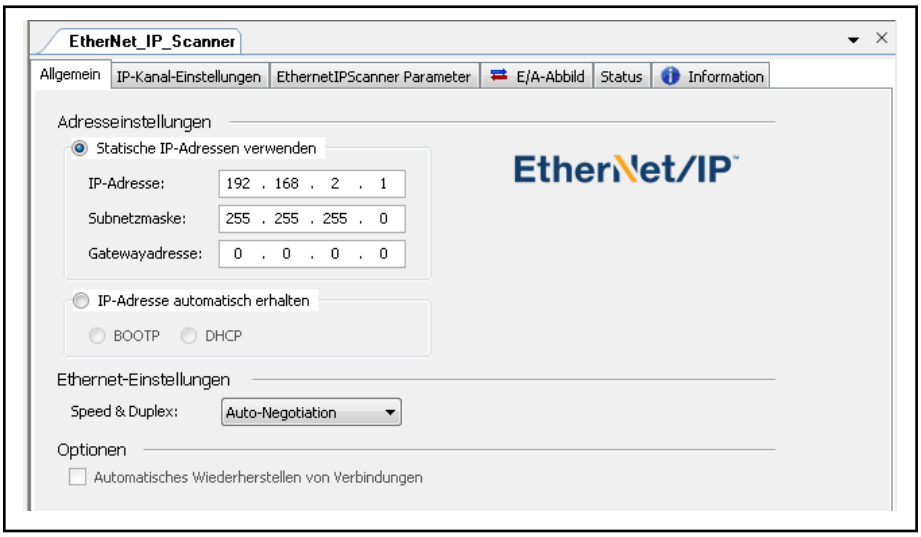


Abb. 7-9: EtherNet/IP Scanner, Registerkarte "Allgemein"

In diesem Fenster müssen die IP-Adresse, die Subnet-Maske und die Gateway-Adresse eingestellt werden. Wäre der Schalter "IP-Adresse automatisch über" eingeschaltet, würde der Scanner seine Adresse von einer übergeordneten Steuerung oder einem BootP/DHCP-Server zugeteilt bekommen.

Die MAC-Adresse des Scanners befindet sich auf dem Gerät.

Registerkarte 'IP-Kanal-Einstellungen'

Siehe "Registerkarte 'IP-Kanal-Einstellungen'" auf Seite 33.

Registerkarte 'EthernetIP Scanner Parameter'

In der Registerkarte "EthernetIP Scanner Parameter" werden die Kommunikationsparameter des EtherNet/IP Scanner angezeigt.

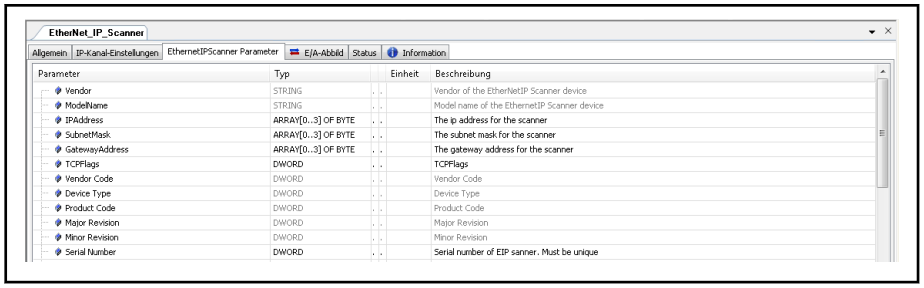


Abb. 7-10: EtherNet/IP Scanner, Registerkarte "EthernetIP Scanner Parameter"

Einige Parameter (schwarze Schrift) können an dieser Stelle geändert werden.

Dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.

Aufruf: IndraWorks Menü **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor**

Eventuelle weitere Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

Registerkarte 'E/A-Abbild'

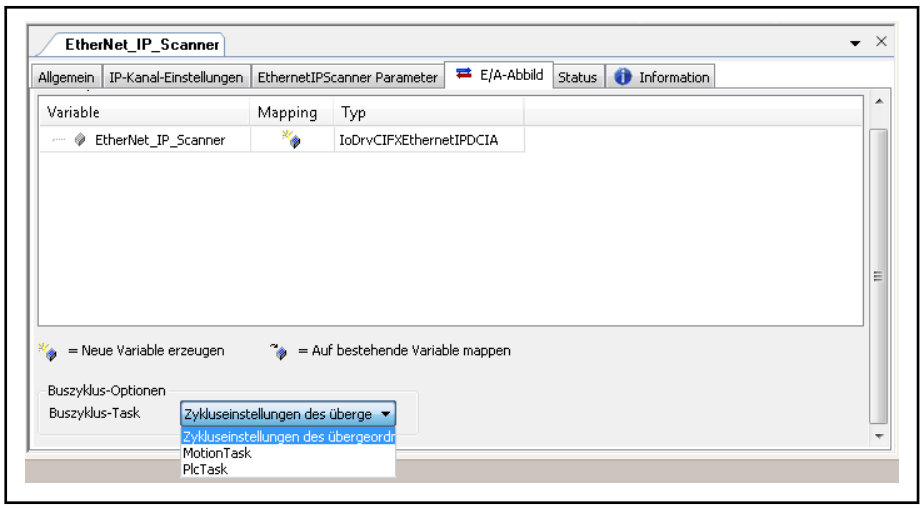


Abb. 7-11: EtherNet/IP Scanner, Registerkarte "E/A-Abbild"

IEC-Objekte: Mit der Definition des EtherNet/IP Scanners werden die Bibliotheken "IoDrvCIFXPROFINETDCIA.library" bzw. RIL_EtherNetIP.library automatisch angezogen. Der zur Realisierung des EtherNet/IP Scanners erforderliche Speicherraum kann somit definiert werden.

Buszyklus-Optionen

Buszyklus-Task: Durch die Auswahl einer Buszyklus-Task kann der Zyklus des Abbildtausches für den EtherNet/IP Scanner an eine spezielle Task gebunden werden. Es ist sinnvoll in dieser Task auch die IO-Daten des Masters zu verarbeiten.

Standardeinstellung: Zykluseinstellung des übergeordneten Busses verwenden

Diese Einstellung bewirkt, dass die in "Buszyklus-Optionen" der Steuerung (Doppelklick auf die konkrete Steuerung im Projektnavigator, SPS-Einstellungen) vorgenommene Taskeinstellung für den konkreten Bus übernommen wird.

Registerkarte 'Status'

Die Registerkarte "Status" im "EtherNet/IP-Editor" zeigt Status-Informationen (z.B. "Running", "Stopped") und spezifische Diagnosemeldungen aus dem Gerät. Zusätzlich werden die Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt.



Abb. 7-12: EtherNet/IP Scanner, Registerkarte "Status" (keine Diagnosemeldung vorhanden)

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

7.5.2 EtherNet/IP Scanner, Adapter hinzufügen

Adapter hinzufügen

Dem Scanner können ein oder mehrere EtherNet/IP Adapter hinzugefügt werden. Dies ist mittels Drag&Drop aus der Gerätebibliothek, über den Menüpunkt **Ethernet_IP_Scanner ▸ Hinzufügen ▸ ...** oder alternativ "Hinzufügen" im Kontextmenü des Scanners möglich.

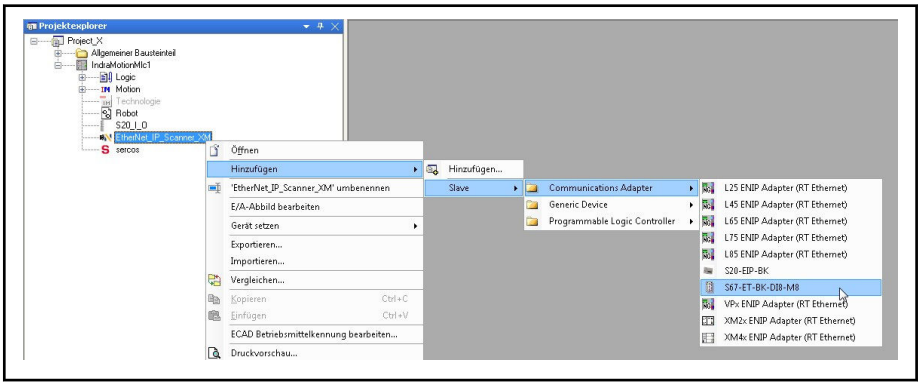


Abb. 7-13: Adapter einfügen

Registerkarte 'Allgemein'

Bei der Konfiguration eines Adapters sind die Zielsystemeinstellungen einzutragen.

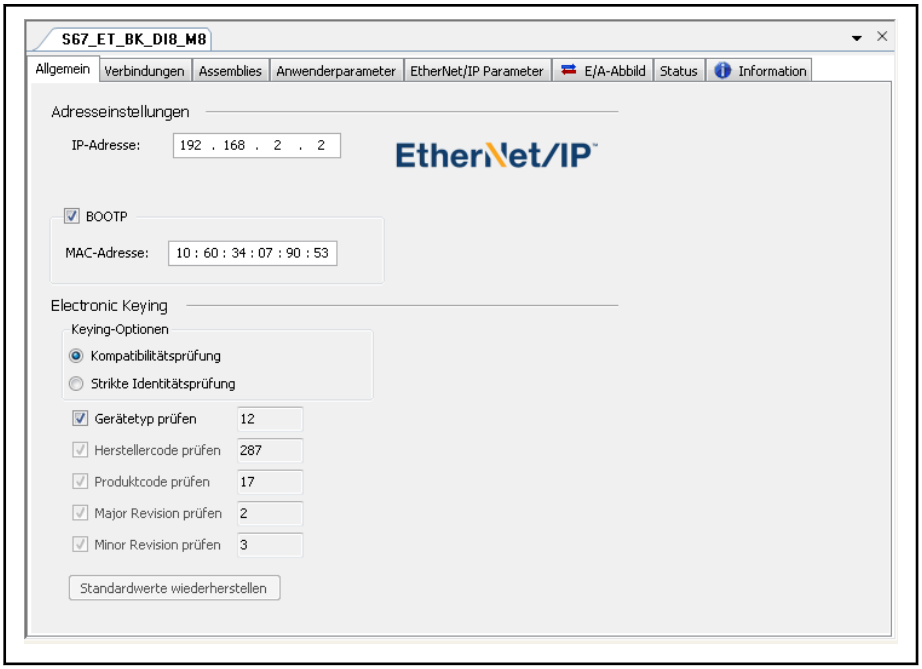


Abb. 7-14: EtherNet/IP Scanner, Adapter - Registerkarte "Allgemein"

Adresseinstellung	Die Adressierung eines Adapters erfolgt über seine IP-Adresse. Diese Adresse ist hier einzugeben
BOOTP	Hat der Adapter noch keine IP-Adresse, kann die konfigurierte IP-Adresse mittels "BOOTP" dem Adapter zugewiesen werden. Die MAC-Adresse des Adapters hier einzugeben. **Hinweis**: Bei einem Gerätetausch ist bei Verwendung von "BOOTP" die MAC-Adresse des neuen Adapters hier im IndraWorks-Projekt zu aktualisieren!
Electronic Keying	Mit dem "Electronic Key" wird ein EtherNet/IP Gerät eindeutig identifiziert. Diese Werte werden aus der Gerätebeschreibung (EDS-Datei) des Adapters übernommen. *Keying-Optionen (ab 14V08)* • **Kompatibilitätsprüfung (empfohlen)**: im Adapter muss nicht der exakt identische Electronic Key vorliegen wie hier in der Konfiguration. Der Adapter prüft selbst, wenn eine abweichende Version vorliegt, ob diese kompatibel ist, und EtherNet/IP Kommunikation wird dann erlaubt. Ist das Kompatibilitätsbit gesetzt, dürfen Hersteller-Code, Produkt-Code u. Major-Revision nicht „0“ bzw. nicht deaktiviert sein! • **Genaue Identitätsprüfung**: bei dieser Option werden beim EtherNet/IP Verbindungsaufbau dann alle Attribute des Electronic Keys geprüft, bei denen hier im Konfigurationsdialog das "Häkchen" aktiviert ist. Ein deaktivierter Wert wird nicht geprüft; üblicherweise wird die Minor Revision ignoriert.

Registerkarte 'Verbindungen'

Nach der Zielsystemeinstellung sind die Verbindungen einzutragen. Dies geschieht über die Schaltfläche <Verbindung hinzufügen>.

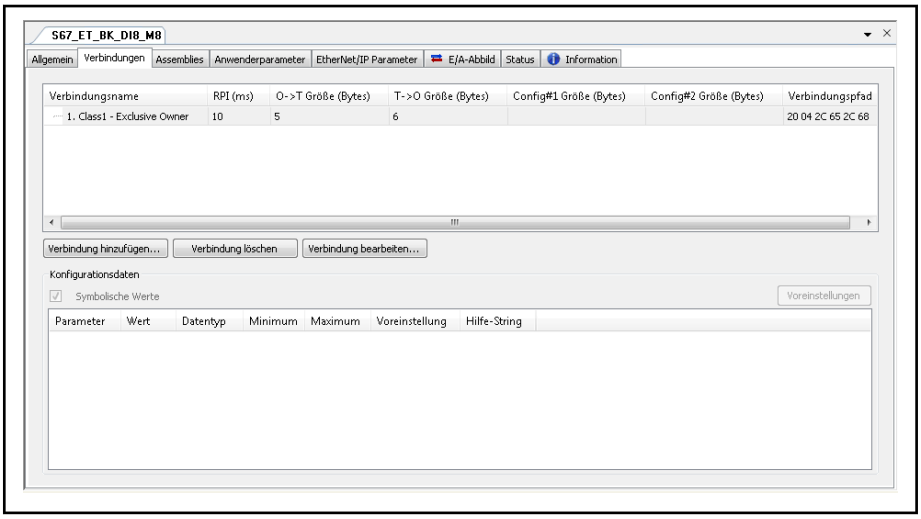


Abb. 7-15: EtherNet/IP Scanner, Adapter - Registerkarte "Verbindungen" (der Eintrag wurde vom nachstehenden Hilfsdialog erzeugt)

In diesem Fenster werden die Verbindungen konfiguriert. Im dargestellten Beispiel wurde eine "Class1 Exclusive Owner" Verbindung eingerichtet, mit 6 Byte Scanner-Ausgangsdaten (O->T) und 8 Byte Scanner-Eingangsdaten (T->O).

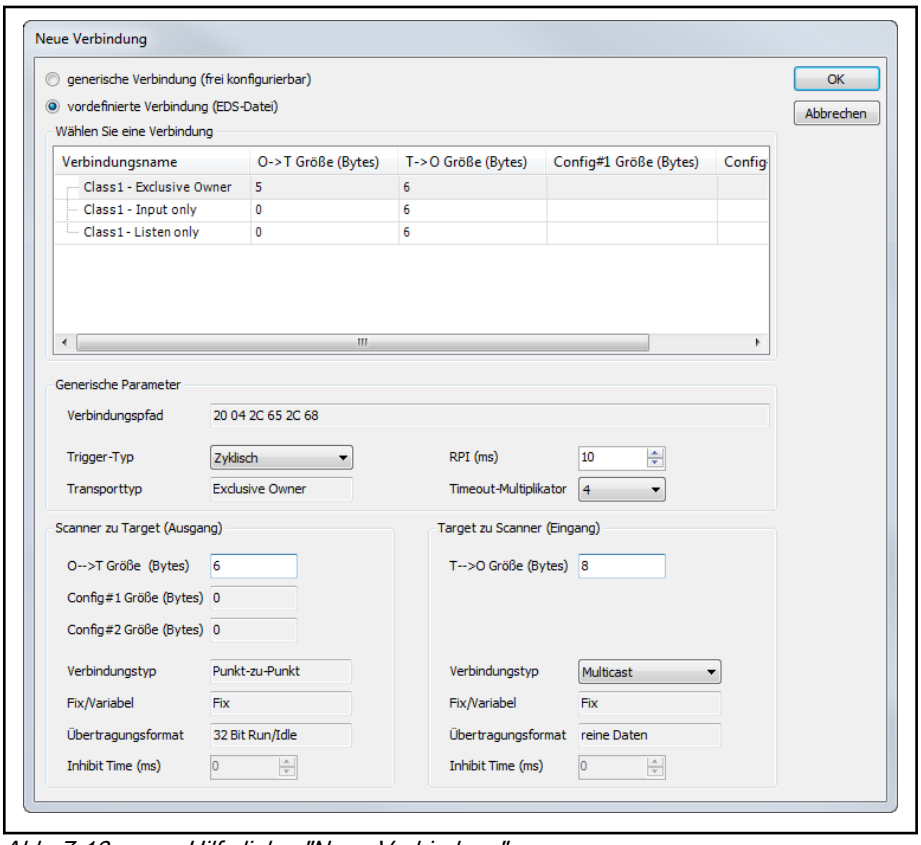


Abb. 7-16: Hilfsdialog "Neue Verbindung"

Es kann eine generische Verbindung angelegt werden, in der alle Verbindungsparameter frei konfigurierbar sind (nur für Experten!).

Es wird aber empfohlen die vordefinierten Verbindungen aus der Gerätebeschreibung (EDS-Datei) zu verwenden. Hier sind dann viele Verbindungseigenschaften bereits fest eingestellt (nicht editierbar) und die editierbaren Parameter können nur in dem erlauben Wertebereich eingestellt werden.

☞ Einige Adapter-Geräte senden herstellerspezifische "Status-Daten" in der zyklischen Kommunikation zum Scanner. Die Scanner Eingangsdatenlänge "T→O Größe" muss dann um die Länge der mitgesendeten "Status-Daten" erhöht werden.

Weitere Informationen hierzu unter Hinweise zur EtherNet/IP Verbindungslänge (O => T / T => O Größe) auf Seite 172

☞ Die Verbindungsparameter Trigger-Typ, RPI und Timeout-Multiplier haben meist schon "sinnvolle" Defaultwerte. Beim Modus (Verbindungstyp) wird "Punkt-zu-Punkt" empfohlen.

Die Schaltfläche <Verbindung löschen> erlaubt das Löschen einer markierten Verbindung (nach Rückfrage).

Die Schaltfläche <Verbindung bearbeiten> erlaubt die Nachbearbeitung einer markierten Verbindung.

Hierbei wird man nur gefragt, ob man die gewählte Verbindung wirklich löschen möchte.

Beschreibung der Verbindungsparameter:

Triggertyp	In der aktuellen Version des Scanners wird nur der Trigger "Zyklisch" unterstützt. Adapter-Geräte können auch die Trigger "Statusänderung" und "Applikation" unterstützen
Transporttyp	"Exclusive Owner" (Ein- und Ausgangsdaten) oder "Input Only" (nur Scanner Eingangsdaten) oder "Listen Only" (nur Scanner Eingangsdaten)
RPI	"Requested Packet Interval", Zykluszeit der zyklischen I/O-Kommunikation
Timeout-Multiplier	Anzahl der Telegramme, die verloren gehen dürfen
O→T-Größe	Datenlänge vom Originator zum Target (Scanner-Ausgangsdaten)
T→O-Größe	Datenlänge vom Target zum Originator (Scanner-Eingangsdaten)
Config#1-Größe	Datenmenge der Konfigurationsdaten zum Target
Config#2-Größe	Datenmenge der Konfigurationsdaten zum Target
Modus	"Verbindungstyp": Punkt-zu-Punkt, Multicast oder Null
Typ	"Längenformat": Fix oder variable Datenlänge
Übertragungsformat	nur Daten, mit Run/Idle Header oder Heartbeat
Inhibit-Time	Verzögerungszeit, die mindestens zwischen zwei zyklischen I/O-Telegrammen vorliegen muss (nur bei Trigger "Statusänderung" oder "Applikation" relevant, keine Funktion bei "Zyklisch")

Registerkarte 'Assemblies'

In der Registerkarte "Assemblies" werden die Datenstrukturen der Ein- u. Ausgangsdaten angezeigt und können bearbeitet werden.

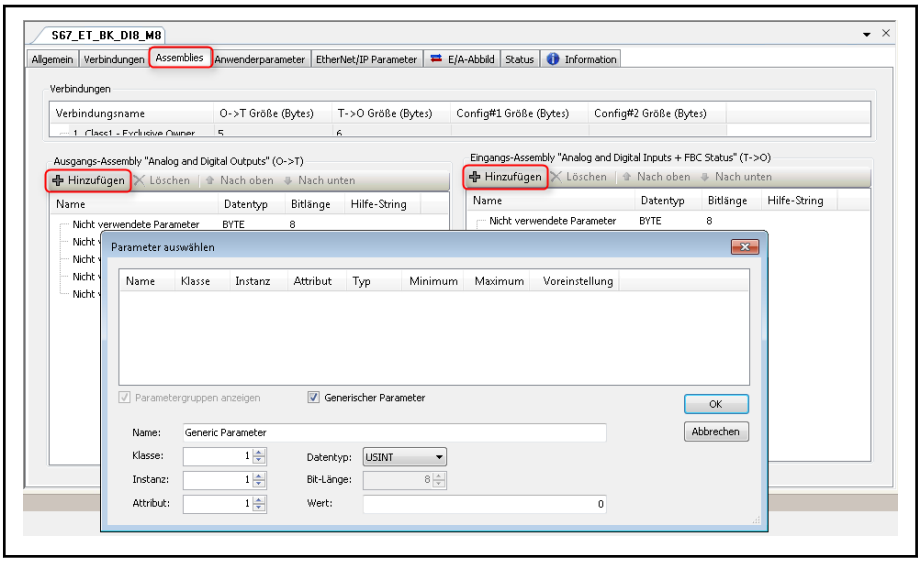


Abb. 7-17: EtherNet/IP Scanner, Adapter - Registerkarte "Assemblies"

Bei manchen EtherNet/IP Adaptern werden die Datenstrukturen und Benennungen in der Gerätebeschreibung (EDS-Datei) definiert. Diese werden dann hier ins I/O-Abbild übernommen und können bei Bedarf auch noch angepasst werden.

Sind in der Gerätebeschreibung keine Datenstrukturen enthalten, wird das E/A-Abbild im Ausgangszustand als Feld eines einfachen Datentyps (BYTE / UINT) dargestellt.

Eigene Datenstrukturen können dann erstellt und entsprechend angeordnet werden ("Hinzufügen", "Löschen", "Nach oben / unten" schieben).

Beim Hinzufügen bzw. Löschen eines Datentyps wird die zugehörige I/O-Länge (O=>T / T=>O Größe) entsprechend angepasst.

Die I/O-Länge der Verbindung muss mit der entsprechenden Datenlänge im Adapter übereinstimmen damit EtherNet/IP-Kommunikation zustande kommen.

Registerkarte 'Status'

Die Registerkarte"Status" im "EtherNet/IP Scanner, Adapter" zeigt Status-Informationen (z. B. "Running", "Stopped") und spezifische Diagnosemeldungen des Gerätes.

Beim EtherNet/IP "Status"-Dialog hat der "Bestätigen"-Button keine Funktionalität.

7.5.3 Das "Generic Device"

Allgemein

Ist für ein Adapter-Gerät keine EDS-Datei vorhanden, kann im Konfigurator des Scanners ein "Generic Device" angelegt werden (das in der Gerätebibliothek bereits vorhanden ist). Hierbei müssen allerdings alle erforderlichen Konfigurationsparameter manuell eingegeben werden, d.h. neben IP-Adresse und ggf. MAC-Adresse sind auch die Werte des "Electronic Key" einzugeben.

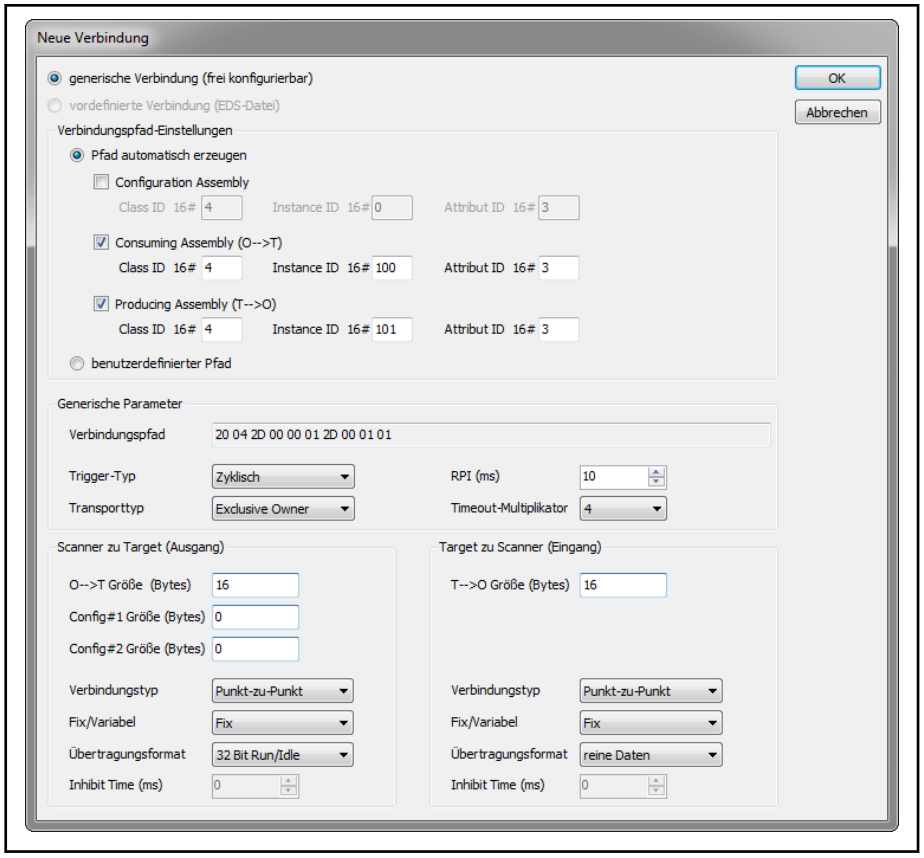


Abb. 7-18: Dialog "Neue Verbindung"

In diesem Fenster können alle zum Verbindungsaufbau notwendigen Daten eingegeben werden.

Das "Generic Device" hat keine vordefinierte Verbindung, sondern eine neue Verbindung muss vollständig manuell eingerichtet werden. Hierzu ist spezifisches Wissen über das Adapter-Gerät erforderlich (z.B. Assembly Instanzen für Verbindungspfad) bzw. auch bei den Verbindungsparametern unterstützt ein Adapter ggf. nur einen eingeschränkten Wertebereich.

Tipps und Empfehlungen zur Konfiguration einer "generischen Verbindung" (frei konfigurierbar)

Verbindungspfad-Einstellungen

Configuration Assembly	Wird selten genutzt und bleibt daher auf "nicht aktiviert" bzw. "leer".
Consuming Assembly (O→T)	Auch als "Output Assembly" bezeichnet. Ausgangsdaten des Scanner, die vom Adapter empfangen werden. Im Feld "Instance ID" ist die entsprechende Output Assembly-Instanz einzutragen.
Producing Assembly (T→O)	Auch als "Input Assembly" bezeichnet. Eingangsdaten im Scanner, die vom Adapter produziert werden. Im Feld "Instance ID" ist die entsprechende Input Assembly-Instanz einzutragen.

Generische Parameter

Trigger-Typ	Bleibt typischerweise auf "Zyklisch".
Transport-Typ	Bleibt typischerweise auf "Exclusive Owner".

RPI (ms)	Die Zykluszeit sollte nicht kleiner als benötigt eingestellt werden, Zykluszeiten ≥ 10 ms sind meist ausreichend.
Timeout-Multiplier	Es wird der Defaultwert 4 empfohlen.

Weitere Verbindungseinstellungen

O→T / T→O Größe	Hier wird die Länge der Verbindung eingetragen. Für das "Output Assembly" (O→T) und das "Input Assembly" (T→O), jeweils Anzahl in Bytes.
Config #1 / Config #2 Größe	Werden nur benötigt, wenn eine "Configuration Assembly" Instanz vorliegt (nur für Experten), ansonsten bleiben diese Werte auf "0".
Verbindungstyp	Hier wird als Einstellung "Punkt-zu-Punkt" empfohlen, was auch als "Unicast" bezeichnet wird. Multicast sollte nur verwendet werden, wenn es erforderlich ist.
Fix/Variabel	Hier sollte die Einstellung "Fix" verwendet werden.
Übertragungsformat	Bei "Scanner zu Target" (Ausgang) wird typischerweise das Format "32 Bit Run/Idle" verwendet. Bei "Target zu Scanner" (Eingang) ist dies abhängig vom Adapter-Gerät: • bei einem allgemeinen I/O Gerät ist das Format typischerweise "reine Daten" • ist der Adapter eine Steuerung kann dies auch "32 Bit Run/Idle" sein
Inhibit Time (ms):	Wird beim Trigger-Typ "Zyklisch" nicht verwendet.

☞ Modulare Geräte, bei denen ein "Slot" konfiguriert werden muss, werden vom Rexroth EtherNet/IP Scanner nicht unterstützt.

7.6 EtherNet/IP, Adapter

7.6.1 Übersicht

Nachstehende Steuerungstypen unterstützen EtherNet/IP Adapter:

- **XM2**

  Anschlussmöglichkeiten:

  1. Über den Engineering-Port
  2. Über die RT-Ethernet Erweiterungsmodule:

     XFE01.1-FB-03 (Master)

     XFE01.1-FB-04 (Slave)

- **XM4**

  Anschlussmöglichkeiten:

  1. Über den Engineering-Port
  2. Über die Onboard-Schnittstelle
  3. Über die RT-Ethernet Erweiterungsmodule:

     XFE01.1-FB-03 (Master)

     XFE01.1-FB-04 (Slave)

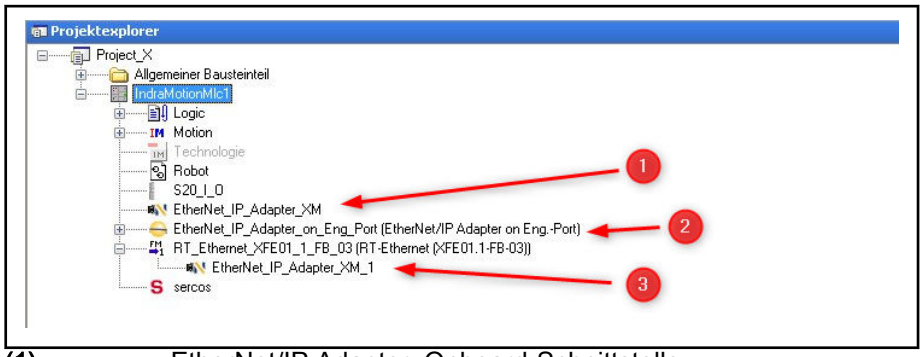
- **VPx**

  Anschlussmöglichkeiten:

  1. Über den Engineering-Port
  2. Über das RT-Ethernet Erweiterungsmodul PFC01.1-FB-13

Die an der Steuerung jeweils vorhandene Schnittstelle wird im Projektexplorer als EtherNet/IP Adapter Objekt dargestellt. Das Objekt kann entweder beim Anlegen der Steuerung oder über den Kontextmenüpunkt **Gerät setzen** als EtherNet/IP Adapter konfiguriert werden.

Zur Lage der Schnittstellen siehe Feldbusse und Feldbuskommunikation auf Seite 11.



(1) EtherNet/IP Adapter, Onboard-Schnittstelle
(2) EtherNet/IP Adapter, Engineering-Schnittstelle, siehe EtherNet/IP, Adapter (Engineering Port) konfigurieren auf Seite 166
(3) EtherNet/IP Adapter, Erweiterungsmodul

Abb. 7-19: EtherNet/IP Adapter-Objekte, noch ohne Beschaltung

EtherNet/IP Adapter

- Registerkarte 'Adapter-Einstellungen' auf Seite 161
- Registerkarte 'E/A-Abbild' auf Seite 163
- Registerkarte 'Feldbusmapping' auf Seite 164
- Registerkarte 'EtherNetIP Parameter' auf Seite 164, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'Status' auf Seite 165
- Registerkarte 'Information' auf Seite 165

7.6.2 Registerkarte 'Adapter-Einstellungen'

In der Registerkarte Adapter-Einstellungen werden die Zielsystemeinstellungen des EtherNet/IP Adapters vorgenommen.

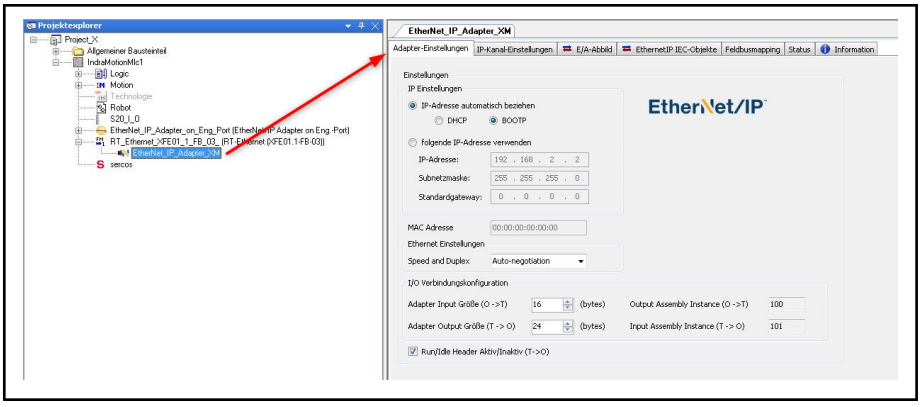


Abb. 7-20: Einstellung des EtherNet/IP Adapters

Einstellungen

IP Einstellungen	
IP-Adresse automatisch beziehen	Die IP-Adresse des EtherNet/IP Adapters kann automatisch bezogen werden: • **DHCP** - IP-Konfiguration von einem DHCP-Server automatisch beziehen. • **BOOTP** - IP-Konfiguration von einem BOOTP-Server automatisch beziehen.
folgende IP-Adresse verwenden	Manuelle Vorgabe der IP-Adresse unter Angabe von: IP-Adresse, Subnetz- Maske und Gateway-Adresse.
MAC Adresse	Die MAC-Adresse des EtherNet/IP Adapters wird automatisch gelesen und kommt lediglich zur Anzeige.
Ethernet Einstellungen	Unter "Speed and Duplex" sind die Geschwindigkeits- und Richtungseinstellungen der Ethernetschnittstelle vorzunehmen. Mögliche Werte sind: • autonegotiation • 10 Mbit / half duplex • 10 Mbit / full duplex • 100 Mbit / half duplex • 100 Mbit / full duplex
MAC Adresse der netX-Schnittstelle	Initial "00:00:00:00:00:00", nur wenn man eingeloggt ist, wird die echte MAC-Adresse angezeigt. Nach dem Ausloggen ist die MAC-Adresse wieder leer.

I/O-Verbindungskonfiguration	Bei EtherNet/IP sind die Input/Output Assembly Instanzen aus der Sicht des Scanners (Originators) wie folgt zu definieren: • **Adapter Input Größe (O -> T)** - Länge der Eingangsdaten in Byte • **Adapter Output Größe (T -> O)** - Länge der Ausgangsdaten in Byte • **Output Assembly Instanz (O -> T)** - Anzeige der Output Assembly Instanz • **Input Assembly Instanz (T -> O)** - Anzeige der Input Assembly Instanz **Hinweis**: Die hier vorgegebene Länge der Ein- und Ausgangsdaten beeinflusst die Einstellmöglichkeit im Register "E/A-Abbild" (Registerkarte 'E/A-Abbildauf Seite 163).
Run/Idle Header Aktiv/ Inaktiv (T->O)	Run/Idle Header (4 Byte) vom Adapter zum Scanner. **Hinweis**: Diese Option muss gewählt werden (Run/Idle Header aktiv), wenn die scannerseitige Konfiguration über die EDS-Datei erfolgt.

7.6.3 Registerkarte 'IP-Kanal-Einstellungen'

Siehe Kap. 4.6 "IP-Kanal bei PROFINET IO & EtherNet/IP (netX)" auf Seite 33.

7.6.4 Registerkarte 'E/A-Abbild'

Die Registerkarte E/A-Abbild erlaubt das Mapping von Variablen zu Datenaustausch mit dem Scanner.

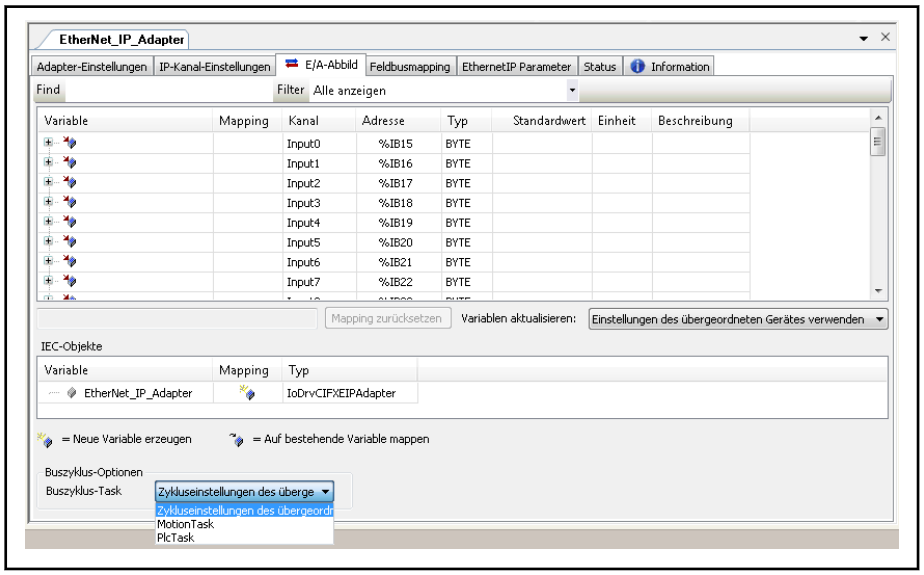


Abb. 7-21: EtherNet/IP Adapter: E/A-Abbild, nach dem Einrichten der Verbindung

Kanäle: Im oberen Teil des Dialoges stehen die in der Konfiguration der Verbindung vorgegebenen Ein- und Ausgänge zur Kommunikation zwischen Scanner und Adapter zum IO-Mapping zur Verfügung.

Das Fenster dient der Zuordnung der Modul-Ein- und Ausgänge zu Variablen, die als (lokale oder) globale Variablen in den einzelnen POUs verwendet werden können.

Im Online-Modus wird der aktuelle Wert der Variablen angezeigt.

Die Zuordnung ist unter Mapping der Feldbus-Ein- und Ausgänge auf Seite 19 beschrieben.

Mapping zurücksetzen: Über diese Schaltfläche können die Mapping-Einstellungen auf die durch die Gerätebeschreibungsdatei definierten Standardwerte zurückgesetzt werden.

Variablen aktualisieren: Festlegung für das Modul , ob die E/A-Variablen in der Buszyklus-Task aktualisiert werden.

Der Default-Wert ist in der Gerätebeschreibung definiert.

- **Einstellungen des übergeordneten Geräts verwenden:** Aktualisierung gemäß den Einstellungen des übergeordneten Geräts.
- **Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet):** Die E/A-Variablen werden in der Buszyklus-Task aktualisiert, falls sie in keiner anderen Task verwendet werden
- **Aktiviert 2 (immer im Buszyklus-Task):** Alle Variablen werden in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

IEC-Objekte: Mit der Definition des EtherNet/IP Adapters werden die Bibliotheken "IoDrvCIFXEIPAdapter.library" bzw. RIL_EtherNetIPAdapter.library automatisch angezogen. Der zur Realisierung des EtherNet/IP Adapters erforderliche Speicherraum kann somit definiert werden.

Buszyklus-Optionen: Durch die Auswahl einer Buszyklus-Task kann der Zyklus des Abbildtausches für den EtherNet/IP Adapters an eine spezielle Task gebunden werden. Es ist sinnvoll in dieser Task auch die IO-Daten des Adapters zu verarbeiten.

7.6.5 Registerkarte 'Feldbusmapping'

Die Registerkarte "Feldbusmapping" ermöglicht den Zugriff auf Variablen der Steuerung von einem übergeordneten Feldbusmaster aus.

Wenn man auf eine Variable der Steuerung von einem übergeordneten Master aus zugreifen will, dann muss dieser Variablen zuerst eine Adresse zugeordnet werden (Mapping). Über diese Adresse kann der übergeordnete Master über azyklische Dienste auf die Daten der Steuerung zugreifen.

Um das Feldbusmapping nutzen zu können, muss die Steuerung als entsprechender Feldbusslave konfiguriert sein.

Details siehe Feldbusmapping, Zugriff vom Feldbusmaster auf Variablen des Feldbusslaves auf Seite 24.

7.6.6 Registerkarte 'EtherNetIP Parameter'

Die Registerkarte "EthernetIP Parameter" zeigt die Kommunikationsparameter des EtherNet/IP Adapters an.

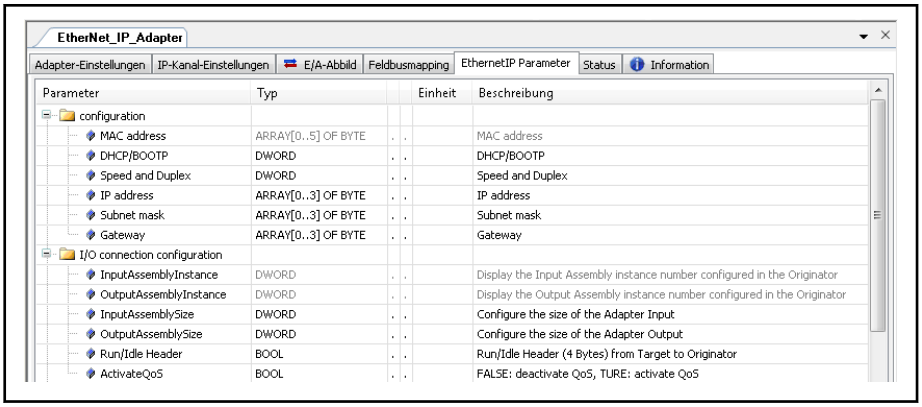


Abb. 7-22: EtherNet/IP Adapter: EtherNetIP Parameter

Einige Parameter (schwarze Schrift) können an dieser Stelle geändert werden.

Dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über IndraWorks Menü **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor**



Eventuelle weitere Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

7.6.7 Registerkarte 'Status'

Die Registerkarte "Status" im "EtherNet/IP Adapter-Geräteeditor" zeigt Status-Informationen (z. B. "Running", "Stopped") und spezifische Diagnosemeldungen aus dem Gerät. Zusätzlich werden die Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt.

7.6.8 Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

7.6.9 EtherNet/IP Scanner-Adapter-Kopplung

MLC-Steuerungen können sowohl als EtherNet/IP Scanner als auch als Adapter konfiguriert werden. Nachfolgend wird die Kopplung zweier Steuerungen auf EtherNet/IP-Basis gezeigt.

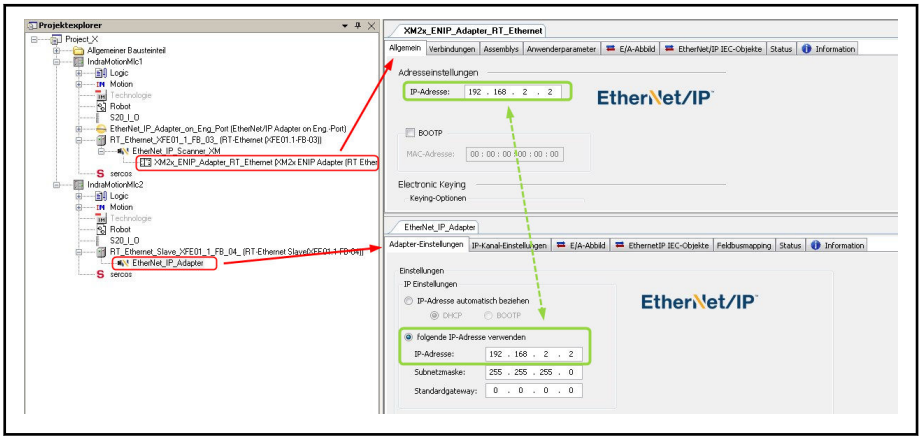


Abb. 7-23: EtherNet/IP Kopplung IP-Adresse

Dem EtherNet/IP Scanner ist genau die Steuerung als Adapter-Gerät hinzuzufügen, die selbst als EtherNet/IP Adapter konfiguriert wurde (hier im Beispiel eine "XM2" Steuerung). Bei der Konfiguration des Adapter-Gerätes unterhalb des Scanners ist die gleiche IP-Adresskonfiguration einzustellen wie bei der Steuerung mit der Adapter-Funktionalität.

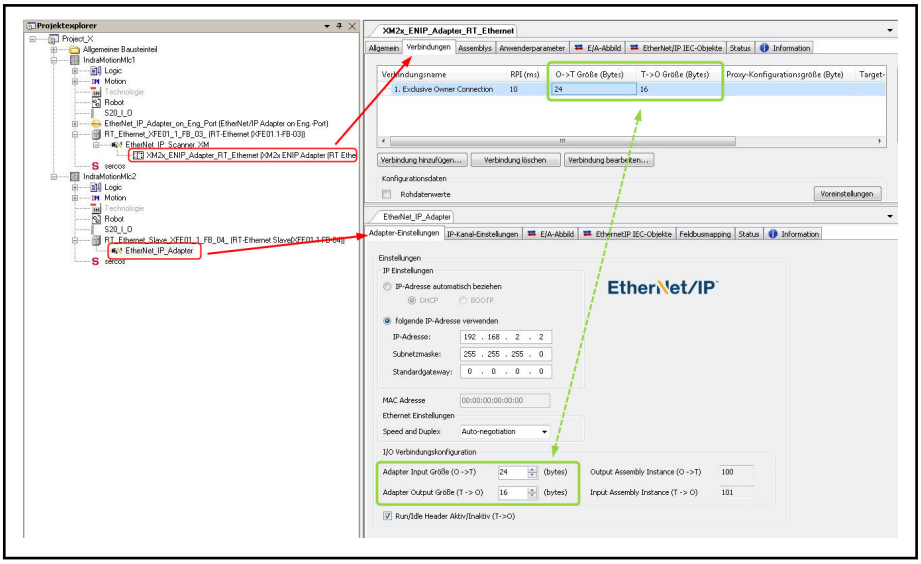


Abb. 7-24: EtherNet/IP Kopplung O <-> T Größe

Wird eine Steuerung als EtherNet/IP Adapter verwendet, kann die Größe der Ein-/Ausgangsdaten variabel eingestellt werden.

Im EtherNet/IP Scanner müssen dann bei der Konfiguration der Verbindung die identischen Größen eingestellt werden. D.h. die Adapter-Ausgangsdaten sind dann die entsprechenden Scanner-Eingangsdaten (T -> O Größe) bzw. die Adapter-Eingangsdaten sind die Scanner-Ausgangsdaten (O -> T Größe).

7.7 EtherNet/IP, Adapter (Engineering Port) konfigurieren

7.7.1 Übersicht

Bei folgenden Steuerungen kann der EtherNet/IP-Adapter am Engineering Port genutzt werden:

- System MLC: XM2, XM4, VPx
- System MTX: XM4

Die an der Steuerung jeweils vorhandene Ethernet-Engineeringschnittstelle wird beim Anlegen der Steuerung im Projektnavigator so konfiguriert, dass sie die Doppelfunktion Engineering-Schnittstelle und EtherNet/IP Adapter-Schnittstelle übernimmt. Die IP-Adresse / Gateway-Einstellung wird übernommen. Die benötigte Konfiguration der Ports erfolgt ohne Zutun des Anwenders.

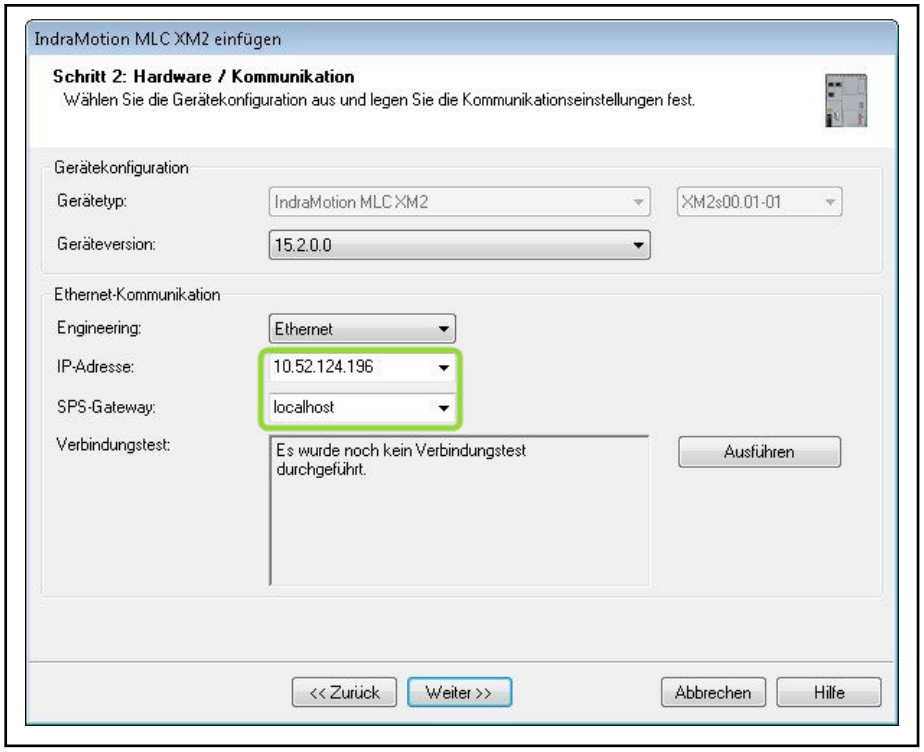


Abb. 7-25: Dialog: Konfiguration der Steuerung, IP-Adresse / SPS-Gateway

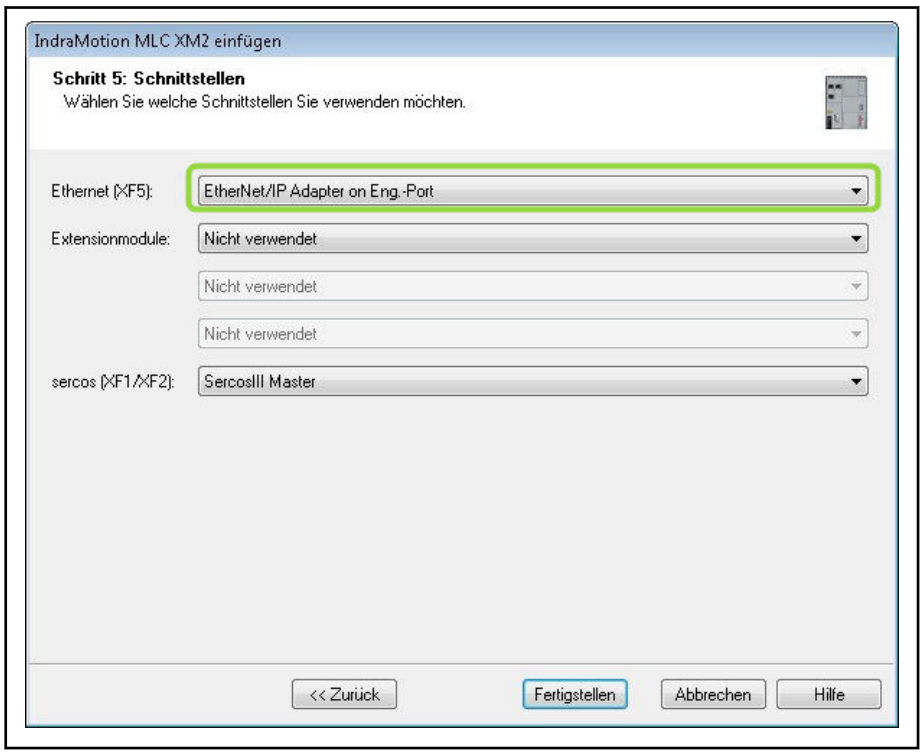


Abb. 7-26: Dialog: Konfiguration der Steuerung, Freigabe der EtherNet/IP Adapter (Engineering-Port)-Funktionalität

Nach erfolgter Freigabe und Abschluss der Steuerungs-Konfiguration erscheint der vorkonfigurierte Ordner im Projektnavigator.

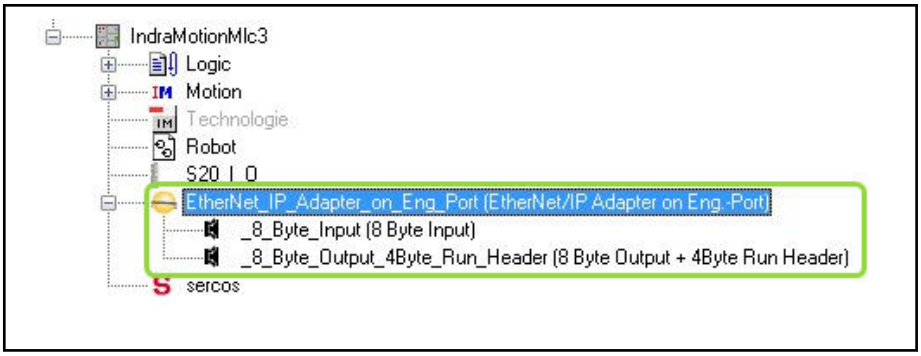


Abb. 7-27: EtherNet/IP Adapter (Engineering)

Die Schnittstelle eines EtherNet/IP Adapter (Engineering Port) zum EtherNet/IP Scanner ist als gemeinsamer Speicherraum ausgeführt. Sie ist adapterseitig mit 8 Byte Eingängen und 8 Byte Ausgängen plus 4 Byte Header vorkonfiguriert.

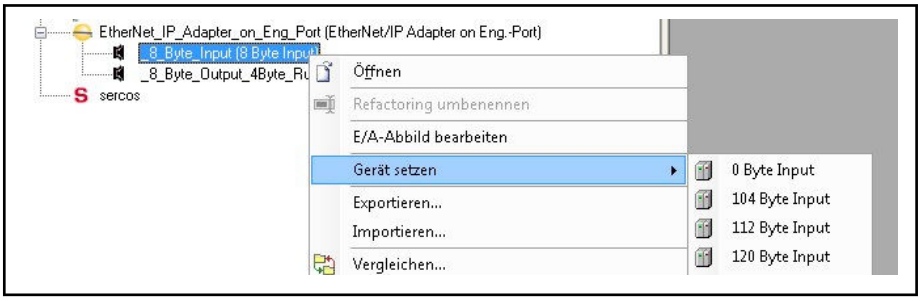


Abb. 7-28: EtherNet/IP Adapter (Engineering-Port) , Veränderung der Größe des Koppelbereiches

1. Eingänge auf der EtherNet/IP Scanner-Seite werden zu Ausgängen auf der EtherNet/IP Adapter-Seite und umgekehrt.
2. Werden Ein- und Ausgangsbereich auf 0 Byte festgelegt, ist keine zyklische Kommunikation möglich.

EtherNet/IP Adapter (Engineering Port)

- *EtherNet/IP Adapter (Engineering Port), Objekt*
  - Registerkarte 'Status' auf Seite 168,
  - Registerkarte 'Information' auf Seite 168.
- *EtherNet/IP Adapter (Engineering Port), Module*
  - Registerkarte 'E/A-Abbild' auf Seite 169,
  - Registerkarte 'Information' auf Seite 169.

7.7.2 Registerkarte 'Status'

Die Registerkarte "Status" im "EtherNet/IP Adapter-Geräteeditor" zeigt Status-Informationen (z. B. "Running", "Stopped") und spezifische Diagnosemeldungen aus dem Gerät. Zusätzlich werden die Diagnosemeldungen, die in den Status-Flags enthalten sind, in "Diag" angezeigt.

7.7.3 Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

7.7.4 EtherNet/IP Adapter (Engineering Port) - Module

Registerkarte 'E/A-Abbild'

Die Registerkarte dient der Zuordnung der Modul-Ein- und Ausgänge zu Variablen, die als lokale oder globale Variablen in den einzelnen Programm-Organisationseinheiten verwendet werden können.

Im Online-Modus wird der aktuelle Wert der Variablen angezeigt.

Die Zuordnung ist unter Mapping der Feldbus-Ein- und Ausgänge auf Seite 19 beschrieben.

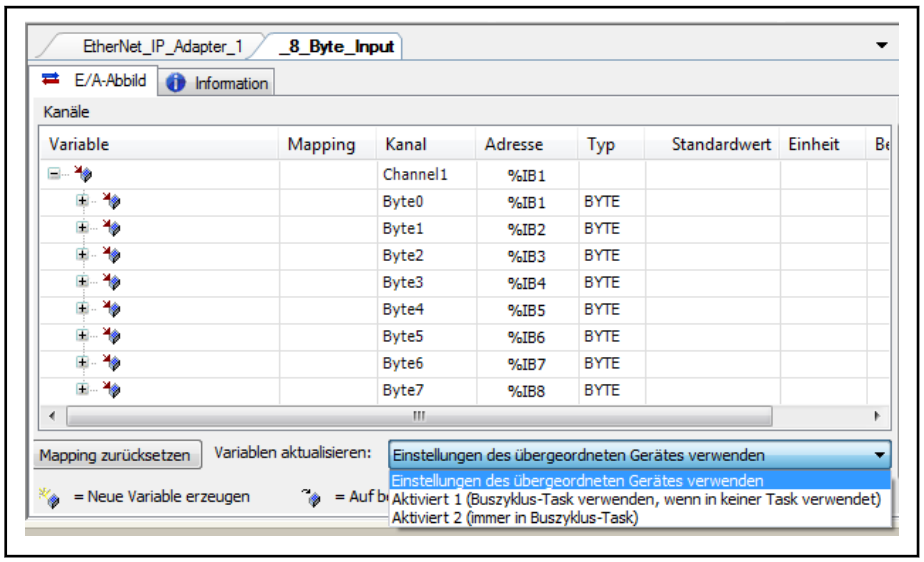


Abb. 7-29: EtherNet/IP Adapter (Engineering Port): E/A-Abbild

Mapping zurücksetzen: Über diese Schaltfläche können die Mapping-Einstellungen auf die durch die Gerätebeschreibungsdatei definierten Standardwerte zurückgesetzt werden.

Variablen aktualisieren: Festlegung für das Modul , ob die E/A-Variablen in der Buszyklus-Task aktualisiert werden.

Der Default-Wert ist in der Gerätebeschreibung definiert.

- **Einstellungen des übergeordneten Geräts verwenden:** Aktualisierung gemäß den Einstellungen des übergeordneten Geräts.
- **Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet):** Die E/A-Variablen werden in der Buszyklus-Task aktualisiert, falls sie in keiner anderen Task verwendet werden
- **Aktiviert 2 (immer im Buszyklus-Task):** Alle Variablen werden in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

7.8 EtherNet/IP, FAQs

7.8.1 Häufig auftretende Probleme bei der EtherNet/IP Konfiguration

Bei einer fehlerhaften EtherNet/IP Verbindung wird eine Fehlermeldung im generischen Geräteeditor des EtherNet/IP Scanners bereitgestellt.

Aufruf:

Offnen Sie den generischen Geräteeditor mittels Doppelklick auf den EtherNet/IP Scanner im Projektexplorer und wechseln Sie zur Registerkarte "Status".

Die folgende Tabelle gibt Hinweise auf mögliche Fehlerursachen und Maßnahmen zur Behebung:

Extended Status Code 0x0114 / 0x0115 / 0x0116 "Vendor ID / Product Code mismatch" "Device Type mismatch" "Revision mismatch"	Die zur Konfiguration des Adapters verwendete Gerätebeschreibungsdatei (EDS) stimmt nicht mit dem vorhandenen Gerät überein. Bitte wenden Sie sich an den Hersteller des Adapter Gerätes um die passende EDS zu erhalten. Die Attribute des vorhandenen Gerätes können z.B. mittels mit dem Funktionsbaustein IL_ENIPGetAttributeSingle (siehe Bibliothek RIL_EtherNetIP) ausgelesen werden: Klasse = 01 ("Identity Object"), Instanz = (in der Regel) 01, Attribute: • 1 = Vendor ID • 2 = Device Type • 3 = Product Code • 4 = Major u. Minor Revision
Extended Status Code 0x0127 / 0x0128 "invalid connection size" (O=>T / T=>O settings)	Siehe Hinweise zur EtherNet/IP Verbindungslänge (O => T / T => O Größe), Seite 172.

Extended Status Code 0x0204 "unconnected request timed out" oder "connection open/close timed out"	1. Überprüfen Sie in der Adapter Konfiguration die IP-Adresse in den Zielsystemeinstellungen: • Ist die IP-Adresse richtig eingegeben? • Wenn BOOTP verwendet wird, ist die MAC Adresse richtig eingegeben? 2. Überprüfen Sie die IP-Adresse im Adapter Gerät (wenn diese z.B. separat bzw. mit eigenem Tool vergeben wird): • Beispielsweise muss beim Rexroth IndraDrive als EtherNet/IP Adapter die "Engineering over IP" Adresse unterschiedlich sein zu konfigurierten EtherNet/IP IP-Adresse. • Wurde die IP-Adresse im Adapter Gerät geändert, ist oft ein Spannungsreset erforderlich, damit die neue IP-Adresse wirksam wird. • Um die IP-Einstellung im Adapter zu testen, kann beispielsweise eine direkte Ethernet-Verbindung zum PC eingerichtet und ein einfacher "Ping" mittels Kommandozeile ausgeführt werden. 3. Überprüfen Sie die IP-Adressen in Ihrem Netzwerk: • Überprüfen Sie, dass keine doppelten IP-Adressen im Netzwerk vergeben wurden bzw. vorhanden sind. • Wenn Switches im Netzwerk verwendet werden, versuchen Sie eine direkte Verbindung zwischen dem Scanner u. den Adapter Geräten. 4. Überprüfen Sie die IP-Adresse in den Scanner-Einstellungen: • Wenn BOOTP oder DHCP für die Scanner IP-Adresse aktiviert ist, aber kein BOOTP / DHCP Server im Netzwerk verwendet wird, muss die Einstellung statische IP-Adresse verwendet werden. • Tritt nach dem Laden bzw. beim Starten einer Änderung der Scanner-Einstellungen z.B. der Fehler "update configuration failed" im Diagnose-Logbuch auf, sind folgende Maßnahmen erforderlich: 1. Ein "Clean up" der Steuerung 2. Das Projekt neu in die Steuerung laden
Extended Status Code 0x0315 "invalid segment in connection path"	Dieser Fehler kann u.a. auch auftreten, wenn die O => T / T => O Größe nicht stimmt, siehe Hinweise zur EtherNet/IP Verbindungslänge (O => T / T => O Größe), Seite 172.
Extended Status Code 0x0320 bis 0x07FF	Diese Statuscodes sind herstellerspezifisch, d.h. weitere Informationen sind in der Dokumentation des Adapter-Gerätes nachzuschlagen.
"Driver not found" oder **"Module not found"**	Es ist ein interner Fehler aufgetreten. Falls ein "Bereinigen" und "Neu-Laden" des SPS-Projektes den Fehler nicht behebt, sind folgende Maßnahmen erforderlich: 1. Ein "Clean up" der Steuerung 2. Das Projekt neu in die Steuerung laden

Diagnose-Logbuch "EtherNet/IP Scanner diagnostic available"	Dieser Logbuch-Eintrag entsteht, wenn bei einer EtherNet/IP Verbindung ein Fehler auftritt. Mögliche Ursachen: • Beim Starten des Systems ist der Scanner schneller betriebsbereit als der Adapter, so dass der Verbindungsaufbau zunächst fehlschlägt (Timeout). Sind alle Geräte kommunikationsbereit, werden die Verbindungen automatisch aufgebaut. • Beim Verbindungsaufbau ist ein Fehler aufgetreten, der Adapter antwortet mit einem CIP Error Code. Details werden im "Status"-Dialog des Adapters angezeigt. • Es sind Fehler während der zyklischen Kommunikation aufgetreten, siehe nächster Abschnitt "I/O Performance".
I/O Performance und **Fehler in der zyklischen Kommunikation**	Hinsichtlich I/O Performance und Netzwerk-Last werden folgende Einstellungen empfohlen: • Es sollten grundsätzlich Point-2-Point (Unicast) Verbindungen verwendet werden; Multicast nur wenn erforderlich. • Das Requested-Packet-Interval (RPI) nur so klein (schnell) einstellen, wie es erforderlich ist. In den meisten Fällen sind 10 bis 20 ms ausreichend. • Das EtherNet/IP Netzwerk sollte getrennt von sonstigen Industrie- u. Office-Netzwerken betrieben werden (ggf. Router verwenden), da Störungen u. Netzwerk-Last sonst die Verfügbarkeit der EtherNet/IP Kommunikation reduzieren kann. Treten Fehler in der zyklischen Kommunikation auf, sind folgende Maßnahmen zu prüfen: • Siehe Empfehlungen bzgl. Verbindungen auf Point-2-Point umstellen und eigenes bzw. getrenntes EtherNet/IP Netzwerk. • Um die Netzwerk-Last zu reduzieren ist bei den RPI Werten der Verbindungen ein größerer Wert einzustellen; hinsichtlich zyklischer Verbindungsabbrüche kann auch der Timeout-Multiplier erhöht werden um die Verfügbarkeit zu verbessern. • Netzwerk-Topologie: werden mehrere Adapter in Linie geschaltet, können Adapter Geräte aufgrund der Netzwerk-Last ggf. nicht mehr alle Telegramme durchleiten. Hier ist dann eine Stern-Topologie z.B. mit Hilfe von Switches zu empfehlen.

Tab. 7-6: Häufig auftretende Probleme bei der EtherNet/IP Konfiguration

7.8.2 Hinweise zur EtherNet/IP Verbindungslänge (O => T / T => O Größe)

Die derzeit auf dem Markt erhältlichen EtherNet/IP Adapter (Buskoppler) variieren zum Teil sehr stark in ihrem Verhalten bzw. in ihrer Konfiguration.

Es gibt Adapter-Geräte, die in den Daten zum Scanner (Scanner-Eingangsdaten) zusätzliche Statusinformationen unterschiedlicher Länge mit senden. Diese Statusdaten können am Anfang oder am Ende der Scanner-Eingangsdaten liegen.

Bei der Konfiguration einer Verbindung sind diese Statusdaten zu berücksichtigen, d.h. die T->O Größe (Scanner-Eingangsdatenlänge) ist um die Länge dieser Statusdaten zu erhöhen. Eine Verbindung mit ungültiger Datenlänge führt zu einem Fehler beim Verbindungsaufbau (keine Kommunikation möglich).

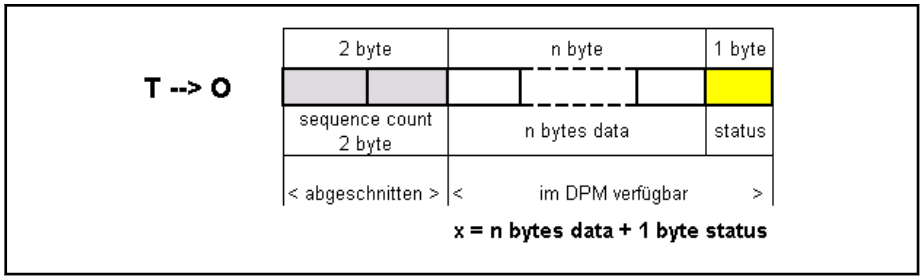


Abb. 7-30: Eingangsdaten, Variante 1

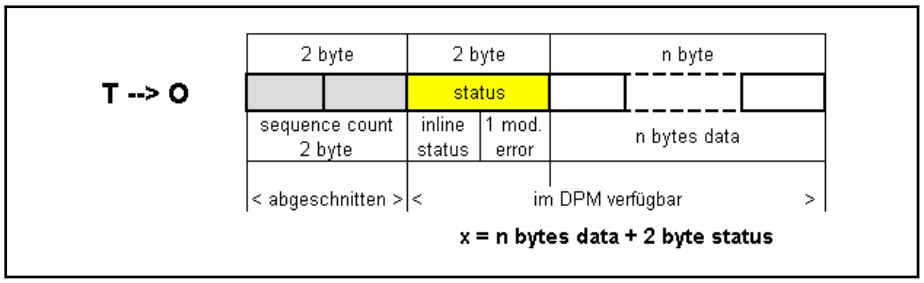


Abb. 7-31: Eingangsdaten, Variante 2

Manche Adapter erzwingen eine gerade Anzahl (2, 4, 6, …) von Eingangs- und/oder Ausgangsdaten, ohne die sie keine Verbindung akzeptieren.

Bei modularen Buskopplern, welche mit Steckmodulen erweitert werden können, werden die einzelnen Module zusammengefasst, um jeweils auf ganze Bytes zu kommen.

Es ist nicht möglich ein z.B. 2Bit-Ausgangsmodul mit einem Byte anzusprechen.

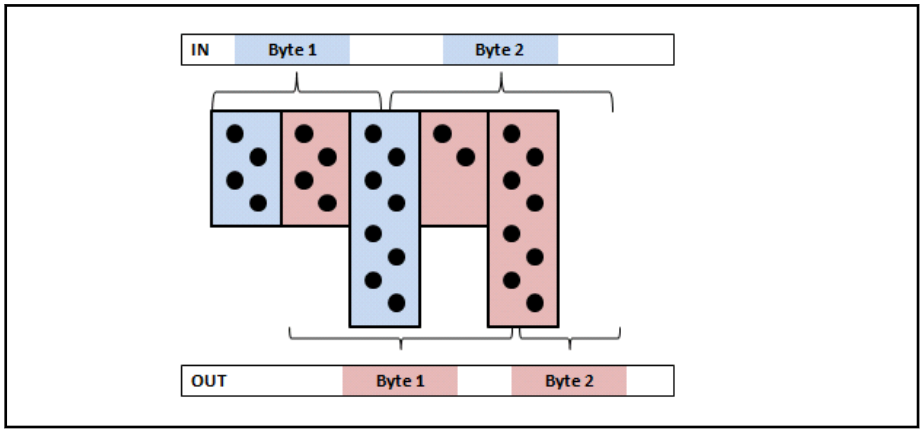


Abb. 7-32: Bytegrenzen bei modularen Buskopplern

Ist die Länge für eine Verbindung bzw. für das Assembly in einer Verbindung nicht bekannt, kann diese Länge auch ausgelesen werden. Zum Beispiel adressiert der Verbindungspfad 20 04 24 03 2C **64** 2C **65** die zwei Assemblies mit den Instanzen 0x64 und 0x65.

Mit dem Funktionsbaustein IL_ENIPGetAttributeSingle (siehe Bibliothek RIL_EtherNetIP) kann bei jedem Assembly die Größe ausgelesen werden: Klasse = 4 (Assembly Objekt), Instanz = Instanz aus dem Verbindungspfad, Attribut = 4 (Größe). Diese ausgelesen Länge ist dann wiederum als O => T bzw. als T => O Größe in der Verbindungskonfiguration einzugeben.

Bei "älteren" Buskopplern muss evtl. der Verbindungspfad "ausführlich" beschrieben werden, d.h. die übliche Kurzschreibweise wurde hier noch nicht unterstützt z.B.

20 04 **24 03** 2C 64 2C 65 gegenüber 20 04 2C 66 2C 69 … .



Bei Bedarf muss die Spezifikation bzw. Anwenderdokumentation des Adapters verwendet bzw. der Hersteller des Adapter-Gerätes kontaktiert werden.

8 EtherCAT

8.1 Features des EtherCAT Master

Funktionspaket Lizenz

Die Funktionalität "EtherCAT Master" ist als **Funktionspaket** lizenziert, d.h. eine entsprechende Lizenz muss erworben werden.

Funktionsumfang

Die aktuelle Implementation des EtherCAT Masters beinhaltet folgende Funktionalität:

- zyklische Kommunikation "Free Run"
- Synchronbetrieb ("Distributed Clocks" / Verteilte Uhren)
- Import der Gerätebeschreibungsdatei (ESI) in IndraWorks Gerätedatenbank
- Offline Konfiguration der Slaves
- Manuelle Konfiguration von SyncUnits
- Online Diagnose in der IndraWorks Oberfläche
- Bus Scan und automatische Konfiguration im IndraWorks Projektbaum
- SPS-Funktionsbausteine (RIL_Ethercat Bibliothek)
  - Funktionsbausteine für Zustandsmaschine
  - Funktionsbausteine für azyklische Kommunikation
  - Funktionsbausteine für Diagnose (u.a. Master-Status, Slave-Status, Konfiguration-/Online-Info)

In Vorbereitung:

- Identifikationsverfahren "Station Alias", "Explicit Device Identification" und "Data Word"

Zielsysteme

Der EtherCAT Master kann ausschließlich mit dem System MLC umgesetzt werden.

Steuerungs-Hardware

Folgende Steuerungs-Hardware wird unterstützt:

- XM21 und XM22

  (ab IndraWorks Version 14V20, auf der onboard Ethernet-Schnittstelle XF1 alternativ zu Sercos, XF2 ohne Verwendung)
- XM42

  (ab IndraWorks Version 14V22 P1, auf der onboard Ethernet-Schnittstelle XF1 alternativ zu Sercos, XF2 ohne Verwendung)
- VPx

  (ab IndraWorks Version 14V22 P1, auf der Ethernet-Schnittstelle XF01 alternativ zu Sercos, XF02 ohne Verwendung)

Siehe auch "Verkabelung" auf Seite 205.

Technische Daten

Eigenschaft / Funktionalität	Wert
Max. Anzahl Teilnehmer (Slaves)	128
Max. Anzahl zyklischer Eingangsdaten des Masters	2048 Bytes [1.)]
Max. Anzahl zyklischer Ausgangsdaten des Masters	2048 Bytes [1.)]
Max. Anzahl zyklischer Eingangsdaten eines Slaves	2048 Bytes [1.)]

Max. Anzahl zyklischer Ausgangsdaten eines Slaves	2048 Bytes [1.)]
Max. Anzahl Daten bei azyklischer Kommunikation	CoE, SoE: 65535 Bytes
Max. Anzahl paralleler azyklischer Aufträge über Funktionsbausteine	CoE, SoE: 128
Baudrate	100 MBit/s
Auto negotiation / autocrossing	Ja
Min. Zykluszeit	1 ms
Unterstützung mehrerer unterschiedlicher Zykluszeiten	Nein
Synchronbetrieb (Distributed Clocks / Verteilte Uhren)	Bus Shift (ab 14V22 P1)
Hot Connect Gruppen (flexible Topologien)	Nein
Mailbox CoE (CAN application layer over EtherCAT)	Ja (ab 14V20)
Mailbox SoE (Sercos drive profile over EtherCAT)	Ja (ab 14V20)
Mailbox EoE (Ethernet over EtherCAT)	Ja (ab 14V22)
Mailbox FoE (File access over EtherCAT)	Nein
Mailbox VoE (Vendor specific protocol over EtherCAT)	Nein
Mailbox AoE (ADS over EtherCAT)	Nein
Slave-to-Slave Kommunikation	Nein
Redundanz	Nein
Feldbus-Gateways v. Fremdhersteller mit proprietärer Konfiguration (z.B. EtherCAT Slave Klemmen EL6xxx von Beckhoff)	Nein
Deaktivieren von Slaves oder Modulen ("Maximalprojekt")	Nein

[1.)] Die max. Anzahl an Prozessdaten des EtherCAT Masters ist 2048 Bytes (= Summe aus zyklischen Ein-/Ausgangsdaten). Es gibt keine Einschränkung hinsichtlich Ein-/Ausgangsdaten eines Slaves.

8.2 EtherCAT Grundlagen

8.2.1 Begriffe und Abkürzungen

EtherCAT	Ethernet for Control Automation Technology. EtherCAT® ist eine eingetragene Marke und patentierte Technologie, lizenziert durch die Beckhoff Automation GmbH, Deutschland. EtherCAT ist ein offener Standard, der international genormt ist und von der EtherCAT Technology Group (ETG) weiterentwickelt wird.
EC / ECAT **ECM**	Abkürzungen für EtherCAT (EC / ECAT) bzw. EtherCAT Master (ECM)
ENI	EtherCAT Network Information: Beschreibung der Buskonfiguration als XML-Datei
ESI	EtherCAT Slave Information: Gerätebeschreibungsdatei als XML-Datei
ESC	EtherCAT Slave Controller: in Hardware integrierte Kommunikationsanschaltung (ASIC, FPGA) oder als reine Softwarelösung

ETG	EtherCAT Technology Group: Nutzerorganisation
EAP	EtherCAT Automation Protocol: Protokoll zur Vernetzung auf der Leitebene
DC	Distributed Clocks Zur Synchronisation der Busteilnehmer wird das Verfahren "Verteilte Uhren" angewendet. Man unterscheidet allgemein folgende Vorgehensweisen: • Master Shift: Der erste DC-Slave am Bus dient als Referenzuhr und Nachregelung des Timers des EtherCAT Masters. • Bus Shift: Der EtherCAT Master stellt die Systemzeit und schreibt diese in den ersten DC-Slave am Bus (Schieben der Buszeit). • Externe Referenzuhr (IEEE 1588)
FSoE	FailSafe over EtherCAT: Safety-Protokoll für EtherCAT
HC **HC-Group**	Hot Connect: Ab- und Zuschalten von Slaves im laufenden Betrieb Hot Connect Gruppe: Slaves können in einer Gruppe zusammengefasst werden. Die Gruppe kann als optional deklariert werden (flexible Topologien).
Mailbox **CoE** **SoE** **FoE** **EoE** **VoE** **AoE**	Azyklische Kommunikation über EtherCAT: CoE: CAN application protocol over EtherCAT SoE: Servo drive profile over EtherCAT (adaptiertes Sercos II Profil für Servoantriebe) FoE: File access over EtherCAT EoE: Ethernet over EtherCAT (z.B. TCP/IP) VoE: Vendor specific protocol over EtherCAT (Hersteller-spezifische Mailbox-Protokolle) AoE: Automation Device Specification (ADS) over EtherCAT (geräte- und feldbusunabhängige Schnittstelle von Beckhoff).
WC / WKC **WcState**	Abkürzungen für WorkingCounter (WC / WKC) bzw. WorkingCounter-Status (WcState): Gültigkeit von Kommando-Ausführungen bzw. von Daten (ein WcState Wert 0 = gültig, 1 = ungültig). Siehe auch "Prozessdaten Status" im Registerkarte 'E/A-Abbild', Seite 236.

8.2.2 Topologie

Ein EtherCAT Netzwerk besteht aus einem Master und bis zu 65535 Slaves.

Topologien können beliebig realisiert werden, z.B. durch den Einsatz von Abzweigklemmen.

Bei der Verkabelung ist zu beachten, dass Slave Geräte genau einen definierten Eingangsport haben. Dies ist in der Regel der Port 0, der üblicherweise mit "IN" beschriftet ist. Die meisten Slaves besitzen 2 Ports, d.h. neben dem IN-Port noch einen OUT-Ausgangsport (Port 1). Bei einem Buskoppler mit E-BUS als Rückwandbus werden die Ethernet-Telegramme über diesen Rückwandbus (E-BUS) weitergeleitet, d.h. jedes Modul (Klemme) stellt einen Slave mit IN- u. OUT-Port dar. Der E-BUS Buskoppler oder Abzweigklemmen haben dementsprechend 3 bzw. 4 Ports (= mehrere Ausgangsports).

Nachfolgende Abbildung zeigt eine exemplarische Topologie mit unterschiedlichen Gerätetypen:

- Slave **(A)** ist ein modularer Buskoppler nach dem "Modular Device Profile" (z.B. Rexroth S20-EC-BK): die Erweiterungsmodule sind mit einem gerätespezifischen Rückwandbus mit dem Buskoppler verbunden. Der Buskoppler stellt ein Slave Gerät im Netzwerk dar und besitzt alle E/A-Daten der Module.
- Slave **(B)** ist ein modulares System mit E-BUS (z.B. Beckhoff EK1100): jedes Modul als auch der Buskoppler und mögliche Abzweig- oder Ver-

längerungsklemmen stellen jeweils ein eigenes Slave Gerät am EtherCAT Netzwerk dar.

- Slave **(C)** ist ein Antrieb. Hinweis: Mehrachsgeräte stellen i.d.R. nur ein Slave Gerät dar.

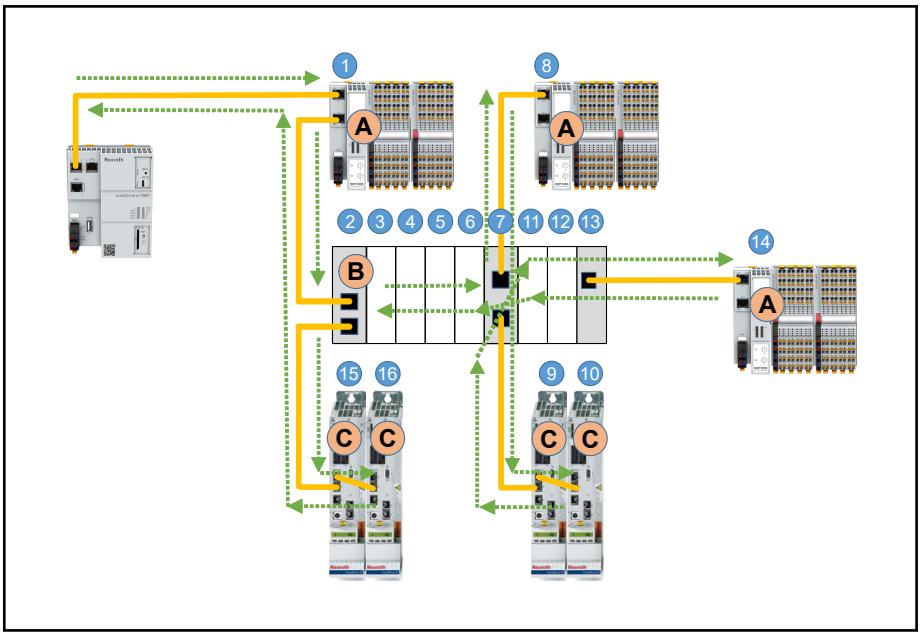


Abb. 8-1: Exemplarische Topologie mit unterschiedlichen Slave Gerätetypen (A/B/C)

Das EtherCAT Telegramm wird vom Master gesendet und durchläuft alle Slave Geräte. Nur in der "Vorwärts"-Richtung findet die Verarbeitung der Ein-/Ausgangsdaten in den EtherCAT Slave Controllern (ESC) statt. Das letzte Slave Gerät sendet das Telegramm dann wieder zurück zum Master. Hierbei wird das Telegramm in den Slaves dann ohne Verarbeitung nur durchgeleitet. Aus logischer Sicht stellt EtherCAT daher unabhängig von der Topologie immer einen Ring dar. In obiger Abbildung zeigen die Nummern (1) bis (16) die logische Reihenfolge, wie das EtherCAT Telegramm die Slave Geräte durchläuft. Bei Abzweigklemmen bzw. bei Geräten mit mehr als 2 Ports ist die Reihenfolge, wie das EtherCAT Telegramm die Ports durchläuft, in der Produktdokumentation des Gerätes nachzuschlagen.

Kabelredundanz

Kabelredundanz kann nur mit einer Ringtopologie erreicht werden. Hierbei wird das letzte Slave Gerät mit dem zweiten Ethernet-Port am Master verbunden.

☞ Der Rexroth EtherCAT Master unterstützt Kabelredundanz **nicht**, d.h. an der Steuerung kann nur der erste Port verwendet werden; z.B. bei der XM2-Steuerung ist der erste Port der Anschluss XF1 (XF2 nicht möglich).

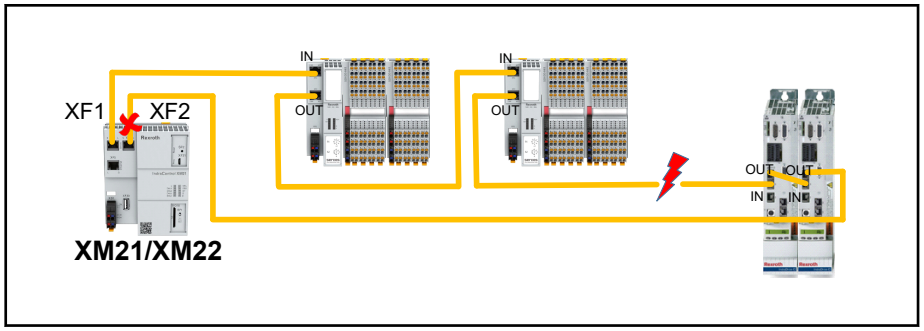


Abb. 8-2: Ringtopologie für Kabelredundanz

8.2.3 Adressierungs- und Identifikationsverfahren

In einem EtherCAT Netzwerk gibt es verschiedene Verfahren zur Adressierung und Identifizierung eines Slave Gerätes.

Auto-Increment Adresse

Die Auto-Increment Adresse entspricht einer topologischen Adresse, d.h. das Slave Gerät wird anhand seiner physikalischen Position im Netzwerk adressiert.

Die Auto-Increment Adresse wird vom Master hauptsächlich bei der Initialisierung der Buskommunikation verwendet. Ab dem Zustand "Pre-Operational" wird die "EtherCAT Adresse" ("Fixed Physical Address") für die Kommunikation verwendet.

Der erste Slave nach dem Master besitzt die Auto-Increment Adresse "0". Für jeden weiteren Slave wird der Wert dekrementiert, d.h. der zweite Slave hat die Adresse "-1" (0xFFFF), der dritte Slave hat die Adresse "-2" (0xFFFE) usw.

Bei der Kommunikation inkrementiert jeder Slave die Auto-Increment Adresse im EtherCAT-Telegramm. Der Slave, der den Wert "0" im Telegramm liest, wird entsprechend adressiert.

EtherCAT Adresse

Die "EtherCAT Adresse" wird in der ETG Spezifikation auch als "Fixed Physical Address" bezeichnet. Bei anderen Herstellern synonym verwendete Begriffe sind zum Beispiel "Slave Address" oder "Station Address".

Die EtherCAT Adresse wird in den Slave Geräten nicht gespeichert, sondern vom Master bei der Bus-Initialisierung jedem Slave zugewiesen. Die Zuordnung, welche EtherCAT Adresse ein bestimmtes Slave Gerät erhält, erfolgt in der EtherCAT Konfiguration entweder über die Auto-Increment Adresse oder bei Hot Connect Gruppen (flexible Topologien) über einen Identifikationswert.

Die EtherCAT Adresse "0" ist für den Master reserviert. Slaves können die Adresse "1" bis "65535" (16 bit) haben. EtherCAT Adressen für die Slaves werden oft ab dem Wert "1001" verwendet.

Identifikationswert

Bei Hot Connect Gruppen (flexible Topologien) oder um Verkabelungsfehler auszuschließen ist es erforderlich, Slave Geräte unabhängig von deren topologischer Position eindeutig zu identifizieren.

Damit einem bestimmten Slave Gerät immer eine festgelegte EtherCAT Adresse zugewiesen werden kann, gibt es drei verschiedene Identifikationsverfahren. Welche Identifikationsverfahren von einem Slave Gerät unterstützt werden (keine Unterstützung, ein oder mehrere Verfahren werden unterstützt) ist in dessen Gerätebeschreibung bzw. Produktdokumentation beschrieben. Die Konfiguration des Identifikationswertes erfolgt beispielsweise über DIP- oder Drehschalter am Slave Gerät.

Identifikationsverfahren:

- (Configured) **Station-Alias**: der Station-Alias wird oft auch als "Second-Slave-Address" (SSA) oder "Second-Station-Address" bezeichnet. Der Wert kann im ESC Register 0x0012 ausgelesen werden. Die Konfiguration erfolgt i.d.R. über das EtherCAT Konfigurationstool, in dem der gewünschte Identifikationswert in das Slave EEPROM geschrieben wird und der EtherCAT Slave Controller dann beim (Neu-) Start diesen Wert in sein Register übernimmt.
- **Explicit Device Identification**: hierbei teilt der Slave auf Anforderung durch den Master seinen Identifizierungswert im AL Status Code Register (0x0134) mit. Dieses Verfahren wird in der ETG.1020 auch als "Requesting ID" bezeichnet.
- **Data Word (/ Input Word)**: hierbei steht der Identifizierungswert in einem vorgegebenen Speicherbereich des EtherCAT Slaves (z.B. Prozessdatenbereich ab ADO 0x1000). Der Address-Offset (ADO) wird aus der Gerätebeschreibung entnommen (Attribut "IdentificationAdo") oder kann ggf. auch eingestellt werden. Dieses Verfahren wird in der ETG.1020 auch als "Direct ID" bezeichnet.

Beispiel In folgendem Beispiel haben die ersten zwei Slaves (EtherCAT-Adresse 1001 u. 1005) keinen Identifikationswert, d.h. die EtherCAT-Adresse wird nur anhand der Topologie-Position (AutoInc-Adresse) vergeben. Wird die physikalische Reihenfolge dieser beiden Geräte (identischer Geräte-Typ) getauscht, z.B. Verkabelungsfehler oder ggf. beabsichtigter Gerätetausch, wird die EtherCAT-Adresse weiterhin gültig entsprechend der Position zugewiesen

Bei den folgenden zwei Slaves (EtherCAT-Adresse 1042 u. 1043) ist ein Identifikationsverfahren konfiguriert, d.h. nur wenn die Identifikationswerte (hier: 2 u. 3) mit der konfigurierten Topologie-Position (AutoInc-Adresse) übereinstimmen, werden die Slaves vom Master in die zyklische Kommunikation aufgenommen. Auf diese Weise wird ein Verkabelungsfehler zwischen diesen beiden Slaves ausgeschlossen. Bei einem beabsichtigen Kreuztausch oder Gerätetausch (Ersatzgerät) muss der Identifikationswert entsprechend gesetzt werden.

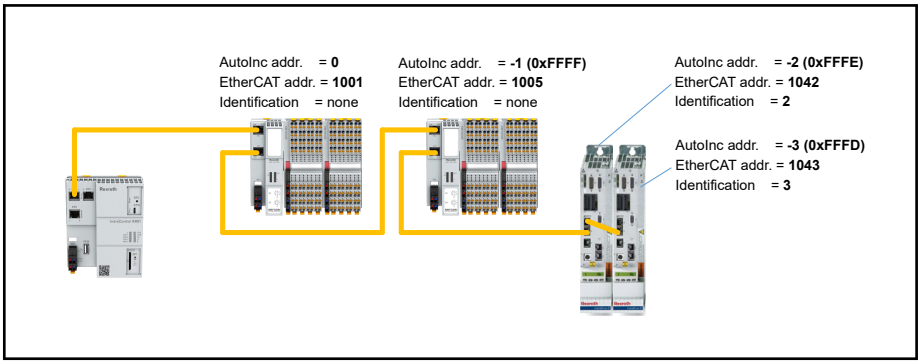


Abb. 8-3: Beispiel für eine EtherCAT Adressierung

8.2.4 Zustandsmaschine

Bei EtherCAT gibt es folgende definierte Kommunikationszustände:

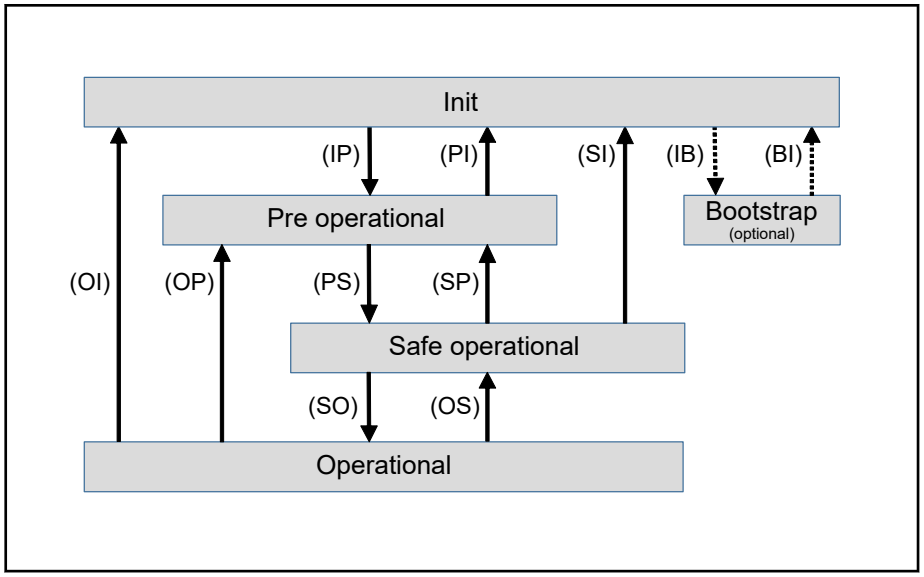


Abb. 8-4: EtherCAT Zustandsmaschine

Zustand	Beschreibung
Init (I)	• keine Kommunikation zur Applikationsschicht • Übergang zu ‚PreOP' (IP): Master konfiguriert Slave-Register, u.a. Adress-Register und Sync-Manager für Mailbox-Kommunikation
Pre-Operational (P)	• ab ‚PreOP' ist Mailbox-Kommunikation zur Applikationsschicht möglich • hier noch keine Prozessdaten-Kommunikation • Übergang zu ‚SafeOP' (PS): Master konfiguriert u.a. Prozessdaten-Mapping (FMMU) und Sync-Manager für Prozessdaten-Kommunikation
Safe-Operational (S)	• Prozessdaten-Kommunikation, allerdings werden nur die Eingangsdaten ausgewertet, die Ausgangsdaten bleiben hier noch im "sicheren" Zustand • DC-Synchronisation wird eingeregelt
Operational (O)	• Prozessdaten-Kommunikation: Ein- und Ausgangsdaten sind gültig • DC-Synchronisation
Bootstrap (B)	• optionaler Zustand in EtherCAT Slave Geräten (empfohlen für Firmware-Update) • der Zustand ‚Bootstrap' kann nur von ‚Init' aus erreicht werden • keine Prozessdaten-Kommunikation • Mailbox-Kommunikation über FoE (ggf. ist FoE auch nur in ‚Bootstrap' möglich => geräte-spezifisch)

Zusätzliche Hinweise:

- ein EtherCAT Slave Gerät kann in einen "kleineren" Zustand als der (Master-) Buszustand geschaltet werden, z.B. ist der EtherCAT Buszustand ‚OP', kann ein Slave individuell in ‚PreOP' geschaltet werden.

- ein Zustandswechsel kann nur vom Master initiiert werden, ansonsten kann ein Slave selbst nur im Fehlerfall seinen Zustand herunterschalten.

8.2.5 Azyklische Kommunikation (Mailbox)

Azyklische Kommunikation wird bei EtherCAT auch als Mailbox Kommunikation bezeichnet und ist erst ab Zustand "Pre-Operational" möglich. Es gibt verschiedene Protokolle, die über EtherCAT getunnelt werden können. Welche Protokolle ein Slave unterstützt, ist der Dokumentation (oder Gerätebeschreibung) des Slave Gerätes zu entnehmen.

Übersicht Mailbox Protokolle

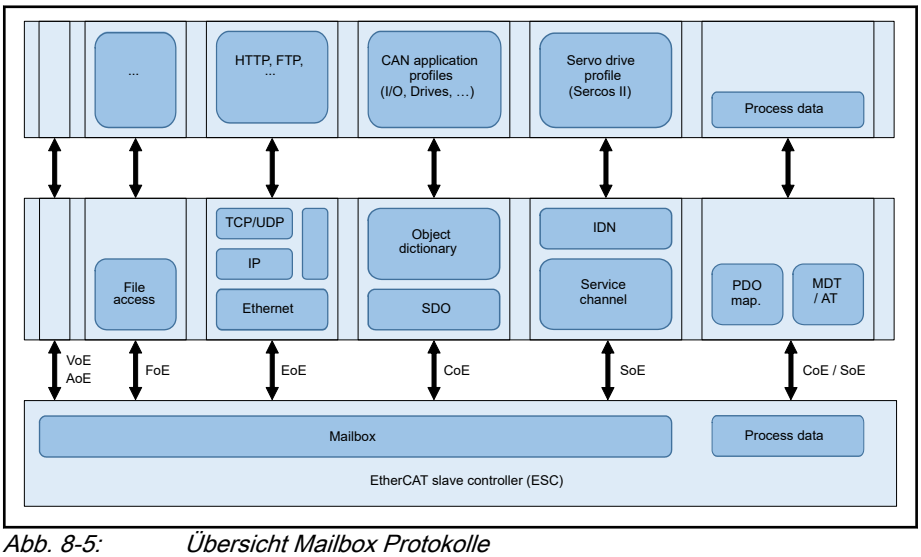


Abb. 8-5: Übersicht Mailbox Protokolle

CoE: CAN application protocol over EtherCAT

CoE Objekte werden auch als "Prozess Daten Objekte" (PDO) oder "Service Daten Objekte" (SDO) bezeichnet. Ein Objekt wird mit "Index" und "Subindex" adressiert, z.B. 0x1018:01. Hat ein Objekt mehr als einen Subindex, dann steht im Subindex "0" die Anzahl der Subindices.

Mit den optionalen CoE Diensten "SDO Information" kann das Objektverzeichnis eines Slaves ausgelesen werden sowie Objektbeschreibungen und Elementbeschreibungen der Subindices.

Siehe auch Kap. "CoE SDO Abortcodes" auf Seite 190.

SoE: Servo drive profile over EtherCAT

SoE ist ein adaptiertes Sercos II Profil für Servoantriebe, d.h. manche Sercos II Parameter haben bei SoE keine Bedeutung oder eine teilweise geänderte Bedeutung. Ein Parameter wird adressiert mit einer 16 bit Identifikationsnummer (IDN: Parametertyp S/P, Parametersatz, Datenblock-Nummer).

Im Unterschied zu Sercos stellen Mehrachsgeräte bei SoE nur ein Slave Gerät dar. Die unterlagerten Antriebe unterscheidet man durch eine Antriebsnummer ("DriveNo": 0 / 1 / ... / 7), die alternativ auch als Kanal ("Channel": A / B / ... / H) bezeichnet wird.

Siehe auch Kap. "SoE Fehlercodes" auf Seite 191.

Folgende IDN haben bei SoE keine Relevanz:	
S-0-0003	Sende-Reaktionszeit AT (T1min)
S-0-0004	Umschaltzeit Senden-Empfangen (TATMT)
S-0-0005	Mindestzeit Istwerterfassung (T4min)
S-0-0009	Anfangsadresse im Master-Daten-Telegramm
S-0-0010	Länge Master-Daten-Telegramm

S-0-0088	TMTSY Erholzeit Empfangen-Empfangen
S-0-0090	TMTSG Kopierzeit Sollwerte
S-0-0127	C0100 Umschaltvorbereitung auf Komm.-Phase 3 Wird beim EtherCAT Zustandsübergang PreOP => SafeOP durchgeführt. Bei einem Fehler kann die Ursache mittels S-0-0021 ausgewertet werden.
S-0-0128	C5200 Umschaltvorbereitung auf Komm.-Phase 4 Wird beim EtherCAT Zustandsübergang SafeOP => OP durchgeführt. Bei einem Fehler kann die Ursache mittels S-0-0022 ausgewertet werden.

Folgende IDN haben bei SoE eine angepasste Bedeutung:	
S-0-0006	Sendezeitpunkt Antriebs-Telegramm (T1) T1 legt die Zeitverschiebung fest vom EtherCAT Sync-Signal bis zum Zeitpunkt, in dem die Applikation neue AT-Daten im ESC-Speicher zur Verfügung stellen soll.
S-0-0014	Schnittstellen-Status Dieser Parameter soll den DL Status, AL Status und AL Status Code von EtherCAT anzeigen.
S-0-0028	Fehlerzähler MST Zähler für fehlende Datagramme der zyklischen Daten, als auch RX-Fehlerzähler und Lost-Link Zähler von EtherCAT.
S-0-0089	T2 Sendezeitpunkt MDT T2 legt die Zeitverschiebung fest vom EtherCAT Sync-Signal bis zum Zeitpunkt, in dem neue Master Daten im ESC-Speicher verfügbar sind.

FoE: File access over EtherCAT

FoE ist ein Protokoll, vergleichbar zu TFTP, um Dateien übertragen zu können. Üblicherweise wird FoE für Firmware-Update verwendet, daher ist FoE ggf. auch nur im EtherCAT Zustand ‚Bootstrap' möglich (geräte-spezifisch).

Siehe auch Kap. "FoE Fehlercodes" auf Seite 193.

EoE: Ethernet over EtherCAT

Mittels EoE kann Standard Ethernet Kommunikation (z.B. TCP/IP) über EtherCAT getunnelt werden. Hierzu wird ein "virtueller Ethernet Switch" im EtherCAT Master benötigt.

Ein IP-fähiges Gerät (z.B. Laptop) kann nicht an einer beliebigen Stelle im EtherCAT Netzwerk angeschlossen werden, sondern hierfür wird ein "Switch Port" benötigt (z.B. spezielle Klemme mit "Switch Port" ⇔ ist kein EtherCAT Port).

Alternativ kann bei einem EtherCAT Slave Gerät auch ein "IP-Port" konfiguriert werden (wenn das Gerät EoE / IP-Port unterstützt, z.B. integrierter Web-Server).

In beiden Fällen benötigt der EtherCAT Slave innerhalb des EtherCAT Netzwerkes eine "virtuelle MAC Adresse", die in der EtherCAT Konfiguration festgelegt wird. Die EoE-Einstellungen werden dann beim Bus-Hochlauf (Übergang Init => PreOP) vom Master in die Slave Geräte geschrieben. EoE ist wie alle Mailbox-Protokolle erst ab Zustand PreOP möglich.

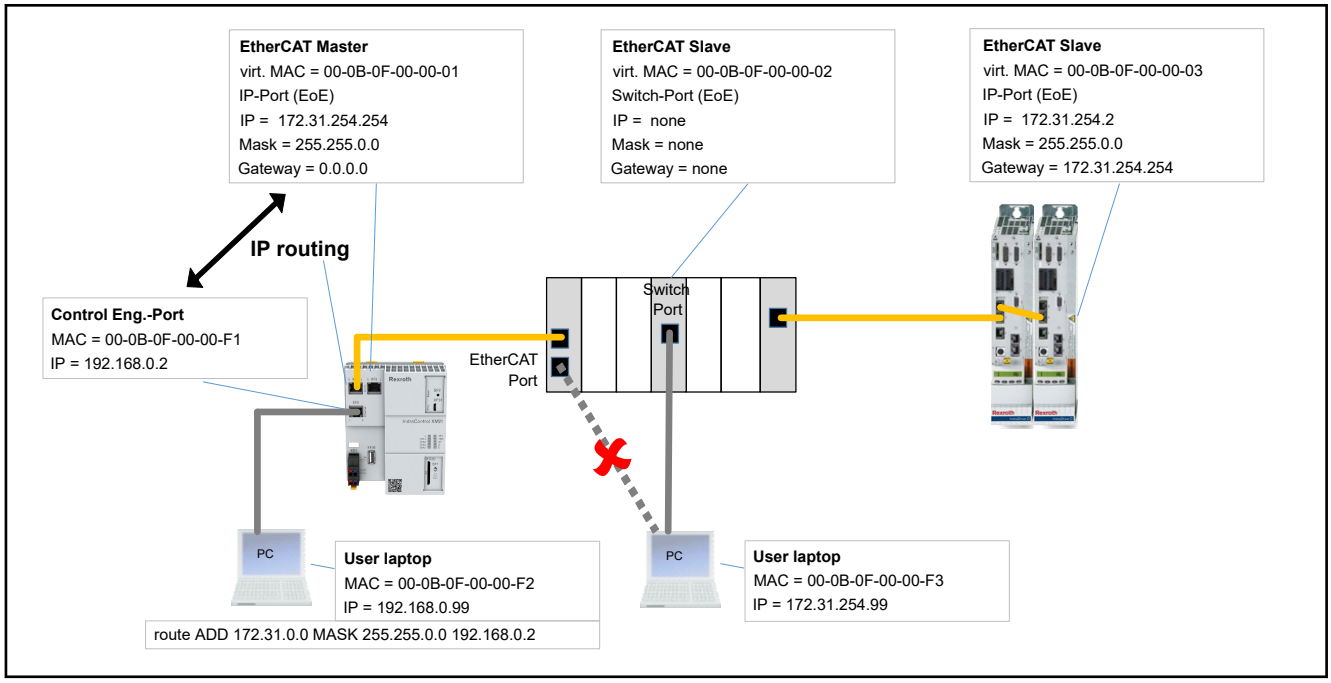


Abb. 8-6: EoE Konfigurations-Beispiel

VoE: Vendor specific protocol over EtherCAT

VoE dient zur Übertragung von Hersteller-spezifischen Protokollen.

AoE: ADS over EtherCAT

AoE ist eine geräte- und feldbusunabhängige Schnittstelle von Beckhoff (ADS: Automation Device Specification).

Slaves ohne Mailbox Kommunikation

EtherCAT Slave Geräte müssen kein Mailbox Protokoll unterstützen. Zum Beispiel unterstützen einfache digitale Ein-/Ausgangsklemmen überhaupt kein Mailbox Protokoll. Bei diesen Slave Geräten können dann nur die Register im EtherCAT Slave Controller (ESC) gelesen und geschrieben werden, was jeder EtherCAT Slave unterstützen muss.

8.2.6 Telegramm-Aufbau

EtherCAT verwendet Standard Ethernet Frames (IEEE 802.3) mit einem eigenen EtherType 0x88A4. Ethernet Telegramme eines anderen Typs werden vom EtherCAT Slave Controller nicht unterstützt und müssen durch EtherCAT getunnelt werden (z.B. EoE).

Der EtherCAT Frame besteht aus mehreren Sub-Telegrammen bzw. Datagrammen, die verschiedene Kommandos enthalten. Beispielsweise ist jede SyncUnit (siehe Kap. 8.2.8 "SyncUnits" auf Seite 185) ein eigenes Datagramm als auch azyklische Kommunikation (Mailbox) oder sonstige Kommandos sind eigene Datagramme.

In einem EtherCAT Frame können bis zu 15 Datagramme zusammengefasst werden. Bei mehr als 15 Datagrammen werden mehrere Frames benötigt.

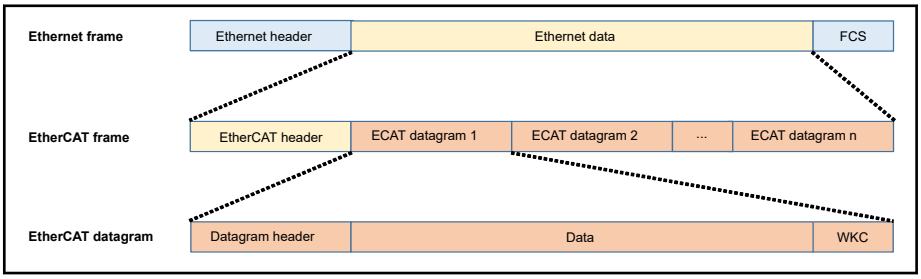


Abb. 8-7: EtherCAT Telegramm-Aufbau

8.2.7 WorkingCounter

Durch die Checksumme des Ethernet Frames (FCS : Frame Check Sequence) wird die korrekte Datenübertragung auf dem Bus sichergestellt.

Zur Überprüfung der korrekten Bearbeitung eines Kommandos in den Slave Geräten besitzt jedes Datagramm einen **WorkingCounter** (WKC). Der WKC wird vom Master mit dem Wert 0 abgeschickt. Jeder adressierte Slave inkrementiert den WKC dann abhängig vom Kommando. Nachdem der Frame dann alle Slaves durchlaufen hat und wieder im Master ankommt, kann der Master den Erwartungswert (Sollwert) des WKC mit dem Istwert vergleichen und somit die korrekte Bearbeitung des Kommandos überprüfen.

Für die Gültigkeit der zyklischen Prozessdaten wird der Status der WorkingCounter berücksichtigt (siehe "PdStatus", Kap. 8.5.12 "Registerkarte 'E/A-Abbild'" auf Seite 236).

8.2.8 SyncUnits

SyncUnit

Eine SyncUnit beschreibt eine logische Gruppierung von zyklischen Daten, die konsistent und synchron ausgetauscht werden sollen. Im einfachsten Fall sind alle EtherCAT Slaves in einer SyncUnit zusammengefasst oder es könnte auch einzelne SyncUnits für jeden Slave geben.

In der Regel werden optionale Slaves (z.B. Slaves, die im laufenden Betrieb "ausfallen dürfen") oder Maschinenmodule in einer eigenen SyncUnit gruppiert. Auf diese Weise besitzen diese Teilnehmer bzw. diese SyncUnit ein eigenes Datagramm und damit einen eigenen WorkingCounter. Zum Beispiel führt ein fehlender Teilnehmer dann nur zu einem ungültigen WorkingCounter dieser SyncUnit und alle anderen SyncUnits haben weiterhin einen gültigen Datenstatus.

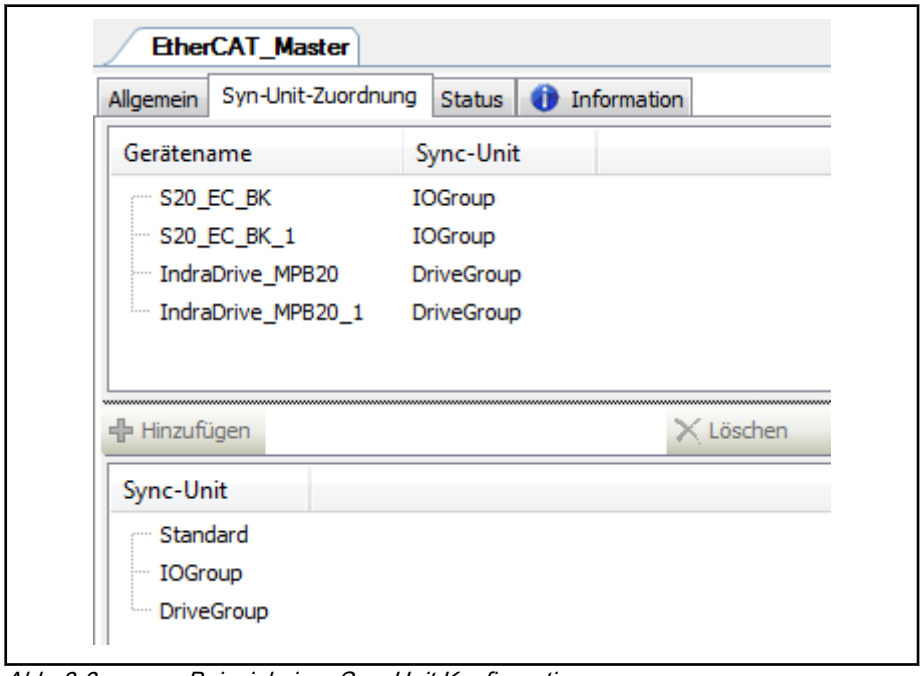


Abb. 8-8: Beispiel einer SyncUnit Konfiguration

Unterschiedliche EtherCAT Zykluszeiten

Mit SyncUnits können auch unterschiedliche EtherCAT Zykluszeiten realisiert werden. Hierbei werden die SyncUnits (= Datagramme) dann in eigenen Frames mit unterschiedlichen Zykluszeiten gesendet. Zum Beispiel werden schnelle I/O-Baugruppen und Antriebsgruppen in einzelnen SyncUnits mit unterschiedlichen Zykluszeiten gruppiert.

Der Rexroth EtherCAT Master unterstützt keine unterschiedlichen EtherCAT Zykluszeiten!

8.2.9 Hot Connect

Hot Connect

Unter Hot Connect (HC) versteht man allgemein, dass während dem laufenden Betrieb (Bus-Zustand "Operational") Slaves ab- und angesteckt werden können. Hierbei sind auch die Anwendungsfälle enthalten, dass mehrere Teilnehmer bzw. Maschinenmodule nicht vorhanden sein müssen bzw. eine Inbetriebnahme eines Teils der Anlage bereits möglich ist.

HC-Gruppe

Durch die Konfiguration von HC-Gruppen werden Slaves automatisch in einer eigenen SyncUnit gruppiert. Eine HC-Gruppe kann als "optional" deklariert werden, so dass der EtherCAT Master eine nicht vorhandene HC-Gruppe nicht als Fehler erkennt (flexible Topologien). Der erste Slave einer HC-Gruppe wird durch ein Identifikationsverfahren (siehe Kap. 8.2.3 "Adressierungs- und Identifikationsverfahren" auf Seite 179) erkannt. Des Weiteren kann bei einer HC-Gruppe auch festgelegt werden, ob diese Gruppe nur an einer bestimmten Topologie-Position (Port des Vorgänger EtherCAT Gerätes) angeschossen werden darf oder eine beliebige Position haben kann.

Der Rexroth EtherCAT Master unterstützt nicht die Konfiguration von HC-Gruppen!

8.2.10 Diagnose

Allgemein

EtherCAT Diagnosen stehen auf verschiedenen Kommunikationsschichten zur Verfügung.

Eine zyklusgenaue Diagnose wird im Master generiert. Dies umfasst zum einen den (Master-) Bus Zustand, als auch die Auswertung des WorkingCounter (WKC). Durch die Auswertung des WKC wird zyklusgenau detektiert, ob ein Datagramm (z.B. zyklische Prozessdaten) korrekt bearbeitet wurde und der Datenstatus gültig ist. Alle Slaves, die in einem Datagramm (z.B. SyncUnit) mit fehlerhaftem WKC adressiert sind, haben somit dann einen ungültigen Datenstatus (siehe "PdStatus", Kap. 8.5.12 "Registerkarte 'E/A-Abbild'" auf Seite 236).

Eine detaillierte Diagnose, ob z.B. ein Slave einen EtherCAT Zustandsfehler hat oder nicht mehr verbunden ist bzw. evtl. ein Link-Problem vorliegt (Port-Status), wird dann vom Master azyklisch ermittelt.

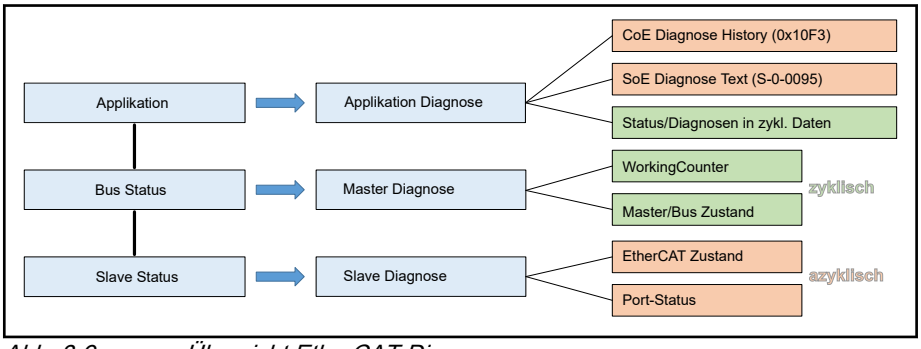


Abb. 8-9: Übersicht EtherCAT Diagnosen

Auf der Applikationsebene haben viele Slaves Statusmeldungen oder einfache Diagnosen-Daten bereits in den zyklischen Daten verfügbar. Detail-Diagnosen können dann azyklisch ausgelesen werden, z.B. bei CoE Slaves gibt

es optional das Diagnose History Objekt (0x10F3) oder bei SoE Slaves den Diagnose-Text (S-0-0095).

Für die Auswertung der Diagnosen im SPS-Programm stehen beim Rexroth EtherCAT Master verschiedene Funktionsbausteine in der RIL_Ethercat Bibliothek zur Verfügung.

Manche CoE Slaves unterstützen auch CoE Emergency Nachrichten. Diese Emergency Nachrichten haben größtenteils aber nur geräte-spezifische Informationen und auch eine Quittierung ist nicht erforderlich. Daher werden diese i.d.R. nur ins Steuerungs-Logbuch eingetragen.

AL Status Codes

Der "AL Status Code" (AL: Application Layer) beschreibt den Fehlercode eines Slave Gerätes, wenn eine angeforderte Zustandsänderung nicht ausgeführt werden kann oder wenn der Slave einen internen Fehler entdeckt.

Der Rexroth EtherCAT Master liest im Fehlerfall automatisch den AL Status Code (ESC Register 0x0134:0x0135) aus und quittiert die Fehler-Indikation im Slave. Beim nächsten Zustandswechsel des Slaves wird der ausgelesene AL Status Code im Master zurückgesetzt.



Tritt ein unspezifizierter Fehler auf oder eine Konfiguration wird als ungültig erkannt, sind Details in der Produkt-Dokumentation des Slave Gerätes nachzuschlagen oder der Hersteller des Slave Gerätes zu kontaktieren.

In nachfolgender Tabelle werden folgende Abkürzungen verwendet:

- I = Zustand "Init"
- B = Zustand "Bootstrap"
- P = Zustand "Pre-Operational"
- S = Zustand "Safe-Operational"
- O = Zustand "Operational"

AL Status Code	Beschreibung	Aktueller Zustand oder Zustandsänderung	Resultierender Zustand
0x0000	Kein Fehler	Alle	Aktueller Zustand
0x0001	Unspezifizierter Fehler	Alle	Alle
0x0002	Kein Speicher	Alle	Alle
0x0003	Ungültige Geräte-Konfiguration	P → S	P
0x0004	Ungültige Revision	P → S	P
0x0005	Reserviert aus Kompatibilitäts-Gründen	-	-
0x0006	SII / EEPROM Daten passen nicht mit Firmware zusammen	I → P	I
0x0007	Firmware Update nicht erfolgreich	Alle	Alle
0x000E	Lizenz Fehler	Alle	I
0x0011	Ungültige Zustandsänderung angefordert, siehe Kap. 8.2.4 "Zustandsmaschine" auf Seite 180	I → S, I → O, P → O, P → B, S → B, O → B	Aktueller Zustand
0x0012	Unbekannter Zustand angefordert	Alle	Aktueller Zustand

AL Status Code	Beschreibung	Aktueller Zustand oder Zustandsänderung	Resultierender Zustand
0x0013	Zustand Bootstrap nicht unterstützt	I → B	I
0x0014	Keine gültige Firmware	I → P	I
0x0015	Ungültige Mailbox Konfiguration (Bootstrap)	I → B	I
0x0016	Ungültige Mailbox Konfiguration (PreOP)	I → P	I
0x0017	Ungültige Sync Manager Konfiguration	P → S, S → O	Aktueller Zustand
0x0018	Keine gültigen Eingänge verfügbar	O, S → O	S
0x0019	Keine gültigen Ausgänge möglich	O, S → O	S
0x001A	Synchronisations-Fehler (verschiedene Ursachen)	O, S → O	S
0x001B	Sync Manager Watchdog	O, S	S
0x001C	Ungültige Sync Manager Typen	O, S, P → S	S
0x001D	Ungültige Sync Manager Konfiguration (Ausgänge)	O, S, P → S	S
0x001E	Ungültige Sync Manager Konfiguration (Eingänge)	O, S, P → S	P
0x001F	Ungültige Watchdog Konfiguration	O, S, P → S	P
0x0020	Slave benötigt Kalt-Start	Alle	Aktueller Zustand
0x0021	Slave benötigt Zustand "Init"	B, P, S. O	Aktueller Zustand
0x0022	Slave benötigt Zustand "Pre-Operational"	S, O	S, O
0x0023	Slave benötigt Zustand "Safe-Operational"	O	O
0x0024	Ungültiges Mapping Eingangsdaten	P → S	P
0x0025	Ungültiges Mapping Ausgangsdaten	P → S	P
0x0026	Inkonsistente Einstellungen (allgemein)	P → S	P
0x0027	Free Run nicht unterstützt	P → S	P
0x0028	Sync Modus nicht unterstützt	P → S	P
0x0029	Free Run benötigt Sync Manager im 3 Puffer Modus	P → S	P
0x002A	Background Watchdog	S, O	P
0x002B	Keine gültigen Ein- und Ausgangsdaten	O, S → O	S
0x002C	Fataler Sync Fehler, Sync0 oder Sync1 Signal nicht mehr empfangen.	O	S
0x002D	Kein Sync Signal, Slave wartet in "Safe-Operational" auf Sync0/Sync1 Signal (S → O Timeout).	S → O	S
0x002E	Zykluszeit zu klein, die eingestellte EtherCAT Zykluszeit wird vom Slave nicht unterstützt.	S → O	S

AL Status Code	Beschreibung	Aktueller Zustand oder Zustandsänderung	Resultierender Zustand
0x0030	Ungültige DC Sync Konfiguration	O, S → O, P → S	S, P
0x0031	Ungültige DC Latch Konfiguration	O, S → O, P → S	S, P
0x0032	PLL Fehler, Master nicht synchronisiert, aber mind. ein DC Event empfangen.	O, S → O	S
0x0033	DC Sync IO Fehler, mehrere Synchronisations-Fehler, Master u. Slave nicht mehr synchronisiert.	O, S → O	S
0x0034	DC Sync Timeout Fehler mehrere Synchronisations-Fehler, zu viele SM Events ausgelassen	O, S → O	S
0x0035	Ungültige DC Zykluszeit	P → S	P
0x0036	DC Sync0 Zykluszeit nicht funktionsfähig mit Applikation	P → S	P
0x0037	DC Sync1 Zykluszeit nicht funktionsfähig mit Applikation	P → S	P
0x0041	Mailbox AoE	B, P, S, O	Aktueller Zustand
0x0042	Mailbox EoE	B, P, S, O	Aktueller Zustand
0x0043	Mailbox CoE	B, P, S, O	Aktueller Zustand
0x0044	Mailbox FoE	B, P, S, O	Aktueller Zustand
0x0045	Mailbox SoE	B, P, S, O	Aktueller Zustand
0x004F	Mailbox VoE	B, P, S, O	Aktueller Zustand
0x0050	EEPROM kein Zugriff	Alle	Alle
0x0051	EEPROM Fehler	Alle	Alle
0x0052	Externe Hardware nicht betriebsbereit	Alle	Alle
0x0060	Slave führte Neustart durch	Alle	I
0x0061	Geräte Identifikationswert aktualisiert	P	P
0x0070	Ungültige Modul-Konfiguration, bei einem Slave mit Modular-Device-Profile stimmen die konfigurierten Module (0xF030) nicht mit den physikalischen Modulen (0xF050) überein.	P → S	P
0x00F0	Applikations-Steuerung verfügbar	I	I
< 0x8000	Reserviert	-	-
0x8000 - 0xFFFF	Hersteller-spezifisch	-	-

Siehe auch:

- Kap. 8.8.4 "S20-EC-BK: Buskoppler schaltet nicht in OP (Hinweise zu AL Status Codes)" auf Seite 243

CoE SDO Abortcodes

Bei azyklischer CoE Mailbox Kommunikation können folgende standardisierte SDO Abortcodes auftreten. Diese werden auf Fehlercodes der Steuerung abgebildet (→ ECM Fehlercode).

SDO Abortcode	Beschreibung	ECM Fehlercode
-	Ecat CoE: Protokoll nicht unterstützt	F0ED0032
0x0503 0000	Ecat SDO: Toggle Bit kein Zustandswechsel	F0ED0040
0x0504 0000	Ecat SDO: SDO Protokoll Zeitüberschreitung	F0ED0041
0x0504 0001	Ecat SDO: Client/Server ungültiger Kommandobezeichner	F0ED0042
0x0504 0002	Ecat SDO: Ungültige Blockgröße (nur im Block Modus)	F0ED0043
0x0504 0003	Ecat SDO: Ungültige Sequenznummer (nur im Block Modus)	F0ED0044
0x0504 0004	Ecat SDO: CRC Fehler (nur im Block Modus)	F0ED0045
0x0504 0005	Ecat SDO: Kein freier Speicher	F0ED0046
0x0601 0000	Ecat SDO: Nicht unterstützter Zugriff auf ein Objekt	F0ED0047
0x0601 0001	Ecat SDO: Lesezugriff auf ein nur schreibbares Objekt	F0ED0048
0x0601 0002	Ecat SDO: Schreibzugriff auf ein nur lesbares Objekt	F0ED0049
0x0601 0003	Ecat SDO: Schreibzugriff auf Subindex fehlgeschlagen	F0EE0004
0x0601 0004	Ecat SDO: Complete Access nicht unterstützt	F0EE0005
0x0601 0005	Ecat SDO: Objekt größer als Mailbox	F0EE0006
0x0601 0006	Ecat SDO: Objekt in RxPDO gemappt, Download blockiert	F0EE0007
0x0602 0000	Ecat SDO: Objekt ist nicht im Objektverzeichnis vorhanden	F0ED004A
0x0604 0041	Ecat SDO: Objekt kann nicht in ein PDO gemappt werden	F0ED004B
0x0604 0042	Ecat SDO: PDO Länge überschritten	F0ED004C
0x0604 0043	Ecat SDO: Allgemeine Inkompatibilität (Parameter)	F0ED004D
0x0604 0047	Ecat SDO: Allgemeine Inkompatibilität (Geräteintern)	F0ED004E
0x0606 0000	Ecat SDO: Zugriff fehlgeschlagen (Hardwarefehler)	F0ED004F
0x0607 0010	Ecat SDO: Datentyp und Längenkennung differieren	F0ED0050
0x0607 0012	Ecat SDO: Falscher Datentyp (Parameter zu groß)	F0ED0051
0x0607 0013	Ecat SDO: Falscher Datentyp (Parameter zu klein)	F0ED0052
0x0609 0011	Ecat SDO: Subindex existiert nicht	F0ED0053
0x0609 0030	Ecat SDO: Parameterwert außerhalb gültigem Bereich	F0ED0054
0x0609 0031	Ecat SDO: Parameterwert größer als Maximalwert	F0ED0055
0x0609 0032	Ecat SDO: Parameterwert kleiner als Minimalwert	F0ED0056

SDO Abortcode	Beschreibung	ECM Fehlercode
0x0609 0033	Ecat SDO: Module Ident Liste unpassed Ungültige Modul-Konfiguration, bei einem Slave mit Modular-Device-Profile (MDP) stimmen die konfigurierten Module (0xF030) nicht mit den physikalischen Modulen (0xF050) überein.	F0ED005E
0x0609 0036	Ecat SDO: Maximalwert kleiner als Minimalwert	F0ED0057
0x0800 0000	Ecat SDO: Allgemeiner Fehler	F0ED0058
0x0800 0020	Ecat SDO: Daten übertragen oder speichern fehlgeschlagen	F0ED0059
0x0800 0021	Ecat SDO: Fehler Datenübertragung (lokale Steuerung)	F0ED005A
0x0800 0022	Ecat SDO: Fehler Datenübertragung (Gerätezustand)	F0ED005B
0x0800 0023	Ecat SDO: Objektverzeichnis existiert nicht	F0ED005C
-	Ecat SDO: Unbekannter Fehlercode	F0ED005D

SoE Fehlercodes

Bei azyklischer SoE Mailbox Kommunikation können folgende standardisierte SoE Fehlercodes auftreten. Diese werden auf Fehlercodes der Steuerung abgebildet (→ ECM Fehlercode).

Sercos Fehlercode	Beschreibung	ECM Fehlercode
-	Ecat SoE: Protokoll nicht unterstützt	F0ED0035
-	Ecat SoE: Ungültiger Zugriff auf Element 0	F0ED0078
0x1001	Ecat SoE: Nicht vorhanden	F0ED0079
0x1009	Ecat SoE: Ungültiger Zugriff auf Element 1	F0ED007A
0x2001	Ecat SoE: Kein Name	F0ED007B
0x2002	Ecat SoE: Übertragungsfehler, Name zu kurz	F0ED007C
0x2003	Ecat SoE: Übertragungsfehler, Name zu lang	F0ED007D
0x2004	Ecat SoE: Name kann nicht geändert werden (read only)	F0ED007E
0x2005	Ecat SoE: Name ist zur Zeit schreibgeschützt	F0ED007F
0x3002	Ecat SoE: Übertragungsfehler, Attribut zu kurz	F0ED0080
0x3003	Ecat SoE: Übertragungsfehler, Attribut zu lang Übertragungsfehler, Attribut zu lang	F0ED0081
0x3004	Attribut kann nicht geändert werden (nur lesbar)	F0ED0082
0x3005	Attribut ist zur Zeit schreibgeschützt	F0ED0083
0x4001	Keine Einheit	F0ED0084
0x4002	Übertragungsfehler, Einheit zu kurz	F0ED0085
0x4003	Übertragungsfehler, Einheit zu lang	F0ED0086
0x4004	Einheit kann nicht verändert werden (nur lesbar)	F0ED0087
0x4005	Einheit ist zur Zeit schreibgeschützt	F0ED0088
0x5001	Kein Min-Wert	F0ED0089

Sercos Fehlercode	Beschreibung	ECM Fehlercode
0x5002	Übertragungsfehler, Min-Wert zu kurz	F0ED008A
0x5003	Übertragungsfehler, Min-Wert zu lang	F0ED008B
0x5004	Min-Wert kann nicht geändert werden (nur lesbar)	F0ED008C
0x5005	Min-Wert ist zur Zeit schreibgeschützt	F0ED008D
0x6001	Kein Max-Wert	F0ED008E
0x6002	Übertragungsfehler, Max-Wert zu kurz	F0ED008F
0x6003	Übertragungsfehler, Max-Wert zu lang	F0ED0090
0x6004	Max-Wert kann nicht geändert werden (nur lesbar)	F0ED0091
0x6005	Max-Wert ist zur Zeit schreibgeschützt	F0ED0092
0x7001	Kein Betriebsdatum	F0ED0093
0x7002	Übertragungsfehler, Betriebsdatum zu kurz	F0ED0094
0x7003	Übertragungsfehler, Betriebsdatum zu lang	F0ED0095
0x7004	Betriebsdatum nicht änderbar (nur lesbar)	F0ED0096
0x7005	Betriebsdatum ist zur Zeit schreibgeschützt	F0ED0097
0x7006	Betriebsdatum ist kleiner als Min-Wert	F0ED0098
0x7007	Betriebsdatum ist größer als Max-Wert	F0ED0099
0x7008	Ungültiges Betriebsdatum	F0ED009A
0x7009	Betriebsdatum ist schreibgeschützt (Passwort)	F0ED009B
0x700A	Betriebsdatum ist schreibgeschützt (zykl. konf.)	F0ED009C
0x700B	Ungültige Adressierung (Datencontainer, Liste)	F0ED009D
0x700C	Betriebsdatum ist schreibgeschützt (anderweitig)	F0ED009E
0x7010	Befehlsfolge bereits aktiv	F0ED009F
0x7011	Befehlsfolge nicht unterbrechbar	F0ED0100
0x7012	Befehlsfolge zur Zeit nicht ausführbar	F0ED0101
0x7013	Befehlsfolge nicht ausführbar, Param. ungültig	F0ED0102
-	Falsche Antriebsnummer in Antwort	F0ED0103
-	Falsche IDN in Antwort	F0ED0104
-	Fragment verloren	F0ED0105
-	Empfangspuffer voll	F0ED0106
-	Kein Datenstatus	F0ED0107
-	Kein Standardwert	F0ED0108
-	Übertragungsfehler, Standardwert zu lang	F0ED0109
-	Standardwert kann nicht geändert werden	F0ED010A
-	Ungültige Antriebsnummer	F0ED010B
-	Allgemeiner Fehler	F0ED010C
-	Kein Element adressiert	F0ED010D

FoE Fehlercodes

Bei azyklischer FoE Mailbox Kommunikation können folgende standardisierte FoE Fehlercodes auftreten. Diese werden auf Fehlercodes der Steuerung abgebildet (→ ECM Fehlercode).

FoE Fehlercode	Beschreibung	ECM Fehlercode
-	Mailbox FoE nicht unterstützt	F0ED0034
-	Mailbox FoE nicht unterstützt in Bootstrap	F0ED010F
0x8000	Herstellerspezifischer FoE Fehler (nicht definiert)	F0ED0060
0x8001	Nicht gefunden	F0ED0061
0x8002	Zugriff verweigert	F0ED0062
0x8003	Laufwerk voll	F0ED0063
0x8004	Nicht erlaubt	F0ED0064
0x8005	Falsche Paketnummer	F0ED0065
0x8006	Bereits vorhanden	F0ED0066
0x8007	Nutzer nicht vorhanden	F0ED0067
0x8008	Nur in Bootstrap	F0ED0068
0x8009	Dateiname in Bootstrap ungültig	F0ED0069
0x800A	Keine Zugriffsrechte	F0ED006A
0x800B	Programmfehler	F0ED006B
0x800C	Falsche Prüfsumme	F0ED006C
0x800D	Firmware passt nicht zur Hardware	F0ED006D
0x800E	Reserviert	-
0x800F	Datei nicht gefunden	F0ED006F
0x8010	Datei-Header fehlt	F0EE0001
0x8011	Fehler Flash-Speicher	F0EE0002
0x8012	Datei inkompatibel	F0EE0003
-	Datei größer als maximal erlaubt	F0ED017A

8.2.11 Synchronisation

Allgemein

Für jeden EtherCAT Slave kann individuell ein Synchronisations-Modus eingestellt werden. Welche Modi von einem Slave Gerät unterstützt werden, ist der Dokumentation bzw. der Gerätebeschreibung des Slave Gerätes zu entnehmen. Man unterscheidet allgemein folgende Synchronisations-Modi:

- Free Run (keine Synchronisation)
- Synchronisation mit SM-Event (SM : SyncManager [1])

[1] *Die Hauptaufgabe des SyncManagers (SM) ist es, die Daten-Konsistenz im DPRAM des ESC sicherzustellen. Es kann bis zu 16 unabhängige SyncManager Kanäle geben. In der Standard-Konfiguration werden 4 verwendet: SM0 = Mailbox-Out, SM1 = Mailbox-In, SM2 = Prozessdaten-Out, SM3 = Prozessdaten-In.*

- Synchronisation mit DC-Sync (DC : Distributed Clocks / "Verteilten Uhren")

Folgendes Beispiel zeigt die DC-Konfiguration des Rexroth IndraDrive MPx20 SoE. Der Konfigurator zeigt die vordefinierten Konfigurationen aus der Gerätebeschreibung (ESI) an. Neben dem "Free Run" Modus werden drei DC-Modi mit unterschiedlichen "Sync0Shift"-Zeiten angeboten (Zeitverschiebung im Slave zum Sync0 Event).

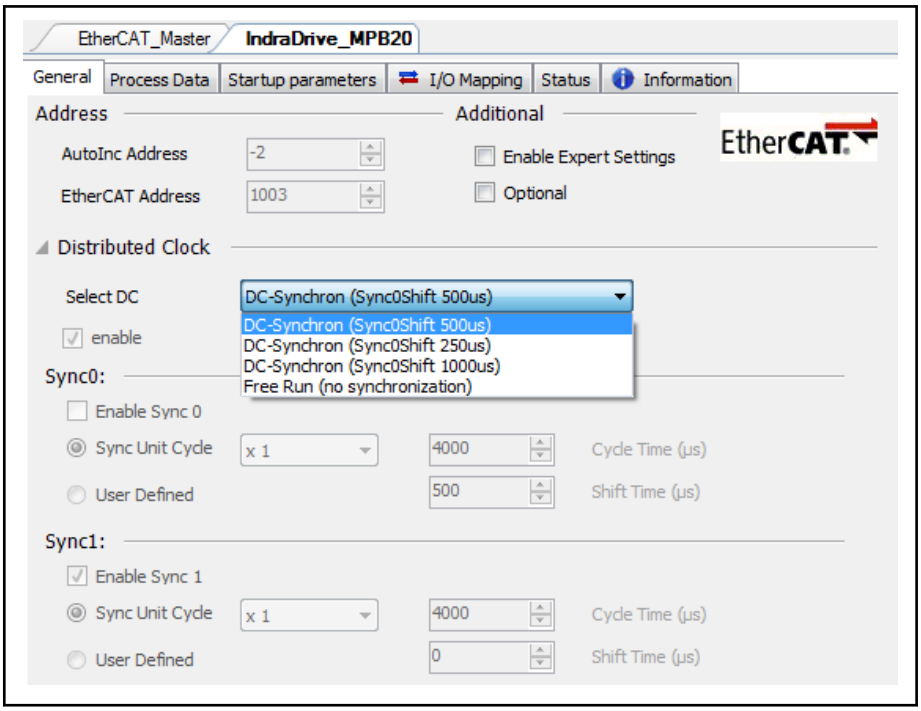


Abb. 8-10: DC-Konfiguration mit Rexroth IndraDrive MPx20 SoE

Free Run

Der EtherCAT Kommunikationszyklus und die Ein-/Ausgangsdaten-Verarbeitung im Slave Gerät laufen unabhängig voneinander. D.h. der lokale Applikationszyklus im Slave Gerät kann schneller oder langsamer sein als der EtherCAT Buszyklus.

In folgenden Abbildungen werden mit den "In" / "Out"-Balken jeweils die Kopier- und Verarbeitungszeiten in den Slave-Geräten dargestellt.

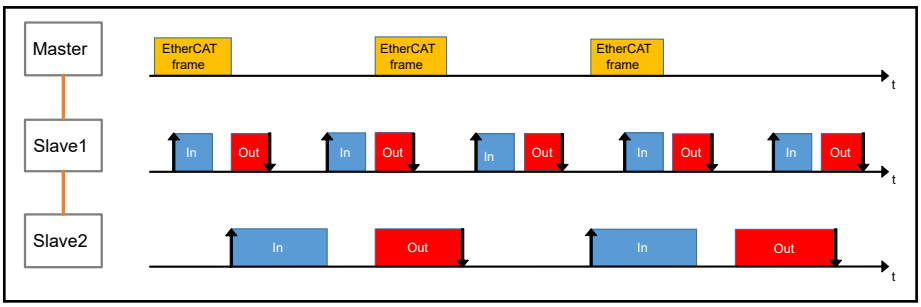


Abb. 8-11: Modus "Free Run"

Zudem kann ein SM-Event zur Synchronisation nach Telegramm-Empfang im Slave verwendet werden.

Bei CoE sind u.a. folgende Parameter des SyncManager Objekts 0x1C3m relevant (mit der Standard-Konfiguration "SM2/3" für die Prozessdaten gilt: 0x1C32 für Ausgänge u. 0x1C33 für Eingänge):

SI	Beschreibung	Zugriff	Verwendung	Erklärung / Defaultwert
1	Synchronization Type	R / RW	M	Synchronisations-Typ: 0x00: Free Run
2	Cycle Time	R / RW	O	Lokale Zykluszeit des Applikations-Controllers in ns
4	Synchronization Types supported	R	M	Unterstützte Synchronisations-Typen: Bit 0 = 1 : Free Run unterstützt
5	Minimum Cycle Time	R	C	Minimale Zykluszeit: maximale Laufzeit des lokalen Applikationszyklus (erforderlich, wenn SI=2 geschrieben werden kann)

SI = Subindex, R = Read only, RW = Read/Write, M = erforderlich, O = optional, C = bedingt

Synchronisation mit SM-Event

Diese Synchronisation wird oft auch als "SM2/3"-Event bezeichnet, da die SyncManager 0/1 üblicherweise für Mailbox Out/In verwendet werden und SyncManager 2/3 dann für die Prozessdaten Out/In.

Die Ausgangsdaten-Verarbeitung wird im Slave durch den Empfang des EtherCAT Telegramms getriggert durch den SyncManager. Der Zeitpunkt, wann die Ausgangsdaten gültig werden, kann variieren aufgrund von Jitter beim Sendezeitpunkt des Masters, durch die Laufzeit des Telegramms bei der Übertragung, als auch durch Verarbeitungszeiten in den Slaves. Für die Übernahme der Eingangsdaten kann optional ein "Input Shift" konfiguriert werden (geräte-individuell), damit die Eingangsdaten beim nächsten SM-Event (nächster Telegramm-Durchlauf) möglichst "aktuell" sind.

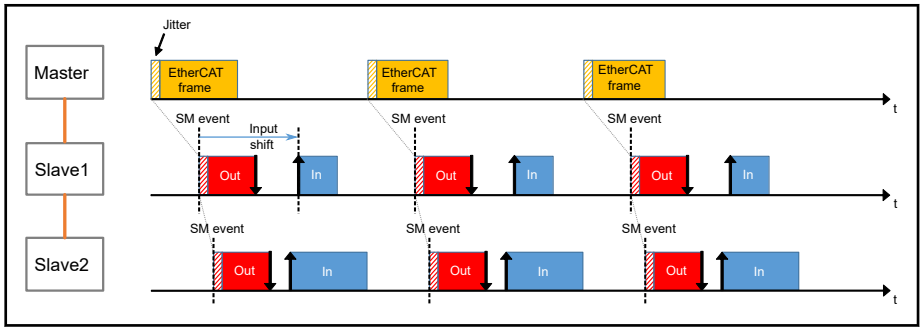


Abb. 8-12: Modus "Synchronisation mit SM-Event"

Bei CoE sind u.a. folgende Parameter des SyncManager Objekts 0x1C3m relevant (mit der Standard-Konfiguration "SM2/3" für die Prozessdaten gilt: 0x1C32 für Ausgänge u. 0x1C33 für Eingänge):

SI	Beschreibung	Zu-griff	Ver we nd un g	Erklärung / Defaultwert
1	Synchronization Type	RW	M	Synchronisations-Typ: Ausgänge (0x1C32): 0x01 : Synchron mit SM2 Event Eingänge (0x1C33): 0x01 : Synchron mit SM3 Event (wenn keine Ausgänge verfügbar sind) 0x22 : Synchron mit SM2 Event (wenn Ausgänge verfügbar sind)
2	Cycle Time	R / RW	O	EtherCAT Kommunikationszyklus in ns (Zeit zwischen zwei SM Events)
3	Shift Time	RW	-/ M	Zeitverschiebung Eingangsdaten-Übernahme (“Shift Input Latch“) (nur Eingänge / 0x1C33) in ns
4	Synchronization Types supported	R	M	Unterstützte Synchronisations-Typen: Bit 1 = 1: Synchron SM unterstützt
5	Minimum Cycle Time	R	M	Minimale Zykluszeit, die vom Slave unterstützt wird (maximale Zeit des lokalen Applikationszyklus in ns)
8	Get Cycle Time	RW	C	wird im SM- und DC-Modus bei einer variablen Zykluszeit verwendet: Bit 0: Messung der lokalen Zeit gestartet Bit 1: Reset der Fehlerzähler
11	SM-Event missed	R	O	Dieser Fehlerzähler wird inkrementiert, wenn das SM Event nicht innerhalb der erwarteten Zeit ankommt (in der Folge können die Daten nicht mehr kopiert werden)
12	Cycle Time Too Small	R	M	Dieser Fehlerzähler wird inkrementiert, wenn die Zykluszeit zu klein ist, so dass der lokale Applikationszyklus nicht fertig ausgeführt werden kann (Eingangsdaten können bis zum nächsten SM Event nicht aufbereitet werden)

14	RxPDO Toggle Failed	R	O	Dieser Fehlerzähler wird inkrementiert, wenn der Slave RxPDO Toggle unterstützt und keine Statusänderung vom Master mehr empfangen wurde
32	Sync Error	R	C	Synchronisations-Fehler: sollte unterstützt werden, wenn SI = 11 vorhanden ist; kann in TxPDO gemappt werden. Wert 0 : kein Sync Fehler oder Sync nicht unterstützt Wert 1 : Sync Fehler

SI = Subindex, R = Read only, RW = Read/Write, M = erforderlich, O = optional, C = bedingt

Synchronisation mit DC-Sync

Für das Synchronisations-Verfahren "Verteilte Uhren" (DC : Distributed Clocks) gibt es allgemein verschiedene Möglichkeiten die Referenzuhr festzulegen:

- Master Shift: Der erste DC-Slave am Bus dient als Referenzuhr und Nachregelung des Timers des EtherCAT Masters.
- Bus Shift: Der EtherCAT Master stellt die Systemzeit und schreibt diese in den ersten DC-Slave am Bus (Schieben der Buszeit).
- Externe Referenzuhr (IEEE 1588)

Auf Basis der Referenzuhr wird in allen Teilnehmern eine globale Systemzeit eingeregelt. Mit Hilfe dieser Systemzeit werden in den EtherCAT Slave Controllern dann entsprechende "DC Sync" Events generiert zur Verarbeitung der Ein- u. Ausgangsdaten.

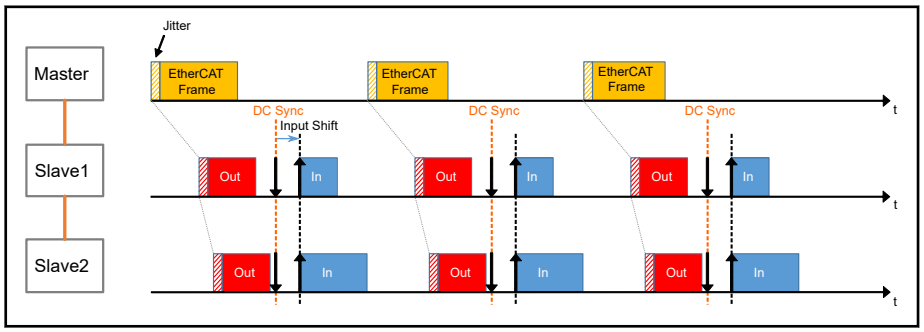


Abb. 8-13: Modus "Synchronisation mit DC-Sync" (vereinfachte Darstellung)

Neben dem Sync-Event "Sync0" wird i.d.R. noch ein untergeordnetes Sync-Event "Sync1" unterstützt (Funktionalität eines Standard EtherCAT Slave Controllers (ESC)). Welche Sync-Modi ein Slave Gerät unterstützt, sollte in dessen Gerätebeschreibungsdatei (ESI) enthalten sein oder ist in der Produkt-Dokumentation nachzuschlagen. Man unterscheidet allgemein folgende DC-Modi:

1. Synchron mit Sync0 Event
2. Synchron mit Sync0 Event, Input Shift mit Sync1 Event
3. Synchron mit Sync0 Event, Output Shift mit Sync1 Event
4. Synchron mit Sync1 Event
5. DC mit untergeordneten Zyklen (Sync1 Event nur jeden n-ten Sync0 Zyklus)

Der Zeitpunkt des Sync-Events wird im EtherCAT Master global eingestellt als "Sync Shift Time" (oder „Sync Offset“) bezogen auf den Master Applikati-

onszyklus. Für jeden Slave kann dann individuell eine Sync0 und Sync1 Zykluszeit eingestellt werden. Diese Sync-Zykluszeit ist ein Vielfaches des EtherCAT Kommunikationszyklus. Zudem ist auch eine Verschiebung ("Shift Time") bezogen auf den globalen Sync-Zeitpunkt möglich (siehe Abb. 8-14 "Slave Konfiguration der DC Einstellungen" auf Seite 198).

☞ Die DC Einstellungen sind normalerweise vordefinierte Konfigurationen aus der Gerätebeschreibung (ESI) eines Slaves und sollten nur von Experten geändert werden!

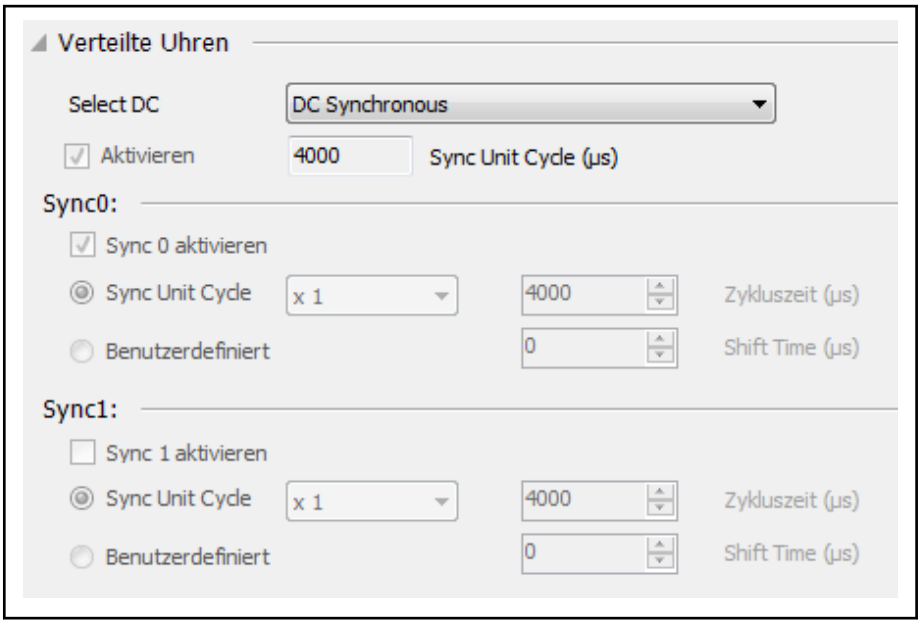


Abb. 8-14: Slave Konfiguration der DC Einstellungen

Bei CoE sind u.a. folgende Parameter des SyncManager Objekts 0x1C3m relevant (mit der Standard-Konfiguration "SM2/3" für die Prozessdaten gilt: 0x1C32 für Ausgänge u. 0x1C33 für Eingänge):

SI	Beschreibung	Zugriff	Verwendung	Erklärung / Defaultwert
1	Synchronization Type	RW	M	Synchronisations-Typ: 0x02: DC Synchron mit Sync0 Event 0x03: DC Synchron mit Sync1 Event
2	Cycle Time	R	O	Zykluszeit Sync0 Event, Zeit zwischen zwei Sync0 Events in ns (Register 0x09A3:0x09A0) Bei untergeordneten Zyklen: Zeit zwischen zwei Sync1 Events in ns (Register 0x09A7:0x09A4) Die Festlegung der Zykluszeiten Sync0/1 und ggf. Shift-Zeiten von Sync0/1 erfolgt im Konfigurator.

3	Shift Time	RW	M	Zeitverschiebung in ns: Ausgänge: Zeit zwischen Sync0 Event und "Outputs valid" (Initialisiert mit 0x1C32:09) Eingänge: Zeit zwischen Sync0 Event und "Input Latch" (initialisiert mit 0x1C33:05 – 0x1C33:06)
4	Synchronization Types supported	R	M	Unterstützte Synchronisations-Typen: Bit 4-2 : unterstützte DC-Modi 001 = DC Sync0 010 = DC Sync1 100 = untergeordneter Zyklus Bit 5-6: Shift Einstellungen 00 = Output Shift nicht unterstützt 01 = Input / Output Shift mit lokalem Timer 10 = Input / Output Shift mit Sync1
5	Minimum Cycle Time	R	M	Minimale Zykluszeit, die vom Slave unterstützt wird (maximale Zeit des lokalen Applikationszyklus in ns)
6	Calc and Copy Time	R	M	Berechnungs- und Kopier-Zeit in ns: minimale Zeit zw. SM2 Event und Sync0 Event. bei Sync1: minimale Zeit zw. Sync0 und Setzen der Ausgänge
8	Get Cycle Time	RW	C	wird im SM- und DC-Modus bei einer variablen Zykluszeit verwendet Bit 0: Messung der lokalen Zeit gestartet Bit 1: Reset der Fehlerzähler
9	Delay Time	R	M	Hardware Verzögerungszeit in ns: Zeit zwischen Trigger (Sync0 oder Sync1 Event) und Gültigkeit der Daten im Prozess (z.B. elektrische Signale)
11	SM-Event missed	R	O	Dieser Fehlerzähler wird inkrementiert, wenn das SM Event nicht innerhalb der erwarteten Zeit ankommt (in der Folge können die Daten nicht mehr kopiert werden)
12	Cycle Time Too Small	R	M	Dieser Fehlerzähler wird inkrementiert, wenn die Zykluszeit zu klein ist, so dass der lokale Applikationszyklus nicht fertig ausgeführt werden kann (Eingangsdaten können bis zum nächsten SM Event nicht aufbereitet werden)

13	Shift Time Too Short	R	O	Dieser Fehlerzähler wird inkrementiert, wenn das SM Event zu spät empfangen wird, so dass das Sync0 Event empfangen wird bevor "Calc+Copy Time" fertig ist (Master Telegramm zu spät; Konfigurations-Problem)
14	RxPDO Toggle Failed	R	O	Dieser Fehlerzähler wird inkrementiert, wenn der Slave RxPDO Toggle unterstützt und keine Statusänderung vom Master mehr empfangen wurde.
15	Minimum cycle distance	R	O	Minimaler Abstand zwischen zwei SM Events in ns (um Jitter zu überwachen)
16	Maximum cycle distance	R	O	Maximaler Abstand zwischen zwei SM Events in ns (um Jitter zu überwachen)
17	Minimum SM Sync distance	R	O	Minimaler Abstand zwischen SM Event und Sync0 Event in ns (um Jitter zu überwachen)
18	Maximum SM Sync distance	R	O	Maximaler Abstand zwischen SM Event und Sync0 Event in ns (um Jitter zu überwachen)
32	Sync Error	R	C	Synchronisations-Fehler: sollte unterstützt werden, wenn SI = 11 oder SI = 13 vorhanden ist; kann in TxPDO gemappt werden. Wert 0 : kein Sync Fehler oder Sync nicht unterstützt Wert 1 : Sync Fehler

SI = Subindex, R = Read only, RW = Read/Write, M = erforderlich, O = optional, C = bedingt

Es gelten folgende Randbedingungen bei den DC Modi:

(1) Synchron mit Sync0 Event

Optionale Zeitverschiebungen von "Outputs valid" und "Input Latch" durch Shift Time (0x1C3m:03).

Synchronization Type:

- Outputs (0x1C32:01) = 2 (Sync0 Event)
- Inputs (0x1C33:01) = 2 (Sync0 Event)

Cycle Time:

- Sync0 Cycle Time (0x1C3m:02) **>** Min Cycle Time (0x1C3m:05)

Shift Time:

- Output Shift Time (0x1C32:03) **>** Output Delay Time (0x1C32:09)
- Input Shift Time (0x1C33:03) **>** [Min Cycle time (0x1C32:05) – Input Calc and Copy Time (0x1C33:06)]
- Input Shift Time (0x1C33:03) **<** [Sync0 Cycle Time (0x1C33:02) – Input Calc and Copy Time (0x1C33:06)]

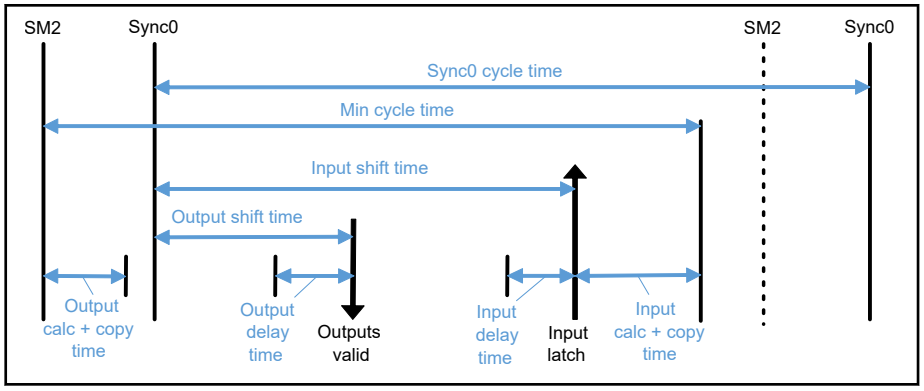


Abb. 8-15: Randbedingungen (1) Synchron mit Sync0 Event

(2) Synchron mit Sync0 Event, Input Shift mit Sync1 Event

Zeitverschiebung von "Input Latch" mit Sync1 Event (im Vergleich zur "Shift Time" wird hier der "Start Latch" Zeitpunkt beschrieben, d.h. ohne "Input Delay Time").

Synchronization Type:

- Outputs (0x1C32:01) = 2 (Sync0 Event)
- Inputs (0x1C33:01) = 3 (Sync1 Event)

Cycle Time:

- Sync0 Cycle Time (0x1C3m:02) **>** Min Cycle Time (0x1C3m:05)

Shift Time:

- Shift Sync0 **<>** Sync1 (Konfig.) **>=** [Min Cycle Time (0x1C3m:05) – Output Calc and Copy Time (0x1C32:06) – Input Calc and Copy Time (0x1C33:06) – Input Delay Time (0x1C33:09)]

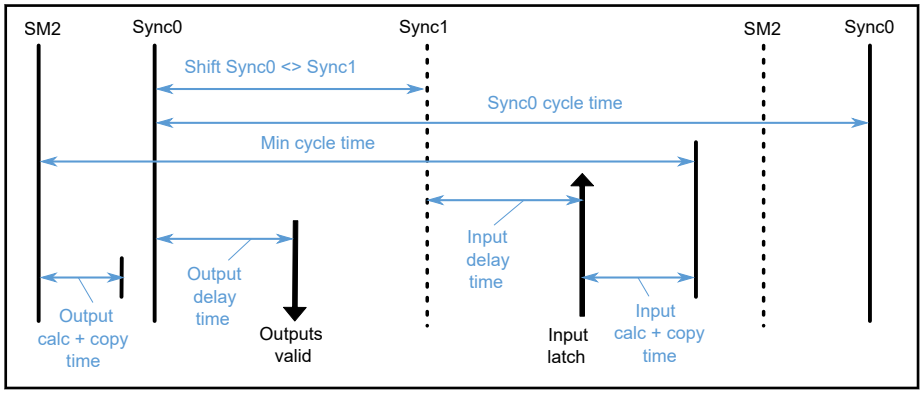


Abb. 8-16: Randbedingungen (2) Synchron mit Sync0 Event, Input Shift mit Sync1 Event

(3) Synchron mit Sync0 Event, Output Shift mit Sync1 Event

Zeitverschiebung von "Outputs valid" mit Sync1 Event (im Vergleich zur "Shift Time" wird hier der "Start Outputs" Zeitpunkt beschrieben, d.h. ohne "Output Delay Time").

Synchronization Type:

- Outputs (0x1C32:01) = 3 (Sync1 Event)
- Inputs (0x1C33:01) = 2 (Sync0 Event)

Für weitere Details zur Konfiguration "Output Shift mit Sync1 Event" bitte an die ETG wenden.

(4) Synchron mit Sync1 Event

Zeitverschiebungen von "Outputs valid" und "Input Latch" durch Sync1 Event.

Synchronization Type:

- Outputs (0x1C32:01) = 3 (Sync1 Event)
- Inputs (0x1C33:01) = 3 (Sync1 Event)

Cycle Time:

- Sync0 Cycle Time (0x1C3m:02) **>** Min Cycle Time (0x1C3m:05)

Shift Time:

- Shift Sync0 **<>** Sync1 (Konfig.) **>=** Output Calc and Copy Time (0x1C32:06)

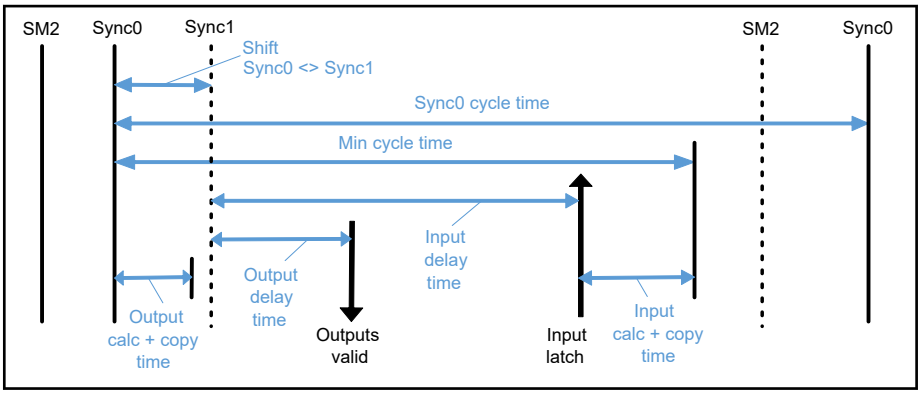


Abb. 8-17: Randbedingungen (4) Synchron mit Sync1 Event

(5) DC mit untergeordneten Zyklen

Das Sync0 Event dient als Trigger für den lokalen Applikationszyklus (z.B. schnelle Regelungen), während das Sync1 Event für einen (langsameren) Kommunikationszyklus verwendet wird. D.h. die Sync1 Zykluszeit ist ein Vielfaches der Sync0 Zykluszeit (Konfig.).

Synchronization Type:

- Outputs (0x1C32:01) = 3 (Sync1 Event)
- Inputs (0x1C33:01) = 3 (Sync1 Event)

Cycle Time:

- Cycle Time (0x1C3m:02) ist Sync1 Zykluszeit

Shift Time:

- Die Zeitverschiebungen von "Outputs valid" und "Input Latch" können nur durch "Input / Output Shift Time" (0x1C3m:03) festgelegt werden.

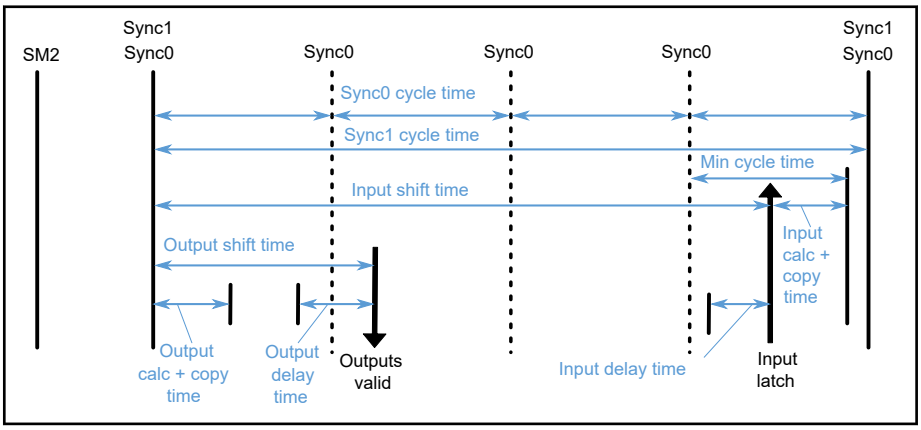


Abb. 8-18: Randbedingungen (5) DC mit ungeordneten Zyklen

8.2.12 EtherCAT Slave Controller

Aufbau

Nachfolgende Abbildung zeigt eine schematische Darstellung eines EtherCAT Slave Controllers (ESC).

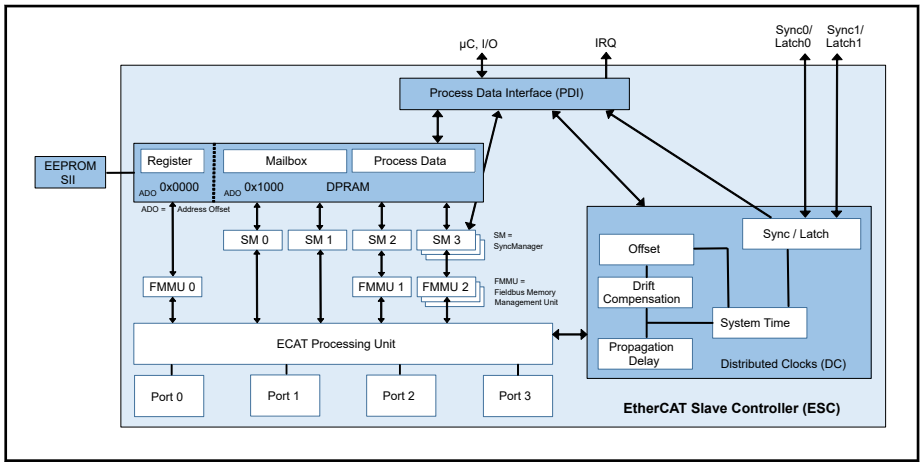


Abb. 8-19: EtherCAT Slave Controller Aufbau

Register

Die Register liegen im lokalen Adressbereich 0x0000 bis 0x0FFF des ESC und werden u.a. verwendet für

- Konfiguration des ESC, z.B. EtherCAT Adresse („Fixed Physical Address“), SyncManager, FMMU, DC.
- Zugriff auf EEPROM / SII (Slave Information Interface), dieses enthält u.a. die Geräte-Identifikation sowie Defaultwerte für die ESC Konfiguration.
- Steuer- und Status-Daten der EtherCAT Zustandsmaschine.

SyncManager (SM)

Die SyncManager stellen die Datenkonsistenz im DPRAM des ESC sicher, d.h. gleichzeitiger Zugriff auf Speicherbereiche wird verhindert; z.B. wenn die ECAT Processing Unit (Telegrammverarbeitung) und die lokale Applikation gleichzeitig auf einen Speicherbereich im DPRAM zugreifen wollen.

Die SM Konfiguration ist ab Register ADO 0x0800.

Es sind bis zu 16 unabhängige SyncManager Kanäle möglich:

- ein Slave ohne Mailbox hat typischerweise 2 SyncManager:
  - SM0 = Prozessdaten-Ausgänge (oder Eingänge, wenn es keine Ausgänge gibt),
  - SM1 = Prozessdaten-Eingänge.
- ein Slave mit Mailbox hat typischerweise 4 SyncManager:
  - SM0 = Mailbox-Out,
  - SM1 = Mailbox-In,
  - SM2 = Prozessdaten-Ausgänge (oder Eingänge, wenn es keine Ausgänge gibt),
  - SM3 = Prozessdaten-Eingänge.
- Prozessdaten einer Richtung (Eingänge oder Ausgänge) können auch in mehrere SyncManager aufgeteilt sein; z.B. wenn die Datenkonsistenz

einzelner Speicherbereiche separat überwacht werden soll. Hierbei spricht man dann auch von "SyncUnits" im Slave.

Es werden verschiedene Typen unterschieden:

- Mailbox (azyklische Kommunikation): 1 Puffer System mit Handshake Mechanismus.
- Prozessdaten: 1 oder 3 Puffer System, mit 3 Puffer können unabhängige Lese- und Schreibzugriffe stattfinden; z.B. für Free Run Kommunikation, wenn die lokale Applikation und die ECAT Telegrammverarbeitung unabhängig voneinander laufen.

Fieldbus Memory Management Unit (FMMU)

Die Prozessdaten werden auf dem EtherCAT Bus einem logischen Adressraum zugeordnet (bis zu 4 Gigabyte), der sich über beliebig viele Slaves verteilen kann. Die FMMU ist für das Mapping einer globalen logischen Adresse auf die lokale physikalische Adresse des ESC zuständig.

Die FMMU Konfiguration erfolgt ab Register ADO 0x0600.

Es sind bis zu 16 unabhängige FMMU Kanäle möglich, z.B. separate FMMUs für Datenrichtung und Speicherbereiche.

Distributed Clocks (DC)

Zur Synchronisation der EtherCAT Teilnehmer wird das Verfahren der "Verteilten Uhren" mit Festlegung einer globalen Referenzuhr angewendet.

Die Systemzeit, die am 1. Jan. 2000 um 00:00 Uhr beginnt (Einheit 1 ns / 64 Bit), wird in allen Geräten eingeregelt, d.h. der Drift und der Offset der lokalen ESC Uhr zur Referenzuhr wird entsprechend kompensiert (Genauigkeit < 100 ns).

Die DC Konfiguration erfolgt ab Register ADO 0x0900.

Die DC Einheit generiert entsprechende Events (Sync / Latch) für die Ein-/Ausgangsdaten-Verarbeitung. Das zweite Sync Signal (Sync1) kann mit einer definierten Verschiebung nach dem Sync0 Event konfiguriert werden. Die Latch Signale werden z.B. für "Time Stamping" der Systemzeit verwendet.

Kommandos

Es gibt verschiedene EtherCAT Kommandos für alle Adressierungsverfahren. Neben separaten Lese- und Schreib-Kommandos gibt es auch kombinierte Befehle, in denen Lese- u. Schreibzugriffe mit einem Datagramm ausgeführt werden können. Diese kombinierten Zugriffe sind effizienter, werden aber nicht von allen Slave Geräten unterstützt.

Wert	Code	Name	Beschreibung
0	NOP	No Operation	Slave ignoriert den Befehl.
1	APRD	Auto Increment Read	Slave Adressierung mit Auto-Inkrement Adresse. Der Slave, der Adresse 0 empfängt, verarbeitet die Daten. Jeder Slave inkrementiert die Adresse um 1.
2	APWR	Auto Increment Write	
3	APRW	Auto Increment Read Write	
4	FPRD	Fixed Physical Read	Slave Adressierung mit konfigurierter EtherCAT Adresse („Fixed Physical Address“).
5	FPWR	Fixed Physical Write	
6	FPRW	Fixed Physical Read Write	
7	BRD	Broadcast Read	Alle Slaves führen jeweils eine OR –Verknüpfung der empfangenen Datagrammdaten mit dem adressierten Speicherbereich durch und Schreiben das Ergebnis an die gleiche Stelle des Datagramms.
8	BWR	Broadcast Write	Die Daten werden in alle Slaves geschrieben.

Wert	Code	Name	Beschreibung
9	BRW	Broadcast Read Write	Alle Slaves führen jeweils eine OR –Verknüpfung der empfangenen Datagrammdaten mit dem adressierten Speicherbereich durch und Schreiben das Ergebnis an die gleiche Stelle des Datagramms und in den Speicher. (Dieser Befehl wird gewöhnlich nicht verwendet).
10	LRD	Logical Memory Read	Slave Adressierung mit logischer Speicheradresse, d.h. alle Slaves, deren FMMU mit der Adresse des logischen Speicherabbilds aus dem Datagramm konfiguriert sind, verarbeiten die Daten.
11	LWR	Logical Memory Write	
12	LRW	Logical Memory Read Write	
13	ARMW	Auto Increment Read Multiple Write	Slave Adressierung mit Auto-Inkrement Adresse. Der Slave, der Adresse 0 empfängt, fügt dem Datagramm Eingangsdaten hinzu („Read“), alle anderen Slaves entnehmen dem Datagramm an derselben Stelle Ausgangsdaten („Multiple Write“). Jeder Slave inkrementiert die Adresse um 1.
14	FRMW	Fixed Physical Read Multiple Write	Der Slave mit der konfigurierten EtherCAT Adresse („Fixed Physical Address“) fügt dem Datagramm Eingangsdaten hinzu („Read“), alle anderen Slaves entnehmen dem Datagramm an derselben Stelle Ausgangsdaten („Multiple Write“).
15-255	-	-	Reserviert.

Abhängig vom Kommandotyp wird der WorkingCounter des Datagramms inkrementiert:

Kommandotyp	Ergebnis	Inkrementierung
Lesen	Lesen nicht erfolgreich (oder Slave nicht adressiert)	nein
	Lesen erfolgreich	+ 1
Schreiben	Schreiben nicht erfolgreich (oder Slave nicht adressiert)	nein
	Schreiben erfolgreich	+ 1
Lesen + Schreiben	Lesen und Schreiben nicht erfolgreich (oder Slave nicht adressiert)	nein
	Lesen erfolgreich	+ 1
	Schreiben erfolgreich	+ 2
	Lesen und Schreiben erfolgreich	+ 3

8.3 EtherCAT Konfiguration Übersicht

Verkabelung

Der Rexroth EtherCAT Master unterstützt keine Kabelredundanz. Es kann am EtherCAT Master nur 1 Ethernet-Port für die EtherCAT Verkabelung verwendet werden:

Bei XM2-Steuerungen kann nur der Port XF1 verwendet werden (Port XF2 nicht möglich).

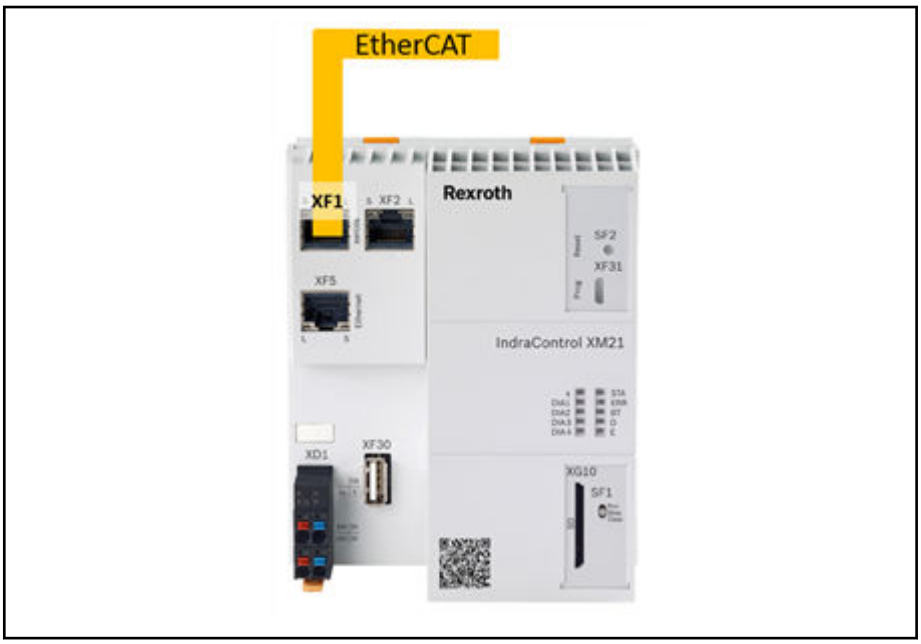


Abb. 8-20: EtherCAT-Verkabelung an einer XM2-Steuerung

Bei der XM4-Steuerung kann nur der Port XF1 verwendet werden (Port XF2 nicht möglich).

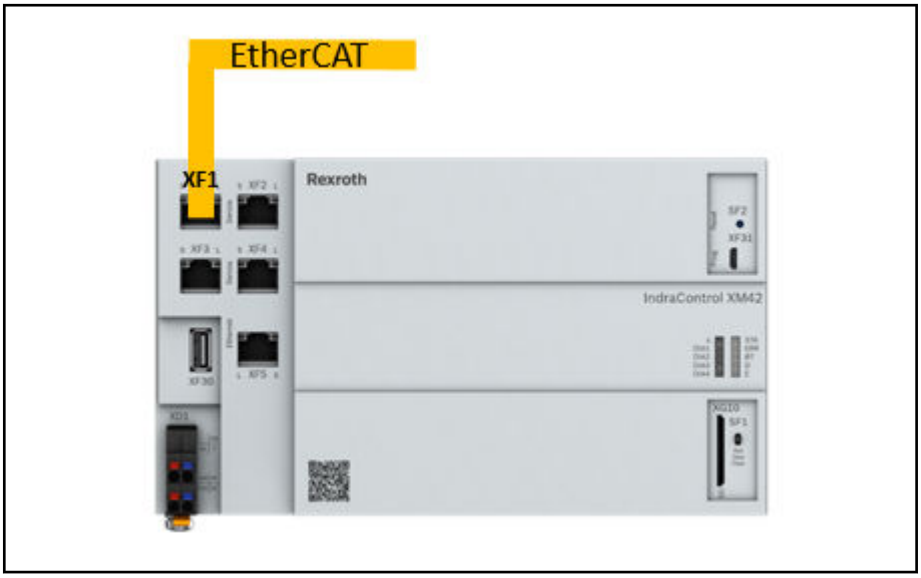


Abb. 8-21: EtherCAT-Verkabelung an einer XM42-Steuerung

Bei der VPx-Steuerung kann nur der Port XF01 verwendet werden (Port XF02 nicht möglich).

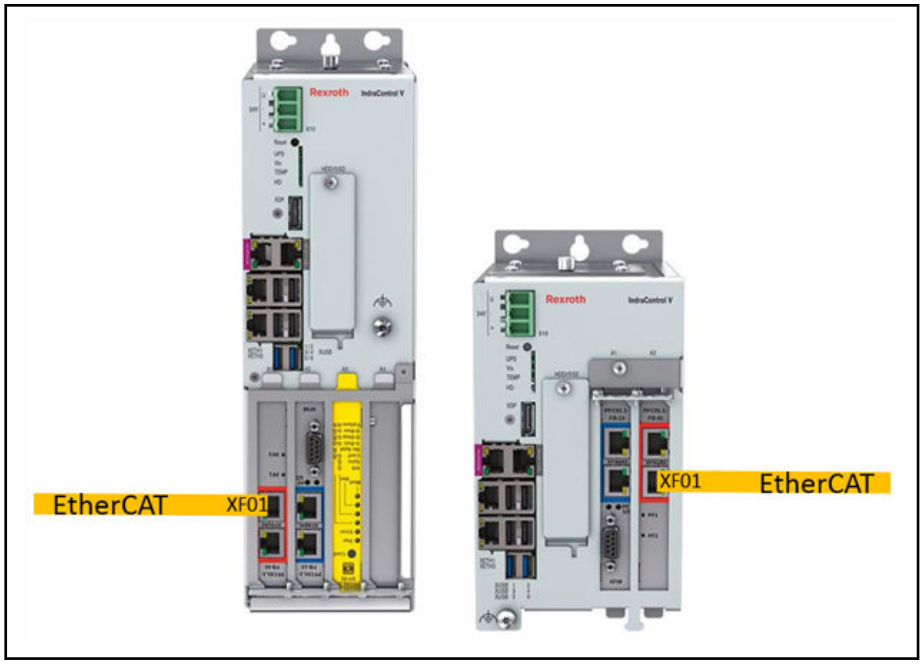


Abb. 8-22: EtherCAT-Verkabelung an einer VPx-Steuerung

Funktionspaket EtherCAT Master

Die Funktionalität "EtherCAT Master" ist als Funktionspaket lizenziert, d.h. eine entsprechende Lizenz muss erworben werden.

Das Funktionspaket "EtherCAT Master" kann wie bei sonstigen Funktionspaketen bereits beim Anlegen einer Steuerung aktiviert werden oder auch nachträglich über die "Eigenschaften" der Steuerung.

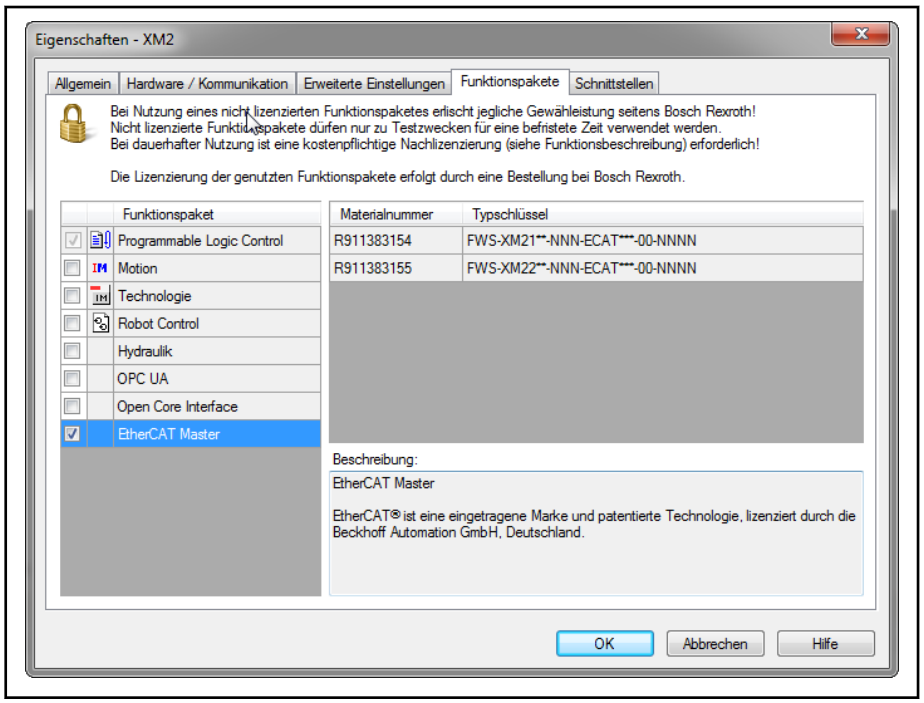


Abb. 8-23: Funktionspaket EtherCAT Master

Download der Konfiguration

Die EtherCAT Buskonfiguration wird durch SPS-Einloggen auf die Steuerung übertragen und der EtherCAT Master wird automatisch gestartet.

Der Zielzustand des EtherCAT Masters nach Download (oder nach Hochlauf der Steuerung) kann im Master Registerkarte 'Allgemein', Seite 209 gesetzt werden.

Wird das Funktionspaket Motion im Steuerungsprojekt verwendet, muss nach dem Setzen des EtherCAT Masters im Steuerungsprojekt einmalig die Motion-Konfiguration auf die Steuerung geladen werden. Hierbei wird die Sercos Schnittstelle für EtherCAT umkonfiguriert, andernfalls kommt beim SPS-Download der EtherCAT Fehler "ungültige Schnittstelle".

Übersicht Konfigurations-Dialoge

In den nachfolgenden Kapiteln wird die Konfiguration des EtherCAT Masters und der Slave Geräte im IndraWorks-Projekt beschrieben. Die Konfigurations-Dialoge öffnen sich jeweils durch einen Doppelklick auf den Master- bzw. Slave-Objekten im IndraWorks-Projektbaum.

- Kap. 8.4 "EtherCAT Master konfigurieren" auf Seite 209
  - Kap. 8.4.1 "EtherCAT Master, Gerät setzen" auf Seite 209
  - Kap. 8.4.2 "Registerkarte 'Allgemein'" auf Seite 209
  - Kap. 8.4.3 "Registerkarte 'Sync-Unit-Zuordnung'" auf Seite 210
  - Kap. 8.4.4 "Registerkarte 'Distributed Clocks'" auf Seite 211
  - Kap. 8.4.5 "Registerkarte 'Busdiagnose'" auf Seite 214
  - Kap. 8.4.6 "Registerkarte 'EoE'" auf Seite 216
  - Kap. 8.4.7 "Registerkarte 'CoE'" auf Seite 216
  - Kap. 8.4.8 "Registerkarte 'Diagnose'" auf Seite 217
  - Kap. 8.4.9 "Registerkarte 'Information'" auf Seite 217
- Kap. 8.5 "EtherCAT Master, Slaves konfigurieren" auf Seite 217
  - Kap. 8.5.1 "Slaves und Module hinzufügen" auf Seite 217
  - Kap. 8.5.2 "Registerkarte 'Allgemein'" auf Seite 219
  - Kap. 8.5.3 "Registerkarte 'SyncManager'" auf Seite 224
  - Kap. 8.5.4 "Registerkarte 'Prozessdaten Expertenmodus'" auf Seite 225
  - Kap. 8.5.5 "Registerkarte 'Prozessdaten'" auf Seite 227
  - Kap. 8.5.6 "Registerkarte 'Startparameter'" auf Seite 228
  - Kap. 8.5.7 "Registerkarte 'EoE-Einstellungen'" auf Seite 232
  - Kap. 8.5.8 "Registerkarte 'Online'" auf Seite 233
  - Kap. 8.5.9 "Registerkarte 'ESC Register'" auf Seite 233
  - Kap. 8.5.10 "Registerkarte 'CoE'" auf Seite 234
  - Kap. 8.5.11 "Registerkarte 'Diagnose'" auf Seite 235
  - Kap. 8.5.12 "Registerkarte 'E/A-Abbild'" auf Seite 236
  - Kap. 8.5.13 "Registerkarte 'Information'" auf Seite 238
  - Kap. 8.5.14 "SoE Parametereditor" auf Seite 238
- Kap. 8.6 "EtherCAT Master, Bus scannen" auf Seite 239
- Kap. 8.7 "EtherCAT Master, Bus synchrone SPS-Verarbeitung" auf Seite 239
- Kap. 8.8 "EtherCAT FAQs" auf Seite 240

8.4 EtherCAT Master konfigurieren

8.4.1 EtherCAT Master, Gerät setzen

Der EtherCAT Master kann alternativ zur Sercos Schnittstelle ausgewählt werden.

Dies kann entweder bereits beim Anlegen einer Steuerung bei der Auswahl der Schnittstellen eingestellt werden, oder bei einer bereits vorhandenen Steuerung im IndraWorks-Projekt im Kontextmenü des Sercos-Knotens mittels "Gerät setzen" eingestellt werden.

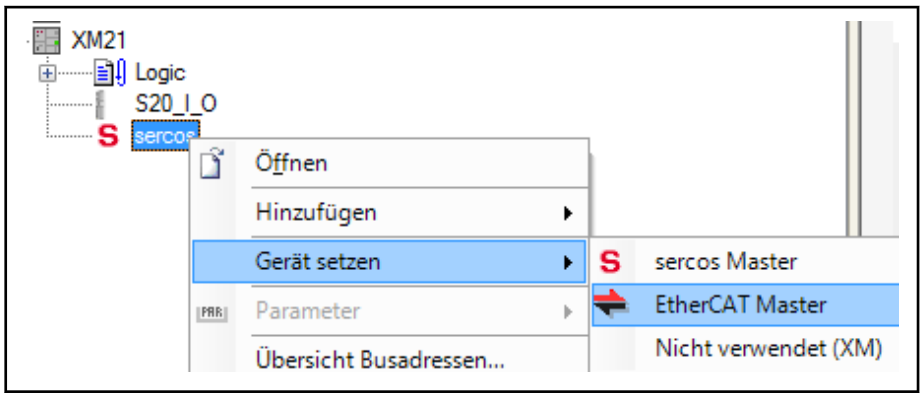


Abb. 8-24: Kontextmenü "Gerät setzen" EtherCAT Master

Bei dem Steuerungstyp "VPx" wird der Befehl "Gerät setzen" nicht unterstützt!

Der EtherCAT Master ist bei der Auswahl der Steuerungs-Schnittstellen auszuwählen, z. B. über das Kontextmenü des Steuerungs-Knotens **Eigenschaften** ▸ **Schnittstellen**.

8.4.2 Registerkarte 'Allgemein'

In dieser Registerkarte können allgemeine Einstellungen des EtherCAT Masters gesetzt werden.

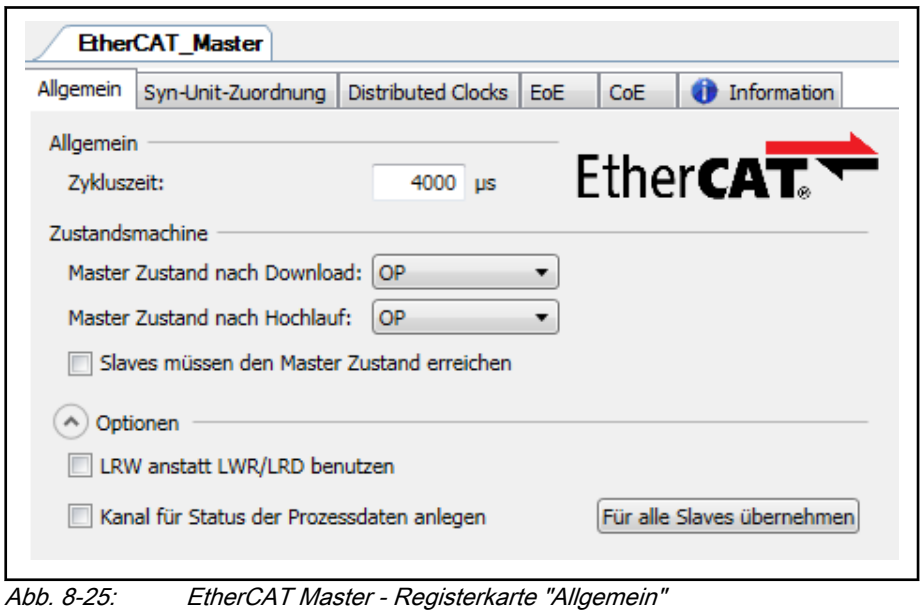


Abb. 8-25: EtherCAT Master - Registerkarte "Allgemein"

Allgemein

Zykluszeit	Buszykluszeit der EtherCAT Kommunikation.

Zustandsmaschine

Master Zustand nach Download	Zielzustand des Masters nach Download der Konfiguration.
Master Zustand nach Hochlauf	Zielzustand des Masters nach Hochlauf der Steuerung.
Slaves müssen den Master Zustand erreichen	Ist diese Option aktiviert, müssen **alle** Slaves den Master Zustand erreichen. Ist diese Option deaktiviert, kann der Master einen Bushochlauf (z.B. Zielzustand OP) ausführen, auch wenn nicht alle Slaves den Zielzustand erreichen, oder Slaves am EtherCAT-Bus fehlen.

Optionen

LRW anstatt LWR/LRD verwenden	Bei aktivierter Option werden anstelle von separaten Lese- (LRD) und Schreib-Befehlen (LWR) kombinierte Lese-/Schreib-Befehle (LRW) verwendet. **Achtung!** • **Nicht alle Slaves unterstützten LRW** • **Bei Kabelredundanz kann LRW nicht verwendet werden**
Kanal für Status der Prozessdaten anlegen	Bei aktivierter Option wird beim Hinzufügen eines Slaves ein Eingangskanal "PdStatus" in dessen E/A-Abbild angelegt. Siehe auch Kap. 8.5.12 "Registerkarte 'E/A-Abbild'" auf Seite 236. Mit dem Befehl "Für alle Slaves übernehmen" wird die Einstellung für alle vorhandenen Slaves übernommen.

8.4.3 Registerkarte 'Sync-Unit-Zuordnung'

In dieser Registerkarte können SyncUnits manuell erstellt und Slaves zugeordnet werden.

Siehe auch Grundlagen Kap. 8.2.8 "SyncUnits" auf Seite 185.

Die voreingestellte SyncUnit "Standard" ist immer vorhanden und kann nicht gelöscht werden. Alle Slaves sind zunächst dieser SyncUnit "Standard" zugeordnet. Es können dann manuell weitere SyncUnits erstellt und jeweils Slaves zugeordnet werden.

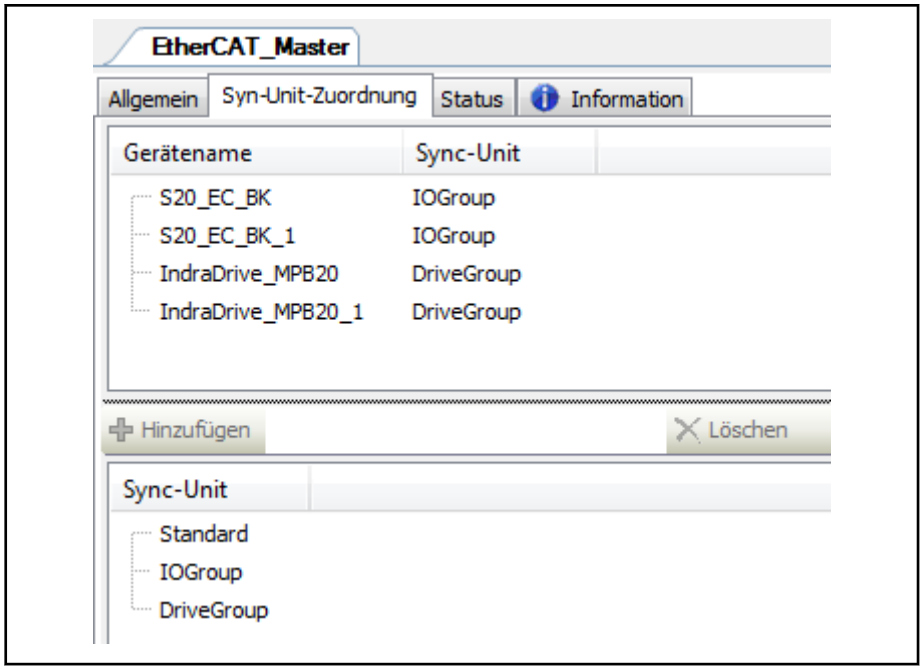


Abb. 8-26: EtherCAT Master - Registerkarte "Sync-Unit-Zuordnung"

Gerätename	Name des Slaves aus dem Projektbaum
Sync-Unit	Zuordnung des Slaves zu einer SyncUnit. Durch Doppelklick auf das SyncUnit Feld eines Slaves (rechts neben dem Gerätenamen) kann eine der konfigurierten SyncUnits ausgewählt werden.
Hinzufügen	Wenn ein Name im Textfeld rechts von "Hinzufügen" eingegeben wird, wird mit „Hinzufügen“ eine SyncUnit mit diesem Namen erstellt.
Löschen	Löscht die im unteren Fenster ausgewählte SyncUnit. Die voreingestellte SyncUnit "Standard" kann nicht gelöscht werden. Wird eine SyncUnit gelöscht, zu dieser noch Slaves zugeordnet sind, werden diese Slaves wieder der SyncUnit "Standard" zugewiesen.

8.4.4 Registerkarte 'Distributed Clocks'

In dieser Registerkarte können die "Distributed Clocks" (Verteilte Uhren) Einstellungen des EtherCAT Masters gesetzt werden.

Zudem werden im eingeloggten Zustand Diagnosen zu den Distributed Clocks angezeigt.

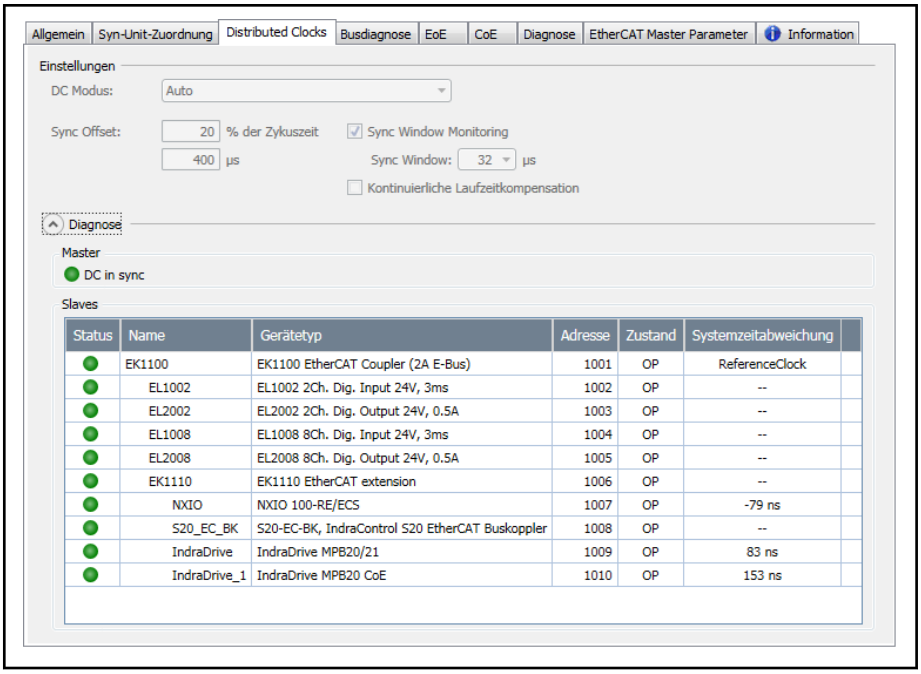


Abb. 8-27: "Busdiagnose"

Einstellungen

DC Modus	• **Automatic**: ist bei mindestens einem Slave DC aktiviert, so wird automatisch "Bus Shift" eingestellt. • **Free Run**: zyklische Kommunikation, ohne Uhrensynchronisation (d.h. es findet keine Regelung der "Verteilten Uhren" statt). • **Bus Shift**: Synchronbetrieb der Slaves, bei denen DC aktiviert ist. Bus Shift: Der EtherCAT Master stellt die Systemzeit und schreibt diese in den ersten DC-Slave am Bus (Schieben der Buszeit).
Sync Offset	Der "Sync Offset" ist die zeitliche Verschiebung vom Sync0 Impuls in den DC-Slaves zum Sendezeitpunkt des zyklischen Datentelegramms im Master. Die Einstellung erfolgt prozentual zur Zykluszeit. Eine typische Einstellung ist z.B. 20 %.

Sync Window Monitoring	Bei aktivierter Option, wird die Synchronisation der Verteilten Uhren in allen Slaves überwacht. Die "Systemzeit-Differenz" (Reg. 0x092C) wird zyklisch von jedem Slave ausgelesen. Mit der Eingabe "Sync Window" wird die maximal zulässige Abweichung zur Systemzeit (Referenzuhr) vorgegeben . Wenn bei allen Slaves die Systemzeit-Differenz kleiner ist als das Überwachungsfenster, gilt die Synchronisation als eingeregelt ("DC in Sync").
Kontinuierliche Laufzeitmessung	"Continous Propagation Compensation": Bei aktivierter Option, wird die Laufzeit des EtherCAT Telegrams zyklisch am Bus gemessen. Auf diese Weise kann die Laufzeitkompensation kontinuierlich angepasst werden. Bei EtherCAT Buskonfigurationen, bei denen sich die Topologie im Betriebsmodus nicht ändert, sollte diese Option deaktiviert werden.

Diagnose

Die DC-Diagnose wird nur im eingeloggten Zustand angezeigt.

Master: es gilt "DC in Sync", wenn bei allen Slaves die Systemzeit-Differenz kleiner ist als das Überwachungsfenster (siehe auch "Sync Window Monitoring" bei den DC-Einstellungen).

Slaves: neben den vordefinierten Spalten können mittels Kontextmenü (rechter Mausklick) in der Tabellenüberschrift weitere Werte eingeblendet werden.

Status	"grün": OK, "rot": es liegt ein Fehler vor, siehe Registerkarte "Busdiagnose".
Name	Konfigurierter Gerätename des Slaves
Gerätetyp	Gerätetyp-Beschreibung des Slaves (aus ESI)
Adresse	EtherCAT Adresse
Zustand	EtherCAT Zustand des Slaves
Systemzeit-Differenz	Systemzeit-Differenz in ns (ESC Register 0x092C) (dieser Wert ist nur gültig, wenn DC konfiguriert ist) Der erste Slave am Bus, bei dem DC aktiviert ist, stellt die Referenzuhr.
SlaveDelay	Zeitverzögerung "hinter" dem Slave in ns (dieser Wert ist nur gültig, wenn DC konfiguriert ist)
PropagationDelay	Übertragungsverzögerung in ns (ESC Register 0x0928) (dieser Wert ist nur gültig, wenn DC konfiguriert ist)

DC	Zeigt an, ob der EtherCAT Slave Controller DC unterstützt (32 / 64 Bit). Hinweis: die Anzeige, dass DC unterstützt wird, bedeutet nicht, dass für die Slave Applikation auch DC aktiviert werden kann in der Konfiguration (dies wird in der ESI vorgegeben).
Hersteller /-kennung **Produktcode, Version,** **Seriennummer**	Identitätsinformation (EEPROM Offset 0x0008)

8.4.5 Registerkarte 'Busdiagnose'

Diese Registerkarte wird nur im eingeloggten Zustand angezeigt.

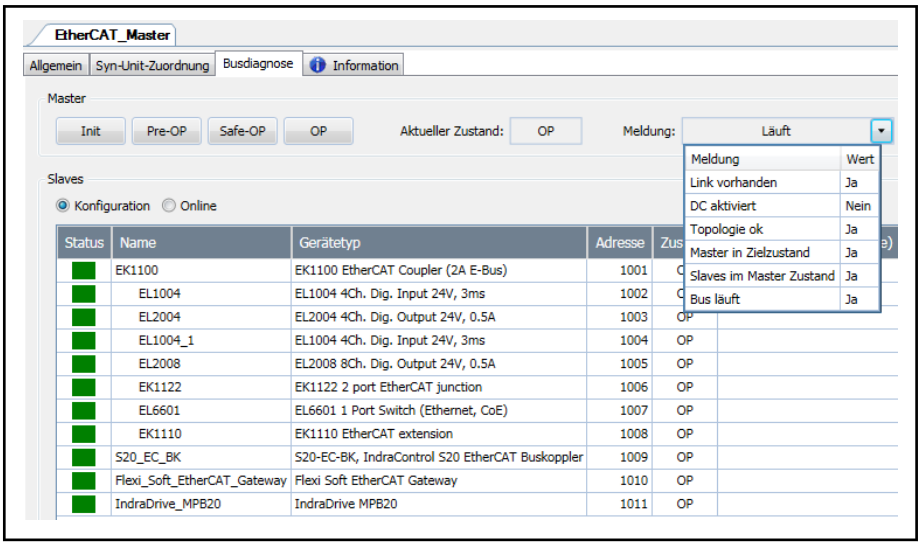


Abb. 8-28: EtherCAT Master - Registerkarte "Busdiagnose"

Mit den Schaltflächen "Init" / "Pre-OP" / "Safe-OP" / "OP" kann der Bus-Zustand beim EtherCAT Master geschaltet werden. Zudem wird der aktuelle Bus-Zustand angezeigt.

Im Feld "Meldung" können folgende Diagnosen angezeigt werden:

Text	Beschreibung
Link nicht vorhanden	Kein Link am EtherCAT Master.
DC nicht in Sync	Die Synchronisation der Verteilten Uhren (DC) ist nicht innerhalb des konfigurierten Überwachungsfensters (nur wenn DC aktiviert ist).
Topologie nicht OK	Die konfigurierten Slaves stimmen nicht mit den am Bus vorhandenen Slaves überein, z.B. es fehlen Slaves oder die Konfiguration der Topologie stimmt nicht mit der realen Topologie am Bus überein. Siehe auch in nachfolgender Tabelle EtherCAT Zustand "nicht verbunden". Diese Diagnose ist zudem auch gesetzt, wenn mehr Slaves am Bus vorhanden sind als konfiguriert sind.
Master nicht im Zielzustand	Der angeforderte EtherCAT Master-Zustand konnte nicht erreicht werden.

Slaves nicht im Master-Zustand	Ein oder mehrere Slaves sind nicht im Master-Zustand.
Slaves im Fehlerzustand	Ein, oder mehrere Slaves melden einen Fehler (z.B. AL Status Code).
Master nicht in OP	Der EtherCAT Master ist nicht im Zustand OP.
Bus läuft	Der EtherCAT Master und alle Slaves sind in gültiger zyklischer Kommunikation (Zustand OP).

Unter "Slaves" kann zwischen zwei alternativen Ansichten umgeschaltet werden.

- Bei "Konfiguration" werden alle konfigurierten Slave-Geräte angezeigt.
- Bei "Online" werden alle aktuell am Bus vorhanden Slave-Geräte angezeigt. Diese Ansicht ist insbesondere dann nützlich, wenn ein "Topologie-Fehler" vorliegt, d.h. die konfigurierte Topologie im IndraWorks-Projektbaum nicht mit der Verkabelung der Slaves am Bus übereinstimmt.

Die Spalten der Tabelle haben folgende Bedeutung:

Status	"grün": OK, "rot": es liegt ein Fehler vor, der Slave ist nicht verbunden oder siehe Spalte "Diagnose"
Name	Konfigurierter Gerätename des Slaves. Ein nicht-konfigurierter Slave wird in der "Online" Ansicht mit dem Namen "Slave_x" angezeigt.
Gerätetyp	Gerätetyp-Beschreibung des Slaves (aus ESI). Ein Slave, dessen ESI in der IndraWorks Gerätedatenbank nicht installiert ist, wird in der "Online" Ansicht als "Unbekanntes Gerät" angezeigt.
Adresse	EtherCAT Adresse
Zustand	EtherCAT Zustand des Slaves Hinweis: der Zustand "nicht verbunden" bedeutet, dass der konfigurierte Slave keinem Teilnehmer am Bus zugeordnet werden konnte, d.h. entweder ist das Gerät physikalisch nicht am Bus verbunden oder die Topologie-Position des Gerätes am Bus stimmt nicht mit der Konfiguration überein. Tipp: bei einem Topologie-Fehler • vergleiche die Konfiguration der Topologie im IW-Projektbaum mit der "Online"-Ansicht der Slaves • siehe auch "Konfiguration der Topologie" auf Seite 218
Diagnose	Diagnose, z.B. EtherCAT Zustandsfehler (siehe Kap. "AL Status Codes" auf Seite 187)

Slave Zustand schalten

Mittels rechter Mausklick (Kontextmenü) auf einen Slave kann der Zustand von diesem Slave individuell geschaltet werden.

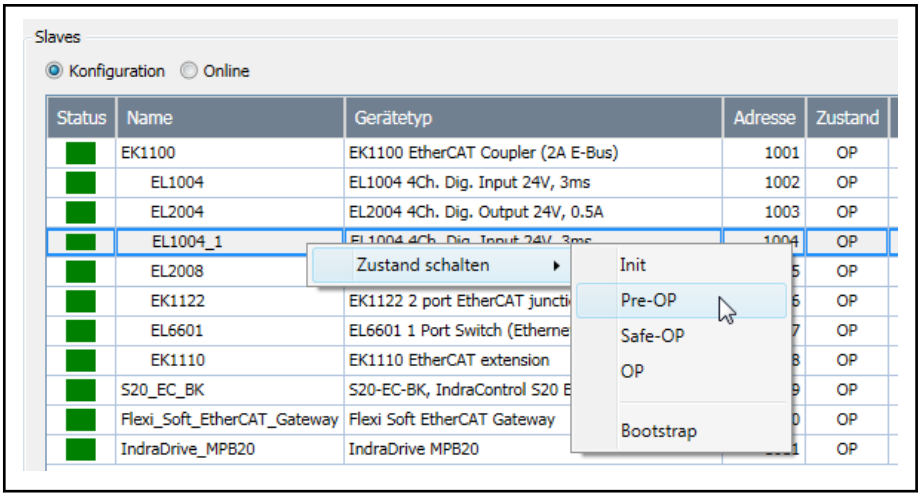


Abb. 8-29: Registerkarte "Busdiagnose" - Slave Zustand schalten

8.4.6 Registerkarte 'EoE'

In dieser Registerkarte kann ein virtueller Ethernet-Port im EtherCAT Master aktiviert werden.

Durch das Setzen der IP-Adresse für den EtherCAT Master wird gleichzeitig auch das IP-Routing zum Engineering-Port der Steuerung aktiviert.

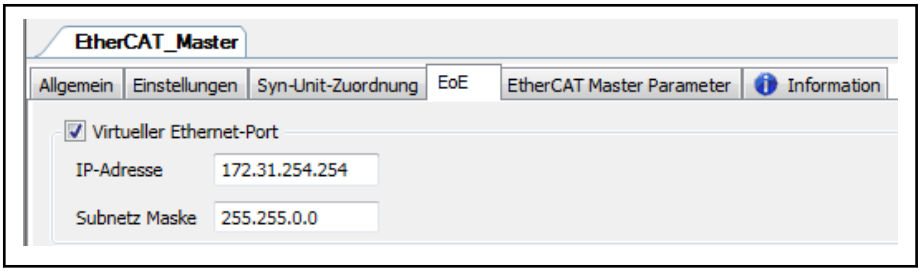


Abb. 8-30: EtherCAT Master - Registerkarte "EoE"

Es müssen gültige IP-Einstellungen entsprechend der Netzwerk-Konfiguration festgelegt werden.

Siehe auch

- Grundlagen: "EoE: Ethernet over EtherCAT" auf Seite 183
- Slave Konfiguration: Kap. 8.5.7 "Registerkarte 'EoE-Einstellungen'" auf Seite 232

8.4.7 Registerkarte 'CoE'

Diese Registerkarte wird nur im eingeloggten Zustand angezeigt.

Es werden die CoE Objekte des EtherCAT Masters für Diagnosezwecke angezeigt.

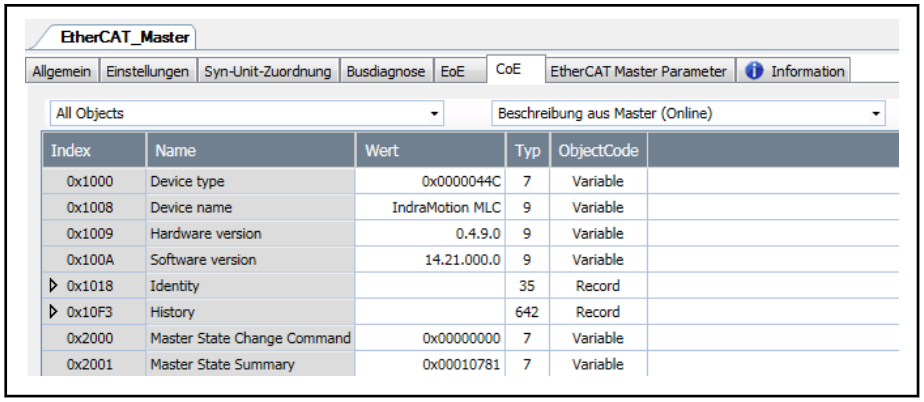


Abb. 8-31: EtherCAT Master - Registerkarte "CoE"

8.4.8 Registerkarte 'Diagnose'

Diese Registerkarte wird nur im eingeloggten Zustand angezeigt.

Es wird das CoE Objekt "Diagnosis History" (0x10F3) des EtherCAT Masters dargestellt.

Für eine ausführliche Beschreibung des Dialogs siehe Kap. 8.5.11 "Registerkarte 'Diagnose'" auf Seite 235.

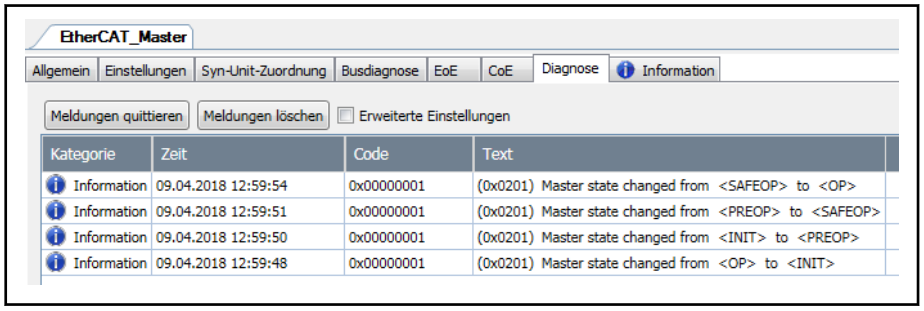


Abb. 8-32: EtherCAT Master - Registerkarte "Diagnose"

8.4.9 Registerkarte 'Information'

Diese Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen: u.a. Name, Hersteller, Kategorie, Version und Beschreibung.

8.5 EtherCAT Master, Slaves konfigurieren

8.5.1 Slaves und Module hinzufügen

Slaves hinzufügen

Über das Kontextmenü "Hinzufügen" auf dem EtherCAT Master Objekt können Slave Geräte dem Master hinzugefügt werden.

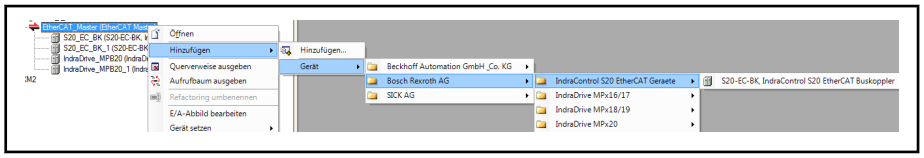


Abb. 8-33: EtherCAT Master Kontextmenü "Hinzufügen Slave"

Alternativ können Geräte auch mittels Drag&Drop aus der IndraWorks Geräte-Bibliothek dem EtherCAT Master hinzugefügt werden.

Module hinzufügen

Module bei einem Slave mit Modular-Device-Profile (z.B. Rexroth S20-EC-BK) werden in gleicher Weise über das Kontextmenü "Hinzufügen" auf dem Slave Objekt dem Buskoppler hinzugefügt.

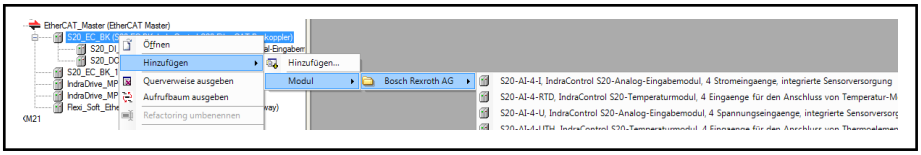


Abb. 8-34: EtherCAT Master Kontextmenü "Hinzufügen Modul"

Slaves am E-BUS hinzufügen

Module (Klemmen) an einem Buskoppler mit E-BUS, zum Beispiel Beckhoff EK1100, werden ebenso mit dem Kontextmenü "Hinzufügen" dem Buskoppler angehängt. Diese E-BUS Module werden im Projektbaum als Unterknoten zum Buskoppler dargestellt, stellen aber hinsichtlich EtherCAT Buskonfiguration ein eigenes Slave Gerät dar und besitzen daher auch alle Dialoge eines Slaves.

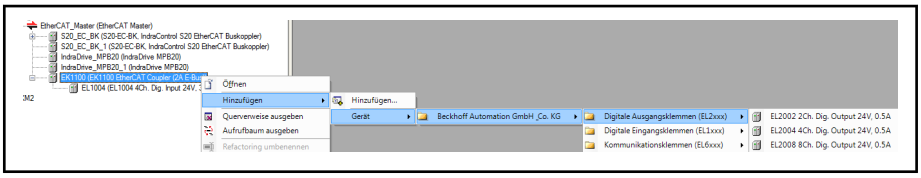


Abb. 8-35: EtherCAT_Buskoppler Kontextmenü "Hinzufügen Modul"

Tipp: Eingabedialog "Hinzufügen"

Eine Hilfestellung beim Hinzufügen von Slaves und Modulen bietet der Eingabedialog "Hinzufügen".

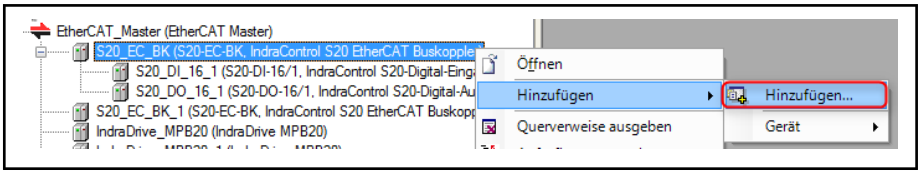


Abb. 8-36: Eingabedialog "Hinzufügen" Aufruf

Dieser Dialog zeigt abhängig von der markierten Stelle im Projektbaum alle Geräte an, die an dieser Stelle hinzufügt werden können.

Zudem kann mit einem Filter sehr einfach nach bestimmten Slaves oder Modulen gesucht werden.

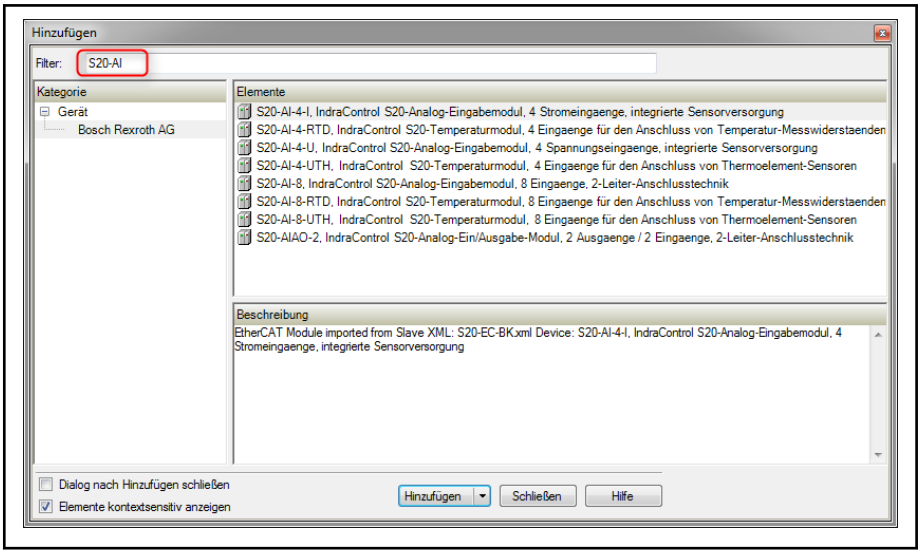


Abb. 8-37: Eingabedialog "Hinzufügen" Ansicht

Konfiguration der Topologie

Beim Hinzufügen der Slaves muss beachtet werden, dass die Topologie des EtherCAT Netzwerkes eindeutig in der Konfiguration abgebildet werden muss. Die physikalische Reihenfolge der Slaves am Bus muss mit der Reihenfolge der Slaves im IW-Projektbaum (Auto-Increment Adresse) überein-

stimmen. Die Auto-Increment Adresse wird vom Konfigurator automatisch "von oben nach unten" vergeben.

Hierbei muss auch die "Vorgänger-Port" Konfiguration übereinstimmen. D. h. insbesondere bei Abzweigklemmen (zum Beispiel Beckhoff EK1122) oder Verlängerungsklemmen (zum Beispiel Beckhoff EK1110) ist darauf zu achten, dass daran angeschlossene Slaves auch im Projektbaum an den entsprechenden unterlagerten Knoten hinzugefügt werden müssen.

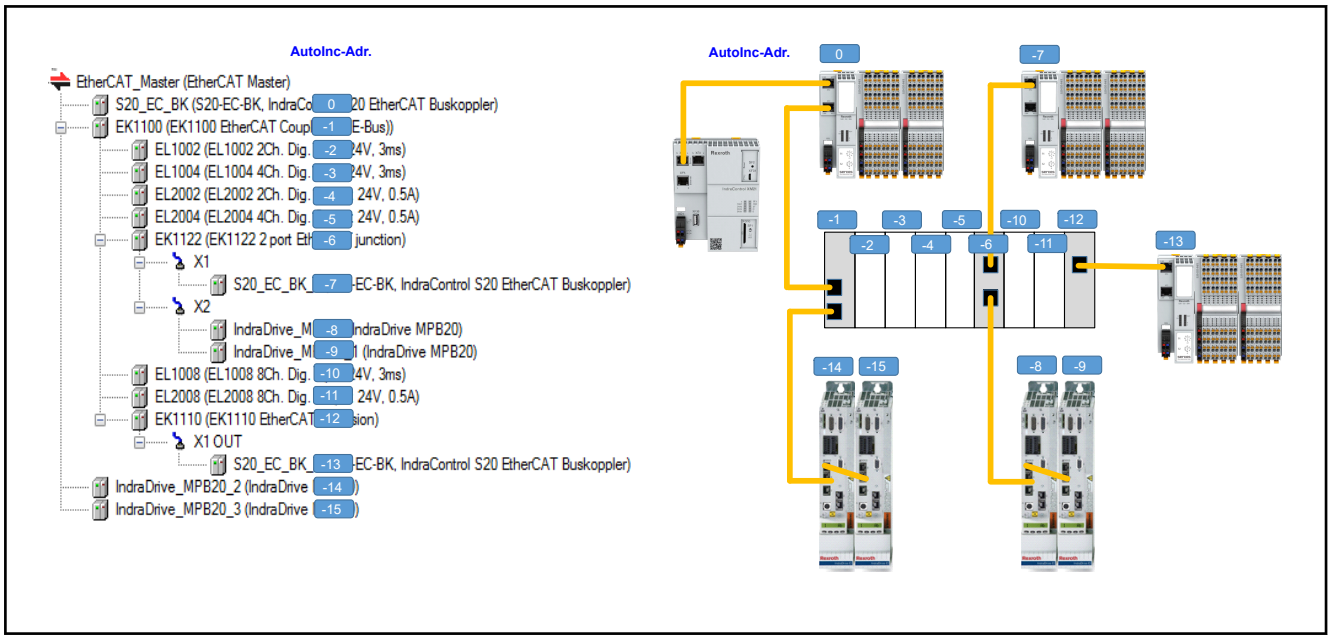


Abb. 8-38: Konfiguration der EtherCAT Topologie im IW-Projektbaum

Die Reihenfolge der Slaves im IW-Projektbaum kann auch mittels Drag&Drop geändert werden.

8.5.2 Registerkarte 'Allgemein'

In dieser Registerkarte werden die Grundeinstellungen für den EtherCAT-Slave konfiguriert.

Die voreingestellten Grundeinstellungen stammen aus der Gerätebeschreibungsdatei (ESI) des Slaves.

Dialog "Adresse und Zusätzlich"

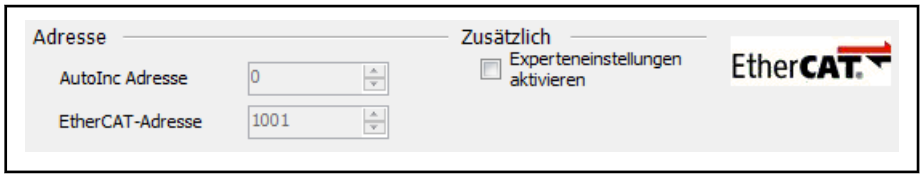


Abb. 8-39: Dialog "Adresse und Zusätzlich"

AutoInc-Adresse	Auto-Increment-Adresse: topologische Adresse des Slaves als negatives Auto-Increment (16 Bit), zum Beispiel: 0, -1 (0xFFFF), -2 (0xFFFE), -3 (0xFFFD) usw. Diese AutoInc-Adresse wird automatisch vom Konfigurator vergeben, in der Reihenfolge wie die Slaves im Projektbaum angelegt werden (von oben nach unten).
EtherCAT-Adresse	EtherCAT-Adresse des Slaves (auch als „Fixed Physical Address“ bezeichnet). Diese Adresse wird im Slave nicht gespeichert, sondern der Master weist diese Adresse dem Slave beim Bus-Hochlauf zu. Siehe auch Grundlagen Kap. 8.2.3 "Adressierungs- und Identifikationsverfahren" auf Seite 179 **Hinweis**: die EtherCAT-Adresse kann aktuell nicht verändert werden, sondern wird vom Konfigurator automatisch vergeben.
Experten-Einstellungen	☑ Es können zusätzliche Konfigurations-Einstellungen vorgenommen werden, u.a. Startup-Überprüfungen, Timeouts, Watchdog u. Identifikation (siehe nachfolgende Beschreibung). Zudem wird die Registerkarte „Expertenmodus Prozessdaten“ freigeschaltet.

Dialog "Diagnose"

Das Diagnose-Feld "Aktueller Status" wird im eingeloggten Zustand angezeigt und derzeit nicht unterstützt.

Dialog "Verteilte Uhren"

Siehe auch Grundlagen Kap. 8.2.11 "Synchronisation" auf Seite 193

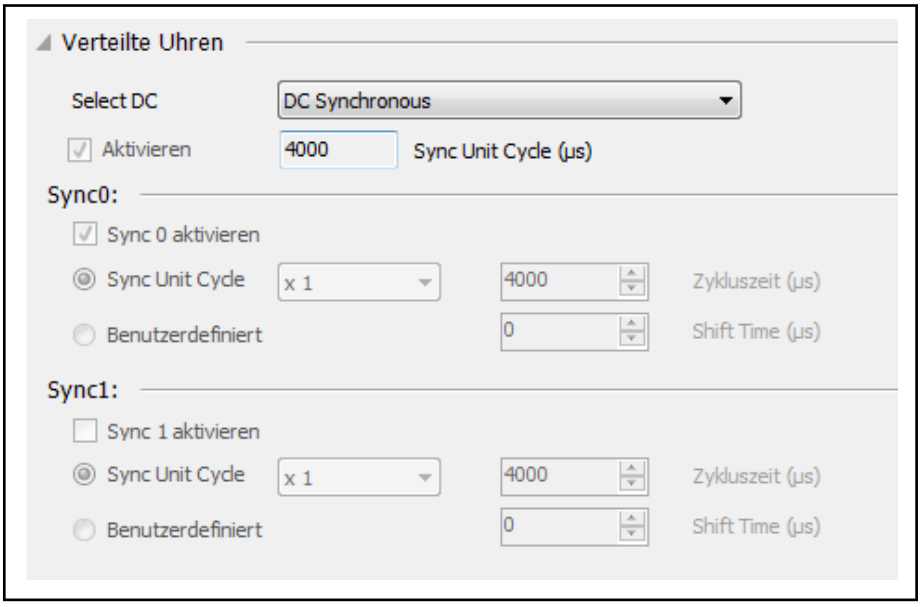


Abb. 8-40: Dialog "Verteilte Uhren"

Verteilte Uhren auswählen	Dropdown-Liste mit vordefinierten Einstellungen für Verteilte Uhren aus der Gerätebeschreibungs-Datei (ESI) des Slaves
Sync Unit Cycle (µs)	Anzeigefeld der EtherCAT Kommunikations-Zykluszeit (diese ist die Basis für die DC-Zykluszeit)

Aktivieren	☑ wenn die Option "Experten-Einstellungen" aktiviert ist, kann eine Benutzer-definierte DC-Einstellung konfiguriert werden.
Sync 0 aktivieren / Sync 1 aktivieren	☑ Sync0 Event im EtherCAT Slave verwenden / ☑ Sync1 Event im EtherCAT Slave verwenden
Sync Unit Cycle (Sync0 / Sync1)	Mit der Dropdown-Liste kann ein Faktor eingestellt werden, mit dem die EtherCAT Kommunikations-Zykluszeit multipliziert wird. Dies entspricht dann der Synchronisations- bzw. DC-Zykluszeit des Sync Events im Slave (=> Anzeigefeld "Zykluszeit (µs)")
Benutzerdefiniert	Die (Sync-) "Zykluszeit (µs)" kann manuell eingestellt werden.
Shift Time	Verschiebung des Sync Events im Slave

Dialog "Startup-Überprüfung und Timeouts"

Diese Dialoge werden nur angezeigt, wenn "Experten-Einstellungen" (siehe oben) aktiviert ist.

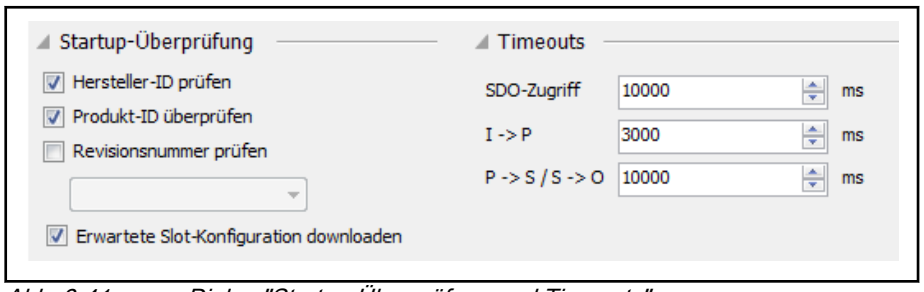


Abb. 8-41: Dialog "Startup-Überprüfung und Timeouts"

Startup-Überprüfung **Die Startup-Überprüfungen dienen dazu, ob der konfigurierte Slave mit dem physikalischen Gerät übereinstimmt. Auf diese Weise wird der Hochlauf einer fehlerhaften Konfiguration verhindert.**	
Hersteller-ID prüfen	☑ beim Bus-Hochlauf wird geprüft, dass die Hersteller-ID des konfigurierten Slaves mit dem physikalischen Gerät übereinstimmt.
Produkt-ID prüfen	☑ beim Bus-Hochlauf wird geprüft, dass die Produkt-ID des konfigurierten Slaves mit dem physikalischen Gerät übereinstimmt.

Revisions-Nummer prüfen	☑ beim Bus-Hochlauf wird die Revisions-Nummer des konfigurierten Slaves mit dem Wert des physikalischen Geräts entsprechend der Auswahl der Dropdown-Liste geprüft. Es sind teilweise unterschiedliche Überprüfungen für Low-Word (LW / Bit 0-15) und High-Word (HW / Bit 16-31) der Revisions-Nummer auswählbar. Hinweis: der Rexroth EtherCAT Master unterstützt nur die "==" Überprüfung, eine ">=" Überprüfung ist bisher nicht in der ETG.2100 (ENI) standardisiert.
Erwartete Slot-Konfiguration downloaden	☑ bei einem Slave mit "Modular Device Profile" (zum Beispiel Rexroth S20-EC-BK) wird die erwartete bzw. konfigurierte Slot-/Modul-Konfiguration auf das Gerät geladen, damit diese im Gerät mit der tatsächlichen Modul-Konfiguration verglichen werden kann.
Timeouts **Timeout-Überwachung für azyklische Dienste u. Zugriffe. Die Standard-Einstellungen stammen in der Regel aus der Gerätebeschreibungsdatei (ESI) des Slaves.**	
SDO-Zugriff	Timeout für SDO-Zugriffe beim Bus-Hochlauf
I -> P	Timeout für Zustands-Übergang Init -> PreOperational
P -> S / S -> O	Timeout für Zustands-Übergang PreOperational -> SafeOperational und SafeOperational -> Operational

Dialog "Watchdog"

Der Watchdog überwacht, dass bei Fehlern in der Prozessdaten-Kommunikation die Ausgänge nach einer voreingestellten Zeit in den sicheren Zustand geschaltet werden. Der EtherCAT Slave Controller (ESC) besitzt Watchdogs für den Sync-Manager (SM) und für das Prozessdaten-Interface (PDI).

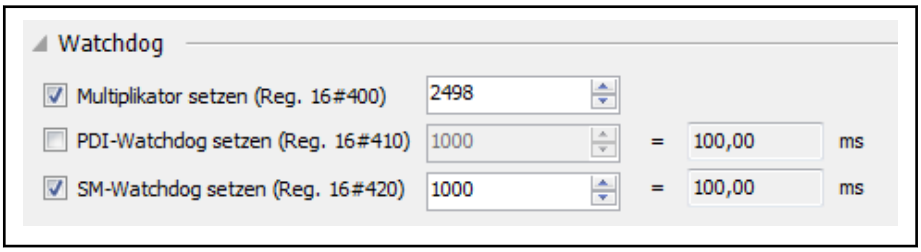


Abb. 8-42: Dialog "Watchdog"

Multiplikator setzen	Der Multiplikator (Register 16#400) teilt den lokalen Takt um den Trigger für die Watchdogs festzulegen: (1 / 25 MHz) * (Multiplikator + 2) Mit der Multiplikator Standard-Einstellung 2498 ergibt sich das Intervall 100 µs (Berechnung: (1 / 25 MHz) * (Multiplikator + 2) = 40 ns * 2500 = 100 µs) Beispiel PDI-Watchdog: mit dem Faktor 1000 entsteht somit ein Watchdog-Timeout von 100 ms. Eine manuelle Anpassung des Multiplikators ist zum Beispiel bei sehr langsamen EtherCAT Kommunikations-Zykluszeiten erforderlich.
PDI Watchdog setzen	Der PDI Watchdog (Register 16#410) überwacht die Prozessdaten-Kommunikation vom ESC zur Applikation (z.B. lokaler Mikrocontroller) des Slaves.
SM Watchdog setzen	Der SM Watchdog (Register (16#420) überwacht die zyklische EtherCAT Kommunikation. Der Watchdog wird bei jeder erfolgreichen EtherCAT Prozessdaten-Kommunikation zurückgesetzt. Wenn der Watchdog aktiviert ist und länger als die eingestellte Überwachungszeit keine EtherCAT Prozessdaten-Kommunikation stattfindet, werden die Ausgänge in den sicheren Zustand gesetzt.

Dialog "Identifikation"

Durch ein Identifikationsverfahren kann ein Slave unabhängig von seiner topologischen Position eindeutig identifiziert werden (zum Beispiel um Verkabelungsfehler auszuschließen).

Typischerweise werden Dreh- oder DIP-Schalter verwendet um einen Identifikationswert an den Slave Geräten einzustellen.

Siehe auch Grundlagen Kap. 8.2.3 "Adressierungs- und Identifikationsverfahren" auf Seite 179.

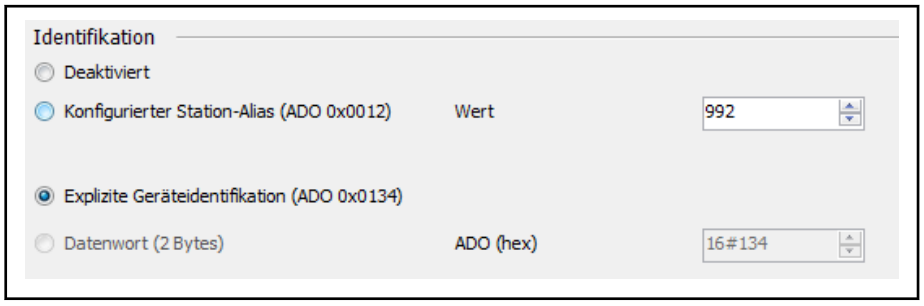


Abb. 8-43: Dialog "Identifikation"

Deaktiviert	Es findet keine Identifikation statt.
Konfigurierter Station-Alias	Identifikation über "(Configured) Station-Alias": Üblicherweise wird über Remote-Vergabe der Wert des Station-Alias in das Slave EEPROM geschrieben. Dieser Wert wird dann vom Slave beim Booten in sein Register ADO 0x0012 geladen (ADO = Address Offset). Beispiel: Dieses Verfahren wird beispielsweise beim Rexroth IndraDrive angewendet, bei dem der Station-Alias alternativ auch über das Display eingestellt werden kann.
Explizite Geräteidentifikation	Identifikation über "Explicit Device Identification": Der Slave teilt dem Master seinen Identifikationswert auf Anfrage im AL Status Code Register (ADO 0x0134) mit. Beispiel: Beim Rexroth S20-EC-BK wird mit dem Drehschalter am Gerät der Identifikationswert für "Explicit Device Identification" eingestellt.
Datenwort	Identifikation über "Data Word" (oder auch als "Input Word" bezeichnet): Der Identifikationswert wird an einem bestimmen Register Adress Offset (ADO) erwartet, der aus der Gerätebeschreibungsdatei (ESI) des Slaves vorgegeben wird oder ggf. auch beliebig eingestellt werden kann (siehe Produktbeschreibung des Slave Gerätes). Beispiel: Beim Beckhoff EK1101 wird mit dem Drehschalter am Gerät der Identifikationswert für "Data Word" eingestellt.
Wert	Dies ist der "Sollwert", der für das ausgewählte Verfahren vorgegeben wird. Dieser Wert muss mit dem eingestellten Wert im Slave Gerät übereinstimmen.
"EEPROM schreiben" und tatsächliche Adresse	Im eingeloggten Zustand wird die Schaltfläche "EEPROM schreiben" und das Feld "Tatsächliche Adresse" angezeigt. **Hinweis**: diese Online-Funktionen werden vom Rexroth EtherCAT Master nicht unterstützt.

8.5.3 Registerkarte 'SyncManager'

Diese Registerkarte wird nur angezeigt, wenn im Slave Dialog "Allgemein" die "Experteneinstellungen" aktiviert sind.

Der Dialog zeigt die SyncManager-Konfiguration für diesen Slave an und erlaubt eine manuelle Änderung → nur für Experten empfohlen!

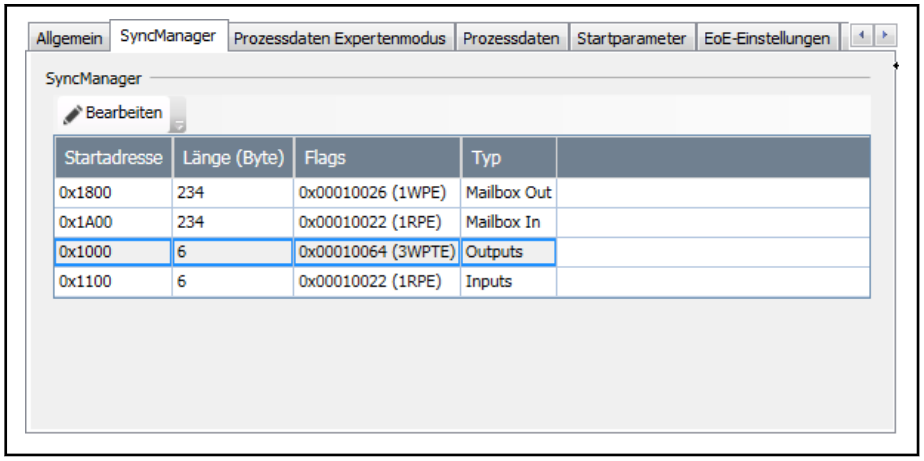


Abb. 8-44: Registerkarte "SyncManager"

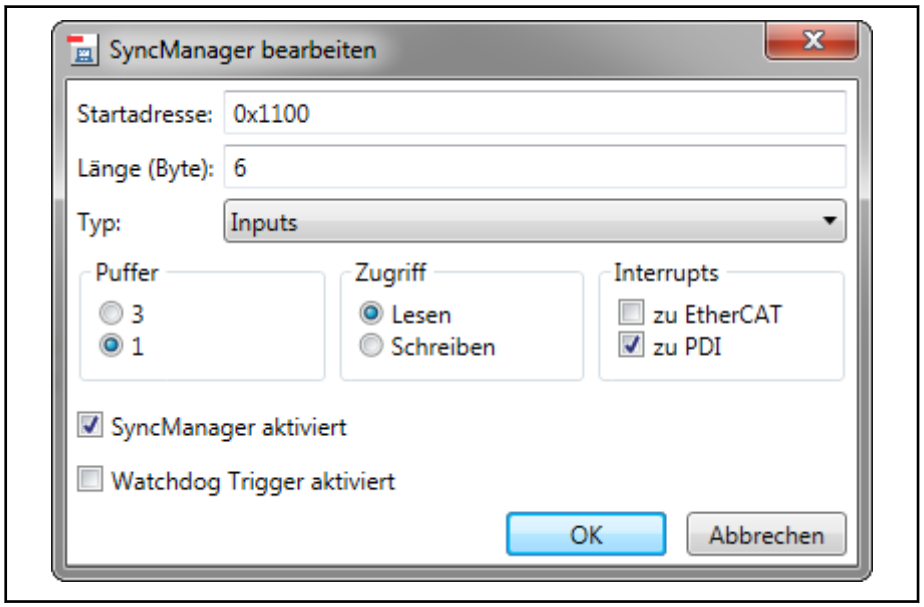


Abb. 8-45: Dialog "SyncManager bearbeiten"

Beispiel:

Bei SoE-Geräten, wie z.B. Rexroth IndraDrive, sind die SyncManager als Default mit "1 Puffer" eingestellt (entsprechend ETG-Spezifikation). In der Betriebsart Free Run kann dies zu Working-Counter Fehlern führen.

Daher wird für den Free Run Betrieb für die zyklischen Daten (Inputs / Outputs) die SyncManager-Einstellung "3 Puffer" empfohlen.

8.5.4 Registerkarte 'Prozessdaten Expertenmodus'

Diese Registerkarte bietet eine detaillierte Ansicht und Experten-Konfiguration der Prozessdaten, die auch im Dialog "Prozessdaten" dargestellt sind. Zusätzlich wird hier der Download der PDO-Zuordnung und der PDO-Konfiguration aktiviert.

Voraussetzung: dieser Dialog wird nur angezeigt, wenn im Slave Dialog "Allgemein" die "Experteneinstellungen" aktiviert sind.

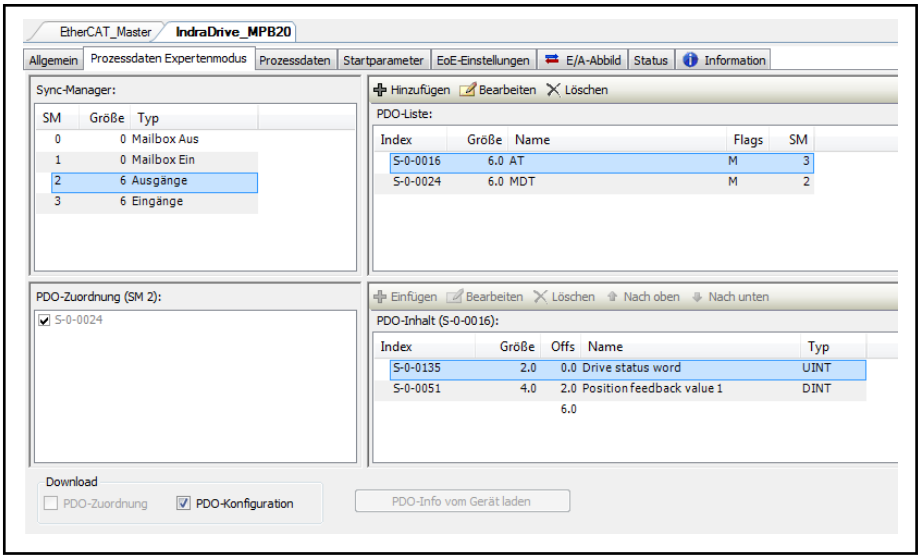


Abb. 8-46: Registerkarte "Prozessdaten Expertenmodus"

Sync-Manager Liste der Sync-Manager (SM) mit Datengröße und PDO-Typ (PDO: Prozess-Daten-Objekt).

PDO-Zuordnung (SM x) Liste der PDOs, die dem angewählten Sync-Manager (im Feld "Sync-Manager") zugeordnet sind.

Wenn eine Checkbox aktiviert wird, werden die PDOs aktiviert und die E/A-Kanäle erzeugt (vergleichbar zur Objekt-Auswahl im Dialog "Prozessdaten")

PDO-Liste Liste der PDOs, die dem angewählten Sync-Manager (im Feld "Sync-Manager") zugeordnet sind.

Über die entsprechenden Befehle in der Titelzeile oder des Kontextmenüs (Hinzufügen, Löschen, Bearbeiten) können weitere PDOs hinzugefügt oder bestehende PDOs editiert oder gelöscht werden (=> siehe "Bearbeite PDO-Liste").

Bearbeite PDO-Liste Dieser Dialog öffnet sich beim "Hinzufügen" oder "Bearbeiten" von PDOs in der "PDO-Liste".

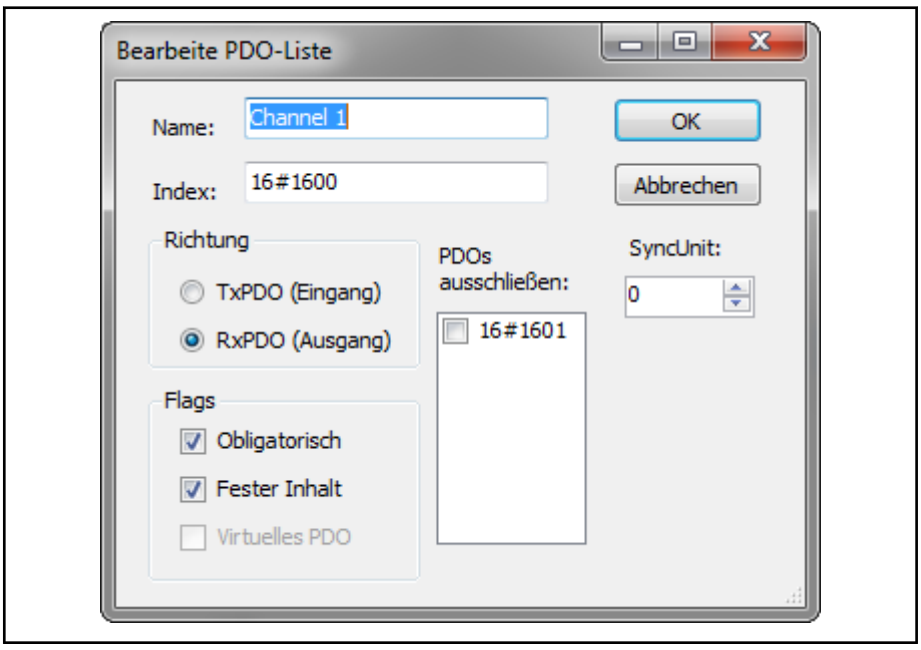


Abb. 8-47: Dialog "Bearbeite PDO-Liste"

Name	Name des PDO
Index	Index des PDO (oder IDN bei SoE)
Richtung	• TxPDO (Eingang in der SPS): PDO wird vom Slave zum Master übertragen • RxPDO (Ausgang in der SPS): PDO wird vom Master zum Slave übertragen
Flags	• Obligatorisch: das PDO ist notwendig und kann in der PDO-Zuordnung nicht deaktiviert werden. • Fester Inhalt: der Inhalt des PDO ist festgeschrieben und kann nicht geändert werden. Es ist dann nicht möglich, Einträge im PDO-Inhalt hinzuzufügen. • Virtuelles PDO: Reserviert für zukünftige Verwendung
PDOs ausschließen	Es ist möglich, eine Ausschluss-Liste festzulegen. Wenn ein PDO in der PDO-Zuordnung aktiviert ist, sind andere inaktiv und können nicht aktiviert werden.
SyncUnit	ID der SyncUnit im Slave (Sync-Manager), welchem das PDO zugeordnet werden soll

PDO-Inhalt

Zeigt den Inhalt des PDO an, das in der "PDO-Liste" angewählt ist.

Über die entsprechenden Befehle in der Titelzeile oder des Kontextmenüs (Hinzufügen, Löschen, Bearbeiten) können Einträge hinzugefügt oder bestehende Einträge editiert oder gelöscht werden.

Die Reihenfolge der PDOs kann über die Befehle "Nach oben" und "Nach unten" angepasst werden.

Download

PDO-Zuordnung	☑ Es werden CoE-Kommandos zum Initialisieren der Sync-Manager Objekte (0x1Cxx) erzeugt und auf den Slave geschrieben.
PDO-Konfiguration	☑ Es werden CoE-Kommandos für das PDO-Mapping (0x16xx u. 0x1Axx) erzeugt und auf den Slave geladen. Normalerweise kommen die Standardwerte aus der ESI-Datei und das Gerät muss dies unterstützen. Wenn beispielsweise ein Gerät eine feste Konfiguration hat, würden es diese Kommandos als fehlerhaft erkennen.

PDO-Info vom Gerät laden

Es wird die aktuelle PDO-Konfiguration aus dem Slave ausgelesen und in die Konfiguration eingetragen. Somit werden die Listen rechts oben und rechts unten komplett gelöscht und mit den gelesenen Daten gefüllt. Das ist immer dann sinnvoll, wenn die ESI-Datei unvollständig ist oder die Konfiguration nur im Slave vorhanden ist.

☞ "PDO-Info vom Gerät laden" wird vom Rexroth EtherCAT Master nicht unterstützt!

8.5.5 Registerkarte 'Prozessdaten'

Diese Registerkarte zeigt die Prozessdaten für die Ein- und Ausgänge des Slaves.

Die Daten stammen aus der Gerätebeschreibungsdatei (ESI) des Slaves.

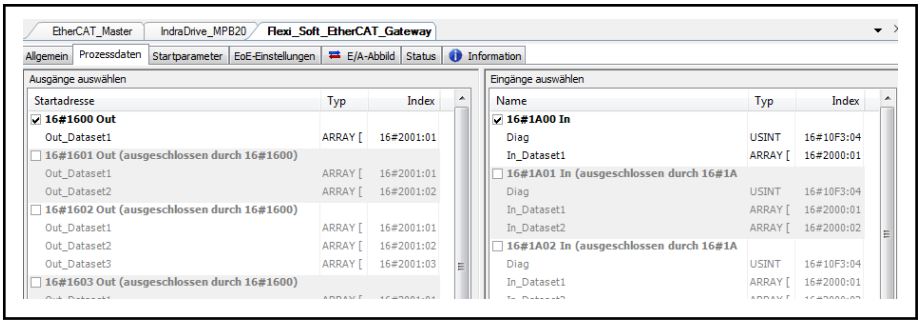


Abb. 8-48: Registerkarte "Prozessdaten"

Ausgänge auswählen

Die Tabelle zeigt die Ausgänge des Slaves definiert durch Startadresse, Typ, Index.

Wenn hier Ausgänge des Slaves aktiviert werden (☑), dann können diese Ausgänge im Dialog "E/A-Abbild" verwendet werden. Ausgangs-Objekte können durch andere (aktivierte) Ausgangs-Objekte ausgeschlossen sein (diese Abhängigkeiten sind in der Gerätebeschreibungsdatei des Slaves vorgegeben).

Eingänge auswählen

Die Tabelle zeigt die Eingänge des Slaves definiert durch Name, Typ, Index.

Wenn hier Eingänge des Slaves aktiviert werden (☑), dann können diese Eingänge im Dialog "E/A-Abbild" verwendet werden. Eingangs-Objekte können durch andere (aktivierte) Eingangs-Objekte ausgeschlossen sein (diese Abhängigkeiten sind in der Gerätebeschreibungsdatei des Slaves vorgegeben).

8.5.6 Registerkarte 'Startparameter'

In dieser Registerkarte werden Parameter definiert, die beim Bus-Hochlauf (bei Übergang PreOP zu SafeOP) in das Slave Gerät geschrieben werden.

Voraussetzung ist, dass das Gerät "CAN application layer over EtherCAT" (CoE) oder "Servo drive profile over EtherCAT" (SoE) unterstützt. Dementsprechend können dann SDOs (Service Data Objects) oder IDNs (Identifikations-Nummern) eingetragen werden.

Manche Module, die unterhalb eines Slaves eingefügt werden, verfügen über eigene Startparameter. Diese Parameter werden dann ebenfalls in dieser Liste angezeigt, können hier jedoch nicht editiert werden. Das Verändern der Parameter erfolgt im Editor des entsprechenden Moduls.

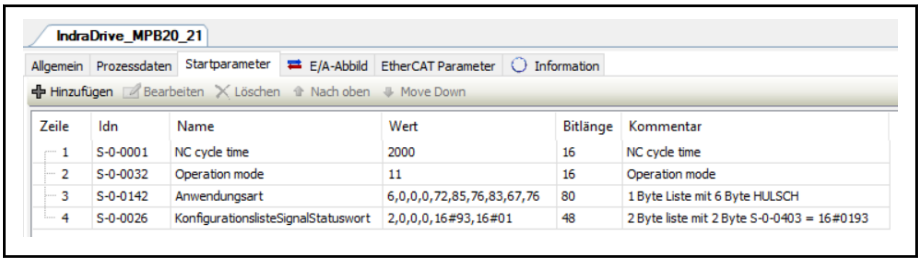


Abb. 8-49: Registerkarte "Startparameter"

Die Reihenfolge (von oben nach unten) in der Tabelle gibt die Reihenfolge vor, wie die SDOs bzw. IDNs zum Slave Gerät übertragen werden.

Zeile	Zeilen-Nummer
Idn	Identifikations-Nummer (SoE)
Index : Subindex	Index u. Subindex des Objekts (CoE)
Name	Name des Parameters
Wert	Wert des Parameters
Bitlänge	Bitlänge des Parameters
Kommentar	Eingabefeld für Kommentar
Nach oben	Verschiebt die selektierte Zeile um 1 Zeile nach oben
Nach unten	Verschiebt die selektierte Zeile um 1 Zeile nach unten
Hinzufügen	Öffnet den Eingabe-Dialog (SDO oder IDN) um Parameter der Startparameter-Tabelle hinzuzufügen
Bearbeiten	Öffnet den Eingabe-Dialog (SDO oder IDN) zum Bearbeiten des selektierten Parameters
Löschen	Entfernt den gerade selektierten Eintrag

IDN hinzufügen / bearbeiten (SoE)

Es kann ein Eintrag aus dem Objektverzeichnis ausgewählt werden (Voraussetzung ist, dass in der Gerätebeschreibungsdatei (ESI) des Slave Gerätes die Parameter beschrieben sind), oder es kann manuell ein Parameter durch die Eingabefelder definiert werden.

Spalten	
Idn	Identifikations-Nummer
Grundwert	Grundwert des IDN, Editierbar (Öffnen durch Doppelklick)
Eingabefelder	
IDN	Parametertyp: S oder P - Parameter
PSet	Parametersatz
Offset	Datenblock-Nummer
Bitlänge	Bitlänge
Wert	Wert
Is List	☑ Als Liste: Wenn die Option "als Liste" angewählt ist, können Sercos Listen angegeben werden. Die Option befindet sich im Dialog zur Auswahl des Objektverzeichnisses, siehe Beispiel "Beispiel 1 Byte Liste" auf Seite 230. Listen werden als Byte Arrays in folgendem Format angegeben: `<Ist Länge (LowByte)>,<Ist Länge (HighByte)>,0,0,<Byte1>,<Byte2>,<Byte3>...` → Das Byte Array muss im "Little Endian" Format angegeben werden.

SDO hinzufügen / bearbeiten (CoE)

Es kann ein Eintrag aus dem Objektverzeichnis ausgewählt werden (Voraussetzung ist, dass in der Gerätebeschreibungsdatei (ESI) des Slave Gerätes

die Objekte beschrieben sind), oder es kann manuell ein SDO durch die Eingabefelder definiert werden.

Spalten	
Flags	Anzeige des Zugriff-Flags: RW ("read write" = lesen und schreiben), RO ("read only" = nur lesen), WO ("write only" = nur schreiben)
Grundwert	Grundwert des Objekts, Editierbar (Öffnen durch Doppelklick)
Eingabefelder	
Name	Name des Objekts
Index	Index des Objekts (16#)
SubIndex	Subindex des Objekts (16#)
Bitlänge	Bitlänge
Wert	Wert
Byte Array	Das Objekt ist eine Liste.

Beispiel "1 Byte Liste"

Auf den Parameter "Anwendungsart" wird beispielsweise der String „Hulsch“ geschrieben (1 Byte Liste).

Der String "Hulsch" entspricht folgenden ASCII Zeichen:

72(H),**85**(u), **76**(l),**83**(s),67(c),**72**(h)

Der String "Hulsch" hat 6 Zeichen.

Somit beträgt die "Istlänge" 6 Byte und es ergibt sich für die Liste das Array: "6,0,0,0,72,85,76,83,67,72"

Das Array muss im Feld "Wert" eingegeben werden und zusätzlich muss die Option "als Liste" angewählt werden, wie im folgenden Bild dargestellt:

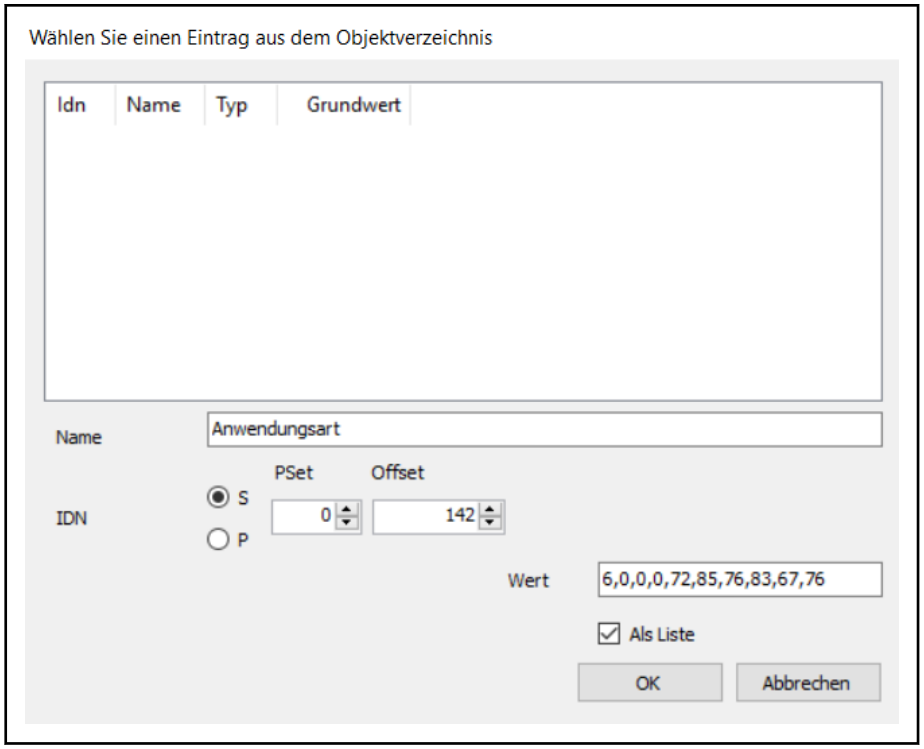


Abb. 8-50:

Beispiel "2 Byte Liste"

In die Parameterliste S26 "Konfigurations-Liste Signal-Statuswort" wird als erstes Listenelement die IDN S-0-403 geschrieben.

S-0-403 enstpricht `0x0193`

Somit ergibt sich für die "Ist Länge" 2Byte = `0x0002`

In das Feld "Wert" ist folgendes einzugeben: 2,0,0,0,16#93,16#01

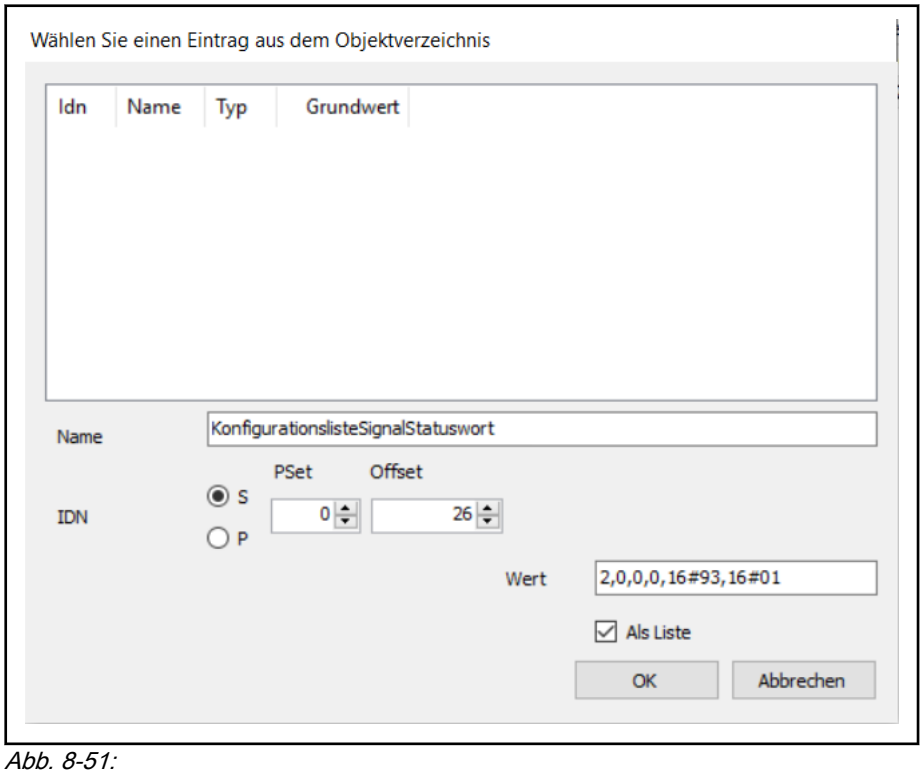


Abb. 8-51:

8.5.7 Registerkarte 'EoE-Einstellungen'

In dieser Registerkarte werden die Kommunikations-Einstellungen für Ethernet over EtherCAT (EoE) vorgenommen. Der Dialog wird nur angezeigt, wenn das Slave Gerät EoE unterstützt und "Experteneinstellungen" im Dialog "Allgemein" aktiviert sind.

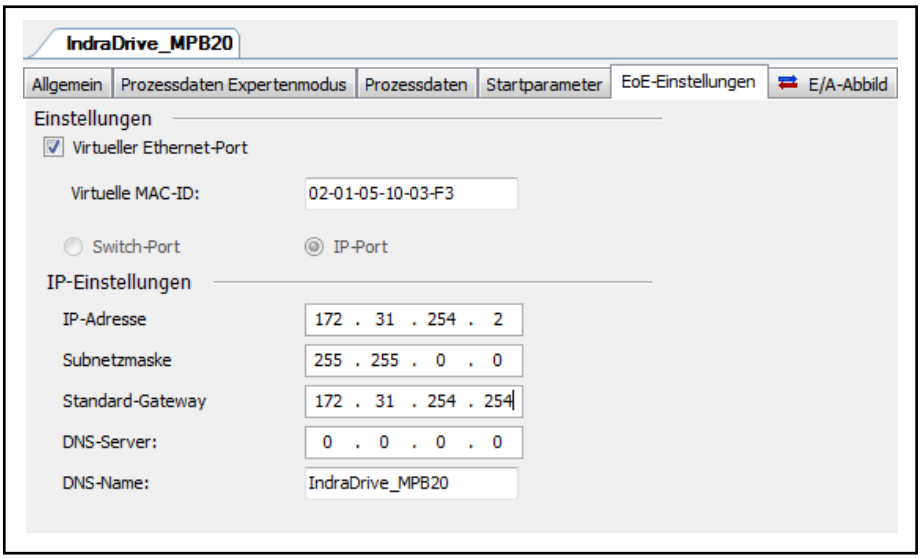


Abb. 8-52: Registerkarte "EoE-Einstellungen"

Registerkarte EoE-Einstellungen

Einstellungen	
Virtueller Ethernet Port	☑: aktiviert die EoE Funktionalität des Slaves. Eine eindeutige Virtuelle MAC-ID muss definiert werden
Virtuelle MAC-ID	Eingabefeld für die Virtuelle MAC-ID
Switch-Port	◉ Gerät arbeitet als Switch. Keine weiteren Netzwerk-Einstellungen erforderlich
IP-Port	◉ Gerät arbeitet als IP-Port. Es müssen IP-Einstellungen vorgenommen werden.
IP-Einstellungen **Es müssen gültige IP-Einstellungen entsprechend der Netzwerk-Konfiguration festgelegt werden.**	
IP-Adresse	IP-Adresse des Slaves im Netzwerk Beispiel: hat der EtherCAT Master die IP-Adresse 172.31.254.254 und die Subnetzmaske ist 255.255.0.0, dann muss die IP-Adresse der Slaves im Bereich zwischen 172.31.0.0 und 172.31.254.253 liegen. Siehe auch • Grundlagen: **"EoE: Ethernet over EtherCAT" auf Seite 183** • Master Konfiguration: Kap. 8.4.6 "Registerkarte 'EoE'" auf Seite 216
Subnetzmaske	Subnetzmaske des Slaves

Standard-Gateway	Standard-Gateway des Slaves (z.B. IP-Adresse des Masters)
DNS-Server	IP-Adresse des DNS-Servers
DNS-Name	Name des DNS-Servers

8.5.8 Registerkarte 'Online'

Diese Registerkarte wird nur im eingeloggten Zustand angezeigt.

Mit diesem Dialog kann der EtherCAT Zustand des Slave Gerätes geschaltet werden.

Der Slave Zustand darf nicht "höher" sein als der Master Zustand, siehe auch Kap. 8.4.5 "Registerkarte 'Busdiagnose'" auf Seite 214.

8.5.9 Registerkarte 'ESC Register'

Diese Registerkarte wird nur im eingeloggten Zustand angezeigt.

Der Dialog zeigt die Register-Werte des EtherCAT Slave Controllers (ESC) an.

Das Auslesen der Werte findet beim Öffnen des Dialogs sowie mittels Button "Aktualisieren" statt. Durch das Auswählen ("Anklicken") einer Adresse kann zudem auch ein Register zum Schreiben ausgewählt werden, siehe unterer Bereich des Dialogs.

⚠ VORSICHT

Durch das Lesen / Schreiben der Register können Meldungen vom Slave an den Master quittiert werden, die für den Master-Stack dann nicht mehr sichtbar sind.

Das Lesen / Schreiben der Register sollte daher nur bei Bedarf mit entsprechenden Expertenwissen erfolgen.

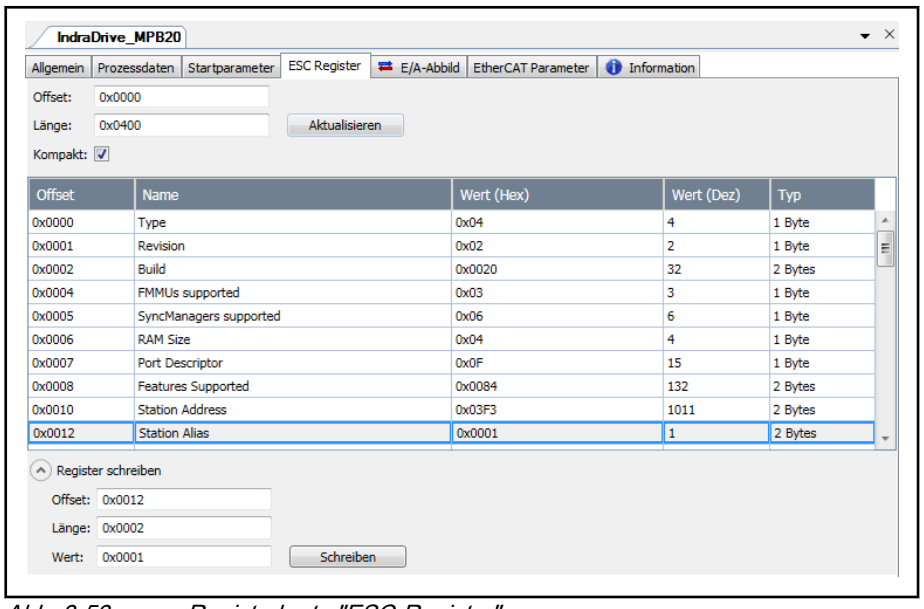


Abb. 8-53: Registerkarte "ESC-Register"

8.5.10 Registerkarte 'CoE'

Diese Registerkarte zeigt Informationen über die CoE Objekte eines EtherCAT Slaves mit CoE Profil an.

Die CoE Objektbeschreibungen (Offline aus ESI oder Online aus Gerät) sind optionale Dienste und werden daher nicht von allen CoE Slaves unterstützt.

Siehe Produktbeschreibung des Slave Gerätes.

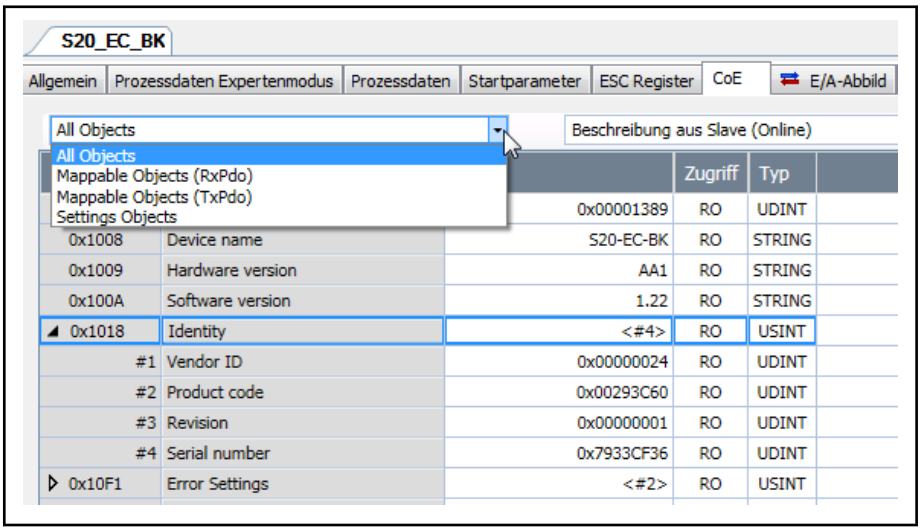


Abb. 8-54: Registerkarte "CoE"

In der rechten oberen Auswahlliste kann ausgewählt werden, ob:

- die Beschreibung aus der Gerätebeschreibung (ESI)[2] des Slaves angezeigt werden soll, oder
- die Beschreibung aus dem Gerät (Online) ausgelesen werden soll.

Bei einem Gerät mit Modular-Device-Profile, z.B. Rexroth S20-EC-BK, sind die Informationen zu den physikalisch vorhandenen Modulen nur Online vorhanden.

Voraussetzung ist, dass das Gerät das Auslesen der Informationen unterstützt (SDO Info-Dienste).

In der linken oberen Auswahlliste kann ausgewählt werden, welche Objekte angezeigt werden sollen[3] :

All Objects	(RxPDO): Objekte, die als Prozessdaten konfiguriert werden können (Ausgangsdaten)
Mappable Objects	(RxPDO): Objekte, die als Prozessdaten konfiguriert werden können (Ausgangsdaten)

[2] *Alternativ zur ESI-Gerätebeschreibung kann auch eine CANopen EDS-Datei ausgewählt werden. Mit dem Button rechts neben der Auswahlliste kann eine EDS-Datei für diesen Slave hinzugefügt werden.*

[3] *Es werden nur die Objektgruppen angezeigt, zu denen vom Slave eine Beschreibung bereit gestellt wird (z.B. ESI / Online Informationen können unterschiedlich sein).*

Backup Objects	Objekte, die im Slave gespeichert werden für einen Gerätetausch
Settings Objects	Objekte, die beim Hochlauf als Startparameter konfiguriert werden können

Im eingeloggten Zustand werden jeweils die aktuellen Werte aus dem Slave Gerät angezeigt.

Zum **Schreiben** der Objekte ist das entsprechende Feld in der Spalte "Wert" zu editieren. Voraussetzung ist, dass das ausgewählte Objekt den Zugriff "RW" hat (RO: "read only" = nur lesen; RW: "read write" = lesen und schreiben).

8.5.11 Registerkarte 'Diagnose'

Diese Registerkarte wird nur im eingeloggten Zustand angezeigt.

Es wird das CoE Objekt "Diagnosis History" (0x10F3) des Slave Gerätes dargestellt. Dieses Diagnose Objekt ist eine optionale Funktionalität und wird nicht von allen CoE Slaves unterstützt.

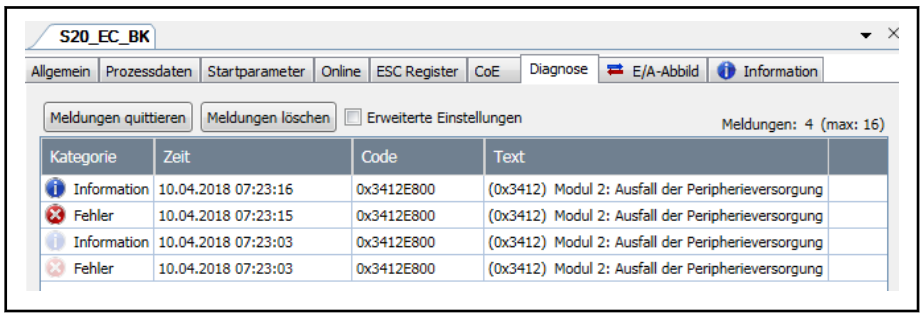


Abb. 8-55: Registerkarte "Diagnose"

Kategorie	Info / Warnung / Fehler Es gibt keine Standardisierung wie "kommende" oder "gehende" Fehler gemeldet werden. Siehe Produktbeschreibung des Slave Gerätes. Quittierte Meldungen werden durch ein ausgegrautes Symbol dargestellt.
Zeit	Der Zeitstempel wird vom Slave gemeldet: • ist "0" (= Datum 01.01.2000 00:00:00), wenn vom Slave kein Zeitstempel unterstützt wird, oder • ist ein lokaler Zeitstempel des Slave Gerätes (angezeigt wird die Zeit umgerechnet auf die Zeit des PCs), oder • entspricht der Uhrzeit der "Distributed Clocks", wenn DC 64 Bit unterstützt wird (z.B. entspricht der Zeit, die auf der Steuerung eingestellt ist).
Code	Diagnosecode
Text	Die Diagnosetexte werden aus der Gerätebeschreibung (ESI) des Slave Gerätes entnommen, d.h. auch die unterstützten Sprachen sind abhängig vom Slave Gerät (z.B. hauptsächlich wird nur Englisch unterstützt).

Tab. 8-1: Spaltenbeschreibung des "Diagnosis History"

Meldungen quittieren / löschen

Das Diagnose Logbuch wird als Ringpuffer betrieben und kann max. 250 Nachrichten speichern (Geräte-spezifisch ggf. auch weniger).

Nachrichten können quittiert und gelöscht werden.

Erweiterte Einstellungen

Es gibt verschiedene optionale Einstellungen, die im Slave Gerät gesetzt werden können. Wird eine bestimmte Option vom Slave nicht unterstützt, dann wird beim De-/Aktivieren ein Schreibfehler gemeldet (z.B. Parameterwert außerhalb gültigem Bereich).

☑ **Info / Warnung / Fehler aktivieren**: es kann de-/aktiviert werden, welche Diagnosetypen ins Logbuch eingetragen werden sollen.

☑ **Emergency aktivieren**: es kann de-/aktiviert werden, ob der Slave neue Diagnosen als CoE Emergency Nachrichten an den Master schicken soll.

☑ **AcknowledgeMode aktivieren**: man unterscheidet "OverwriteMode" und "AcknowledgeMode". Im "OverwriteMode" wird, wenn der Nachrichtenpuffer voll ist, die älteste Nachricht durch eine neue Nachricht überschrieben, unabhängig davon ob die Nachrichten quittiert wurden. Im "AcknowledgeMode" werden unquittierte Nachrichten nicht überschrieben, d.h. wenn der Nachrichtenpuffer voll ist, werden neue Nachrichten nicht gespeichert und gehen verloren.

8.5.12 Registerkarte 'E/A-Abbild'

Diese Registerkarte dient der Zuordnung der Ein- und Ausgänge des EtherCAT Slaves zu Variablen, die als lokale oder globale Variablen in den SPS verwendet werden können.

Im eingeloggten Zustand wird der aktuelle Wert der Variablen angezeigt.

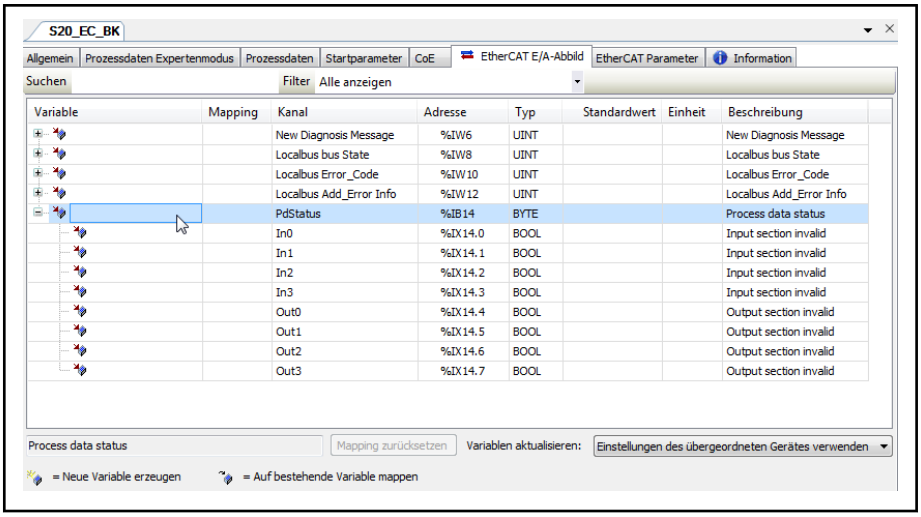


Abb. 8-56: Registerkarte "EtherCAT E/A-Abbild"

Registerkarte EtherCAT E/A-Abbild

Mapping zurücksetzen	Über diese Schaltfläche können die Mapping-Einstellungen auf die durch die Gerätebeschreibungsdatei definierten Standardwerte (i.d.R. leeres Mapping) zurückgesetzt werden
Variablen aktualisieren	Festlegung für das Gerät, ob die E/A-Variablen in der Buszyklus-Task aktualisiert werden: • **Einstellungen des übergeordneten Geräts verwenden**: Aktualisierung gemäß den Einstellungen des übergeordneten Geräts (siehe SPS-Einstellungen). • **Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet)**: Die E/A-Variablen werden in der Buszyklus-Task aktualisiert, falls sie in keiner anderen Task verwendet werden. • **Aktiviert 2 (immer im Buszyklus-Task)**: Alle Variablen werden in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

Prozessdaten Status

Der Prozessdaten Status "PdStatus" kann im Master Registerkarte 'Allgemein', Seite 209 aktiviert werden.

Der Eingang PdStatus zeigt die Gültigkeit der zyklischen Prozessdatenkommunikation an. Der PdStatus kann ungültig werden, wenn:

- kein zyklischer Datenaustausch stattfindet,
- ein zyklisches Datentelegramm nicht empfangen wird,
- ein zyklisches Telegramm mit ungültigen Daten empfangen wird (CRC Fehler)
- der Working Counter des zyklischen Kommandos nicht dem erwarteten Wert entspricht (siehe auch Grundlagen Kap. 8.2.8 "SyncUnits" auf Seite 185).

Der EtherCAT Master unterstützt Slaves mit bis zu 4 Cyclic Input Sync Manager und 4 Cyclic Output Sync Manager. Somit kann es pro Slave 4 Eingangsdatenbereiche und 4 Ausgangsdatenbereiche geben. Diese werden durch die 4 In (In0..In3) bzw. 4 Out (Out0..Out3) Bits des PdStatus abgebildet.

Momentan erzeugt IndraWorks nur Konfigurationen bei denen die 4 möglichen Eingangs- bzw. Ausgangsdatenbereiche eines Slaves zu einem Eingangsdatenstatus Bit In0 bzw. Ausgangsdatenstatus Bit Out0 zusammengefasst werden. Somit wird nur In0 für den Eingangsdatenbereich und Out0 für den Ausgangsdatenbereich wirksam.

Wenn der Prozessdatenaustausch der Eingangsdaten ungültig wird, geht das entsprechende Bit In0 auf TRUE (TRUE = Prozessdatenstatus ist ungültig). Bit Out0 = TRUE signalisiert, dass Ausgangsdaten vom EtherCAT Slave nicht übernommen werden.

Der PdStatus kann wie andere Eingangskanäle auf SPS - Variablen abgebildet werden oder direkt über die %IB Adresse abgefragt werden. Im SPS Programm kann man den Status des Prozessdatenaustausches unabhängig von den Eingangs- und Ausgangsdatenbereiche wie folgt überprüfen.

```
IF PdStatus = 0 THEN
; //valid cyclic data exchange
ELSE
; // invalid cyclic data exchange
END_IF
```

Abb. 8-57: Abfrage des Prozessdaten Status

8.5.13 Registerkarte 'Information'

Diese Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei (ESI) des Slaves stammen: unter anderem Name, Hersteller, Kategorie, Version und Beschreibung.

8.5.14 SoE Parametereditor

Der Parametereditor wird im eingeloggten Zustand im Kontextmenüs eines Slaves angeboten, wenn dieser das Mailbox Protokoll SoE unterstützt; z.B. IndraDrive mit SoE Profil.

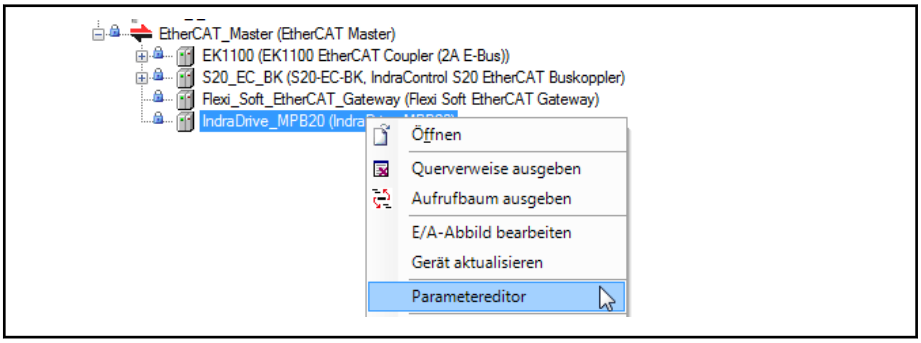


Abb. 8-58: SoE Parametereditor Aufruf

Der Parametereditor erlaubt die Anzeige und das Ändern eines einzelnen SoE Parameters, d.h. es können nur 16 Bit S / P - Parameter editiert werden (32 Bit Sercos III Parameter sind bei SoE nicht möglich).

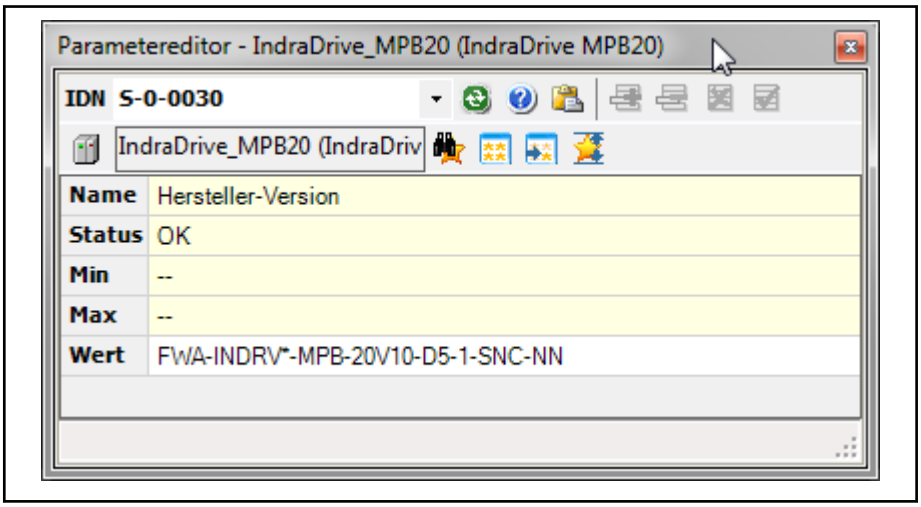


Abb. 8-59: SoE Parametereditor Eingabe

Die Funktionen "Parameter suchen" und "Parametergruppe" werden im SoE Parametereditor noch nicht unterstützt.

8.6 EtherCAT Master, Bus scannen

Funktion

Der "Bus scannen" Dialog zeigt links die Geräte an, die bereits konfiguriert sind, und rechts die Geräte, die aktuell am Bus gefunden wurden.

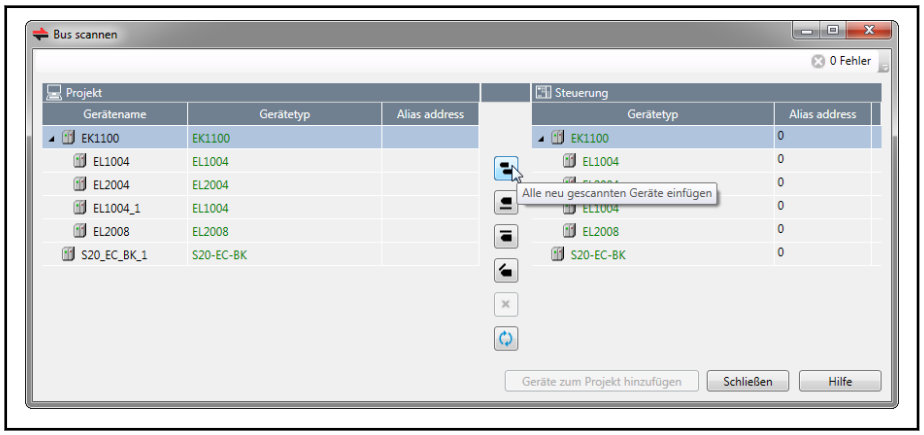


Abb. 8-60: EtherCAT Master - Dialog "Bus scannen"

Mit Hilfe der Buttons in der Dialogmitte können die gefundenen Geräte (rechts) in die Projektkonfiguration (links) übernommen werden. Dabei können entweder alle neuen Geräte eingefügt werden, oder einzelne Geräte markiert und bearbeitet werden (siehe Tooltip-Unterstützung der Buttons).

Aufruf

Im Projektexplorer über das Kontextmenü des EtherCAT Masters

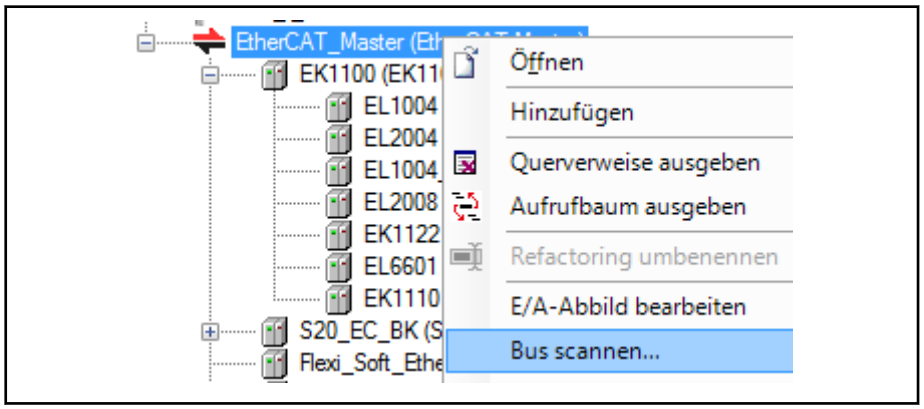


Abb. 8-61: EtherCAT Master - "Bus scannen" Aufruf

8.7 EtherCAT Master, Bus synchrone SPS-Verarbeitung

Für eine EtherCAT Bus synchrone SPS-Verarbeitung muss in der Task-Konfiguration der SPS eine Task mit dem externen Ereignis "EVENT_OP_MODE_ETHERCAT_CYCLIC" angelegt werden.

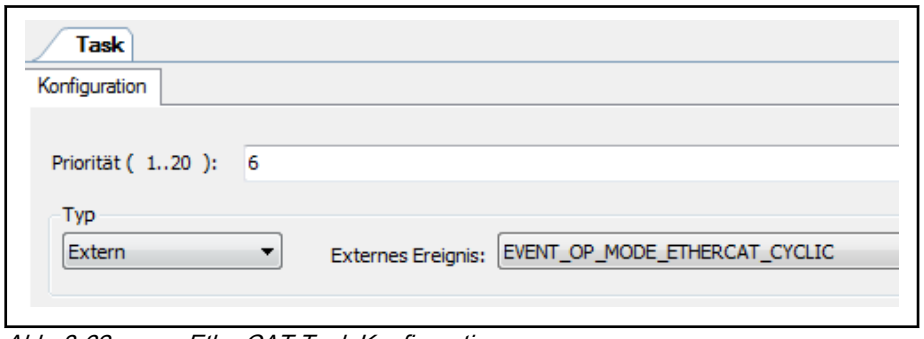


Abb. 8-62: EtherCAT Task Konfiguration

Die POUs, die mit dieser Task verknüpft sind, werden dann synchron zum EtherCAT Kommunikationszyklus aufgerufen. Es gilt folgender Ablauf:

- Der EtherCAT Stack wird vom System-Timer in der eingestellten EtherCAT Zykluszeit getriggert.
- Die zyklischen Ein- u. Ausgangsdaten werden verarbeitet (Empfangen u. Senden des EtherCAT Telegramms)
- Der EtherCAT Stack triggert die SPS-Task.



Derzeit gibt es keine Überwachung, ob beim nächsten EtherCAT-Event die verknüpfte SPS-Task Ausführung fertig ist. Der Standard Watchdog der SPS-Task kann unabhängig davon zur Überwachung der SPS-Task Ausführung verwendet werden.

- Jitter in der SPS Task-Ausführung haben keinen Einfluss auf die EtherCAT Telegramm Übertragung.

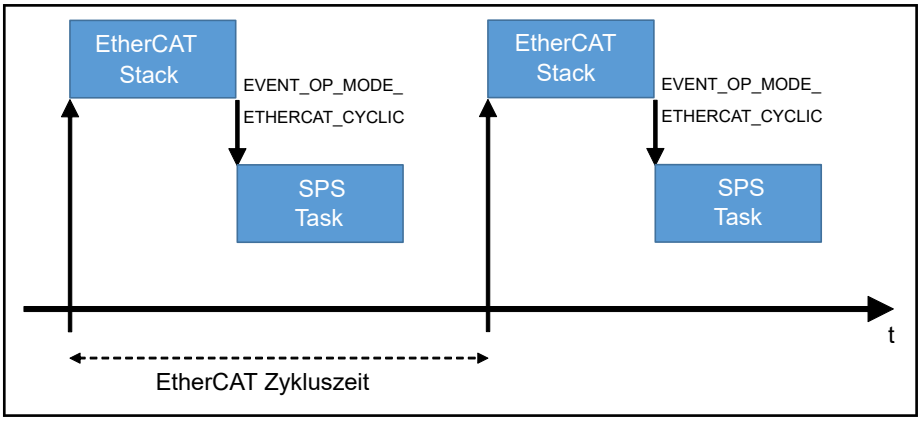


Abb. 8-63: EtherCAT Task Aufruf

8.8 EtherCAT FAQs

8.8.1 Übersicht

In den nachfolgenden Kapiteln sind Hinweise zu folgenden FAQs beschrieben:

- IndraDrive: Wie verbindet man IndraWorks Ds über EtherCAT?, Seite 240
- IndraDrive: Wie konfiguriert man SoE mit Free Run?, Seite 242
- S20-EC-BK: Buskoppler schaltet nicht in OP (Hinweise zu AL Status Codes), Seite 243
- S20-EC-BK: Wie erstellt man eine Anlaufparametrierung für die S20-Module?, Seite 244

8.8.2 IndraDrive: Wie verbindet man IndraWorks Ds über EtherCAT?

IndraWorks Ds kann eine Kommunikationsverbindung zum IndraDrive beim Rexroth EtherCAT Master entweder über MLPI (mit SoE) oder über EoE herstellen.

Variante A.) "IndraWorks Ds hier ..." auf Slave Knoten in IndraWorks

Der Aufruf "IndraWorks Ds hier ..." im Kontextmenü des IndraDrive Slave Knotens unterhalb eines EtherCAT Master wird nur bei IndraDrive mit SoE Profil angeboten.

Hierbei wird eine MLPI Verbindung zur MLC genutzt, die mittels Mailbox SoE azyklisch mit den Antrieben kommuniziert.

Für IndraDrive mit CoE Profil werden nachfolgende Varianten C.) und D.) mit EoE empfohlen.

Variante B.) IndraWorks Ds mit MLPI zum EtherCAT Master

In IndraWorks Ds ist im Dialog Verbindungsauswahl, Registerkarte Fremdsteuerung, als Typ "MLC (EtherCAT)" auszuwählen.

Als IP-Adresse ist der Eng.-Port der MLC Steuerung einzugeben. IndraWorks Ds nutzt hierbei eine MLPI Verbindung zur MLC, die mittels Mailbox SoE azyklisch mit den Antrieben kommuniziert.

Die IndraWorks Ds mit MLPI Verbindung kann nur für Antriebe mit SoE Profil verwendet werden. Eine EoE Konfiguration oder IP-Routing Einstellungen auf dem Eng.-PC sind hierbei nicht erforderlich.

Für IndraDrive mit CoE Profil werden nachfolgende Varianten C.) und D.) mit EoE empfohlen.

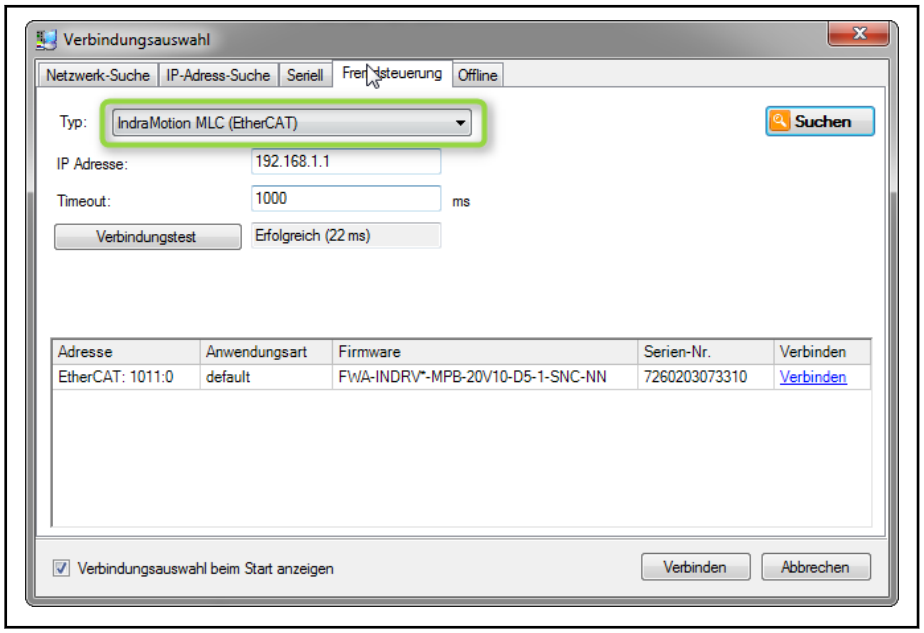


Abb. 8-64: IndraWorks Ds mit MLC-Steuerung (EtherCAT) verbinden

Variante C.) IndraWorks Ds mit EoE und Switch Port

Mit einem Switch Port, z.B. Beckhoff EL6601, können dezentral beliebige Ethernet Gerät an ein EtherCAT Netzwerk angeschlossen werden. Auf diese Weise ist IP Kommunikation mit den EtherCAT Slaves möglich.

Beim IndraDrive muss eine EoE Konfiguration eingestellt werden, bei denen die IP Einstellungen im gleichen Netzwerk sind wie der Eng.-PC, siehe auch EoE Konfigurations-Beispiel, Seite 184.

In IndraWorks Ds kann dann mittels Netzwerk-Suche oder IP-Adress-Suche eine Verbindung hergestellt werden. Ein IP-Routing auf dem Eng.-PC ist nicht erforderlich.

Variante D.) IndraWorks Ds mit EoE und IP-Routing im EtherCAT Master

Der Eng.-PC mit IndraWorks Ds ist mit dem Eng.-Port der Steuerung verbunden und die Steuerung führt ein IP Routing auf EoE in das EtherCAT Netzwerk aus.

Beim IndraDrive und beim EtherCAT Master muss eine EoE Konfiguration eingestellt werden, so dass eine gültige IP Konfiguration im EtherCAT Netzwerk entsteht, siehe auch EoE Konfigurations-Beispiel, Seite 184.

Bevor in IndraWorks Ds dann mittels Netzwerk-Suche oder IP-Adress-Suche eine Verbindung hergestellt werden kann, muss manuell noch die IP Routing-Tabelle des Engineering-PC's mit einem entsprechenden Routing-Eintrag erweitert werden (siehe z.B. Windows Kommandozeile, Befehl "route help").

8.8.3 IndraDrive: Wie konfiguriert man SoE mit Free Run?

1. **IndraDrive als Slave Gerät hinzufügen**

   IndraDrive mit entsprechender Version MPxVyy unterhalb EtherCAT Master im Projektexplorer anlegen, hier am Beispiel MPB20:

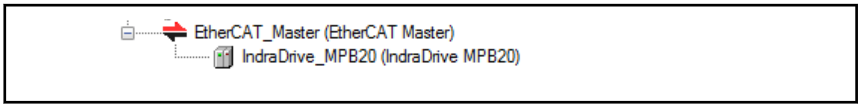


Abb. 8-65: IndraDrive als Slave Gerät

2. **Einstellungen im Slave Dialog "Allgemein"**

   Für Free Run Betrieb ist bei der Einstellung "Verteilte Uhren" die Auswahl "Free Run" auszuwählen. Dies ist ein Einstellwert aus der Gerätebeschreibungsdatei des Slaves (hier: aus IndraDrive MPB20).

   Zudem ist in diesem Dialog noch "Experteneinstellungen aktivieren" zu setzen, damit der zusätzliche Dialog "Prozessdaten Expertenmodus" sichtbar wird.

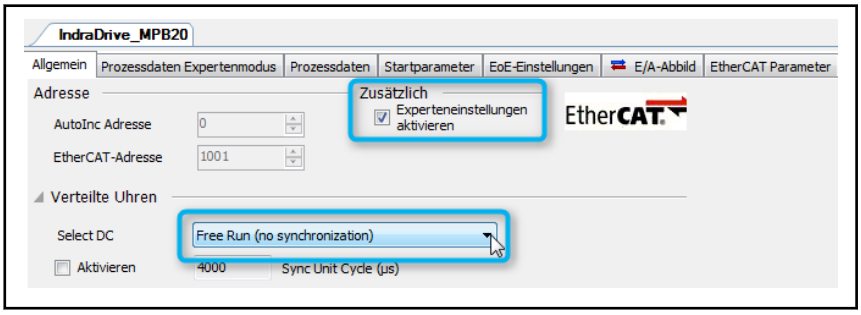


Abb. 8-66: IndraDrive Konfiguration "Allgemein"

3. **Einstellungen im Slave Dialog "Prozessdaten Expertenmodus"**

   In diesem Dialog können die gewünschten Prozessdaten für den Drive manuell konfiguriert werden.

   Im Feld rechts oben "PDO-Liste" wird entweder S-0-0016 (AT) oder S-0-0024 (MDT) ausgewählt, abhängig davon werden rechts unten "PDO-Inhalt" dann die aktuell konfigurierten zyklischen Parameter angezeigt. Mittels den Funktionen Einfügen / Bearbeiten / Löschen können im Dialog rechts unten die gewünschten Parameter für das AT (Antriebstelegramm) bzw. für das MDT (Master-Daten-Telegramm) manuell konfiguriert werden.

   Wenn in der Gerätebeschreibung des Slaves keine vordefinierten Parameter vorhanden sind (wie im nachfolgendem Screenshot dargestellt), dann müssen Name, Parameternummer (IDN) und Datentyp manuell richtig eingegeben werden.

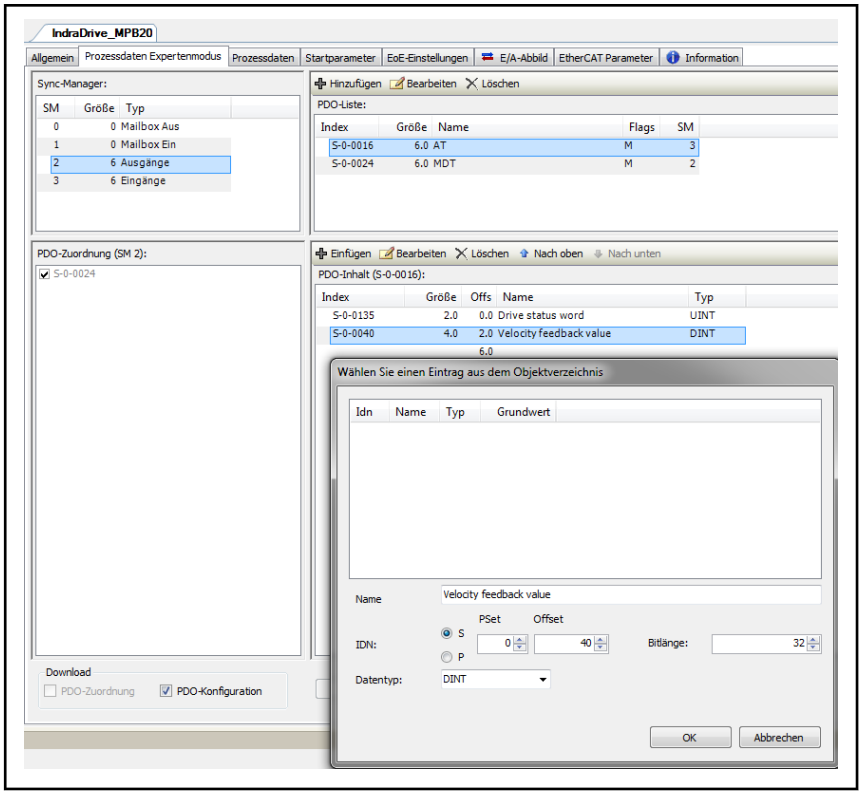


Abb. 8-67: IndraDrive Konfiguration Prozessdaten

4. **Einstellungen im Slave Dialog "Startparameter"**

   Für Free Run Betrieb ist als Startparameter S-0-0002 ("Sercos Zykluszeit") mit dem gleichen Wert anzulegen wie S-0-0001 ("NC-Zykluszeit"). Während S-0-0001 bereits über die Gerätebeschreibung als Startparameter vordefiniert ist, muss S-0-0002 als 16 Bit Parameter manuell hinzugefügt werden.

   Der Wert von S-0-0001 entspricht der EtherCAT Zykluszeit und wird automatisch vom Konfigurator gesetzt. Für S-0-0002 ist der gleiche Wert einzustellen.

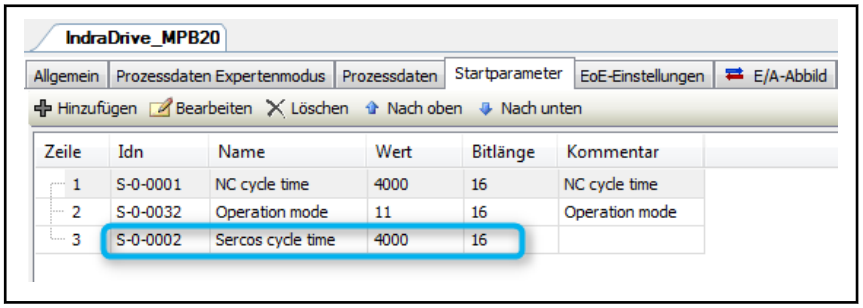


Abb. 8-68: IndraDrive Konfiguration Startparameter

5. **Experteneinstellungen "SyncManager"**

   Die SyncManager-Konfiguration steht bei SoE als Default auf "1 Puffer"-System. Für den Free Run Betrieb wird die Einstellung "3 Puffer" empfohlen, siehe Kap. 8.5.3 "Registerkarte 'SyncManager'" auf Seite 224.

8.8.4 S20-EC-BK: Buskoppler schaltet nicht in OP (Hinweise zu AL Status Codes)

Wenn der EtherCAT Bus in OP geschaltet wird und der S20-EC-BK den Zustand OP nicht erreicht, dann meldet der Buskoppler normalerweise einen AL

Status Code. Dieser AL Status Code wird beispielsweise im EtherCAT Master "Registerkarte 'Busdiagnose', Seite 214" angezeigt und kann verschiedene Ursachen haben:

AL Status Code	Beschreibung
0x001D / 0x001E	Ungültige Sync Manager Konfiguration (Ausgänge) / Ungültige Sync Manager Konfiguration (Eingänge). Abhilfe: Diese Fehler können u.a. als Ursache haben, wenn die konfigurierten Module mit den physikalischen Modulen nicht übereinstimmen. Siehe weitere Hinweise bei AL Status Code 0x0070.
0x0028	Sync Modus nicht unterstützt Der Modus "DC Synchronous" wird nur dann unterstützt, wenn mindestens ein S20-Modul am Buskoppler gesteckt ist, das die Lokalbus-Synchronisation unterstützt, z.B. S20-DI-16/1-HS. Siehe Datenblätter der S20-Module.
0x002B	Keine gültigen Ein- und Ausgangsdaten. Abhilfe: Überprüfen Sie die Steckverbindungen der S20 Bussockel-Module sowohl beim Buskoppler als auch bei den Modulen und starten Sie den Buskoppler dann neu.
0x0070	Ungültige Modul-Konfiguration. Die konfigurierten Module (CoE Objekt 0xF030) stimmen nicht mit den physikalischen Modulen (CoE Objekt 0xF050) überein. Abhilfe: 1. Überprüfen Sie die Modul-Konfiguration im IndraWorks Konfigurator mit dem physikalischen Aufbau. 2. Im Slave Registerkarte 'CoE', Seite 234 können die konfigurierten Modulnummern im Objekt 0xF030 mit den physikalisch gefundenen Modulen im Objekt 0xF050 verglichen werden. In den Objekten 0x9yyy können weitere Informationen zu den vorhandenen Modulen ausgelesen werden. 3. Werden zu einem physikalischen Modul keine Informationen angezeigt, überprüfen Sie die Steckverbindungen der S20 Bussockel Module. 4. Bei einer Änderung des physikalischen Aufbaus oder wenn Steckverbindungen getrennt wurden, muss der Buskoppler neu gestartet werden.

8.8.5 S20-EC-BK: Wie erstellt man eine Anlaufparametrierung für die S20-Module?

Eine Parametrierung, inkl. Anlaufparametrierung, der S20-Module ist in folgender Anwendungsbeschreibung für den S20-EC-BK erklärt:

http://www.boschrexroth.com/various/utilities/mediadirectory/index.jsp?publication=NET&search_query=**DOK-CONTRL-S20*EC*BK**-AP**

9 Sercos III IO

9.1 Features des Sercos III-IO-Master

Die aktuelle Implementation des Sercos III-IO-Master beinhaltet folgende Funktionalitäten:

- zyklischer Datenverkehr
- SDDML-Dateiimport in die Gerätedatenbank
- Busscan von Sercos III-Slaves
- Remote-Adressvergabe
- Automatische Konfiguration der Geräte im Gerätebaum (Offline-/Online-Abgleich der Geräte im Projekt)
- Sercos III-slavebezogener Verbindungsstatus in der Oberfläche
- Diagnose des Sercos III-IO-Master und der Slaves über FB
- azyklische Datenübertragung (Service-Kanal bzw. IP-Kanal) über FB

Funktion / Eigenschaft	Wert
Maximale Anzahl an I/O-Slaves, treiberseitig[1)]	256
Baudrate	100 MBit/s
Auto negotiation/ autocrossing	Ja
Min. Zykluszeit	VEP: 1 ms
Max. Anzahl zyklischer Eingangsdaten	5600 Byte
Max. Anzahl zyklischer Ausgangsdaten	5600 Byte
Max. Anzahl zyklischer Eingangsdaten pro Slave	5600 Byte
Max. Anzahl zyklischer Ausgangsdaten pro Slave	5600 Byte
Max. Anzahl an Modulen pro Slave	61
Max. Telegrammdaten azyklisch pro Slave / Telegram (MTU)	1500 Byte

Tab. 9-1: Technische Daten

9.2 Begriffe und Abkürzungen

Master Der Sercos III-Feldbus-Master wird **Master** genannt.

Slave Die Sercos III-Feldbus-Slaves nennt man **Slaves**.

Sercos-Adresse Das Kriterium zur Adressierung eines Slaves ist die Sercos-Adresse.

SDDML-Datei Dies ist die Gerätebeschreibung, mittels derer IndraWorks die Sercos III-Slaves (typischerweise mit IO-Profil) bekannt gemacht werden.

1) Treiberseitig werden 256 I/O-Ressourcen unterstützt. Diese Zahl wird steuerungshardwareabhängig durch die einzelnen Systemoberflächen reduziert.

9.3 Sercos III konfigurieren

9.3.1 Sercos III konfigurieren, Übersicht

Sercos III ist ein offener und IEC-konformer Universalbus für die ethernetbasierte Echtzeit-Kommunikation. Als Universalbus verfügt Sercos III über Kommunikationskanäle und Geräteprofile für alle gängigen Automatisierungsanwendungen.

Sercos III-Geräte können mit allen Steuerungen genutzt werden, die Sercos-Master-Funktionalität unterstützen. In IndraWorks verfügen diese Steuerungen über den Sercos-Objekt(Onboard)-Knoten (Funktion eines Sercos-Masters).

- Der Master kann durch Slaves und Module erweitert werden, so dass diese anschließend – abhängig von der Gerätebeschreibungsdatei – konfiguriert werden können.
- Vom Sercos-Objekt aus können durch den Kontextmenü-Befehl Buskonfiguration scannen alle am Sercos-Bus laufenden Geräte ermittelt und in das IndraWorks-Projekt übernommen werden.

Sercos III Master

- Sercos III-Master, Übersicht Busadressen auf Seite 247
- Sercos III-Master, Verkabelung und Redundanz (am Beispiel eines MLC-Systems) auf Seite 255
- Registerkarte 'Einstellungen' auf Seite 257

Sercos III Master, Slave hinzufügen

- Sercos III-Master, Slave hinzufügen, Allgemeines auf Seite 259
- Sercos-Slaves aktivieren / deaktivieren auf Seite 261
- Sercos-Slaves / Sercos-Module deaktivieren mittels UserDefs.cfg auf Seite 262, nur System MTX
- Registerkarte 'Allgemein' auf Seite 265
- Registerkarte 'sercos Konfiguration' auf Seite 266, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde
- Registerkarte 'Status' auf Seite 266
- Registerkarte 'Information' auf Seite 266

Sercos III Master, Module zum Slave hinzufügen

- Sercos III Master, Module zum Slave hinzufügen, Allgemeines auf Seite 267
- Sercos-Module aktivieren / deaktivieren auf Seite 267
- Registerkarte 'Sercos Modul' auf Seite 268
- Registerkarte 'Funktionsgruppen' auf Seite 269
- Registerkarte 'Allgemeine Ein- und Ausgänge' auf Seite 269
- Sonder-Register Registerkarte 'Benutzerdefinierte Parameter' auf Seite 271
- Registerkarte 'Oversamplingauf Seite 271
- Registerkarte 'Sercos Module Konfiguration' auf Seite 273, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde

- Registerkarte 'E/A-Abbild' auf Seite 274
- Registerkarte 'Information' auf Seite 275

9.3.2 Sercos III-Master, Übersicht Busadressen

Der Dialog "Übersicht Busadressen" dient zur übersichtlichen Darstellung und Bearbeitung der Adressen von Sercos-Geräten. Er kann über das Kommando "Übersicht Busadressen" im Kontextmenü des Sercos-Knotens im Projektexplorer aufgerufen werden.

Im Dialog werden alle Sercos-Geräte mit ihren wesentlichen Eigenschaften tabellarisch aufgelistet. Die Adressen können nach Auswahl der entsprechenden Zelle bearbeitet werden. Zellen die nicht bearbeitet werden können, sind ausgegraut. Die aktuell ausgewählte Zelle wird farblich hervorgehoben.

Die Daten der Tabelle werden auf Gültigkeit geprüft. Dabei kann es zu Fehlern und Warnungen kommen, die durch ein entsprechendes Symbol in der ersten Spalte gekennzeichnet werden. Im Tooltip auf dem Symbol werden die konkreten Fehler bzw. Warnungen angezeigt. Solange mindestens ein Fehler ansteht, ist das **Übernehmen** der Änderungen bzw. das Beenden des Dialogs mit **OK** nicht möglich.

Alle Änderungen werden erst dann wirksam, wenn auf **Übernehmen** gedrückt wird oder der Dialog mit **OK** verlassen wird.

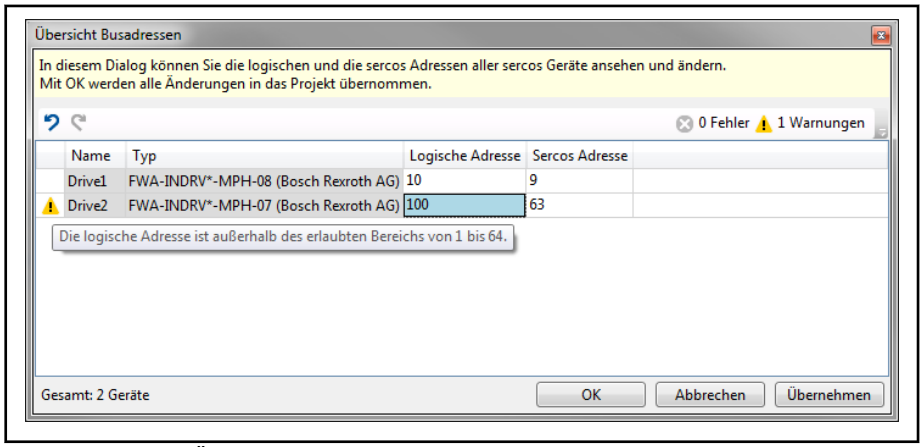


Abb. 9-1: Übersicht Busadressen

Oberhalb der Tabelle befindet sich eine Symbolleiste mit folgenden Funktionen:

Symbol	Name	Kurztaste	Bedeutung
	Rückgängig	Strg+Z	Wurde in der Tabelle Änderungen durchgeführt, können diese Änderungen schrittweise durch Auswahl dieser Funktion rückgängig gemacht werden. Diese Funktion steht nicht mehr zur Verfügung, nachdem die Änderungen mit **Übernehmen** übernommen wurden
	Wiederholen	Strg+Y	Wurden Änderungen rückgängig gemacht, können diese Änderungen schrittweise durch Auswahl dieser Funktion wiederhergestellt werden. Diese Funktion steht nicht mehr zur Verfügung, nachdem die Änderungen mit **Übernehmen** übernommen wurden

Sym-bol	Name	Kurz-taste	Bedeutung
	Fehler		Die Daten der Tabelle werden auf Gültigkeit geprüft. An dieser Stelle wird die Anzahl der gefundenen Fehler angezeigt. Die Sercos-Geräte mit den Fehlern sind durch das gleiche Symbol in der ersten Spalte der Tabelle gekennzeichnet. Solange mindestens ein Fehler ansteht, ist das Übernehmen der Änderungen bzw. das Beenden des Dialogs mit **OK** nicht möglich
	Warnungen		Die Daten der Tabelle werden auf Gültigkeit geprüft. An dieser Stelle wird die Anzahl der gefundenen Warnungen angezeigt. Die Sercos-Geräte mit den Warnungen sind durch das gleiche Symbol in der ersten Spalte der Tabelle gekennzeichnet

Tab. 9-2: Funktionen der Symbolleiste

9.3.3 Sercos III-Master, Sercos-Busdiagnose

Ein Doppelklick auf den Sercos-Master-Knoten im Projektnavigator öffnet das Diagnosefenster für den Sercos-Bus.

Bezüglich dieser Diagnose weisen alle Sercos-Teilnehmer, IndraDrive wie auch Sercos-IOs ein ähnliches Verhalten auf:

Sercos-Teilnehmer besitzen zur Diagnose:

- S-0-0390, Diagnose-Nummer
- S-0-0095, Diagnose, als Text
- Sercos IO-Geräte bieten zusätzlich eine IO-spezifische Diagnose-Meldung (S-0-1500.0.32, IO Diagnostic Message)

Klasse 1 Fehler können quittiert werden, durch die Ausführung des Kommandos S-0-0099, C0500 Reset Zustandsklasse 1).

Diese zeigt detaillierte Diagnosen von dem meldenden Sercos IO-Gerät an. Weiterführende Informationen sind in diesem Fall der Dokumentation des Gerätes zu entnehmen.

Details siehe auch Der Sercos-IO-Statuscode ist enthalten in: auf Seite 250.

Die Anzeige des Sercos III-Diagnosefenster ist auf den Anwendungszweck des jeweiligen Bosch Rexroth Steuerungssystems optimiert. Deshalb gibt es in einzelnen Details Unterschiede in Abhängigkeit des verwendeten Steuerungssystems (MLC, MTX...).

Nachstehend werden die wesentlichen Diagnoseschritte am Beispiel eines MLC-Systems gezeigt.

1. Im IW-Engineering in dem betreffenden Projekt den Dialog/Tab-Reiter "E/A-Diagnose" am Sercos-Knoten öffnen, (Doppelklick auf das Sercos-Symbol im IW-Projektbaum).
2. Mit der IndraLogic-Applikation einloggen (SPS eingeloggt, ist die Voraussetzung für die nächsten Schritte).
3. Fehlermeldendes Sercos IO-Gerät auswählen
4. Im Detailbereich auf **Erweiterter Gerätestatus**, **Geräte-Identifikation** oder **Geräte-Kommunikation** klicken.

Auswahlhilfe

4.1 **Erweiterter Gerätestatus**: zeigt alle allgemeinen Diagnosen an, und bieten einen guten Einstiegspunkt

4.2 **Geräte-Identifikation**: zeigt Diagnosen zum Geräte/Hardwareaufbau an, dieser Dialog enthält wertvolle Informationen für die Erstinbetriebnahme oder die Beseitigung von Modulausfällen

4.3 **Geräte-Kommunikation**: zeigt alle Sercos-verbindungsrelevanten Diagnosen an

5. **Erweiterter Gerätestatus ▸ Diagnostic number**

   Darunter können durch Erweitern/Aufklappen der Elemente, der Fehlertyp, die Fehlerklasse und der Status-Code angezeigt.

   Der Status-Code, wird textuell interpretiert und gibt einen ersten Anhaltspunkt.

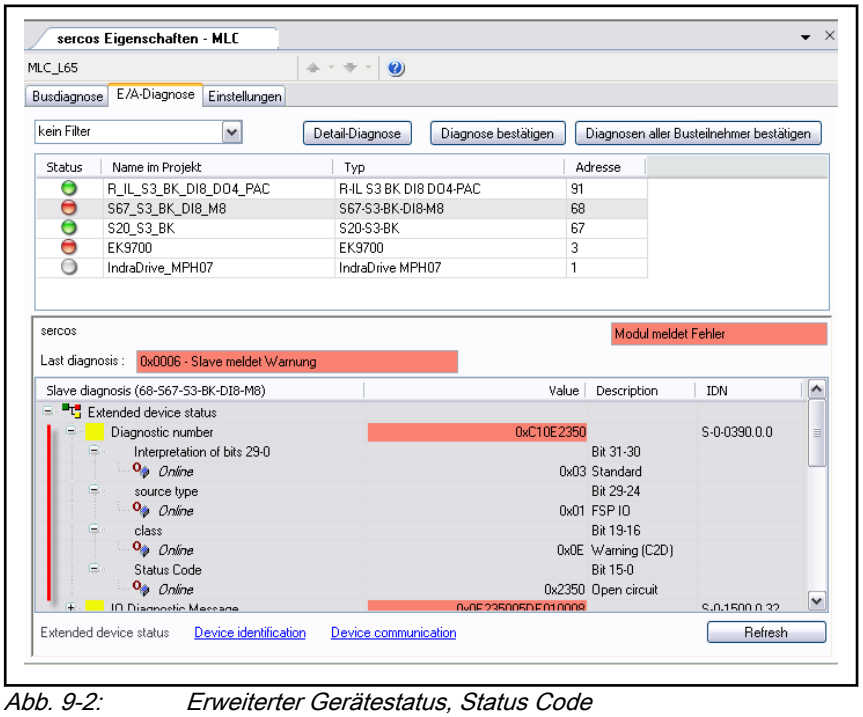


Abb. 9-2: Erweiterter Gerätestatus, Status Code

6. **Erweiterter Gerätestatus ▸ IO Diagnostic Message** ergänzt die **Diagnostic number** um die IO-Gerät relevanten Diagnosen, wie Slot- und Kanalnummer. Damit lässt sich an einem modularen Sercos-Gerät das entsprechende Modul und dessen Kanal identifizieren.

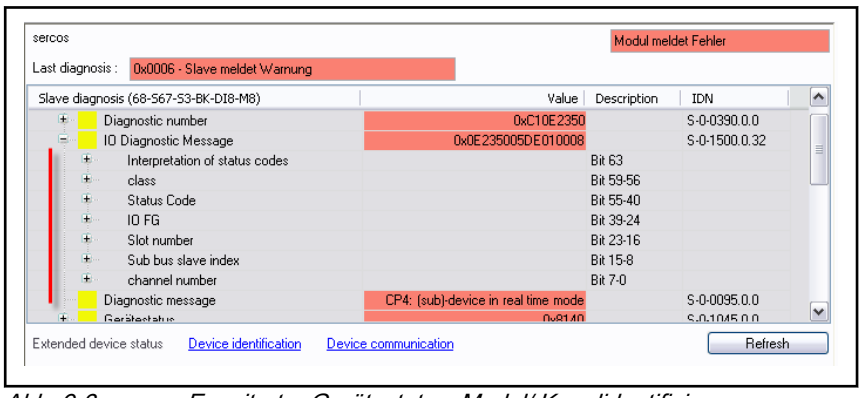


Abb. 9-3: Erweiterter Gerätestatus, Modul/ Kanalidentifizierung

7. **Geräte-Identifikation** wird interessant, wenn Module verkehrt projektiert, installiert (gesteckt) oder defekt sind.

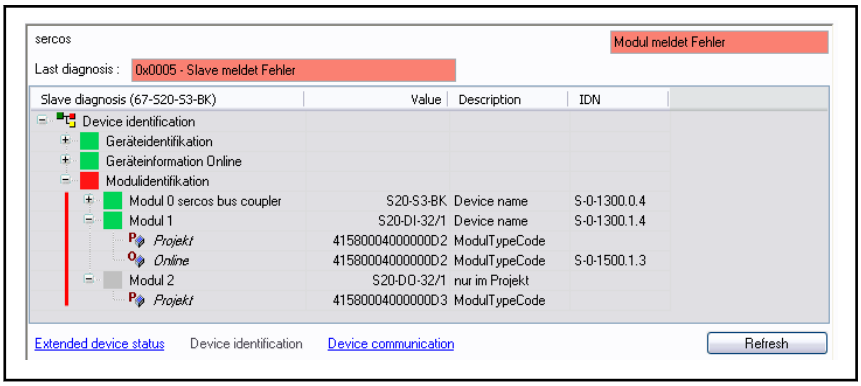


Abb. 9-4: Geräte-Identifikation, Modul-Identifikation

Der Sercos-IO-Statuscode ist enthalten in:

- S-0-0390, Diagnose-Nummer

  Quelltyp (Bit 29-24) | Klasse (Bit 19-16) | Statuscode (Bit 15-0)

  z.B. 16#C10E2350 ⇒ C1 | 0E | 2350
- S-0-1500.0.32 I/O diagnostic message

  Klasse (Bit 59-56) | Statuscode (Bit 55-40) | Funktionsgruppe (Bit 39-24) | Slotnummer (Bit 23-16) | Subbus-Slave (Bit 15-8) | Kanalnummer (Bit 7-0)

  z.B. 16#0E235005DE010008 ⇒ 0E | 2350 | 05DE | 01 | 00 | 08

Wert	Beschreibung
Statuscodes mit den Diagnoseklassen Betriebszustand (0xC10A), Warnung (0xC10E), Fehler (0xC10F)	
16#0	Kein Fehler
16#1000	Allgemeiner Fehler
16#2000	Strom
16#2100	Strom auf der Geräteeingangsseite
16#2110	Kurzschluss/Erdschluss
16#2120	Erdschluss
16#2130	Kurzschluss
16#2136	Kurzschluss an VCC
16#2137	Kurzschluss an Erde
16#2200	Strom innerhalb des Gerätes
16#2213	Überstrom beim Hochlauf
16#2214	Überstrom im Betrieb
16#2220	Dauerüberstrom
16#2230	Kurzschluss/ Erdschluss
16#2240	Erdschluss
16#2250	Kurzschluss
16#2300	Strom auf der Geräteausgangsseite

Wert	Beschreibung
Statuscodes mit den Diagnoseklassen Betriebszustand (0xC10A), Warnung (0xC10E), Fehler (0xC10F)	
16#2310	Dauerüberstrom
16#2320	Kurzschluss/Erdschluss
16#2330	Erdschluss
16#2340	Kurzschluss
16#2344	Ausgang überlastet
16#2345	Sensorversorgung überlastet
16#2346	Kurzschluss an VCC
16#2347	Kurzschluss an Erde
16#2350	Drahtbruch
16#2360	Drahtbruch
16#3000	Spannung
16#3100	Netzspannung (Spannung an der Geräteeingangsseite -Spannungsversorgung)
16#3110	Netzüberspannung
16#3120	Netzunterspannung
16#3130	Phasenfehler
16#3134	Phasensequenz
16#3140	Netzfrequenz
16#3141	Netzfrequenz zu hoch
16#3142	Netzfrequenz zu niedrig
16#3200	Spannung innerhalb des Gerätes
16#3210	Überspannung innerhalb des Gerätes
16#3220	Unterspannung innerhalb des Gerätes
16#3230	Ladefehler
16#3300	Ausgangsspannung (Spannung an der Geräteausgangsseite, z. B. Versorgungs- oder Arbeitsspannung für externe Teile)
16#3310	Ausgangsüberspannung
16#3320	Ausgangsunterspannung
16#3400	Versorgungsspannung
16#3410	Sensorversorgung
16#3411	Sensorversorgung Unterspannung
16#3412	Sensorversorgung fehlt
16#3420	Aktorversorgung
16#3421	Aktorversorgung Unterspannung
16#3422	Aktorversorgung fehlt
16#4000	Temperatur

Wert	Beschreibung
Statuscodes mit den Diagnoseklassen Betriebszustand (0xC10A), Warnung (0xC10E), Fehler (0xC10F)	
16#4100	Umgebungstemperatur
16#4110	Umgebungsübertemperatur
16#4120	Umgebungsuntertemperatur
16#4130	Temperatur Ansaugluft
16#4140	Temperatur Abluft
16#4200	Gerätetemperatur
16#4210	Geräteübertemperatur
16#4220	Geräteuntertemperatur
16#4300	Externe Temperatur (z.B. des Antriebs)
16#4310	Externe Übertemperatur (z.B. des Antriebs)
16#4320	Externe Untertemperatur (z.B. des Antriebs)
16#4400	Anschlusstemperatur
16#4410	Anschlussübertemperatur
16#4420	Anschlussuntertemperatur
16#5000	Gerätehardware (nur im Geräteinneren)
16#5010	Bauteilfehler
16#5100	Versorgung
16#5110	Niederspannungsversorgung (für alle internen Spannungen 24V, 5V, 3,3V, 3,0V, 2,5V, 1,8V)
16#5120	Luftversorgung
16#5150	Initiatorversorgung (Fehler in der Versorgung eines externen Sensors, der aus dem Geräteinneren versorgt wurde)
16#5151	Kurzschluss
16#5160	Versorgung der peripheren Geräte (Fehler der Versorgung für externe Geräte, die aus dem Geräteinneren versorgt werden)
16#5200	Steuergerät/Messkreis
16#5210	Messkreise
16#5220	Computerschaltkreise
16#5230	Kommunikation (innerhalb des Gerätes)
16#5300	Bedieneinheit
16#5400	Leistungsteil
16#5410	Ausgangsstufen
16#5420	Chopper (Unterbrecher)
16#5430	Eingangsstufen
16#5440	Schütze
16#5450	Sicherungen
16#5500	Kommunikation mit Erweiterungsmodulen

Wert	Beschreibung
Statuscodes mit den Diagnoseklassen Betriebszustand (0xC10A), Warnung (0xC10E), Fehler (0xC10F)	
16#6000	Gerätesoftware
16#6010	Software-Reset (Watchdog)
16#6100	Interne Software (Firmware)
16#6200	Applikationssoftware
16#6300	Datensatz fehlerhaft
16#6310	Fehlender Parameter
16#6320	Parameterfehler
16#6330	Parameter noch nicht initialisiert
16#6800	Konfiguration
16#7000	Erweiterungsmodul(e) defekt
16#7100	Leistung
16#7200	Messkreis
16#7300	Sensor
16#7400	Computerschaltkreis
16#7500	Kommunikation
16#7600	Datenspeicher
16#7610	RAM
16#7620	EPROM
16#7630	EEPROM
16#7700	Leitungsbruch / Kabelfehler
16#7710	Drahtbruch Kabel zum Sensor
16#8000	Überwachung
16#8100	Kommunikation
16#8110	Prozessdatenüberwachung
16#8120	Hostüberwachung
16#8200	Regelung
16#8210	Hohe Systemabweichung (Soll > Ist, herstellerspezifisch), Abweichung länger als zulässig vorhanden
16#8211	Maximaler Stellwert erreicht
16#8220	Systemabweichung (Soll > Ist, herstellerspezifisch), Abweichung länger als zulässig vorhanden
16#8221	Maximaler Stellwert erreicht
16#8900	Sensoren
16#8910	Gemessener Wert oberhalb des Wertebereiches
16#8920	Gemessener Wert unterhalb des Wertebereiches
16#8A00	Aktor

Wert	Beschreibung
Statuscodes mit den Diagnoseklassen Betriebszustand (0xC10A), Warnung (0xC10E), Fehler (0xC10F)	
16#8B00	Vorbeugende Wartung notwendig (Zustandsüberwachung)
16#9000	Externer Fehler
16#A000	Subbus-Slave
16#A001	Subbus-Slave fehlt
16#A002	Falscher Subbus-Slave vorhanden
16#A003	Kompatibler Ersatz (der erwartete Subbus-Slave wurde durch einen kompatiblen ersetzt)
16#A004	Fehler Anzahl der Subbus-Geräte (zu viele Geräte)
16#A010	Gerätefehler
16#A012	Anwendung auf dem Subbus-Slave nicht bereit
16#A013	Subbus-Geräte-Reset
16#A020	Kommunikationsfehler
16#A021	Subbus-Fehler - Timeout (Break)
16#A022	Mehrfacher Übertragungsfehler auf dem Subbus
16#A023	Subbus E/A-Datenkommunikationsfehler
16#A024	Datenkommunikationsfehler beim Subbus-Management
16#A030	Subbus-Konfigurationsfehler
16#A031	E/A-Datenkonfiguration doppelt abgebildet
16#A040	Allgemeiner Fehler
16#A041	Subbus-Hardwarefehler
16#A042	Subbus-Firmwarefehler
16#A043	Subbus läuft asynchron (im Bezug auf den sercos Buszyklus)
16#B000	Lokalbus (Systembus der modularen Station)
16#B001	Lokalbus-Slave fehlt
16#B002	Falscher Lokalbus angeschlossen
16#B003	Kompatibler Ersatz (der erwartete Lokalbus-Slave wurde durch einen kompatiblen Slave ersetzt)
16#B004	Fehler bei der Anzahl der Lokalbus-Teilnehmer (zur viele Teilnehmer)
16#B010	Teilnehmerfehler
16#B012	Anwendung auf dem Lokalbus-Slave nicht bereit
16#B013	Lokalbus-Teilnehmerreset
16#B020	Kommunikationsfehler
16#B021	Lokalbusfehler - Timeout (Unterbrechung)
16#B022	Mehrfacher Übertragungsfehler auf dem Lokalbus
16#B023	Lokalbus E/A-Datenkommunikationsfehler
16#B024	Datenkommunikationsfehler beim Lokalbus-Management
16#B030	Konfigurationsfehler auf dem Lokalbus

Wert	Beschreibung
Statuscodes mit den Diagnoseklassen Betriebszustand (0xC10A), Warnung (0xC10E), Fehler (0xC10F)	
16#B031	E/A-Datenkonfiguration doppelt abgebildet
16#B040	Allgemeiner Fehler
16#B041	Hardwarefehler auf dem Lokalbus
16#B042	Firmwarefehler auf dem Lokalbus
16#B043	Lokalbus läuft asynchron (im Bezug auf den sercos Buszyklus)
16#F000	Zusätzliche Funktionen

Value	Description
Status codes with diagnosis class "procedure command specific state" (0xC10C)	
16#A100	S-0-1508.x.67 Sub-bus Tunnel Command
16#A101	S-0-1508.x.67 Sub-bus Tunnel Command failed. Unsupported Sub-bus Tunnel PDU version
16#A102	S-0-1508.x.67 Sub-bus Tunnel Command failed. Unsupported Sub-bus SDU Type
16#A103	S-0-1508.x.67 Sub-bus Tunnel Command failed. Unsupported Sub-bus Tunnel SDU version
16#A104	S-0-1508.x.67 Sub-bus Tunnel Command failed. Sub-bus slave index wrong
16#A105	S-0-1508.x.67 Sub-bus Tunnel Command failed. Wrong length of SDU
16#A106	S-0-1508.x.67 Sub-bus Tunnel Command failed. Wrong length of Command SDU
16#A107	S-0-1508.x.67 Sub-bus Tunnel Command failed. Answer PDU too large
16#A108	S-0-1508.x.67 Sub-bus Tunnel Command failed. Buffer overflow
16#BC00	S-0-1500.x.12 Rearrangement of IO resource
16#BC01	Forbidden insertion/replacement - An attempt was made to insert/replace a generic IO function group with an invalid/forbidden slot number
16#BC02	Forbidden insertion/replacement - An attempt was made to insert/replace the function group IO bus coupler (S-0-1500.x.0)
16#BC03	Forbidden insertion/replacement - An attempt was made to insert/replace an undefined generic IO function group
16#BC04	Forbidden insertion/replacement - The list of inserted/replaced function groups (S-0-1500.x.4/11) contains duplicate entries
16#BC05	Forbidden insertion/replacement - The list of inserted/replaced function groups (S-0-1500.x.4/11) is not sorted in an increasing order (slot number, data block number)
16#BC0F	Forbidden insertion/replacement - Manufacturer-specific reason

9.3.4 Sercos III-Master, Verkabelung und Redundanz (am Beispiel eines MLC-Systems)

Um die Fehlertoleranz gegenüber EMV-Störungen und Kabeldefekten zu erhöhen wird empfohlen eine Sercos-Ringtopologie zu verwenden. Dadurch steht eine redundante Sercos-Kommunikation zu allen Sercos-Geräten zur Verfügung. Bei einer Ringunterbrechung werden die Verbindungsdaten, die Sercos-Geräte direkt miteinander austauschen, über den Master kopiert. Die Verbindungsdaten vom Master zu den Sercos-Geräten und zurück sind davon nicht betroffen.

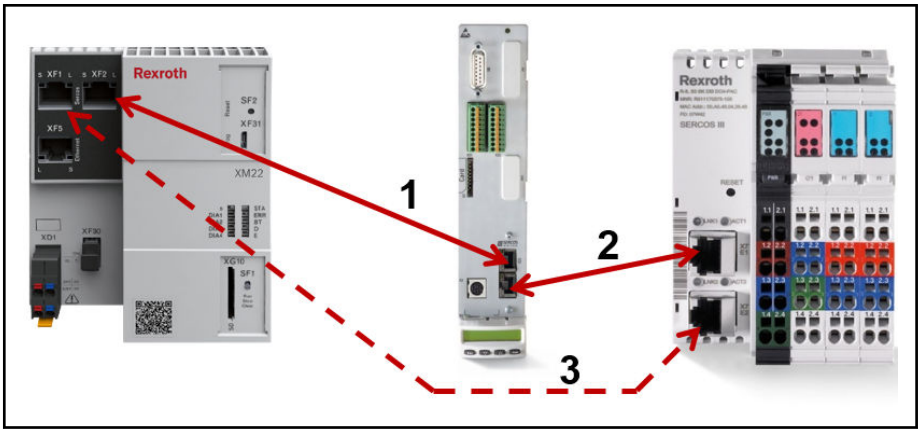


Abb. 9-5: Ringtopologie verdrahten

- **Ringtopologie:** Durch Verdrahten der Kabel 1, 2 und des optionalen Kabels 3 wird eine redundante Ringtopologie hergestellt.
- **Linientopologie:** werden nur die Kabel 1 und 2 installiert, ist keine Redundanz vorhanden.
- **Topologie Doppellinie:** werden nur die Kabel 1 und 3 installiert, ist ebenfalls keine Redundanz vorhanden.

Busdiagnose Ringbruch

Im Grundzustand erwartet der MLC Sercos-Master eine Ringtopologie. Wird diese im Sercos-Phasenhochlauf nicht erkannt oder wird durch Störungen die Ringtopologie im laufenden Betrieb unterbrochen, so wird die Warnung "E00B0179, sercos III: Redundanzfehler" ausgegeben. Im Dialog Sercos-Busdiagnose wird die Ringunterbrechung angezeigt.

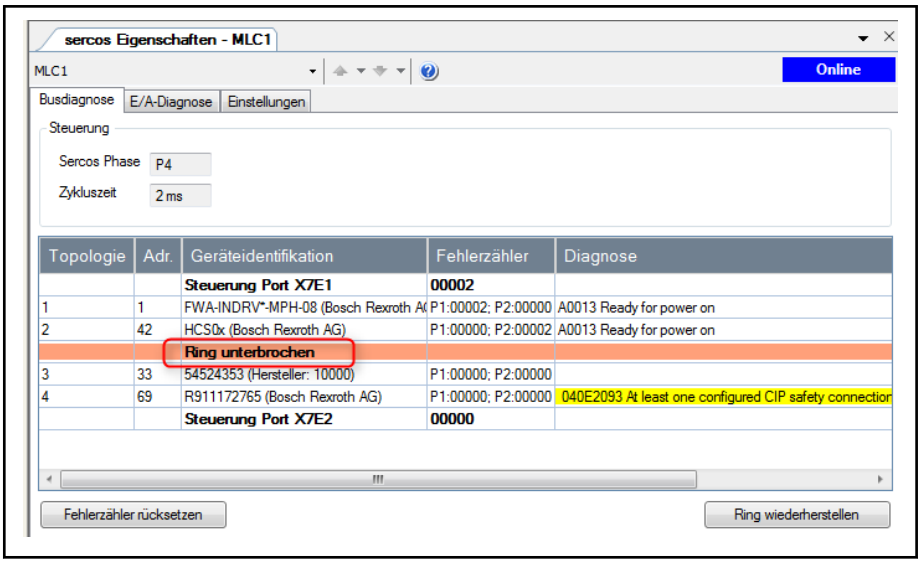


Abb. 9-6: Busdiagnose bei Topologie Doppellinie (Abb. oben: Kabel 3 fehlt)

Der Button <Fehlerzähler rücksetzen> löscht die angezeigten Fehlerzähler aller Geräte.

In der Spalte "Fehlerzähler" wird für Sercos-Geräte der Parameter S-0-1035.0.0 "Fehlerzähler Port 1 und Port 2" angezeigt. Links vom Strichpunkt steht der Fehlerzähler Port2, rechts der Fehlerzähler Port1. In der Zeile **Steuerung Port X7E1** wird vom Parameter C-0-0535, sercos Master, Fehlerzähler für Port1 und Port2, das Low-Word angezeigt, in der Zeile **Steuerung Port X7E2** das High-Word des C-0-0535.

Mit Hilfe des Buttons <Ring wiederherstellen> kann nach Reparatur der Unterbrechung der Ring wieder logisch geschlossen werden.

Wurde die Ringunterbrechung durch ein ausgefallenes Gerät verursacht, so kann der Ring nicht geschlossen werden. Nur mit einer Phasenumschaltung CP0->CP2 wird das ausgefallene Gerät wieder in die Kommunikation mit aufgenommen. Mit der Phasenumschaltung wird der Ring wieder geschlossen.

Soll eine Linien- oder Doppellinientopologie verwendet werden, kann die Warnung im Einstellungen-Dialog abgeschaltet werden. Der Dialog Einstellungen ist nur verfügbar, wenn online geschaltet wurde.

Registerkarte 'Einstellungen'

Der Dialog erlaubt die Vorgabe

- einer erlaubten Anzahl von Telegrammausfällen
- sowie der Sercos-Zykluszeit für das gesamte Sercos-Objekt

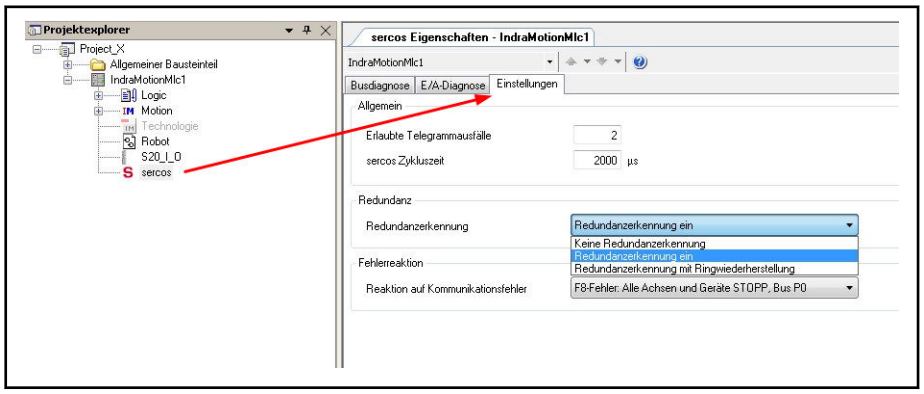


Abb. 9-7: Sercos-Einstellungen

Sercos-Einstellungen zur Redundanz

- **Keine Redundanzerkennung:** Die Warnung "E00B0179, sercos III: Redundanzfehler" wird nicht ausgegeben. Die Statusinformationen werden weiterhin gebildet und Ringwiederherstellung durch den Anwender ist möglich.
- **Redundanzerkennung ein (Default):** Die Warnung "E00B0179, sercos III: Redundanzfehler" wird bei Ringbruch ausgegeben. Ringwiederherstellung durch den Anwender ist möglich.
- **Redundanzerkennung mit Ringwiederherstellung:** Die Warnung "E00B0179, sercos III: Redundanzfehler" wird bei Ringbruch ausgegeben. Es wird automatisch versucht den Ring wieder zu schließen (Ring wiederherstellen auf Seite 258).

Sercos-Einstellungen zur Fehlerreaktion auf Kommunikationsfehler

- **F8-Fehler: Alle Achsen und Geräte STOPP, Bus P0** (Defaulteinstellung. Kompatibel zu Vorgängerversionen und zu Sercos 2). Bei Kommunikationsfehlern (z. B. Ungültige AT-Daten) wird ein F8-Fehler (z. B. F80B0016, Ungültige AT-Daten) ausgegeben und damit der Sercos-Bus mit allen angeschlossenen Geräten gestoppt
- **F5-Fehler: Alle Achsen und Geräte STOPP**. Bei Kommunikationsfehler wird ein F5-Fehler (z. B. F50B017A, Ungültige AT-Daten) ausgegeben. Mit diesem F5-Fehler werden alle Motionachsen und das betroffene Gerät gestoppt. Der Sercos-Bus bleibt im Betriebs- bzw. Parametriermodus. SPS-Geräte am Sercos-Bus sind weiterhin betriebsbereit. Eine Fehlerreaktion auf den Ausfall eines Gerätes ist gegebenfalls im SPS-Programm zu realisieren.

☞ Nach Löschen des F5-Fehlers können die nicht betroffenen Achsen wieder verfahren werden. Insbesondere bei gekoppelten Achsen ist eine Fehlerreaktion auf den Ausfall eines Gerätes im SPS Programm zu realisieren.

- **F2-Fehler: Betroffenes Gerät STOPP**. Bei Kommunikationsfehler wird ein F2-Fehler (z. B. F20B017D, Ungültige AT-Daten) ausgegeben. Mit diesem F2-Fehler wird nur das betroffene Gerät gestoppt. Der Sercos-Bus bleibt im Betriebs- bzw. Parametriermodus. Motionachsen und SPS Geräte am Sercos-Bus sind weiterhin betriebsbereit. Eine Fehlerreaktion auf den Ausfall eines Gerätes ist gegebenfalls im SPS-Programm zu realisieren.

  Fällt eine reale Achse aus, die von einer anderen Achse konsumiert wird, kommt an der konsumierenden Achse auch ein Fehler (ungültige Leitachsdaten).

Ring wiederherstellen

Wurden die Enden der Linien bzw. das Ende der Linie mit dem freien Masterport physikalisch wieder korrekt verbunden muss die Sercos-Topologie noch logisch geschlossen werden, damit bei einer Unterbrechung an einer anderen Stelle die Kommunikation unterbrechungsfrei weiter arbeitet.

Die Ringwiederherstellung kann über den Dialog Sercos-Busdiagnose, Button **Ring wiederherstellen** oder

mit einer steigenden Flanke an

`ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].REBUILD_RING`

aus der SPS heraus gestartet werden.

Der Ring ist erfolgreich wiederhergestellt, wenn

`ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ERROR_RED` zurück auf 0 wechselt.

Die Fehlermeldung "F00B0173, Die Topologie 'Ring' konnte nicht hergestellt werden" wird ausgegeben, wenn die Ringwiederherstellung nicht möglich ist.

Ursache	Lösung:
Die Verkabelung zwischen den im Parameter "C-0-0517, sercos Master, Adressen Ringbruch" gemeldeten Geräten ist nicht vorhanden oder fehlerhaft	Verkabelung prüfen
Ein Port der im "C-0-0517, sercos Master, Adressen Ringbruch" gemeldeten Geräte ist defekt	Gerät tauschen
Die Kommunikationsphase ist kleiner CP2	mindestens in CP2 schalten
Mindestens ein Gerät hat die vorgegebene Topologie nicht übernommen	Diagnose der Geräte prüfen
Neue Ringverzögerung konnte nicht an alle Sercos-Geräte übertragen werden	Diagnose der Geräte prüfen
Die letzten Slaves der Linien haben eine falsche Topologie gemeldet. Es liegt entweder kein oder ein falsches Telegramm am inaktiven Port an	Diagnose der Geräte prüfen

SPS Schnittstelle

- SPS-Eingang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ERROR_PORT1`: Störung an X7E1 der Steuerung.

- SPS-Eingang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ERROR_PORT2`: Störung an X7E2 der Steuerung.
- SPS-Eingang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ERROR_RED`: Redundanz nicht vorhanden.
- SPS-Ausgang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].REBUILD_RING`: mit steigender Flanke Ringwiederherstellung anstoßen.
- SPS-Ausgang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ACTUAL_PHASE_BIT0`: Sercos Phase CP0 ist aktiv.
- SPS-Ausgang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ACTUAL_PHASE_BIT1`: Sercos Phase CP1 ist aktiv.
- SPS-Ausgang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ACTUAL_PHASE_BIT2`: Sercos Phase CP2 ist aktiv.
- SPS-Ausgang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ACTUAL_PHASE_BIT3`: Sercos Phase CP3 ist aktiv.
- SPS-Ausgang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].ACTUAL_PHASE_BIT4`: Sercos Phase CP4 ist aktiv.
- SPS-Ausgang `ControlData[LOCAL_CNTRL].arSercos[ML_LOCAL_SERCOS_IFC].PHASE_SWITCH_ACTIVE`: Sercos Phasenumschaltung läuft.

Beteiligte Parameter

- C-0-0506, sercos Master, sercos III, Konfiguration der Funktionsmodi (Bit 7 und Bit 8)
- C-0-0507, sercos Master, Erlaubte Telegrammausfälle
- C-0-0511, sercos Master, Binäre Eingänge, Sollwerte (Bit 4)
- C-0-0514, sercos Master, Status (Bits 1 bis 3)
- C-0-0517, sercos Master, Adressen Ringbruch
- C-0-0518, sercos Master, Position Ringbruch

9.3.5 Sercos III-Master, Slave hinzufügen

Sercos III-Master, Slave hinzufügen, Allgemeines

Die Slaves befinden sich in der Bibliothek "Peripherie" im Ordner "sercos".

Als Slaves kommen dabei kompakte E/A-Module und Busklemmen mit gegebenenfalls fest dazugehörigen E/A-Kanälen zur Anwendung.

Ziehen Sie aus der Bibliothek die benötigten Slaves auf das Sercos-Objekt.

Slaves lassen sich im Projektexplorer so auch zwischen bereits vorhandene Slaves einfügen.

Alternativ können Sie Slaves im Kontextmenü über **Hinzufügen ▸ ...** des Sercos-Objekts hinzufügen.

Hierbei wird das neue Device als letztes Modul unter das Sercos-Objekt eingefügt.

Es ist auch möglich, durch den Kontextmenü-Befehl Buskonfiguration scannen alle am Sercos-Bus laufenden Geräte zu ermitteln und in das IndraWorks-Projekt zu übernehmen.

Ist ein benötigter Slave nicht standardmäßig in der Bibliothek vorhanden, kann er mittels Import seiner SDDML-Datei über das Hauptmenü **Extras** ▸ **Gerätedatenbank...** in die Bibliothek integriert werden.

Sercos III unterscheidet zwei Arten von Slaves zum Anschluss von I/O-Modulen

1. **Kompakt**:

   Bei einem kompakten Slave ist der Modulaufbau fest vorgegeben.

   Nach dem Einfügen eines Slaves in den Projektexplorer sind bei der kompakten Bauweise die Module unterhalb des Slave-Objektknotens bereits vollständig vorhanden. In der Bibliothek sind die Module nicht sichtbar.

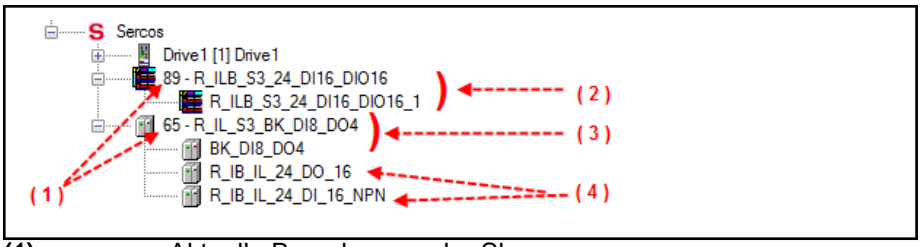
2. **Modular**:

   Der Modulaufbau des Slaves ist variabel.

   Zusätzlich zu einem Festanteil, im Bild unten "BK_DI8_DO4_1" mit 8 digitalen Ein- und 4 digitalen Ausgängen, können weitere Module hinzugefügt werden.

   Die Module können individuell – aber nach Bestückungsvorschrift des Gerätes – angeordnet werden.

   Zum Einfügen von Modulen siehe Sercos III Master, Module zum Slave hinzufügen auf Seite 267.



(1) Aktuelle Busadressen der Slaves
(2) Kompakter Slave mit fest zugeordneten Ein- und Ausgängen
(3) Modularer Slave mit fest zugeordneten Ein- und Ausgängen
(4) Aus- und Eingangsklemmen an den Buskoppler angesteckt

Abb. 9-8: Slaves am Sercos III Master-Objekt

Sercos III erlaubt ferner einen Antrieb als SPS-Gerät hinzu zufügen:

Ein Antrieb als SPS-Gerät hat, im Gegensatz zu einer Achse, keine Motionfunktionalität und keine A-Parameter.

Der Antrieb wird von der SPS kommandiert und ist frei programmierbar.

Für Sercos ist ein IndraDrive-Einzelachsgerät ab MPx18VRS als HCS0x in der Bibliothek der Peripheriegeräte "sercos" zu finden (IndraDrive-Doppelachsgeräte als HMD01).

Eine detailliere Beschreibung zu "Antrieb als SPS-Gerät" ist enthalten in:

Rexroth IndraLogic MLC 15VRS Funktionsbeschreibung

DOK-MLC***-FUNC****V15-APRS-DE-P, R911393304

Die Dokumentation dient der Beschreibung der Wizards, Kontextmenüs, Dialoge, Steuerungsinbetriebnahme, Gerätekonfiguration, Funktionalitäten des Systems MLC.

Übersicht Sercos-Adressen...

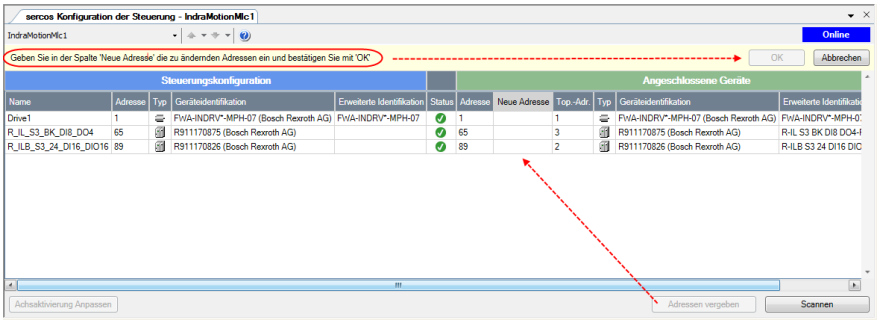


Abb. 9-9: Übersicht Sercos-Adressen

Das Scannen des Sercos-Busses kann nur in der Phase P2 erfolgen, also P4 -> P0 -> P2 -> P4.

Sercos-Slaves aktivieren / deaktivieren

Die an einer Steuerung konfigurierten Sercos-Slaves können im Projektexplorer aktiviert bzw. deaktiviert werden.

Dies erfolgt durch Anwählen / Abwählen der Schaltfläche, die am Icon des Sercos-Slaves erscheint.

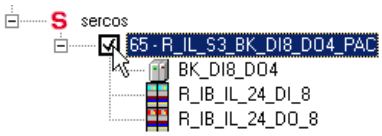


Abb. 9-10: Aktivieren / Deaktivieren von Sercos-Slaves

Ist der Haken gesetzt, ist der Busteilnehmer aktiviert. Er wird in der Diagnose des Busses berücksichtigt.

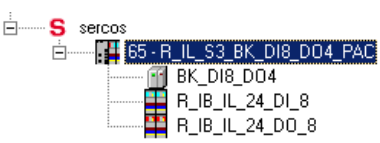


Abb. 9-11: Sercos-Slave als aktivierter Teilnehmern

Ist der Haken nicht gesetzt, ist der Busteilnehmer deaktiviert. Er wird in der Diagnose des Busses nicht berücksichtigt.

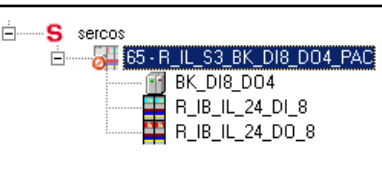


Abb. 9-12: Sercos-Slave als deaktivierter Teilnehmer

Wird ein neuer Sercos-Slave angelegt, ist er per Voreinstellung aktiviert.

Beim Laden der Konfiguration werden alle aktivierten Sercos-Slave berücksichtigt. Deaktivierte Sercos-Slave werden ignoriert.

Ein aktivierter jedoch nicht vorhandener Busteilnehmer blockiert das Schalten des Busses in den Zustand P4, die Gerätekonfiguration wird abgeglichen.

Siehe Sercos III-Master, Übersicht Busadressen auf Seite 247.

Ein deaktivierter nicht vorhandener Busteilnehmer erzeugt in der Diagnose keine Fehlermeldung.

Siehe auch Sercos-Slaves / Sercos-Module deaktivieren mittels UserDefs.cfg auf Seite 262

Sercos-Slaves / Sercos-Module deaktivieren mittels UserDefs.cfg

Durch die Funktion können Sercos-Slaves bzw. Sercos-Module, die in IndraWorks konfiguriert sind, durch Definition in einer auf der Steuerungs-FlashCard abliegenden Datei deaktiviert werden.

Dadurch werden Diagnose-Meldungen für Sercos-Slaves bzw. Sercos-Module verhindert, die zwar konfiguriert, aber nicht vorhanden sind.

Aktivierung der Funktion

Die Funktion wird durch das Vorhandensein einer gültigen Datei "UserDefs.cfg" auf der Steuerungs-FlashCard in folgenden Partitionen aktiviert:

Bei XM- und VPx-Steuerungen: **\USER**

Optional können durch Einträge in der "cfg-Datei" eines Systems ein anderer Pfad und Dateiname definiert werden.

Beispiel:

```
[IOSpecialDefs]
IODeactFilePath=MyFilePath
IODeactFileName=MyFileName
```

Die Datei "UserDefs.cfg" kann Einträge für weitere Busse enthalten:

Inline-Module, Sercos, Profinet IO, Profibus DP.

Zeitpunkt des Deaktivierens

Das Auswerten der Datei "UserDefs.cfg" und das Deaktivieren der dort eingetragenen Sercos-Slaves / Sercos-Module erfolgt zum Zeitpunkt und im Kontext des internen Events "PrepareUpdateConfiguration".

Das Event tritt bei folgenden Aktionen ein:

- Anlauf nach Power-ON
- Reset (warm)
- Reset (kalt)
- Programm laden nach Reset (Ursprung)

Struktur der Datei "UserDefs.cfg"

Die Datei ist strukturiert in ein- oder mehrfach vorhandene Sektion(en) und den darin enthaltenen Einträgen.

- **Sektionen**: Die Bezeichnung einer Sektion ist in '**[**' und '**]**' geklammert. Eine Sektion ist begrenzt durch den Beginn der nächsten Sektion oder durch das Ende der Datei.

  Die Bezeichnung besteht aus dem Symbol **SercosIO.Master.** und dem konstanten Index **0** (kompatibel zum Deaktivieren eines Profibus-Slaves).

  Beispiel:

  ```
  [SercosIO.Master.0]
  ```

- **Sonderfunktion**: Die Definition des Zeichens '**X**' für den Index anstelle einer Ziffer führt dazu, dass diese Sektion in der Auswertung ignoriert wird.

  Es ist möglich, für den Master-Index **0** mehrere Sektionen zu definieren. Die darin enthaltenen Einträge werden alle dem Sercos-Master zugeordnet.

  In Kombination mit der oben beschriebenen "X-Funktion" lassen sich damit unterschiedliche Gruppen von Slaves in einfacher Weise behandeln.

- **Slave-Eintrag**: Ein Eintrag zum Deaktivieren eines Sercos-Slaves besteht aus dem Schlüsselwort mit Zuweisungszeichen `SlaveDeactAddr=` und der 1- bis 2-stelligen Angabe der Slave-Adresse in dezimaler Form.

  Beispiel:

  ```
  SlaveDeactAddr = 28
  ```

  Der zulässige Wertebereich ist **1…255**.

  Der Wert **0** ist ebenfalls zulässig, aber unwirksam.

- **Modul-Eintrag**: Zum Deaktivieren wahlfreier Module unterhalb eines Slaves werden

  das Schlüsselwort `SlaveAddr=` und die Sercos-Adresse, sowie

  das Schlüsselwort `DeactModule=` und durch Komma getrennt, die zu deaktivierenden Modulnummern von **1** bis **n** angegeben.

  Die Modulnummer **0** ist die Busanschaltung des Slaves selbst und darf hier nicht angegeben werden (es ist auch nicht sinnvoll, nicht entfernbare Module zu deaktivieren).

  Beispiel:

  ```
  SlaveAddr = 3, DeactModule=3,4,7
  ```

- **Letzte Modul-Einträge**: Zum Deaktivieren der **n** letzten konfigurierten Module unterhalb eines Slaves werden

  das Schlüsselwort `SlaveAddr=` und die Slave-Adresse, sowie

  das Schlüsselwort `DeactLastModules=` und die zu deaktivierende Anzahl von Modulen angegeben.

  Damit werden die letzten (=untersten) Module im Konfigurationsbaum deaktiviert.

  Beispiel:

  ```
  SlaveAddr = 3, DeactLastModules=2
  ```

- **Kommentare**: Als Kommentarzeichen ist das Semikolon (';') definiert. Ab der Position des Semikolons bis zum Ende der Zeile wird der Text nicht ausgewertet.

Größe der Datei "UserDefs.cfg"

Entsprechend der gegebenen Größe der Datei wird zu deren Auswertung Speicher allokiert. Die zulässige Größe ist daher praktisch nicht begrenzt.

Beispiel der Datei "UserDefs.cfg"

```
;------------ Master.0 -----------
[SercosIO.Master.0]
SlaveDeactAddr= 20
SlaveDeactAddr= 32
SlaveDeactAddr= 35
SlaveDeactAddr= 56
;--------------------------------
SlaveAddr= 40, DeactModule= 2, 4
SlaveAddr= 55, DeactLastModules= 3
;--------------------------------
```

Inbetriebnahme von Sercos-IO-Geräten/Modulen am Sercos-Bus

Eine Umparametrierung von Sercos-IO-Geräten/Modulen ist teilweise nur in der Sercos-Phase P2 möglich und bedingt ein erneutes Übersetzen und Laden der SPS.

Änderung bzw. Anpassung an Sercos-IO-Geräten/ Modulen setzt nachstehende Arbeitsschritte (Empfehlung) voraus:

- Sercos-Bus in Phase P2 schalten (Achtung, geht immer nur über P0!)
- mit der Steuerung Offline gehen
- mit der SPS ausloggen
- Änderung/Anpassung am Sercos-IO-Gerät/Modul vornehmen
- mit der Steuerung Online gehen (ggf. Konfigurationsabgleich durchführen)
- mit der SPS übersetzen & einloggen
- Sercos in Phase P4 schalten

Sercos III Master, Sercos III-Slave konfigurieren

Klicken Sie zum Öffnen des Editorfensters im Projektexplorer doppelt auf das Sercos III-Objekt.

In den Registerkarten des generischen Geräteeditors werden Sie über die Konfiguration des gesamten Slaves informiert und können sie gegebenenfalls modifizieren.



Die Adresse eines Sercos III-Slaves können Sie über den Kontext-Menüpunkt des Sercos III-Knotens **Sercos Konfiguration** ändern (Übersicht Sercos-Adressen... auf Seite 261).

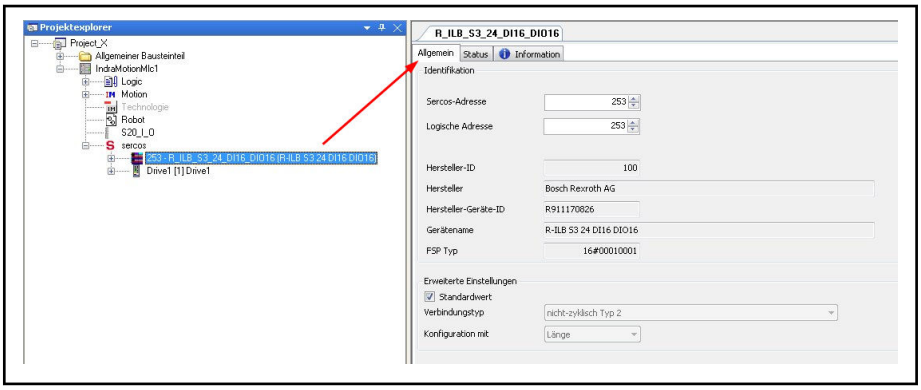


Abb. 9-13: Sercos III-Slave-Objekt konfigurieren

Sercos III Master, Slave hinzufügen

- Sercos-Slaves aktivieren / deaktivieren auf Seite 261
- Sercos-Slaves / Sercos-Module deaktivieren mittels UserDefs.cfg auf Seite 262
- Registerkarte 'Allgemein' auf Seite 265
- Registerkarte 'sercos Konfiguration' auf Seite 266, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.
- Registerkarte 'Status' auf Seite 266
- Registerkarte 'Information' auf Seite 266

Registerkarte 'Allgemein'

Die Registerkarte "Allgemein" enthält Einstellungen und Werte, die je nach Gerät von der Gerätebeschreibung abhängig sind.



Abb. 9-14: Sercos III-Slave: Registerkarte "Allgemein"

Identifikation

Im Bereich "Identifikation" finden Sie spezifische Informationen zum Sercos III-Slave, der den Slave eindeutig identifiziert.

Sercos-Adresse:	Busspezifische Adresse des Sercos III-Slaves
Logische Adresse:	Logische Adresse des Sercos III-Slaves
Hersteller-ID	Herstellerkennnummer des Sercos III-Slave
Hersteller	Herstellername des Sercos III-Slave, z. B. Rexroth
Hersteller-Geräte-ID	Identifikationsnummer, die für den Sercos III-Slave vom Hersteller vergeben wurde
Gerätename:	Bezeichnung des Sercos III-Slave vom Hersteller
FSP-Typ:	Diese Nummer definiert gerätespezifische Funktionen des Sercos III-Slaves, z. B. Eigenschaften des Gerätes

Erweiterte Einstellungen

Im Bereich "Erweiterte Einstellungen" befinden sich Einstellmöglichkeiten, die die Konfiguration oder das Verhalten des Slaves beeinflussen.

Es handelt sich in diesem Bereich um Einstellungen, die nur von einem Sercos-Experten vorgenommen werden sollten.

Wenn der Haken (Standardwert) entfernt wird, können die Einstellungen geändert werden.

- **Verbindungstyp:** Hier kann der Verbindungstyp eingestellt werden. In Abhängigkeit der Möglichkeiten des jeweiligen Slaves stehen folgende Verbindungstypen zur Auswahl:
  - **nicht zyklisch Typ 2**

    Die Verbindung erfolgt nicht zyklisch und ohne Zeitüberwachung. Dies ist die Standardeinstellung
  - **nicht zyklisch Typ 1 (mit Zeitüberwachung)**

    Die Verbindung erfolgt nicht zyklisch und mit Zeitüberwachung

  - **taktsynchron**

    Die Verbindung erfolgt synchron zum Takt des Masters

- **Konfiguration mit:** Hier kann ausgewählt werden, ob die Sercos-Verbindung eines Slaves über **Länge** erfolgt oder über **IDN**.

  Diese Auswahl existiert nur bei Teilnehmern, die das IO-Profil und variable Verbindungskonfiguration unterstützen.

Registerkarte 'sercos Konfiguration'

Diese Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn in **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde.



Die Schaltfläche <Parameter schreiben> erscheint nur im Onlinemode.

Registerkarte 'Status'

Die Registerkarte "Status" zeigt Status-Informationen (z.B. "Läuft" (Bus aktiv), und "n/a" (keine Information vorhanden)) und spezifische Diagnosemeldungen aus dem jeweiligen Gerät, auch zur verwendeten Karte und dem internen Bussystem.



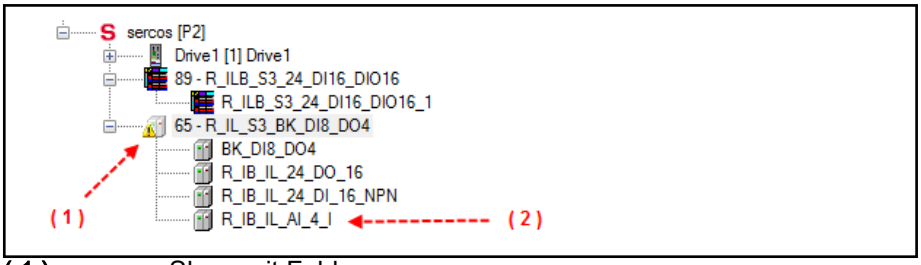
Änderungen im Bus sind nur Offline

Voraussetzung für das Online-Gehen des Sercos III-Busses sind:

1. Konfiguration der Sercos-Teilnehmer in Projekt und Steuerung stimmt nach Type, Reihenfolge und Sercos-Adresse überein.
2. Alle Module laufen fehlerfrei.

   Dies kann für den konkreten Slave im Fenster "Status" durch Betätigung von "Device identification" und "Extended device identification" ermittelt werden.
3. Sercos-Zustand: P4

Fehlerhafte Slaves (vom ersten beginnend) werden im Projektbaum mit einem gelben Warndreieck und einem Fehler-Tooltip, versehen.



(1) Slave mit Fehler
(2) Klemme nur im Projekt, nicht an der Steuerung

Abb. 9-15: Sercos-Bus, Fehlermeldung im Projektnavigator

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

9.3.6 Sercos III Master, Module zum Slave hinzufügen

Sercos III Master, Module zum Slave hinzufügen, Allgemeines

Die zum jeweiligen Slave (Busklemme) passenden Module (Klemmen) befinden sich in der Bibliothek "Peripheriegeräte", im Ordner "Sercos III" unterhalb des jeweiligen Slaves:



I/O-Module lassen sich nur in modular aufgebaute Slaves einfügen (Sercos III unterscheidet zwei Arten von Slaves zum Anschluss von I/O-Modulen auf Seite 260).

Ziehen Sie aus der Bibliothek die benötigten Module in das Slave-Objekt.

Neue Module lassen sich im Projektexplorer auch zwischen bereits vorhandene Module einfügen.

Alternativ können Sie Module im Kontextmenü über **Hinzufügen** ▸ **Modul** ▸ **...** des Slaves hinzufügen.

Hierbei wird das neue Modul als letztes Modul unter den Slave eingefügt.

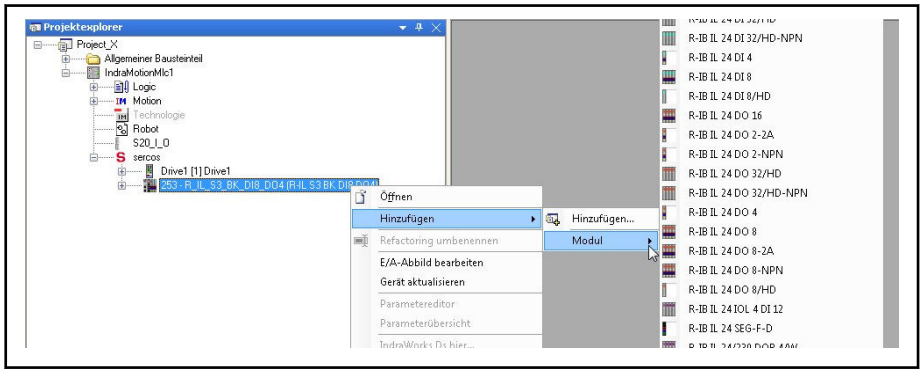


Abb. 9-16: Sercos III, Module zum Slave hinzufügen

Klicken Sie zum Öffnen des Editorfensters im Projektexplorer doppelt auf das gewünschte Modul.

In den Dialogen werden Sie über die Konfiguration des Moduls informiert und können sie gegebenenfalls modifizieren.

Die Einstellungen in allen Registern des Dialoges werden von der Gerätebeschreibungsdatei des Moduls vorgegeben. In dieser Datei ist definiert, mit welchem Wert die Einstellung vorbelegt ist und ob sie editierbar ist.

Sercos III-Master, Module zum Slave hinzufügen

- Registerkarte 'Sercos Modul' auf Seite 268
- Registerkarte 'Funktionsgruppen' auf Seite 269
- Sonder-Register Registerkarte 'Benutzerdefinierte Parameter' auf Seite 271,
- Registerkarte 'Sercos Module Konfiguration' auf Seite 273, dieses Fenster dient Service-Zwecken und ist nur sichtbar, wenn **Extras** ▸ **Optionen** ▸ **IndraLogic 2G** ▸ **Geräteeditor** die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde
- Registerkarte 'E/A-Abbild' auf Seite 274
- Registerkarte 'Information' auf Seite 275

Sercos-Module aktivieren / deaktivieren

Die an einem Sercos-Slave konfigurierten Sercos-Module können im Projektexplorer aktiviert bzw. deaktiviert werden.

Dies erfolgt durch Anwählen / Abwählen der Schaltfläche, die am Icon des Sercos-Moduls erscheint.

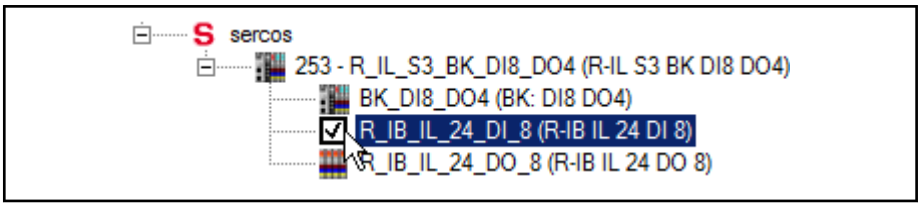


Abb. 9-17: Aktivieren / Deaktivieren von Sercos-Modulen

Ist der Haken gesetzt, ist das Modul aktiviert. Es wird in der Diagnose des Busses berücksichtigt.

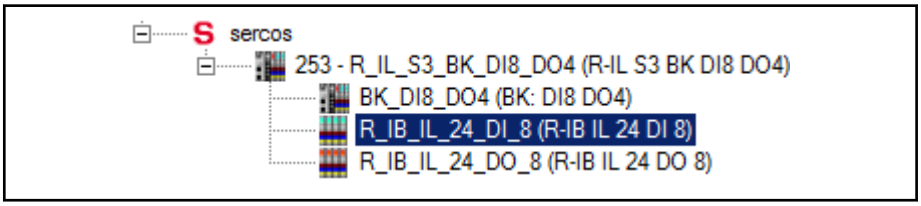


Abb. 9-18: Sercos-Modul als aktivierter Teilnehmer

Ist der Haken nicht gesetzt, ist das Modul deaktiviert. Es wird in der Diagnose des Busses nicht berücksichtigt.

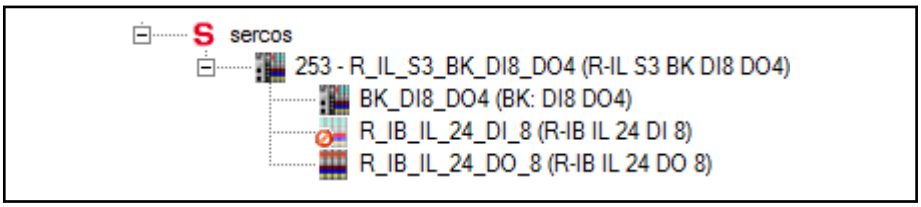


Abb. 9-19: Sercos-Modul als deaktivierter Teilnehmer

Wird ein neues Sercos-Modul angelegt, ist es per Voreinstellung aktiviert.

Beim Laden der Konfiguration werden alle aktivierten Sercos-Module berücksichtigt. Deaktivierte Sercos-Module werden ignoriert.

Ein aktiviertes jedoch nicht vorhandenes Modul blockiert das Schalten des Busses in den Zustand P4, die Gerätekonfiguration wird abgeglichen.

Siehe Sercos III-Master, Übersicht Busadressen auf Seite 247.

Ein deaktiviertes nicht vorhandenes Modul erzeugt in der Diagnose keine Fehlermeldung.

Siehe auch Sercos-Slaves / Sercos-Module deaktivieren mittels UserDefs.cfg auf Seite 262

Registerkarte 'Sercos Modul'

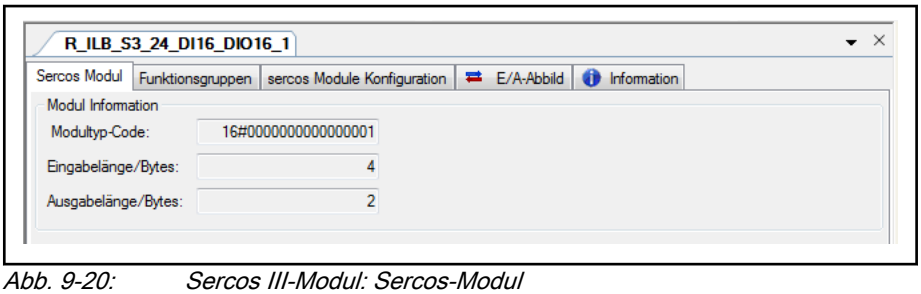


Abb. 9-20: Sercos III-Modul: Sercos-Modul

Modulinformation

Modultyp-Code:	Für jedes Modul existiert eine eindeutige Modultyp-Kennzeichnung des jeweiligen Herstellers. Je nach Hersteller, kann diese Nummer eine Hexadezimalzahl oder eine Auftragsnummer sein
Eingangslänge/Bytes:	Gibt die Eingangslänge des Moduls in Byte an
Ausgangslänge/Bytes:	Gibt die Ausgangslänge des Moduls in Byte an

Registerkarte 'Funktionsgruppen'

Die Registerkarte "Funktionsgruppen" stellt Information über die Ein- und Ausgänge der Funktionsgruppe bereit. Für alle Einstellungen im vorliegenden Dialog gilt, das sie nicht editierbar sind.

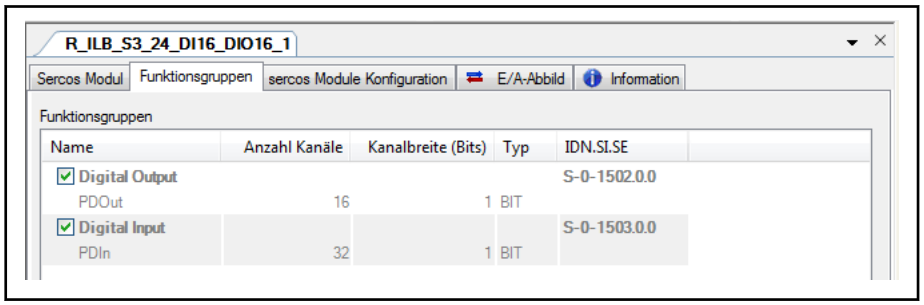


Abb. 9-21: Sercos III-Modul: Funktionsgruppen

Name:	Name des Kanals, nicht editierbar
Kanalanzahl:	Anzahl der unterstützten Kanäle für das Modul.
Kanalbreite (Bits):	Bit-Größe der einzelnen Kanäle, nicht editierbar
Typ:	Anzahl der Strukturelemente (SE) der IO-Funktionsgruppen. Abgängig von den IO-Funktionen ab
IDN.SI.SE:	
• IDN	Identifikationsnummer, z. B. 15xx
• SI (Structure Instance)	Die SI-Nummer ist identisch mit der Einschubnummer der Module Modul 1 hat die SI-Nummer 1 Bei fixen Modulen hat Modul 1 die SI-Nummer 0
• SE (Structure Element)	Anzahl der Strukturelemente (SE) der IO-Funktionsgruppen. Abgängig von den IO-Funktionen

Registerkarte 'Allgemeine Ein- und Ausgänge'

Die Registerkarte erscheint anstelle des Reiters "Funktionsgruppen", wenn das Modul ein Antrieb, ein Geber oder ein Versorger ist. Er dient der Konfiguration von Ein- und Ausgangsverbindungen des Moduls.

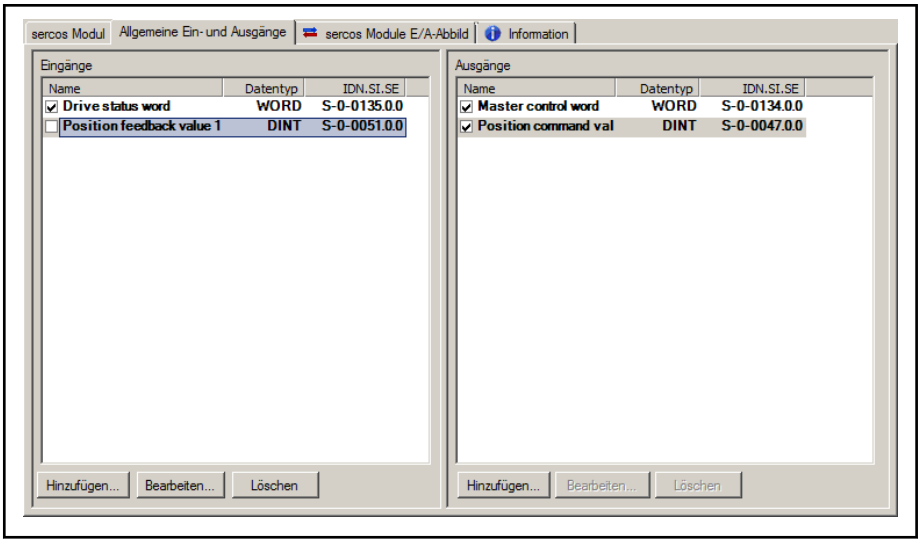


Abb. 9-22: Sercos III-Modul: Allgemeine Ein- und Ausgänge

Die aktuell definierten Ein- und Ausgangsverbindungen können mit einem Mausklick auf die vorangestellte Box aktiviert oder deaktiviert werden. Wenn eine Verbindung aktiviert oder deaktiviert wird, werden die entsprechenden Kanäle im E/A-Abbild eingefügt bzw. entfernt.

Betätigen Sie die Schaltfläche "Hinzufügen", um einen neuen Eingang oder Ausgang hinzuzufügen, bzw. die Schaltfläche "Bearbeiten" um die gerade ausgewählte Verbindung zu bearbeiten. In beiden Fällen öffnet sich der Dialog "Eingabekonfiguration" bzw. "Ausgabekonfiguration".

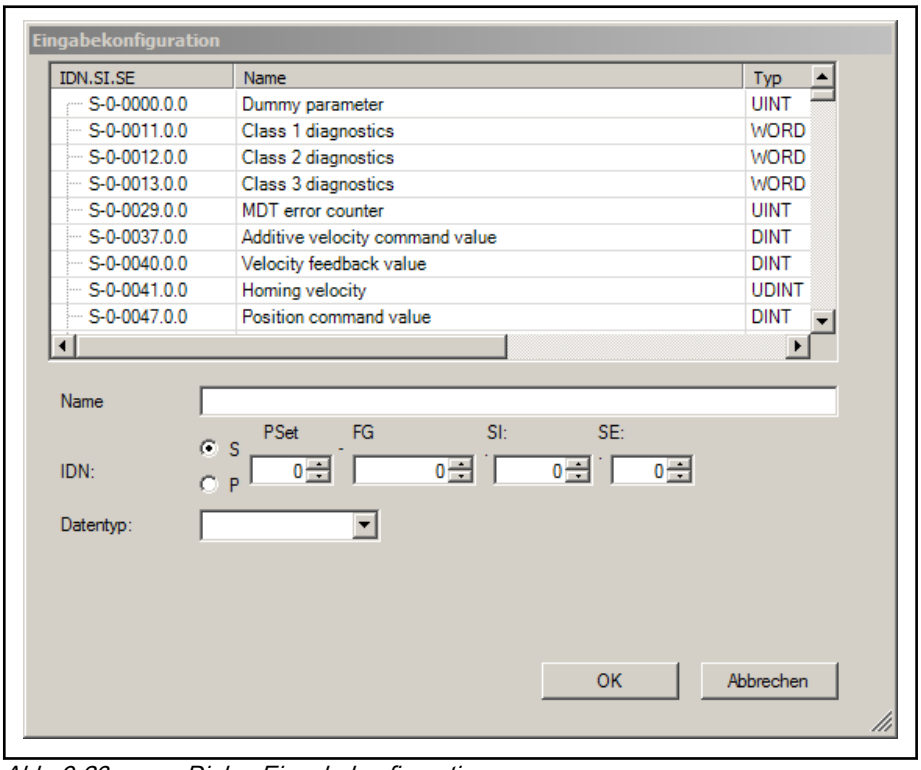


Abb. 9-23: Dialog Eingabekonfiguration

Dieser Dialog zeigt eine Liste der Ein- und Ausgangsobjekte, die in der Gerätebeschreibung des Moduls definiert sind. Sie können neue Ein- oder Ausgänge hinzufügen, indem sie einen oder mehrere Einträge aus der Liste aus-

wählen und mit "OK" bestätigen. Außerdem ist auch eine freie Eingabe möglich.

Um eine Verbindung aus der Liste der Eingänge- bzw. Ausgänge zu entfernen, wählen Sie den Eintrag aus und betätigen sie die Schaltfläche "Löschen".

Registerkarte 'Benutzerdefinierte Parameter'

Diese (Sonder-) Registerkarte wird zur Voreinstellung von Parametern des Moduls durch den Anwender genutzt und wird nur für Module angezeigt, bei denen diese Voreinstellung erforderlich oder möglich ist.

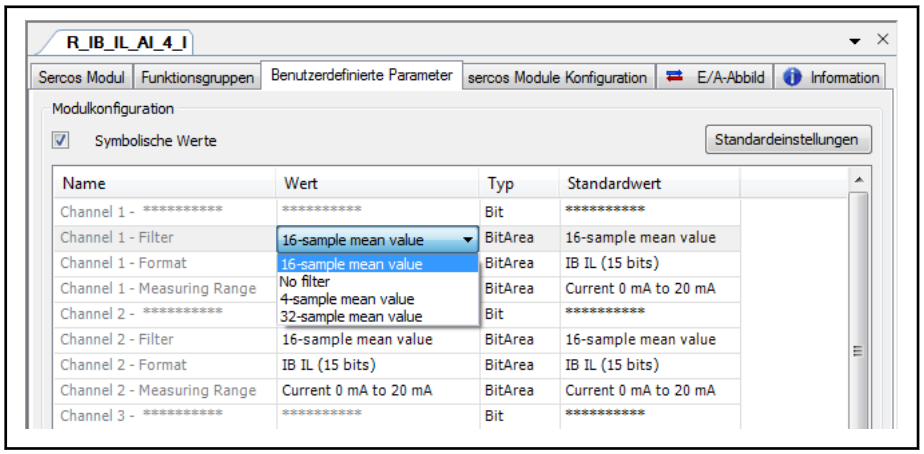


Abb. 9-24: Sercos III-Modul: Benutzerdefinierte Parameter

Modulkonfiguration

Im Beispiel handelt es sich um ein Modul mit vier Analogeingängen.

Für Analogeingang AI4I werden Filter, Format und Messbereich ausgewählt, in dem die Eingangsinformation zu erwarten ist.

Für jeden "Wert" wird dazu ein Auswahlfenster, welches auf einem Aufzählungsdatentyp basiert , angeboten.

Symbolic Values: entscheidet, ob der Wert als Klartext oder als Ganzzahl (16) angezeigt wird.

Standard: setzt alle Parameter zurück auf deren Initialwert.

Registerkarte 'Oversampling'

Die Registerkarte "Oversampling" wird zur Konfiguration der Oversampling-Funktionalität genutzt. Er ist nur sichtbar bei Modulen, die diese Funktionalität unterstützen.

Mit der Oversampling-Funktion können besondere Eingangskanäle schneller eingelesen werden, als es die Buszykluszeit erlauben würde. Es können z.B. analoge Eingänge in 125µs gelesen werden, obwohl die Zykluszeit des Busses nur 1ms beträgt.

Es sind bei dieser Einstellung 8 Datenkanäle im IO-Mapping verfügbar. Die Signale werden zusätzlich zu den normalen Signalen übertragen. Es kann somit gleichzeitig der normale Eingang als auch die overgesampelten Signale verwendet werden.

Die Oversampling-Konfiguration ist in zwei Bereiche gegliedert. Im oberen Bereich sind die verfügbaren Oversampling-Einheiten des Moduls aufgelistet. Diese können dort konfiguriert werden:

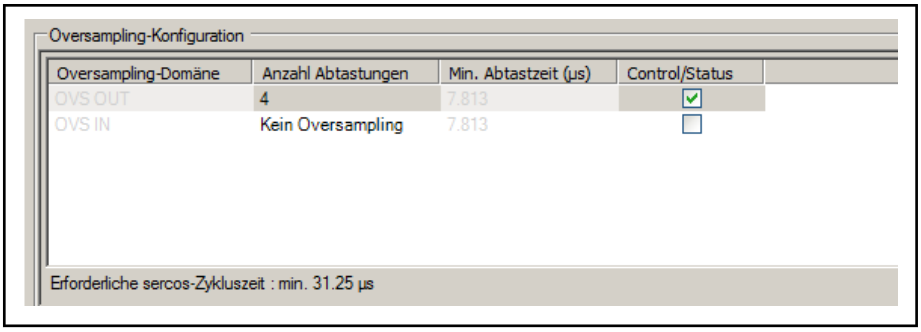


Abb. 9-25: Konfiguration der Oversampling-Domänen

- **Oversampling-Domäne:** Name der Oversampling-Einheit. Jede Einheit kann separat konfiguriert werden.
- **Anzahl der Abtastungen:** Hier können Sie die Anzahl der Abtastungen innerhalb eines Buszyklus einstellen. Die einzustellenden Werte hängen von dem jeweiligen Modul ab. Es kann Module geben, die nur bestimmte feste Verhältnisse (z.B. 1, 2, 4, 8 usw.) erlauben.
- **Min. Abtastzeit (µs):** Dies ist das kleinstmögliche Abtastintervall, das von der Oversampling-Domäne unterstützt wird.
- **Control/Status:** Ist das Häkchen gesetzt, wird das Control- und Statuswort mit übertragen. Die entsprechenden E/A-Kanäle werden im Dialog "E/A-Abbild" hinzugefügt (siehe ① in Abb. 9-27 "E/A-Abbild mit Kanälen für Oversampling" auf Seite 273).
- **Erforderliche Sercos-Zykluszeit:** Zeigt die Sercos-Zykluszeit an, die mindestens erforderlich ist, damit die minimale Abtastzeit nicht unterschritten wird.

  Beispiel:

  Wenn das Gerät eine minimale Abtastzeit von 7,81µs hat und die Anzahl der Abtastungen auf 16 eingestellt ist, dann muss die Sercos-Zykluszeit mindestens 125µs betragen.

Wird im oberen Bereich eine Oversampling-Domäne ausgewählt, so werden im unteren Bereich die Kanäle dieser Oversampling-Domäne angezeigt.

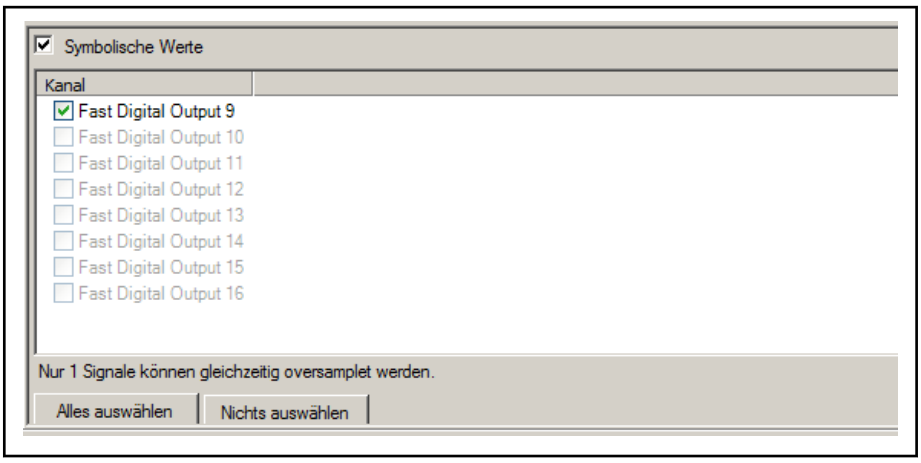


Abb. 9-26: Auswahl der abzutastenden Kanäle

Sie können Oversampling für einen Kanal aktivieren, indem Sie das Häkchen setzen. Abhängig von der eingestellten Anzahl der Abtastungen werden entsprechend viele Kanäle im E/A-Abbild hinzugefügt (siehe ② in nachstehendem Bild).

Das ursprüngliche normale Signal bleibt weiterhin bestehen und wird im Kanalnamen mit "oversampled" gekennzeichnet (siehe ③ in nachstehendem Bild).

Es ist vom jeweiligen Modul abhängig, wie viele Kanäle gleichzeitig abgetastet werden können.

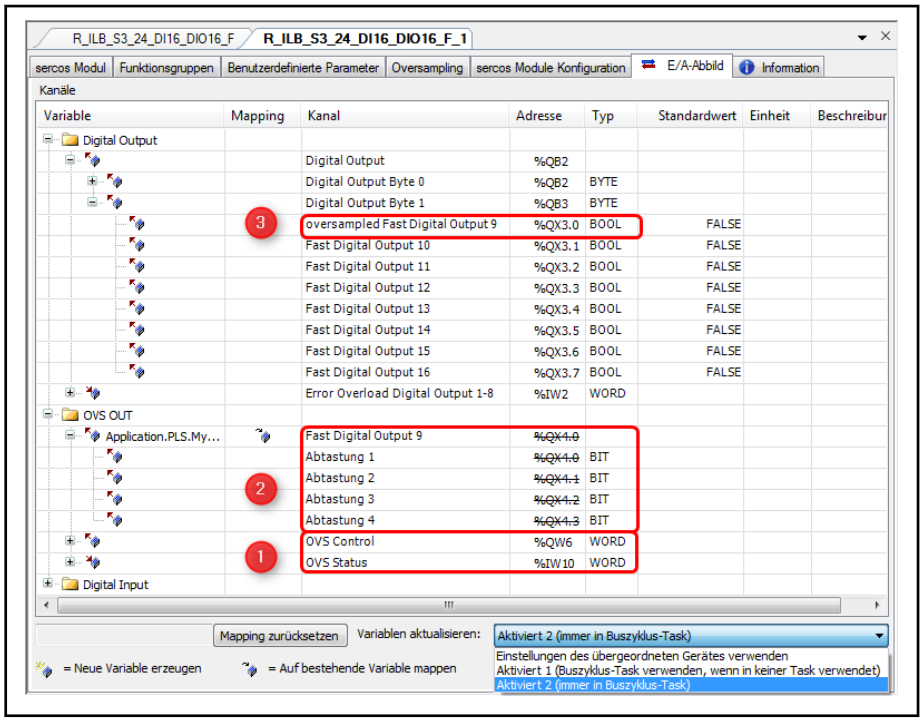


Abb. 9-27: E/A-Abbild mit Kanälen für Oversampling

Registerkarte 'Sercos Module Konfiguration'

Die Registerkarte dient Service-Zwecken und ist nur sichtbar, wenn die Option "Generische Konfigurationseditoren anzeigen" aktiviert wurde. Aufruf über das Indraworks Menü **Extras ▸ Optionen ▸ IndraLogic 2G ▸ Geräteeditor**.

☞ Eventuelle Änderungen in den editierbaren Parametern stimmen Sie bitte mit dem Service ab.

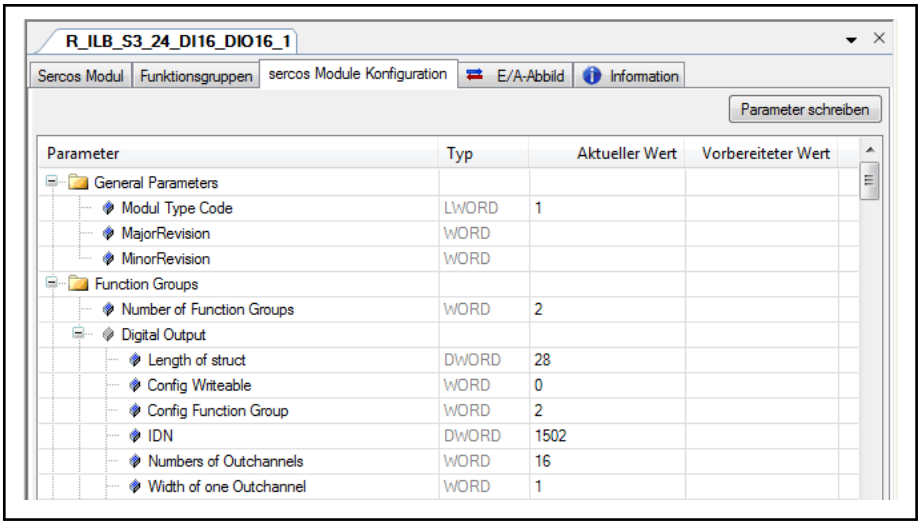


Abb. 9-28: Sercos III-Modul: Sercos Module Konfiguration

Die Schaltfläche <Parameter schreiben> erscheint nur im Online-mode.

Registerkarte 'E/A-Abbild'

Die Registerkarte dient der Zuordnung der Sercos III-Modul-Ein- und Ausgänge zu Variablen, die als lokale oder globale Variablen in den einzelnen POUs verwendet werden können.

Im Online-Modus wird der aktuelle Wert der Variablen angezeigt.

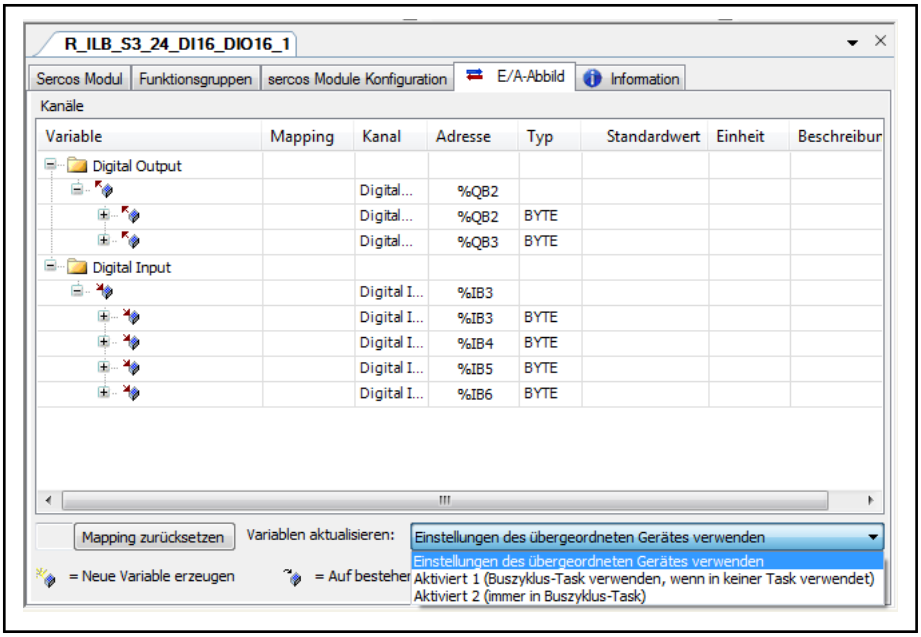


Abb. 9-29: Sercos III-Modul: E/A-Abbild

Mapping zurücksetzen: Über diese Schaltfläche können die Mapping-Einstellungen auf die durch die Gerätebeschreibungsdatei definierten Standardwerte zurückgesetzt werden.

Variablen aktualisieren: Festlegung für das Modul , ob die E/A-Variablen in der Buszyklus-Task aktualisiert werden.

Der Default-Wert ist in der Gerätebeschreibung definiert.

- **Einstellungen des übergeordneten Geräts verwenden:** Aktualisierung gemäß den Einstellungen des übergeordneten Geräts.
- **Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet):** Die E/A-Variablen werden in der Buszyklus-Task aktualisiert, falls sie in keiner anderen Task verwendet werden
- **Aktiviert 2 (immer im Buszyklus-Task):** Alle Variablen werden in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

Die Zuordnung ist unter Mapping der Feldbus-Ein- und Ausgänge auf Seite 19 beschrieben.

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

10 Modbus RTU

10.1 Modbus-Buszyklus-Task

Allgemeine Informationen

Grundsätzlich werden für jede IEC-Task die verwendeten Eingangsdaten am Beginn jeder Task gelesen (1) und die geschriebenen Ausgangsdaten am Ende der Task (3) zum E/A-Treiber übertragen. Die Implementierung im E/A-Treiber ist entscheidend für den weiteren Transport der E/A-Daten. Sie ist verantwortlich dafür, in welchem Zeitraster und zu welchem Zeitpunkt die tatsächliche Übertragung auf das jeweilige Bussystem stattfindet.

Die Buszyklustask der SPS kann für alle Feldbusse global in den SPS-Einstellungen festgelegt werden. Bei einigen Feldbussen können Sie diese jedoch unabhängig von der globalen Einstellung verändern. Als Buszyklustask wird automatisch die Task mit der kürzesten Zykluszeit verwendet (Einstellung: unspezifiziert in den SPS-Einstellungen). In dieser Task werden normalerweise die Nachrichten auf den Bus gesendet.

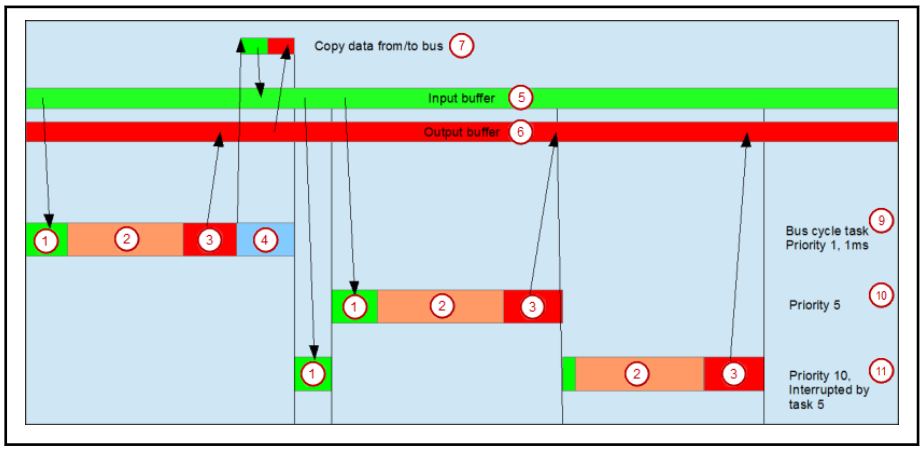


Abb. 10-1: Modbus - Buszyklus-Task

1	Lesen der Eingänge vom Eingangspuffer
2	IEC-Task
3	Schreiben der Ausgänge in den Ausgangspuffer
4	Buszyklus
5	Eingangspuffer
6	Ausgangspuffer
7	Daten vom/zum Bus kopieren
9	Buszyklustask, Priorität 1, 1ms
10	Buszyklustask, Priorität 5
11	Buszyklustask, Priorität 10, unterbrochen durch Task 5

Taskverwendung

In der Registerkarte Taskaufstellung steht Ihnen eine Übersicht der verwendeten E/A-Kanäle, der eingestellten Buszyklustask und der Verwendung der Kanäle zur Verfügung

⚠ WARNUNG

Wenn ein Ausgang in verschiedenen Tasks geschrieben wird, so ist der Zustand undefiniert, da dieser jeweils überschrieben werden kann.

Bei Verwendung von gleichen Eingängen in verschiedenen Tasks kann es passieren, dass sich der Eingang während der Abarbeitung einer Task verändert. Dies geschieht, wenn die Task durch eine Task mit höherer Priorität unterbrochen und damit das Prozessabbild erneut eingelesen wird. Abhilfe: Am Beginn der IEC-Task die Eingangsvariablen in Variablen kopieren und dann im weiteren Code nur noch mit den lokalen Variablen arbeiten.

Fazit: Eine Verwendung von gleichen Ein- und Ausgängen in mehreren Tasks ist nicht sinnvoll und kann in manchen Fällen zu unvorhergesehenen Zuständen führen.

Siehe auch:

- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278
- Weiterführende Dokumentation zum IndraLogic Programmiersystem:

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

  Kapitel:

  – Registerkarte 'Taskaufstellung'
  – Registerkarte 'SPS-Einstellungen'

10.2 Modbus-Konfigurator

Ein Modbus-Netzwerk besteht aus einem Modbus-Master und einem oder mehreren Modbus-Slaves. Es können maximal 32 Slaves unter einem Modbus-Master eingehängt werden.

Modbus-Geräte können in IndraLogic über die serielle Schnittstelle unter Verwendung des Geräts Modbus COM Port eingebunden werden. Dabei kann die IndraLogic Runtime als Modus-Master (1) oder Modbus-Slave fungieren. Im Folgenden wird dieser Modbus-Slave "Modbus Device" genannt. Bei Modbus seriell wird die Betriebsart Modbus RTU unterstützt.

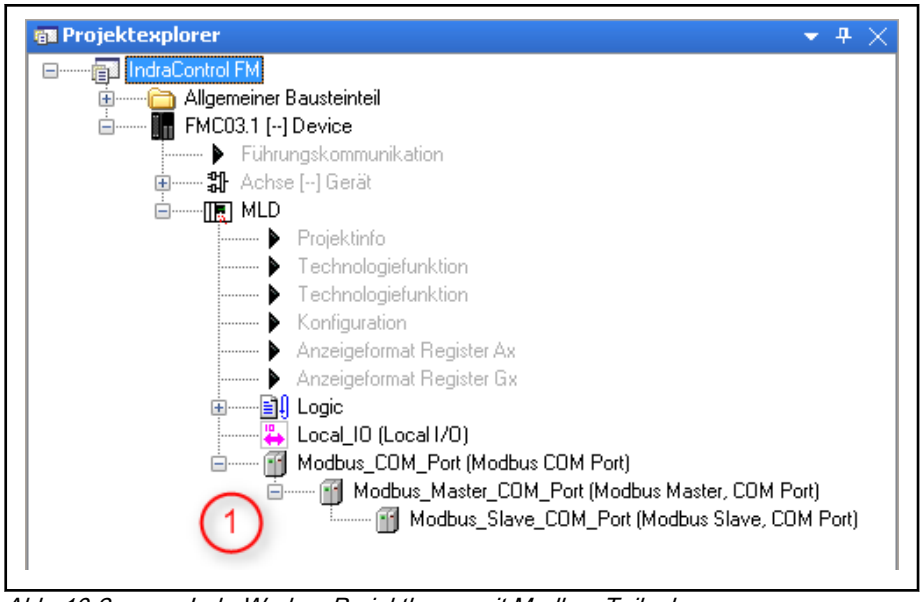


Abb. 10-2: IndraWorks - Projektbaum mit Modbus Teilnehmern

Sie können über die Modbus-Konfigurationsseiten zum Kommunikations-Parameter konfigurieren und die sog. Modbus-Kanäle anlegen. Hinter einem Modbus-Kanal steht ein einzelnes Modbus-Kommando (Daten lesen/schreiben) sowie die zugehörigen I/O-Kanäle.

Die allgemeinen Einstellungen zu den Geräteeditoren finden Sie in den Geräteeditor-Optionen.

Siehe auch:

- Geräteeditor-Optionen in der IndraLogic Anwendungsbeschreibung: DOK-IWORKS-IL2GPRO*V14-APRS-DE-P
- Kap. 10.3 "Modbus Master" auf Seite 280
- Kap. 10.4 "Modbus Slave Device" auf Seite 284

Unterstützte Funktionscodes

Wenn es nicht durch den speziellen Typ der Steuerung eingeschränkt wird, unterstützt der Master grundsätzlich folgende Funktionscodes:

- FC01 Read Coils
- FC02 Read Discrete Inputs
- FC03 Read Holding Registers
- FC04 Read Input Registers
- FC05 Write Single Coil
- FC15 Write Multiple Coils
- FC06 Write Single Register
- FC16 Write Multiple Registers
- FC23 Read/Write Multiple Registers

Zugriff auf Modbus-Geräte aus der Applikation

Wenn Sie ein Modbus-Gerät in den Gerätebaum einhängen, wird automatisch eine Instanz des zugehörigen Funktionsbausteins angelegt. Der Variablenname der Instanz entspricht dem Name des Geräts im Gerätebaum. Mit diesem Funktionsbaustein können Sie aus der Applikation auf die Funktionen der Modbus-Geräte zugreifen. Über die Ausgänge dieses Funktionsbausteins können Sie zum Beispiel den Status des letzten Modbus-Kommandos abfragen (ob erfolgreich oder mit Fehler).

Sie finden den Variablennamen in der Registerkarte E/A-Abbild des zugehörigen Geräteeditors unter IEC-Objekte.

Für Modbus Master (RTU/TCP)

Zusätzlich/Alternativ zu den im Konfigurator vordefinierten Modbus-Kommandos (und dem implizit erzeugten IO-Mapping) können über den Baustein ModbusRequest auch programmatisch Modbus-Requests ausgeführt werden.

10.3 Modbus Master

10.3.1 Registerkarte 'Modbus Serieller Port - Allgemein'

Die Registerkarte im Konfigurator des Modbus Serieller Port enthält die Grundeinstellungen für die Kommunikation über die serielle Schnittstelle.

COM-Port	Nummer der seriellen Schnittstelle
Baudrate	Parameter der seriellen Schnittstelle
Parität	
Daten-Bits	
Stop-Bits	

Siehe auch: Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.3.2 Registerkarte 'Modbus Master - Allgemein'

Die Registerkarte enthält die Grundeinstellung für die Modbus Kommunikation.

Modbus-RTU/ASCII

Übertragungsmodus	• RTU: Übertragung in binärer Form • ASCII: Übertragung im ASCII-Code (wird derzeit noch nicht von allen Treibern unterstützt)
Response Timeout [ms]	Zeitintervall, in dem der Master auf die Antwort eines Slave-Knotens wartet. Antwortet der Knoten nicht innerhalb dieser Zeitspanne, so liegt am impliziten Slave-FB ein Fehler an. Der für die Zeitspanne eingetragene Wert ist zugleich der Standardwert für jeden Knoten. Sie können dennoch für jeden Knoten innerhalb seiner Slave-Konfiguration einen auf diesen Knoten spezifisch abgestimmten Wert setzen.
Zeit zwischen den Frames [ms]	Die Zeit zwischen den Frames ist die Sendepause des Master zwischen der letzten Antwort (bzw. einer Zeitüberschreitung) und der nächsten Anfrage. Verwenden Sie diesen Parameter um die Kommunikation weniger störungsanfällig zu machen.
Automatischer Neustart Kommunikation	☑: Nach einem Kommunikationsfehler bestätigt IndraLogic automatisch den Fehler und versucht, mit der Ausführung der Modbus-Kommandos fortzufahren. ☐: Der Fehler muss über den Slave-Funktionsbaustein explizit bestätigt werden.

Siehe auch:
- Kap. 10.3.4 "Registerkarte 'Modbus Slave - Allgemein'" auf Seite 281
- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.3.3 Registerkarte 'ModbusTCP Master - Allgemein'

Die Registerkarte enthält die Grundeinstellungen für die Kommunikation im Netzwerk.

Modbus TCP

Response Timeout	Zeitintervall, in dem der Master auf die Antwort eines Slave-Knotens wartet. Antwortet der Knoten nicht innerhalb dieser Zeitspanne, so liegt am impliziten Slave-FB ein Fehler an. Der für die Zeitspanne eingetragene Wert ist zugleich der Standardwert für jeden Knoten. Sie können dennoch für jeden Knoten innerhalb seiner Modbus TCP-Slave-Konfiguration einen auf diesen Knoten spezifisch abgestimmten Wert setzen
Socket Timeout	Maximale Dauer, für die auf einkommende TCP/IP-Pakete gewartet wird. Für diese Dauer kann die Buszyklus-Task blockiert sein. (z.B. wenn ein Modbus TCP Slave getrennt wird)

Siehe auch:

- Kap. 10.3.4 "Registerkarte 'Modbus Slave - Allgemein'" auf Seite 281
- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.3.4 Registerkarte 'Modbus Slave - Allgemein'

Die Registerkarte enthält die Grundeinstellungen für die Kommunikation über die serielle Schnittstelle.

Modbus-RTU/ASCII

Slave-Adresse [1..247]	Adresse eines seriellen Modbus-Gerätes; Wert zwischen 1 und 247.
Response Timeout [ms]	Zeitintervall, in dem der Master auf die Antwort eines Slave-Knotens wartet. Es ist speziell auf diesen Slave-Knoten abgestimmt und überschreibt die allgemeine Response Timeout Einstellung des zugehörigen Masters.

Siehe auch:

- Kap. 10.3.2 "Registerkarte 'Modbus Master - Allgemein'" auf Seite 280
- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.3.5 Registerkarte 'ModbusTCP Slave - Allgemein'

Die Registerkarte enthält die Grundeinstellungen für die Kommunikation im Netzwerk.

Modbus-TCP

Slave IP-Adresse:	
Unit-ID [1..247]	Keine Bedeutung für Modbus TCP-Geräte. Dient der Adressierung serieller Modbus-Geräte, die über eine Modbus TCP/Serial-Gateway angesprochen werden.
Response Timeout	Zeitintervall, in dem der Master die Antwort dieses Slaves abwarten wird. Sie ist speziell auf diesen Slave-Knoten abgestimmt und überschreibt die allgemeine Response Timeout Einstellung des zugehörigen Masters.
Port	Port Nummer (TCP/IP) des Slaves. Für Modbus ist 502 Standard.

☞ Die Kommunikationseinstellungen können zur Laufzeit durch die Methode `UpdateCommunicationSettings` verändert werden. Diese Methode ist für den implizit erzeugten FB dieses Modbus-Slaves verfügbar.

Siehe auch:

- Kap. 10.3.2 "Registerkarte 'Modbus Master - Allgemein'" auf Seite 280
- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.3.6 Registerkarte 'Modbus Slave - Kanal'

Die Registerkarte dient zur Definition von Slave-Kanälen.

Jeder Kanal repräsentiert eine einzelne Modbus-Anfrage.

Wenn Sie auf die Schaltfläche "Kanal hinzufügen" klicken erscheint ein Dialog, in dem Sie neue Kanäle für diesen Modbus-Slave erzeugen können. Sie können in der Gerätebeschreibungs-Datei des Modbus-Slaves Beschreibungen zu einzelnen Modbus-Registern oder ganze Kanäle vordefinieren.

Kanal hinzufügen	Öffnet den Dialog "Modbus-Kanal"

Dialog 'Modbus-Kanal'

Name	Ein String, der den Namen des Kanals enthält
Zugriffstyp	• Read Coils (Funktionscode 03) • Read Discrete Inputs (Funktionscode 04) • Read Holding Registers (Funktionscode 03) • Read Input Registers (Funktionscode 04) • Read Single Coil (Funktionscode 05) • Write Single Register (Funktionscode 06) • Write Multiple Coils (Funktionscode 15) • Write Multiple Registers (Funktionscode 16) • Read/Write Multiple Registers (Funktionscode 23)
Trigger	• "CYCLIC": die Abfrage erfolgt periodisch. • "RISING_EDGE": die Abfrage erfolgt in Reaktion auf eine steigende Flanke der booleschen Trigger Variablen. Die Triggervariable wird im Register E/A-Abbild definiert.
Zykluszeit (ms)	Bei Trigger = CYCLIC: Abfrage-Intervall Hinweis: Das Abfrage-Intervall sollte gleich der Zykluszeit der Applikation sein oder ein Vielfaches von ihr betragen.
Kommentar	Beschreibung des Kanals

READ Register

Offset	Start-Adresse, ab welcher gelesen werden soll (Wertebereich 0-65535)
Länge	Anzahl zu lesender Register (bei Wort-Zugriff) oder Anzahl der zu lesenden Discrete Inputs (bei Bit- Zugriff).
Fehlerbehandlung	Definiert was im Fall eines Kommunikationsfehlers mit den Daten geschehen soll • Auf ZERO setzen • Letzten Wert beibehalten

WRITE Register

Offset	Nummer des Registers, in das geschrieben werden soll (Wertebereich 0-65535)
Länge	Anzahl der zu schreibenden Register (= Words) Wertebereich des Parameters ist abhängig vom Funktionscode:

Beispiel:

Im nachfolgenden Beispiel definiert die erste Zeile eine kombinierte Read-Write Operation (Funktionscode 23). Sie liest ein Wort aus dem "Holding Register" mit Abstand `'16#0001'` und schreibt zwei Worte in das Register mit Abstand `'16#0003'` schreibt. Die Operation wird ausgeführt, sobald die auf der Registerkarte E/A-Abbild definierte Trigger-Variable eine steigende Flanke aufweist.

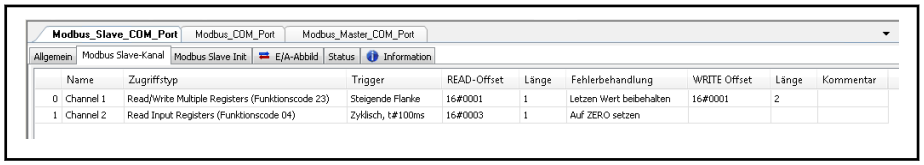


Abb. 10-3: Dialog Modbus Slave-Kanal

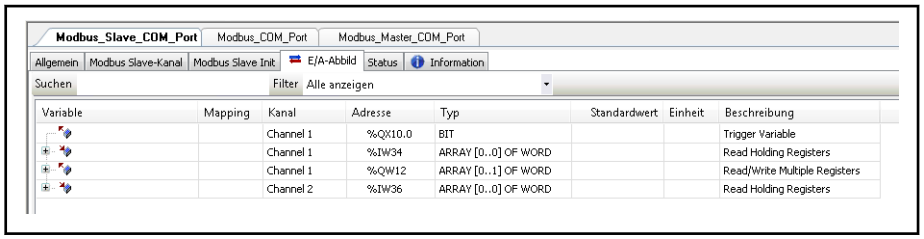


Abb. 10-4: Dialog Modbus Slave E/A-Abbild

Siehe auch:

- Kap. 10.3.4 "Registerkarte 'Modbus Slave - Allgemein'" auf Seite 281
- Kap. 10.3.5 "Registerkarte 'ModbusTCP Slave - Allgemein'" auf Seite 281

10.3.7 Registerkarte 'Modbus Slave Init'

Die Registerkarte dient zur Definition von Initialisierungskommandos.

Initialisierungskommandos werden einmalig beim Starten des Busses oder beim Aktiveren des Slaves (Setzen des "Enabled"-Flag der Slave-Instanz) ausgeführt.

Beim Einrichten oder Editieren eines Slave Initialisierungswertes sind folgende Parameter in den zugehörigen Dialogen zur Verfügung gestellt:

"Nach oben" "Nach unten "	Die Reihenfolge der Kanäle legt auch die Reihenfolge der Initialisierung der Variablen fest.
"Neu"	Öffnet den Dialog "Initialisierungswert". Darin werden die Initialisierungskommandos festgelegt.

Dialog 'Initialisierungswert'

"Zugriffstyp"	• Write Single Coil (Funktionscode 05) • Write Single Register (Funktionscode 06) • Write Multiple Coils (Funktionscode 15) • Write Multiple Registers (Funktionscode 16)
"Register Offset "	Nummer des Registers, in das geschrieben werden soll (Wertebereich 0-65535).
"Länge"	Anzahl der zu schreibenden Register (= Words) Wertebereich des Parameters ist abhängig vom Funktionscode.

"Initialisierungswert"	Initialisierungswert für Register
"Kommentar"	Möglichkeit zur kurzen Beschreibung der Daten

Siehe auch: Kap. 10.3.4 "Registerkarte 'Modbus Slave - Allgemein'" auf Seite 281

10.4 Modbus Slave Device

10.4.1 Registerkarte 'Modbus Serial Device - Allgemein'

Die Registerkarte enthält die Grundeinstellung für die Kommunikation im Netzwerk.

Unit ID	Unit-ID des Slaves
Timeout	☑: Aktivieren der Zeitüberwachungsfunktion. Die Timeout-Spanne wird in Millisekunden angegeben. Werte werden in Schrittgrößen von 500 ms eingestellt. Wenn für die angegebene Zeitspanne kein Schreib-Kommando empfangen wird, werden die Ausgänge auf 0 gesetzt.
Holding Registers (`%IW`)	Anzahl der Holding-Register: Mögliche Werte: 1-4096 (Maximale Anzahl kann in der Gerätebeschreibung eingegrenzt werden).
Input Registers (`%QW`)	Anzahl der Input-Register: Mögliche Werte: 1-4096 (Maximale Anzahl kann in der Gerätebeschreibung eingegrenzt werden).

Siehe auch Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.4.2 Registerkarte 'ModbusTCP Slave Device - Allgemein'

Die Registerkarte enthält die Grundeinstellung für die Kommunikation im Netzwerk.

Konfigurierte Parameter

"Timeout "	☑: Aktivieren der Zeitüberwachungsfunktion. Die Timeout-Spanne wird in Millisekunden angegeben. Werte werden in Schrittgrößen von 500 ms eingestellt. Wenn für die angegebene Zeitspanne kein Schreib-Kommando empfangen wird, werden die Ausgänge auf 0 gesetzt.
"Slave Port "	Port-Nummer des Slaves.
"Unit-ID "	Optional: Unit-ID des Slaves. Nur für TCP/RTU-Gateway.
"Holding Registers (`%IW`) "	Anzahl der Holding-Register: Mögliche Werte: 1-4096 (Maximale Anzahl kann in der Gerätebeschreibung eingegrenzt werden).
"Input Registers (`%QW`)"	Anzahl der Input-Register: Mögliche Werte: 1-4096 (Maximale Anzahl kann in der Gerätebeschreibung eingegrenzt werden).

"IP-Master Adresse"	IP-Adresse des Masters; Beispiel: "192 . 168 . 0 . 1" Ob die Option im Dialog angezeigt wird, hängt von der Gerätebeschreibung ab.
"IO Mapping aktiv"	☑: Aktivieren des I/O Scanners. Ob die Option im Dialog angezeigt wird, hängt von der Gerätebeschreibung ab.

☞ Änderungen der Eingangs-/Ausgangs-Assembly-Größe wirken sich unmittelbar auf die Anzahl der konfigurierbaren Parameter im E/A-Abbild aus.

Datenmodell

"Coils"	Definition der Startadresse des jeweiligen Modbus-Datenbereichs.
"Discrete Inputs"	
"Holding Register"	
"Input Register"	
"Holding- und Input-Register-Datenbereich überlagert"	☑: Überlagerung des Prozessabbildes des Holding- und des Input-Registers. Dies ist z.B. erforderlich, wenn über die Slave-Applikation auf Holding-Register geschrieben wird. Details zum Modbus-Datenmodell siehe Modbus-Protokoll-Spezifikation Kapitel. 4.3 und 4.4 (www.modbus.org)

Siehe auch: Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.5 Registerkarte 'Modbus - Parameter'

Die Registerkarte entspricht der Beschreibung des allgemeinen Konfigurationsdialogs. Der Dialog zeigt die gerätespezifischen Parameter in Listenform an.

Die Registerkarte ist verfügbar, wenn die Option "Generische Konfigurationseditoren" anzeigen aktiviert ist.

☞ Dieser Dialog ist für Testzwecke bestimmt. Werte sollten darin nur von Experten geändert werden.

Siehe auch:

- Kapitel: Dialog 'Optionen' - 'Geräteeditor' in der IndraLogic Anwendungsbeschreibung

  DOK-IWORKS-IL2GPRO*V14-APRS
- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.6 Registerkarte 'E/A-Abbild'

Die Beschreibung entspricht der allgemeinen Beschreibung der Registerkarte "E/A-Abbild".

Siehe auch:

- Kapitel: **Registerkarte '<Gerät> E/A-Abbild'** in der IndraLogic Anwendungsbeschreibung:

  DOK-IWORKS-IL2GPRO*V14-APRS
- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.7 Registerkarte 'Status'

Die Beschreibung entspricht der allgemeinen Beschreibung der Registerkarte "Status".

Siehe:

- Kapitel: **Registerkarte 'Status'** in der IndraLogic Anwendungsbeschreibung

  DOK-IWORKS-IL2GPRO*V14-APRS
- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

10.8 Registerkarte 'Information'

Die Beschreibung entspricht der allgemeinen Beschreibung der Registerkarte "Information".

Siehe:

- Kapitel: **Registerkarte 'Information'** in der IndraLogic Anwendungsbeschreibung:

  DOK-IWORKS-IL2GPRO*V14-APRS
- Kap. 10.2 "Modbus-Konfigurator" auf Seite 278

11 IO-Link

11.1 Einführung und Übersicht

IO-Link ist eine Kommunikationsschnittstelle zur Anbindung intelligenter Sensoren und Aktoren an Automatisierungssysteme. Ein IO-Link-System besteht immer aus einem IO-Link-Master und einem oder mehreren IO-Link-Devices. IO-Link-Master und IO-Link-Device sind dabei über eine Punkt-zu-Punkt-Verbindung verbunden. Das heißt, bei Verwendung eines Masters mit mehreren Devices ergibt sich immer eine sternförmige Topologie. Der IO-Link-Master ist üblicherweise ein Gateway zu einem übergeordneten Feldbus, zur Erstinbetriebnahme werden aber auch USB-IO-Link-Gateways verwendet.

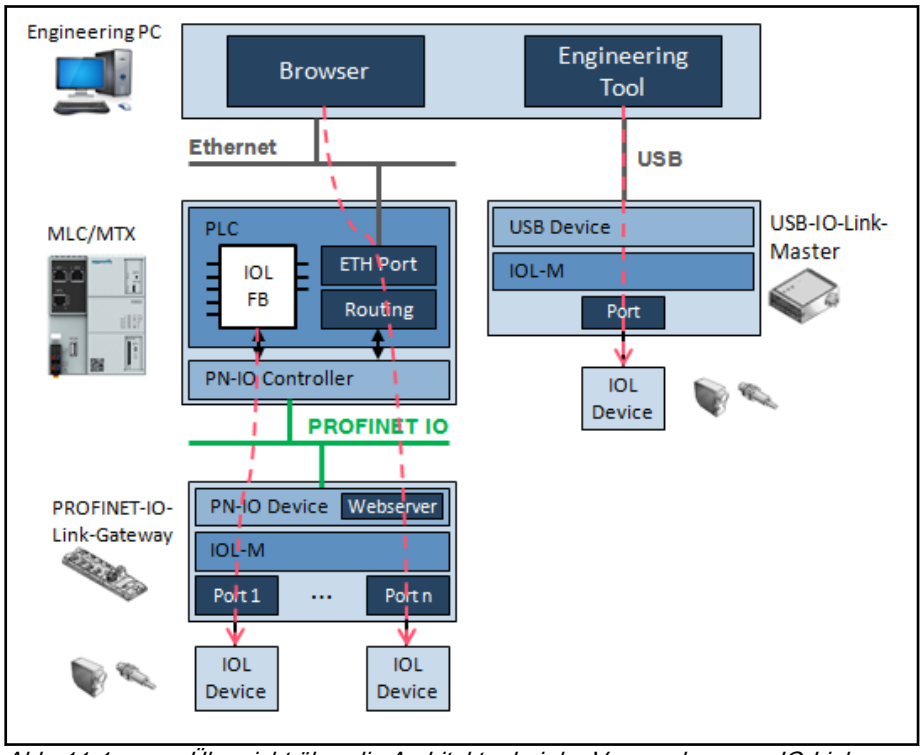


Abb. 11-1: Übersicht über die Architektur bei der Verwendung von IO-Link

11.2 Use Cases für das Engineering von IO-Link-Systemen

11.2.1 Einlesen oder Ausgabe von Analogen Werten mit Hilfe eines IO-Link-Devices

Die Onboard-Ports des IO-Link Master Gateway können entweder nur IO-Link Werte, oder digitale Werte verarbeiten.

Zur Verarbeitung von analogen Werten sind Adapter zur Konvertierung von analog auf IO-Link erforderlich.

Verfügbare IO-Link Interface Module:

Typ	Materialnummer	Funktion
AXL E IOL AI1 I M12 R	R911175300	Einlesen von analogen Stromsensoren
AXL E IOL AO1 I M12 R	R911175301	Analoge Stromausgabe

Typ	Materialnummer	Funktion
AXL E IOL AI1 U M12 R	R911175302	Einlesen von analogen Spannungssensoren
AXL E IOL AO1 U M12 R	R911175303	Analoge Spannungsausgabe
AXL E IOL RTD1 M12 R	R911175304	Einlesen von Temperatur-Widerstandssensoren
AXL E IOL DI8 M12 6P	R911175266	Einlesen von 8 digitalen Eingängen
AXL E IOL DI16 M12 6P	R911175298	Einlesen von 16 digitalen Eingängen
AXL E IOL DO8 M12 6P	R911175299	Ausgeben von 8 digitalen Ausgängen

11.2.2 Erstinbetriebnahme eines IO-Link-Devices

Die Erstinbetriebnahme eines IO-Link-Devices erfolgt am einfachsten mit einem USB-IO-Link-Master, der den Anschluss von IO-Link-Devices an den PC ermöglicht. USB-IO-Link-Master werden von verschiedenen Herstellern mit zugehörigen Engineering-Tools angeboten und dienen zum Einstellen und Testen der Funktionen des IO-Link-Devices. Nach Hinzufügen der Gerätebeschreibungsdatei des IO-Link-Devices (IODD-Datei) können die zyklischen Prozessdaten, sowie die Parameter mit den in der Gerätebeschreibungsdatei hinterlegten Bezeichnungen angezeigt und geändert werden.

Vorgehensweise:

- IO-Link-Device an USB-IO-Link-Master anschließen
- Engineering-Tool des USB-Masters öffnen (ggf. IODD-Datei des IO-Link-Devices importieren)
- IO-Link-Device parametrieren
- IO-Link-Device an das IO-Link-Gateway umstecken und mit der SPS betreiben

11.2.3 Prozessdaten und IO Device Description (IODD) von IO-Link-Devices

☞ IndraWorks unterstützt aktuell keine IO Device Description (IODD)!

Die Interpretation der Prozessdaten (Byteorder) erfolgt mit Hilfe des Datenblatts eines IO-Link-Device.

Gemäß IO-Link Spezifikation müssen Prozessdaten als `BYTE` übertragen werden.

Die Interpretation der Prozessdaten erfolgt in der Applikation, z. B. ob ein `WORD`- Analogwert geswappt werden muss oder nicht. Die Interpretation kann sich zwischen verschiedenen IO-Link Devices unterscheiden.

11.2.4 Serieninbetriebnahme von Maschinen mit IO-Link-Devices

Bei der ProfiNet-Serieninbetriebnahme können zur Übertragung von Konfigurationsparametern auf IO-Devices folgende IO-Link-Backup/Restore-Funktionsbausteine verwendet werden:

- IL_PNIOIoLinkBackupDevice

  Der Funktionsbaustein legt eine Sicherungsdatei auf der Steuerung an, in der die Parametrierung eines IO-Link-Devices gespeichert wird.
- IL_PNIOIoLinkRestoreDevice

Der Funktionsbaustein liest eine Sicherungsdatei auf der Steuerung aus und schreibt die Parametrierung aus der Datei in das entsprechende IO-Link-Device.

Zur Inbetriebnahme einer baugleichen Maschine kann die Sicherungsdatei auf die neue Steuerung geladen werden, z. B. über FTP-Client, oder über die Gerätedatensicherung in IndraWorks.

Die, in der Sicherungsdatei enthaltene Parametrierung kann jetzt durch Ausführung des IO-Link-Restore Funktionsbausteins in das entsprechende IO-Link-Device geschrieben werden.

☞ Bei Sercos ist die Datensicherung und Wiederherstellung über die Sercos Parametersicherung realisiert, d. h. der Data Storage ist in der Sercos-Parametersicherung enthalten.

Vorgehensweise:

- IO-Link-Device an der Maschine zur Erstinbetriebnahme parametrieren
- Backup-Datei auf der Steuerung der Maschine zur Erstinbetriebnahme erstellen
- Backup-Datei auf die Steuerung der baugleichen Maschine kopieren
- Restore der Parametrierung des IO-Link-Devices mittels der kopierten Backup-Datei

11.2.5 Parametrierung von IO-Link-Devices aus dem SPS-Programm

Für die Änderung der Parameter eines IO-Link-Devices aus der SPS-Applikation heraus, können folgende IO-Link-Call-Funktionsbausteine verwendet werden:

- Profinet: IL_PNIOIoLinkCall
- Sercos: IL_IoLinkCall

Die Funktionsbausteine dienen zum Lesen und Schreiben einzelner Parameter eines IO-Link-Devices, was unter Anderem das Ändern von Parametern während des Betriebs ermöglicht, z.B. für eine produktspezifische Parametrierung des IO-Link-Devices.

11.2.6 Diagnose von IO-Link-Devices

Zur Diagnose von IO-Link-Devices gibt es verschiedene Anwendungsfälle. So kann die Parametrierung und Funktion eines IO-Link-Devices am einfachsten mit einem USB-IO-Link-Master überprüft werden. Je nach Gateway kann dieses auch über einen Webserver verfügen. Wird am Engineering-PC ein Routing-Eintrag über die Steuerung zum Feldbus gesetzt, können die Webseiten des Gateways im Browser angezeigt und sowohl das Gateway, als auch das IO-Link-Device diagnostiziert und parametriert werden, ohne dass das IO-Link-Device aus der Maschine ausgebaut werden muss (Übersicht über die Architektur bei der Verwendung von IO-Link, Seite 287).

Weitere Diagnosemöglichkeiten bieten die folgenden Funktionsbausteine:

IO-Link-Call Funktionsbausteine zum Auslesen einzelner Parameter:

- Profinet: IL_PNIOIoLinkCall
- Sercos: IL_IoLinkCall

IO-Link-RemoteDeviceState-Funktionsbaustein zum Auslesen des Gerätestatus, Fehlerzählers und aktuell anstehenden Meldungen:

- Profinet: IL_PNIOIoLinkRemoteDeviceState

11.2.7 Gerätetausch von IO-Link-Devices

Für den Gerätetausch der IO-Link-Devices ist im IO-Link-Master ein Parameter-Server vorhanden. Ist die Parameter-Server-Funktion freigeschaltet und der Upload bzw. Download von Parameter-Serverdaten aktiviert, wird die Parametrierung des IO-Link-Devices nach jeder gültigen Parameteränderung in den Parameter-Server hochgeladen. Ein defektes IO-Link-Device kann während des Betriebs, oder im ausgeschalteten Zustand gegen ein anderes IO-Link-Device ausgetauscht werden. Dies führt zu einer Unterbrechung der Kommunikation mit dem IO-Link-Master, welcher eine Diagnose über den Feldbus ausgibt. Der IO-Link-Master nimmt automatisch die Kommunikation mit dem neuen IO-Link-Device wieder auf und prüft die CRC-Prüfsumme der Parametrierung. Stimmt diese nicht mit dem Parametersatz im Parameter-Server überein, führt der IO-Link-Master selbständig den Download des Parametersatzes durch. Das heißt, dem neuen IO-Link-Device werden die im Parameter-Server gesicherten Daten des defekten IO-Link-Devices übergeben.

☞ Damit dieser automatische Gerätetausch funktioniert, müssen folgende Voraussetzungen im IO-Link-Device gegeben sein:

- IO-Link-Device nach IO-Link Spezifikation 1.1
- IO-Link Index 0x0003 (siehe IO-Link Spezifikation) ist vorhanden und enthält alle zu sichernden Parameter

Wenn die Voraussetzungen nicht erfüllt sind, muss die Parametersicherung für den Gerätetausch im SPS-Programm ausgeführt werden, z. B. mit folgenden Backup/Restore Funktionsbausteinen:

- IL_PNIOIoLinkBackupDevice
- IL_PNIOIoLinkRestoreDevice
- IL_PNIOIoLinkCall
- IL_IoLinkCall

11.3 Profinet IO IO-Link-Gateways

11.3.1 Einführung und Übersicht

Der Aufbau und die Funktion von Profinet IO IO-Link-Gateways wird in der Spezifikation "IO-Link Integration Part 1 – Technical Specification for PROFIBUS and Profinet" beschrieben. Für den Aufbau eines Profinet IO IO-Link-Gateways gibt es zwei Varianten. Variante A ist ein „IO-Link Master Module“, welches über einen Lokal-Bus an einem Profinet-Device angeschlossen wird. Variante B ist ein „IO-Link Compact Master“.

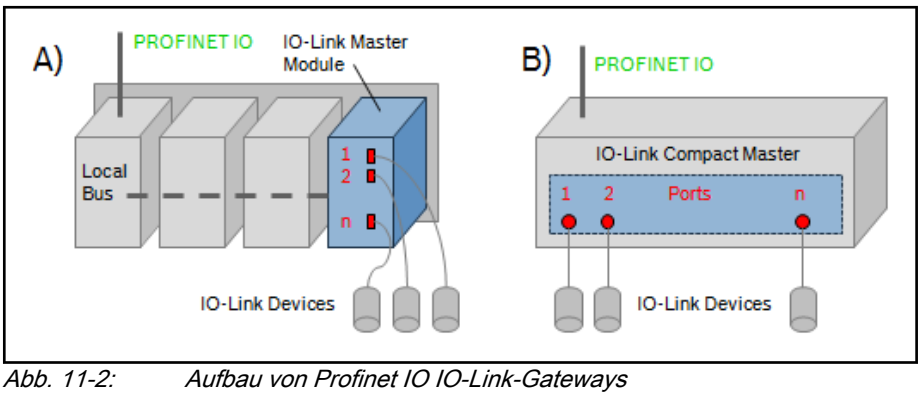


Abb. 11-2: Aufbau von Profinet IO IO-Link-Gateways

Der IO-Link-Master wird über den jeweiligen Slot adressiert. Bei einem Kompaktgerät ist der IO-Link-Master üblicherweise über Slot 1 erreichbar. Für die Belegung der SubSlots gibt es zwei Varianten, das Single-SubSlot-Mapping und das Multi-SubSlot-Mapping. Beim Single-SubSlot-Mapping findet der Zugriff auf alle Devices über den SubSlot 1 statt. Die Anwahl der Devices wird dann über den Port (beginnend mit 1) realisiert. Beim Multi-SubSlot-Mapping ist für jeden Device ein SubSlot vorhanden. Über den Subslot wird auf die zyklischen Ein- und Ausgangsdaten sowie die Diagnosen und Alarme zugegriffen. Die azyklischen Schreib- und Lesevorgänge auf die Parameter erfolgen allerdings ebenfalls über den SubSlot 1 des IOL-Masters und die Angabe des Ports.

Das heißt, bei der Verwendung der Funktionsbausteine zum Schreiben und Lesen von Parametern über azyklische Dienste, wird immer der SubSlot 1 verwendet. Die Devices werden dabei über den Port angewählt. Die zyklischen Daten aller IO-Link-Devices sind beim Single-Subslot-Mapping dem Subslot 1 zugewiesen. Beim Multi-SubSlot-Mapping werden die zyklischen Daten der IO-Link-Devices an den SubSlots konfiguriert.

Welche der Mapping-Varianten in einem Gateway implementiert ist, ist geräte- bzw. herstellerabhängig. Die vorhandenen Funktionen rund um IO-Link arbeiten mit beiden Varianten.

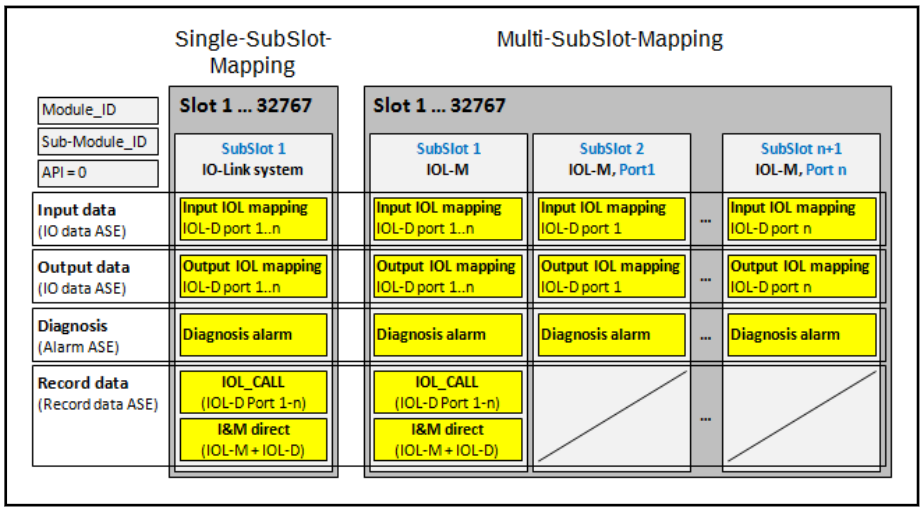


Abb. 11-3: Single-SubSlot-Mapping und Multi-SubSlot-Mapping bei Profinet IO IO-Link-Gateways

11.3.2 Funktionsbausteine für Profinet IO IO-Link-Gateways

Für die Kommunikation mit IO-Link-Devices an Profinet IO IO-Link-Gateways sind in der Bibliothek RIL_ProfinetIO die folgenden Funktionsbausteine vorhanden.

Parameterzugriff

IL_PNIOIoLinkCall: Der Funktionsbaustein dient zum Lesen und Schreiben von Parametern des IO-Link-Devices. Der Zugriff erfolgt über Index und Subindex des jeweiligen Parameters.

Identifikation

IL_PNIOIoLinkIdent: Der Funktionsbaustein dient zur Identifikation des IO-Link-Devices. Über die VendorID und DeviceID kann der Gerätetyp des IO-Link-Device eindeutig identifiziert werden. Dies wird z.B. für Backup/Restore-Funktionen benötigt.

IL_PNIOIoLinkIdentDetails: Der Funktionsbaustein liest zusätzlich auch mehrere Zeichenketten wie den Namen des Herstellers oder die Bezeichnung des IO-Link-Devices aus, die z.B. auf der Maschinen-HMI angezeigt werden können.

Status IL_PNIOIoLinkRemoteDeviceState: Der Funktionsbaustein liest die Diagnosedaten des IO-Link-Devices aus. Dies sind der Fehlerzähler, der aktuelle Zustand des Gerätes sowie die aktuell anstehenden Fehlermeldungen und Warnung.

System Command IL_PNIOIoLinkSystemCommand: Der Funktionsbaustein dient zum Auslösen von Kommandos des IO-Link-Devices, wie z.B. das Wiederherstellen der Werkseinstellungen.

Zugriffsperren IL_PNIOIoLinkSetDeviceAccessLock: Der Funktionsbaustein dient zum Setzen und Rücksetzen der Zugriffsperren im IO-Link-Device. In IO-Link sind vier Zugriffsperren spezifiziert.

- Ist die Schreibsperre für Parameter (ParameterAccessLock) gesetzt, können über die IO-Link-Kommunikation keine Parameter mehr geschrieben werden, mit Ausnahme des Parameters "Device Access Locks" (0x000C). Das Schreiben von Parametern führt dann zu einer Fehlermeldung am Funktionsbaustein IL_PNIOIoLinkCall. Das Lesen von Parametern ist nicht beeinflusst. Auch die Parameter-Server-Funktion des IO-Link-Masters wird nicht blockiert.
- Ist die Sperre des Parameter-Servers (DataStorageLock) gesetzt, ist der Download des Parameter-Servers des IO-Link-Masters gesperrt, d.h. im Parameter-Server gesicherte Daten eines IO-Link-Devices werden bei erkanntem Gerätetausch nicht in das neu angeschlossene IO-Link-Device geschrieben. Der Upload des Parameter-Servers ist nicht betroffen.
- Die Sperre der lokalen Parametrierung (LocalParamLock) ist für IO-Link-Devices mit einer Bedienschnittstelle am Gerät wichtig. Ist diese gesetzt, können über die lokale Bedienung am IO-Link-Device keine Parameter geändert werden.
- Die Sperre der lokalen Bedienung (LocalUserInterfaceLock) ist ebenfalls für IO-Link-Devices mit einer Bedienung lokal am Gerät wichtig. Ist diese Gesetzt, so ist die lokale Bedienung des IO-Link-Devices gesperrt.

IL_PNIOIoLinkGetDeviceAccessLock: Über den Funktionsbaustein kann der Zustand der Zugriffsperren im IO-Link-Device ausgelesen werden.

Backup/Restore IL_PNIOIoLinkBackupDevice: Der Funktionsbausteine dient zum Sichern der Parametrierung eines IO-Link-Devices in einer Datei auf der Steuerung.

IL_PNIOIoLinkRestoreDevice: Der Funktionsbaustein dient zum Laden der Parametrierung aus einer Datei auf der Steuerung in ein IO-Link-Device.

Diese ermöglichen das Sichern der Parametrierung eines IO-Link-Devices in der Steuerung und das Wiederherstellen der Parametrierung in einer baugleichen Maschine, oder nach einem Gerätetausch des IO-Link-Devices.

11.3.3 Beispielhafte Anwendung eines Profinet IO IO-Link-Gateways der Firma Balluff

Im folgenden wird beispielhaft die Konfiguration eines IO-Link Gateways anhand eines Gerätes der Firma Balluff beschrieben. Gateways anderer Hersteller werden in ähnlicher Weise konfiguriert, Unterschiede sind durch die Abbildung des Gateways in der Gerätebeschreibungsdatei verursacht (siehe z.B. oben Single-Slot- bzw. Multi-Slot-Mapping) und werden hier nicht weiter erläutert.



Das Beispiel basiert auf der Firmware des Geräts "Balluff v2.0.0.0 / GSDML-V2.2-Balluf-BNI-PNT-508-105-20120807.xml"

Empfohlen wird die Firmwareversion v2.6.0.0 / GSDML-V2.31-Balluff-BNI-PNT-508-105-Z015-20160901.xml

Hinzufügen des Profinet IO IO-Link-Gateways in das Projekt

Zunächst muss die Gerätebeschreibungsdatei (GSDML-Datei) des Profinet IO IO-Link-Gateways über **Extras** ▸ **Gerätedatenbank** ▸ **Geräte hinzufügen…** hinzugefügt werden. Der Profinet IO Controller wird entweder beim Anlegen der Steuerung oder über den Kontext-Menüpunkt **Gerät setzen** konfiguriert. Das Profinet IO IO-Link-Gateway kann entweder manuell, per drag&drop aus der Gerätebibliothek bzw. über das Kontextmenü **Hinzufügen**, oder wenn es bereits zur Verfügung steht, online durch **Geräte suchen** hinzugefügt werden

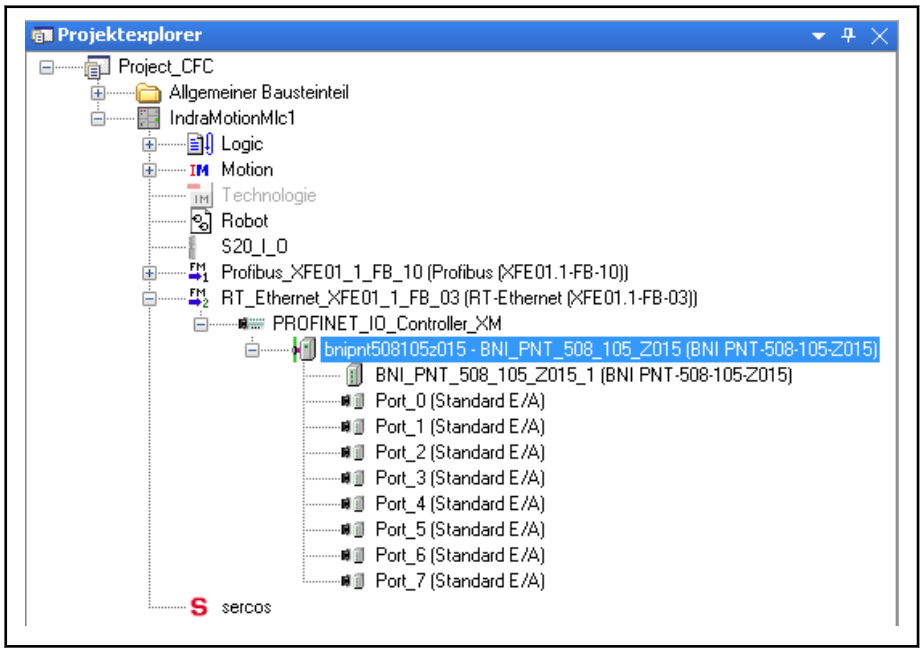


Abb. 11-4: Profinet IO IO-Link-Gateway im Projektexplorer

Konfiguration des Profinet-IO-Link-Gateways

Im IO-Link-Master (Slotnummer 1) kann die Funktion der Ports konfiguriert werden. Für die Ports, an denen IO-Link-Devices angeschlossen werden, muss die Funktion IO-Link ausgewählt werden.



Aktuelle Firmwareversionen, z.B. v2.6.0.0 und GSDML-V2.31-Balluff-BNI-PNT-508-105-Z015-20160901.xml, unterstützten nur noch Schließer, Öffner und Ausgänge.

IO-Link wird automatisch vom Device übernommen, wenn bei "Gerät setzen" ein IO-Link Modul ausgewählt wird.

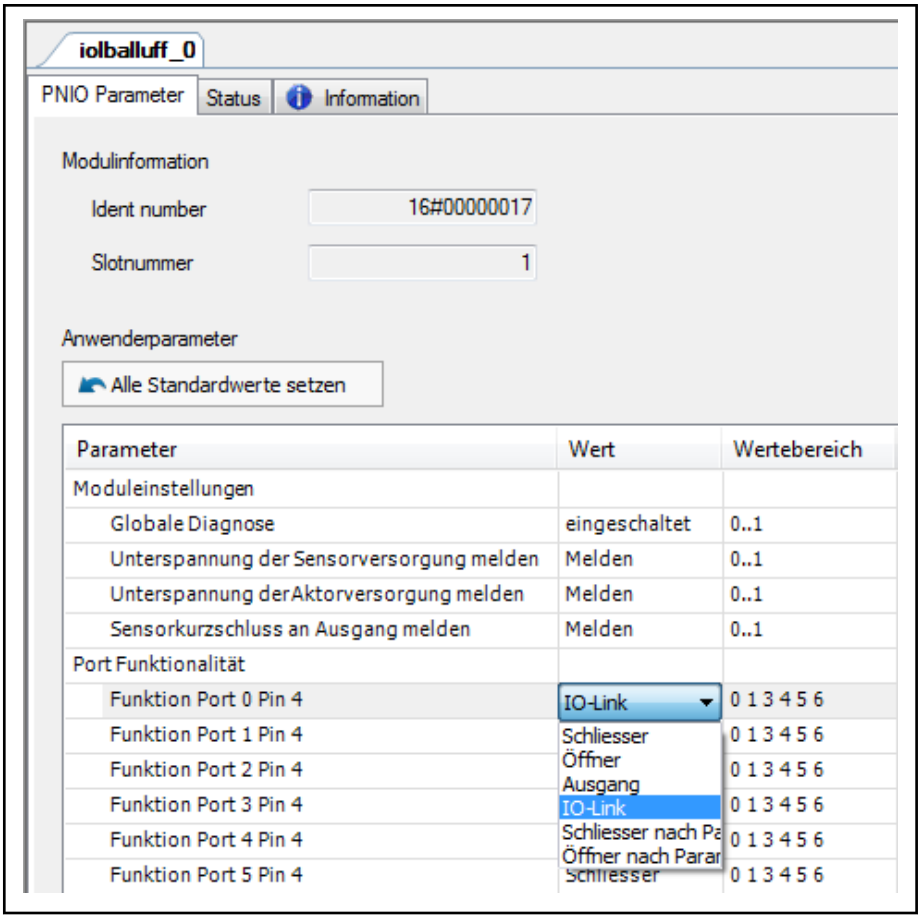


Abb. 11-5: Auswahl der Funktion des Ports als IO-Link

Konfiguration des IO-Link-Devices

Für den Port muss daraufhin das IO-Link-Device mit den jeweiligen Ein- und Ausgangsdaten (aus Steuerungssicht) gesetzt werden. Im Beispiel in Abb. 11-6 "Setzen des IO-Link-Devices" auf Seite 295 werden zyklisch vier Bytes an das IO-Link-Device gesendet. Die Ein- und Ausgangsdaten sind der Dokumentation des IO-Link-Devices zu entnehmen.

Eine Konfiguration von mehr zyklischen Daten, als das IO-Link Device besitzt, generiert keinen Fehler im Anlauf. Die zusätzlich konfigurierten Daten sind dabei lediglich bezüglich der Performance relevant. Eine Konfiguration von weniger zyklischen Daten, als das IO-Link Device besitzt, generiert einen Fehler im Anlauf.

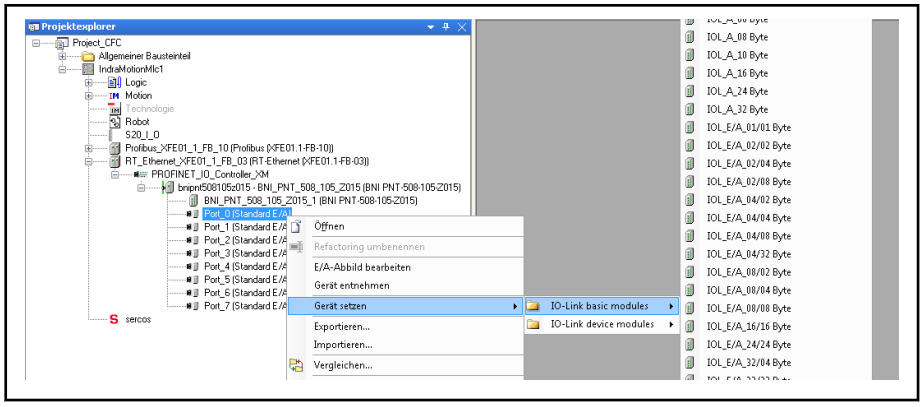


Abb. 11-6: Setzen des IO-Link-Devices

Für das IO-Link-Device muss die Zykluszeit bestehend aus Zykluszeit-Basis und Multiplikator eingestellt werden. Jedes IO-Link-Device hat eine minimale Zykluszeit, welche der Dokumentation des Devices zu entnehmen ist. Weitere wichtige Einstellungen betreffen die Konfiguration des Parameter-Servers. Hier kann die Verwendung des Parameter-Servers vollständig aktiviert/deaktiviert oder die Parameter-Server-Daten gelöscht werden. Außerdem können der Upload der Parametrierung in das Gateway und der Download der Parametrierung in das IO-Link-Device getrennt aktiviert/deaktiviert werden. Empfohlen wird die komplette Aktivierung der Parameter-Server-Funktion (incl. Up-/Download), da nur hierdurch der autarke Gerätetausch ohne Benutzerinteraktion erreicht werden kann.



Mit aktuellen Firmwareversionen, z.B. v2.6.0.0 und GSDML-V2.31-Balluff-BNI-PNT-508-105-Z015-20160901.xml, ist das konfigurieren der Zykluszeit nicht zwingend notwendig.

Die Zyklus-Einstellung kann auf Default belassen werden.

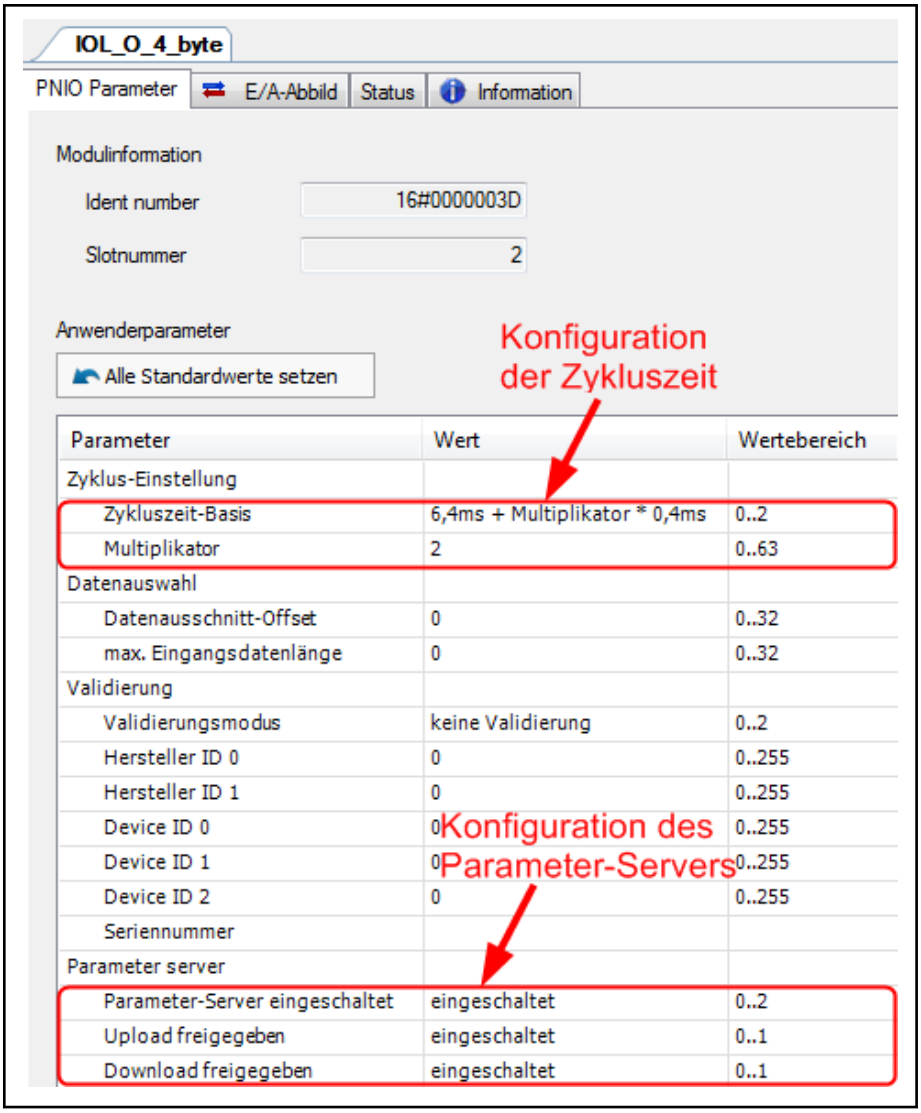


Abb. 11-7: Konfiguration des IO-Link-Devices

11.3.4 Beispielhafte Anwendung des Rexroth Profinet IO IO-Link-Gateways S67E-PN-IOL8-DI4-M12-6P

Dieses Kapitel beschreibt die Konfiguration eines IO-Link Gateways beispielhaft anhand des Gerätes S67E-PN-IOL8-DI4-M12-6P.

☞ Gateways fremder Hersteller werden in ähnlicher Weise konfiguriert, jedoch kann es durch die individuelle Modul-Abbildung in der GSD-Datei[1)] zu Unterschieden in der Modul-Konfiguration kommen.

Siehe Kap. 11.3.1 "Einführung und Übersicht" auf Seite 290.

Voraussetzungen zur Verwendung des IO-Link-Gateways:

- Das IO-Link-Gateway muss in der IndraWorks Gerätedatenbank vorhanden sein.

1) Gerätebeschreibungsdatei

Ist dies nicht der Fall, muss die Gerätebeschreibungsdatei des IO-Link-Gateways in der Gerätedatenbank hinzugefügt werden, siehe Kap. 4.1.2 "Geräte verwalten" auf Seite 15

- Die Steuerung muss mit einem Realtime-Ethernet Funktionsmodul konfiguriert sein, in diesem Beispiel eine XM2-Steuerung mit dem Funktionsmodul XFE01_1_FB_03.

  Die Konfiguration des Funktionsmoduls kann beim Anlegen der Steuerung erfolgen, oder nachträglich im Kontextmenü der Steuerung über den Befehl → **Hinzufügen**

- Das Realtime-Ethernet Funktionsmodul muss mit einem Profinet IO Controller konfiguriert sein.

  Die Konfiguration des Profinet IO Controllers erfolgt über das Kontextmenü des Funktionsmodul-Unterknotens → **Gerät setzen**:

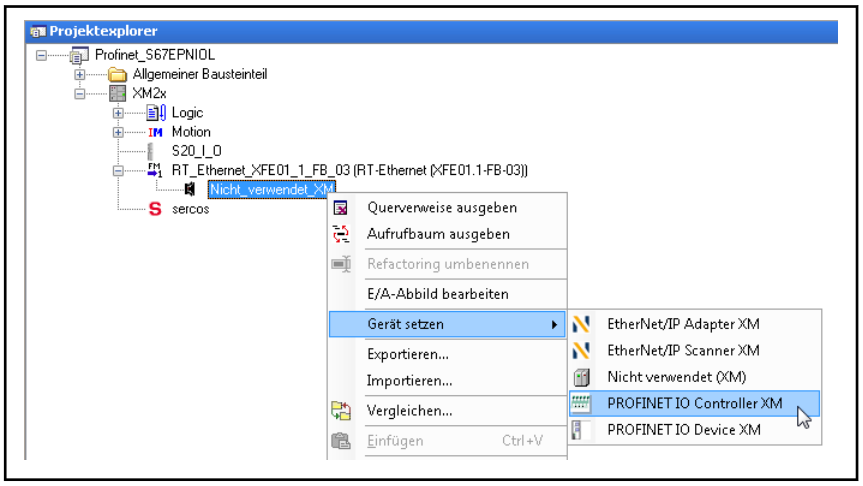


Abb. 11-8:

Hinzufügen des Profinet IO IO-Link-Gateways in das Projekt

Das Profinet IO IO-Link-Gateway kann auf folgende Arten im Projekt hinzugefügt werden:

1. Per drag&drop aus der Gerätebibliothek in der Kategorie **Peripherie ▸ ProfinetIO ▸ E/A**
2. Im Projektexplorer über das Kontextmenü des Profinet IO Controllers **Hinzufügen ▸ Device ▸ E/A**
3. Wenn das Profinet IO IO-Link-Gateway bereits am Bus angeschlossen ist (Online) im Kontextmenü des Profinet IO Controllers über den Befehl **Geräte suchen**, siehe Kap. 6.5.4 "Profinet IO Controller, Geräte suchen" auf Seite 113

Im Falle des S67E-PN-IOL8-DI4-M12-6P wird das Gerät beim Anlegen mit 8 digitalen Eingangsmodulen konfiguriert.

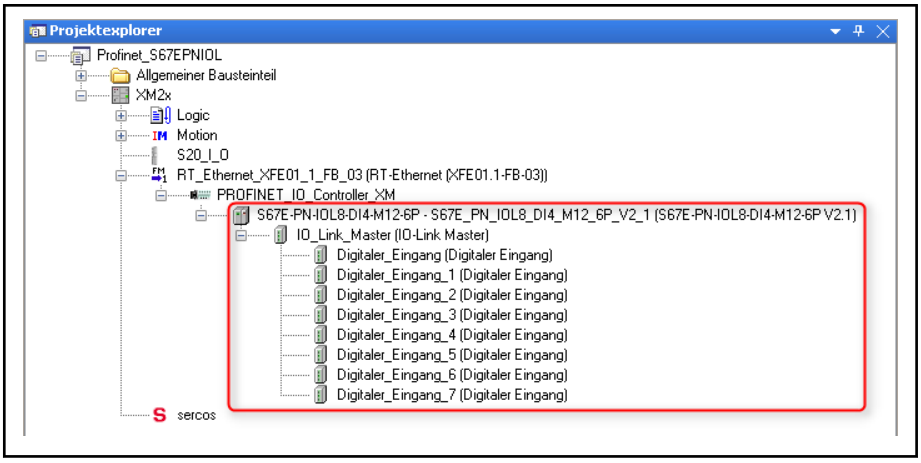


Abb. 11-9: Projektexplorer – Profinet IO IO-Link-Gateway

In dieser Konfiguration können keine weiteren Module hinzugefügt werden, da das Gerät maximal 8 Module unterstützt.

Um einen anderen Modul-Typ konfigurieren zu können, muss zuvor ein bestehendes Modul gelöscht werden.

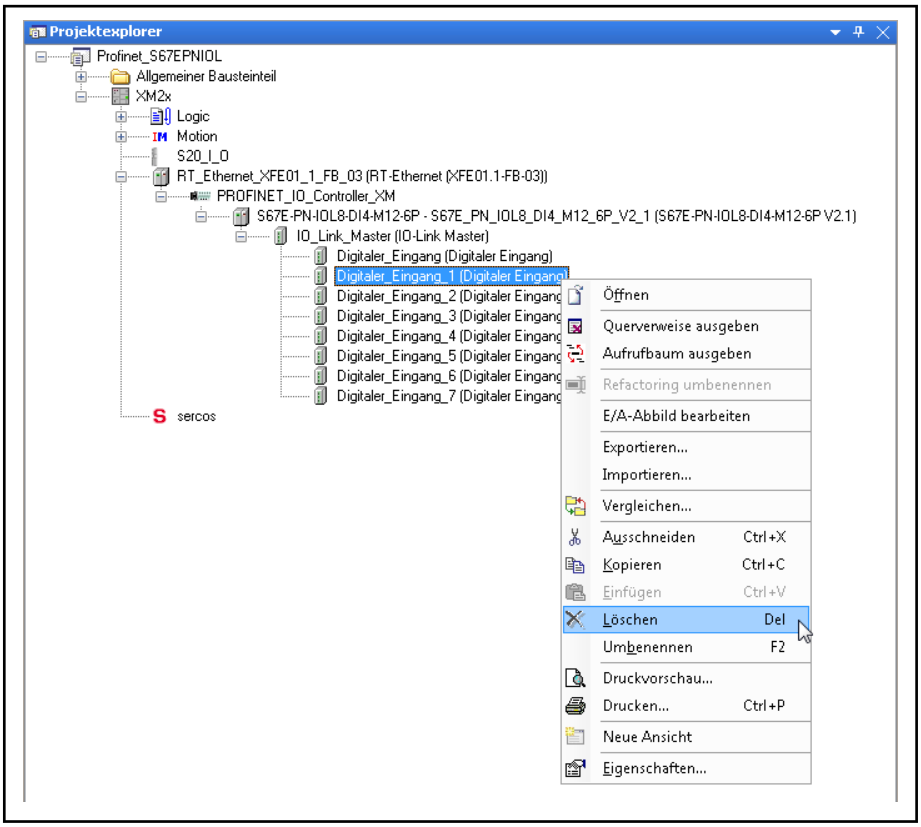


Abb. 11-10:

Nachdem das Modul gelöscht wurde, kann ein anderes Modul über das Kontextmenü des IO-Link Masters hinzugefügt werden: → **Hinzufügen ▸ Modul**

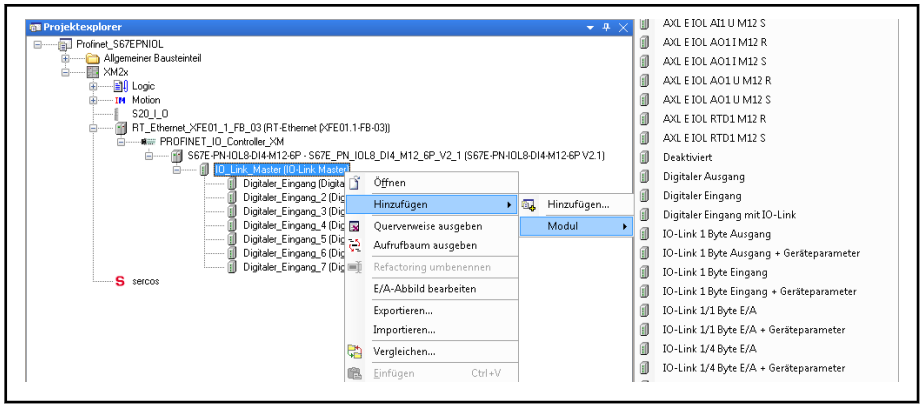


Abb. 11-11: Projektexplorer, IO-Link Modul hinzufügen

Auf diese Weise wird das neue Modul im Projektexplorer unterhalb bestehender Module eingruppiert.

Wenn das neue Modul einer bestimmten Modulposition zugewiesen werden soll, muss das Hinzufügen über folgenden Kontextmenü-Befehl durchgeführt werden: → **Hinzufügen ▸ Hinzufügen...**

Der Dialog "Hinzufügen" öffnet sich.

Der gewünschte Position kann über die Einstellung "**Einfügen vor**" in der eingeblendeten Modul-Liste bestimmt werden.

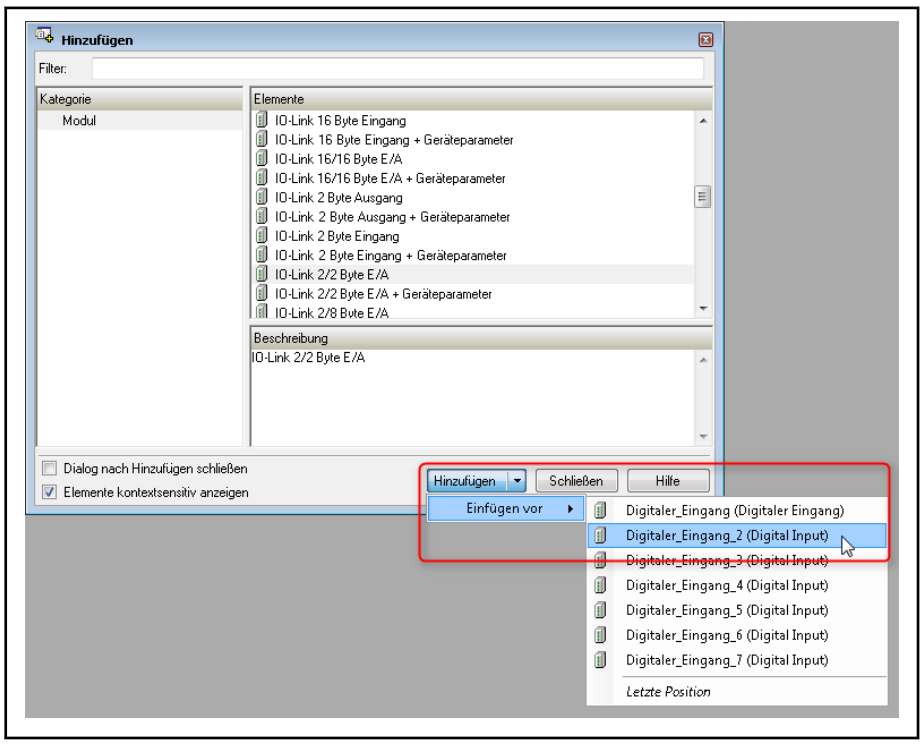


Abb. 11-12: Hinzufügen des IO-Link-Moduls an einer definierten Modul-Position

Konfiguration des IO-Link-Moduls

Ein Kriterium bei der Konfiguration des IO-Link Moduls ist die Prozessdatenbreite des angeschlossenen IO-Link Device (siehe Gerätedokumentation). Im Idealfall ist die Prozessdatenbreite des IO-Link Moduls so konfiguriert, dass sie der Bandbreite des angeschlossenen IO-Link Device entspricht.

- Wenn das IO-Link Modul mit einer zu geringen Prozessdatenbreite konfiguriert ist, wird ein Fehler im Anlauf generiert.
- Wenn das IO-Link Modul mit einer höheren Bandbreite als das Device konfiguriert ist, wird kein Fehler generiert, jedoch wird hierbei die Prozessdatenbandbreite nicht optimal genutzt (Performance relevant).

Konfigurationsbeispiele

IO-Link Modul	IO-Link device	
Prozessdatenbreite		Betrieb
24 Bit	24 Bit	OK
32 Bit	24 Bit	OK (nicht optimal genutzte Datenbandbreite)
24 Bit	**32 Bit**	**Fehler**

Die Zykluszeit wird automatisch, passend zum IO-Link Device je Port gesetzt (schnellste Zykluszeit).

11.4 Sercos IO-Link Gateways

11.4.1 Einführung und Übersicht

Die Sercos IO-Link Inbetriebnahme unterscheidet sich von der Profinet IO-Link Inbetriebnahme. Bei Sercos ist es z. B. nicht erforderlich Slots anzulegen.

Erforderliche Konfigurationen im generischen Geräteeditor des IO-Link Gateways:

- In den Registerkarte "Benutzerdefinierte Parameter" müssen die IO-Link Ports konfiguriert werden.
- In der Registerkarte "Funktionsgruppen" müssen die Längen als Summe aller Ports konfiguriert werden.

Weitere Details zur Konfiguration entnehmen Sie bitte der Dokumentation **"DOK-CONTRL-S67E*S3*IOL-AP"**, Kapitel "Einstellungen an den IO-Link-Ports" → "Netzwerkspezifische Eigenschaften des IO-Link-Masters für Sercos S67E-S3-IOL8-DI4-M12-6P".

Die Vorgehensweise ist für S20-IO und S67E-S3-IO identisch.

11.4.2 Funktionsbausteine für Sercos IO IO-Link-Gateways

Für die Kommunikation mit IO-Link-Devices an Sercos IO-Link-Gateways können die folgenden Funktionsbausteine aus der Bibliothek RIL_IoLink verwendet werden:

Parameterzugriff **IL_IoLinkCall:** Der Funktionsbaustein dient zum Lesen und Schreiben von Parametern des IO-Link-Devices. Der Zugriff erfolgt über Index und Subindex des jeweiligen Parameters.

Identifikation **IL_IoLinkIdent:** Der Funktionsbaustein dient zur Identifikation des IO-Link-Devices. Über die VendorID und DeviceID kann der Gerätetyp des IO-Link-Device eindeutig identifiziert werden.

Backup/Restore Standard Sercos Backup und Restore Funktionsbausteine.

Die Funktionsbausteine ermöglichen das Sichern der Parametrierung eines IO-Link-Devices in der Steuerung und das Wiederherstellen der Parametrie-

rung in einer baugleichen Maschine, oder nach einem Gerätetausch des IO-Link-Devices.

12 Lokales S20 IO

12.1 Features des Lokalen S20 IO

Bei S20 IO Modulen handelt es sich um IO-Klemmen, die direkt an XM- und VPx-Steuerungen und an modulare S20-Buskoppler S20-yy-BK[1] angeschlossen werden können.

Entsprechende Buskoppler sind für Sercos III und Profinet IO verfügbar.

Folgende IO-Typen verfügbar:

- Digitale Eingangsmodule
- Digitale Ausgangsmodule
- Analoge Eingangsmodule
- Analoge Ausgangsmodule
- Einspeise-Modul
- Technologie-Module
  - Zähler-Module
  - Module mit Inkremental-Gebereingängen
  - Module mit SSI-Gebereingängen
- Module zur Temperaturmessung
- Safety IO Module
  - digitale Safety IO Module

Titel	Kennzeichnung
Die Automatisierungsklemmen der Produktfamilie Rexroth S20	DOK-CONTRL-S20*SYS*INS-APxx-DE-P[2] R911335987
Rexroth S20-Buskoppler für Sercos III	DOK-CONTRL-S20*S3*BK**-DAxx-DE-P R911335969
Rexroth S20-Buskoppler für Profibus DP	DOK-CONTRL-S20*PB*BK**-DAxx-DE-P R911342759
Rexroth S20-Buskoppler für Profinet IO	DOK-CONTRL-S20*PN*BK**-DAxx-DE-P R911335967
Rexroth S20-Buskoppler für Safety-Anwendungen (Sercos III, Profibus DP, Profinet IO), jeweils Kap. "Sichere Bussysteme"	Rexroth MTX SafeLogic Systemübersicht DOK-MTX***-SL**SYS*V14-PRxx-DE-P R911341697
	Rexroth MLC SafeLogic Systemübersicht DOK-XLCMLC-SL**SYS*V14-PRxx-DE-P R911341695

Tab. 12-1: Weiterführende Dokumentation

1) *yy - jeweilige Bussystem*

2) *xx - Ausgabe*



Die Eigenschaften und Funktionsweise der verschiedenen S20 IO Module sind in den jeweiligen Modul-Dokumentationen beschrieben.

Informationen zur Verwendung der S20 Safety IO Module entnehmen Sie bitte den entsprechenden Safety -Dokumentationen.

12.2 S20-IO konfigurieren

12.2.1 S20-Objekt und S20-Module, Übersicht

XM-Steuerungen erlauben eine Erweiterung der lokal verfügbaren E/A-Einheiten durch Anreihen von S20-Modulen an der rechten Seite der Steuerung.

Das S20-IO-Objekt ist dazu im Projektexplorer um die gewünschten S20-Module zu erweitern.

S20-IO-Objekt

- Registerkarte 'Einstellungen' auf Seite 304
- Registerkarte 'S20 Module Parameter' auf Seite 307
- Registerkarte 'Feldbus Diagnose' auf Seite 307
- Registerkarte 'E/A-Abbildauf Seite 309
- Registerkarte 'Status' auf Seite 310
- Registerkarte 'Information' auf Seite 312

Zum "S20-IO-Objekt" hinzugefügte S20-IO-Module (Beispiel)

- S20-Module einfügen auf Seite 312
- S20-Module aktivieren / deaktivieren auf Seite 312
- Registerkarte 'Benutzerdefinierte Parameterauf Seite 315
- Registerkarte 'E/A-Abbild' auf Seite 315
- Registerkarte 'Information' auf Seite 316

12.2.2 S20-IO-Objekt

Registerkarte 'Einstellungen'

Der S20-Busmaster auf einer XM-Steuerung unterstützt die folgenden Betriebsarten zur Ansprache der angeschlossenen S20-Module:

1. **Freilaufend:**

   In der Betriebsart "Freilaufend" arbeitet der S20-Lokalbus ohne zeitlichen Bezug zur SPS.

   Die Definition einer Buszyklustask ist in dieser Betriebsart nicht erforderlich (Ausnahme siehe Hinweis unten), die Einstellung des Buszyklustask kann auf der Defaulteinstellung **<unspezifiziert>** bleiben.

   Mit jedem Zyklus der Tasks, die S20 Ein-Ausgänge verwenden, werden die jeweiligen Ausgänge in deren Kontext ausgegeben und Eingänge werden eingelesen.

   Dies gilt für die Default Einstellungen der Option "Variablen immer aktualisieren" in den SPS-Einstellungen der Steuerung und in den Registerkarten E/A-Abbild der zugeordneten S20 Module:

   - Defaultwert:

     Deaktiviert (Aktualisierung nur wenn in einer Task verwendet)

Wenn in der Einstellung "**Variablen immer aktualisieren**" des S20-Busmasters (Steuerung) die Einstellung "**Aktiviert 2 (immer in Buszyklus-Task)**" eingestellt wird, dann werden alle S20 EA's der Buszyklustask zugeordnet und der Austausch aller S20 Ein-/Ausgänge erfolgt nur im Kontext dieser Task.

Siehe:

- "Ereignisgesteuerte SPS-Task als Buszyklustask für den S20-Lokalbus konfigurieren" auf Seite 306
- Kap. 4.5.1 "Beschreibung der Buszyklus-Task" auf Seite 29

Die Aktualisierung der S20-Module durch den S20-Master erfolgt dabei so schnell wie möglich, je nach S20-Modulbestückung.

In der Betriebsart "Freilaufend" wird das externe Ereignis "**EVENT_OP_MODE_S20_CYCLIC**" nicht unterstützt. Wenn das Ereignis dennoch verwendet wird, werden die S20-Module nicht Aktualisiert!

⇒ Eingänge und Ausgänge werden so schnell wie möglich am S20-Modul aktualisiert.

2. **Zyklisch:**

   Für den Betrieb des S20-Lokalbus wird der Zykluszeit-Takt vorgegeben, der auf den Sercos-Takt des Systems "eingerastet" wird.

   Der S20-Takt kann ein Bruchteil, ein Vielfaches oder gleich dem Sercos-Takt sein.

   Der S20-Lokalbus läuft unabhängig von den Sercos-Phasen.

   In Verbindung mit einer extern ereignisgesteuerten SPS-Task wird empfohlen, das externe Ereignis "EVENT_OP_MODE_S20_CYCLIC" zur Synchronisation zu verwenden.

Wenn mehrere SPS-Task verwendet werden, muss die ereignisgesteuerte SPS-Task als Buszyklustask des S20-Masters konfiguriert werden, siehe hierzu "Ereignisgesteuerte SPS-Task als Buszyklustask für den S20-Lokalbus konfigurieren" auf Seite 306.

Der S20-Master initialisiert S20-Module, die "Fast Response" unterstützen d. h. die Ausgänge werden so früh wie möglich ausgegeben und die Eingänge werden so spät wie möglich eingelesen.

⇒ Eingänge und Ausgänge werden im festen Raster am S20-Modul, im Modus "Fast Response", aktualisiert (kaum bis kein Jitter).

3. **Taktsynchron zu Sercos:**

   Der S20-Lokalbus-Takt ist gleich dem Sercos-Takt.

   Der S20-Lokalbus läuft abhängig von den Sercos-Phasen (CP4/BB - EA Datenaustausch; NRT, CP0-CP3 - kein EA-Datenaustausch).

   In Verbindung mit einer extern ereignisgesteuerten SPS-Task wird empfohlen, das externe Ereignis "EVENT_OP_MODE_SERCOS_CYCLIC", oder "EVENT_OP_MODE_S20_CYCLIC" zur Synchronisation zu verwenden.

Wenn mehrere SPS-Task verwendet werden, muss die ereignisgesteuerte SPS-Task als Buszyklustask des S20-Masters konfiguriert werden. siehe hierzu "Ereignisgesteuerte SPS-Task als Buszyklustask für den S20-Lokalbus konfigurieren" auf Seite 306.

Die Erfassung der Eingangssignale und die Ausgabe der Ausgänge an den S20-Modulen finden zum Zeitpunkt T4 statt (wie bei Sercos).

Zur Taktsynchronen Erfassung und Ausgabe der Ein- und Ausgänge sind synchronfähige S20-Module erforderlich, ein Mischbetrieb mit nicht synchronen (asynchronen) S20-Modulen ist möglich.

S20-Module am S20-Lokalbus und Sercos Geräte, welche den synchronen Betrieb unterstützen sind untereinander zum T4 Zeitpunkt synchron. Hierzu muss das externe Ereignis "EVENT_OP_MODE_SERCOS_CYCLIC" für die SPS-Task konfiguriert werden.

⇒ Eingänge und Ausgänge werden im festen Raster am S20-Modul zum Zeitpunkt T4 aktualisiert, bei "EVENT_OP_MODE_SERCOS_CYCLIC" synchron zum Sercos (kaum bis kein Jitter).

Die genauen Zeitpunkte in den Betriebsarten 2 und 3 des Bus-Timings können nur von S20-Modulen eingehalten werden, die für den synchron Betrieb freigegeben sind. Ein Mischbetrieb mit synchronen und nichtsynchronen S20-Modulen ist zulässig.

Ereignisgesteuerte SPS-Task als Buszyklustask für den S20-Lokalbus konfigurieren

Um eine ereignisgesteuerte SPS-Task als Buszyklustask für den S20-Lokalbus zu konfigurieren gehen Sie wie folgt vor:

1. Öffnen Sie den generischen Geräteeditor der Steuerung durch Doppelklick auf den Steuerungsknoten im Projektexplorer und wechseln Sie zur Registerkarte "**SPS-Einstellungen**".
2. Stellen Sie die Konfiguration im Feld "Variablen immer aktualisieren" auf folgende Einstellung ein: "**Aktiviert 2 (immer im Buszyklustask)**".
3. Definieren Sie im Feld "Buszyklus-Optionen Buszyklustask" den gewünschten Task, in der Beispielgrafik "**PlcTask**".

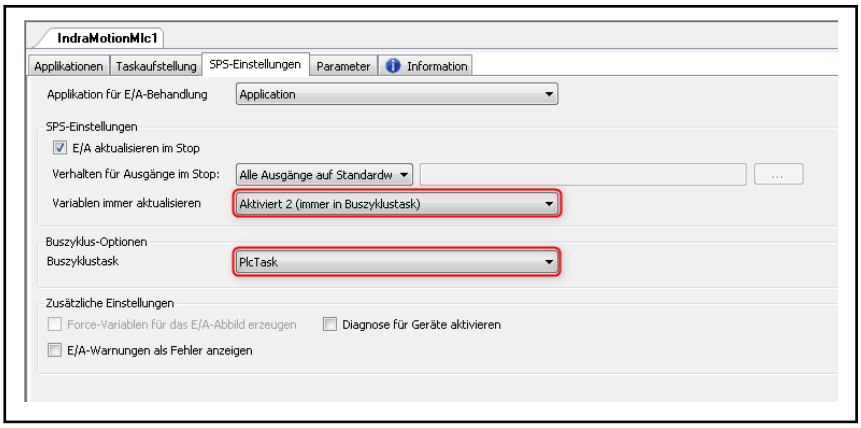


Abb. 12-1:

Die Aktualisierung der S20 Ein-Ausgänge erfolgt jetzt im Kontext der definierten Task.

☞ Die Definition einer spezifischen Task in den Buszyklus-Optionen hat erst dann Wirkung, wenn die Konfiguration im Feld "Variablen immer aktualisieren" aktiviert wird!

Registerkarte 'S20 Module Parameter'

☞ Die Registerkarte "S20 Module Parameter" wird standardmäßig nicht mehr angezeigt.

Um diese Registerkarte anzuzeigen, muss in den Optionen die Anzeige für den "Generischen Geräteeditor" aktiviert werden.

Das Fenster enthält Informationen zur Konfiguration und Diagnose des S20-Busses.

Fensteraufbau

- **Diagnose**: Parametername aus der Gerätebeschreibungsdatei, nicht editierbar.
  - Konfigurationsfehler
  - Masterfehler
  - BusStatus
  - BusInfo
  - BusSyncInfo
  - Modulfehler
- **S20 operating mode**: Betriebsart S20 Lokalbus, nicht editierbar.
- **S20 cycle time**: S20 Lokalbus Zykluszeit, nicht editierbar.
- **Diag Provide Counter**
- **Diag Provide Access**
- **Diag Provide CommonData**
- **Diag Provide DataField 0**
- **Diag Provide DataField 1**
- **Diag Provide DataField 2**
- **Diag Provide DataField 3**

Kann der Parameter bearbeitet werden (dies hängt von der Gerätebeschreibung ab, nicht-editierbare Parameter werden hellgrau dargestellt), kann mit einem Doppelklick auf das Tabellenfeld (oder mit der <Leertaste> bei einem bereits selektierten Feld) ein Eingabefeld oder eine Auswahlliste geöffnet werden, über die der Wert geändert werden kann.

Die Wertübernahme erfolgt durch <Parameter schreiben>.

Handelt es sich bei dem Wert um eine Dateiangabe, öffnet sich der Standarddialog zum Auswählen einer Datei.

Registerkarte 'Feldbus Diagnose'

Die Registerkarte "Feldbus Diagnose" zeigt im Online-Betrieb die am S20-Knoten anliegenden Module mit den zugehörigen Diagnosedaten.

Zum Anzeigen der jeweiligen Diagnosemeldung muss das Modul angewählt werden. Im oberen Teil werden dafür alle Module angezeigt, die auch im Projektbaum vorhanden sind.

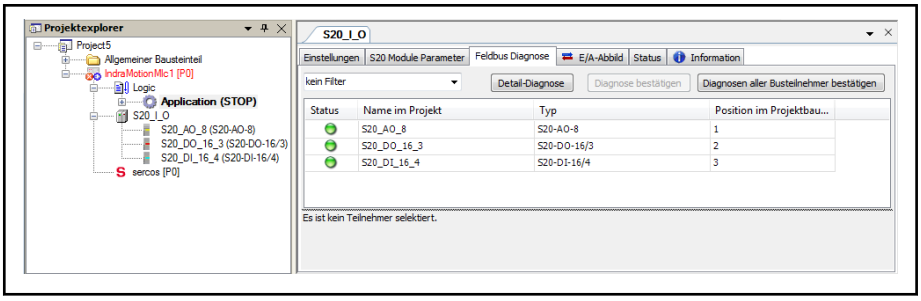


Abb. 12-2: S20 Feldbus Diagnose, Steuerung online, eingeloggt

Im unteren Teil erscheint dann die Meldung zum jeweils ausgewählten Modul. Dazu muss die Steuerung online sein. Ist dies nicht der Fall erscheint ein Hinweistext in dem Bereich, in dem auch die Diagnosemeldungen erscheinen.

Im oberen Teil der Registerkarte werden neben dem Namen und dem Typ der Module auch die Positionsnummern angezeigt. Diese Positionsnummer gibt an, an welcher Stelle sich das jeweilige Modul im Projektbaum befindet. Diese fortlaufende Nummerierung wird vom System automatisch durchgeführt. Wenn sich die Reihenfolge oder die Anzahl der Module ändert wird die Nummerierung automatisch angepasst.

Außerdem wird der Status mit Leuchtdioden, die sich entsprechend verfärben, in der ersten Zeile angezeigt.

Grün heißt, dass keine Fehler vorliegen.

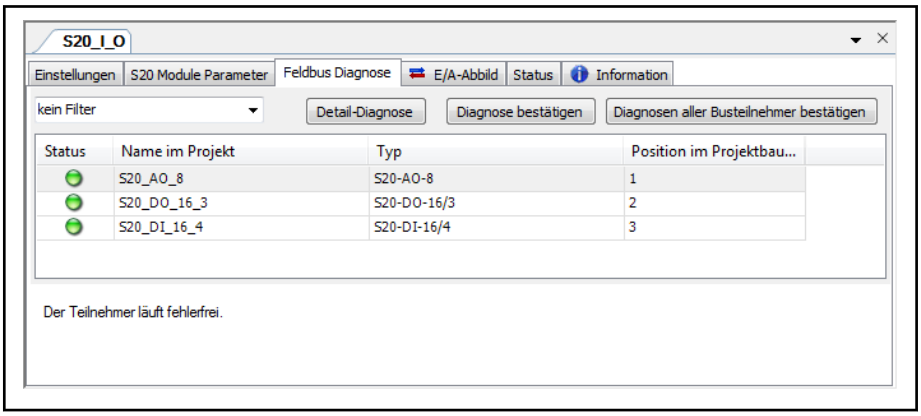


Abb. 12-3: S20 Feldbus Diagnose, Steuerung online, eingeloggt, fehlerfrei

Beispiel einer Diagnose im Fehlerfall

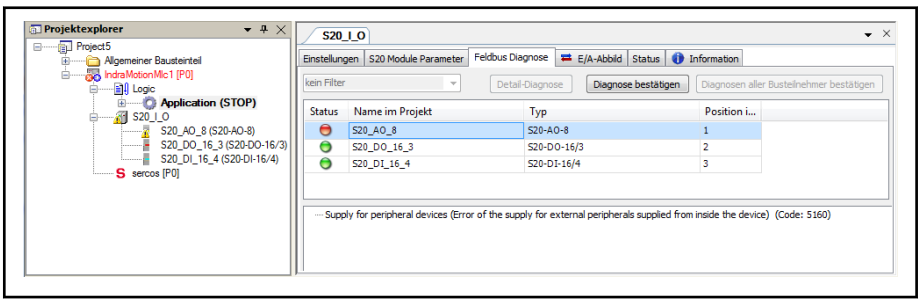


Abb. 12-4: S20 Feldbus Diagnose, Steuerung online, eingeloggt, Fehlerfall

Die Spannungsversorgung zu einem der Module wird unterbrochen, sodass das Modul nicht von der Steuerung erkannt wird. In diesem Fall erscheint am betroffenen Modul ein gelbes Warndreieck, welches Hinweis auf eine vorhandene Diagnose gibt. Die Status-Leuchtdiode des betroffenen Moduls färbt sich rot. Wird das Modul angewählt erscheint im unteren Bereich die Textausgabe, die Angaben der Fehlerursache enthält. Außerdem wird der diag-

nosespezifische Fehlercode mit angegeben. Siehe Kap. 9.3.3 "Sercos III-Master, Sercos-Busdiagnose" auf Seite 248

Eine weitere mögliche Farbe der Leuchtdiode ist grau. Diese Farbe erscheint, wenn entweder die Steuerung noch nicht eingeloggt ist oder ein Modul deaktiviert wurde.

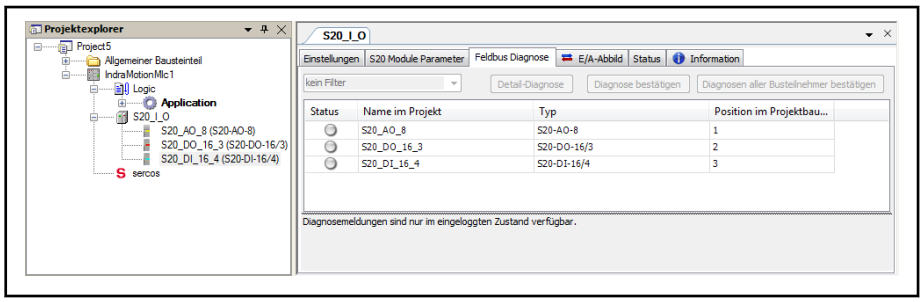


Abb. 12-5: S20 Feldbus Diagnose, nicht eingeloggt

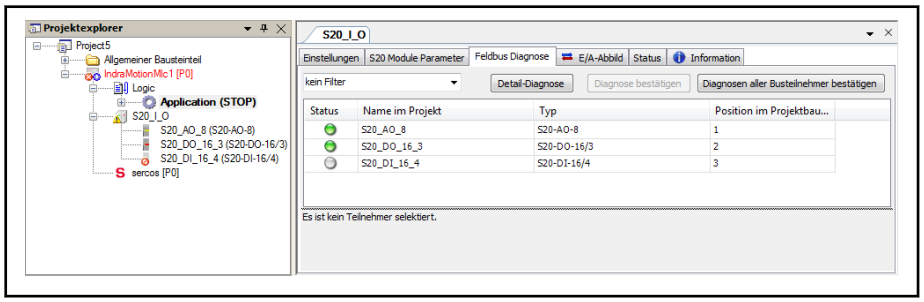
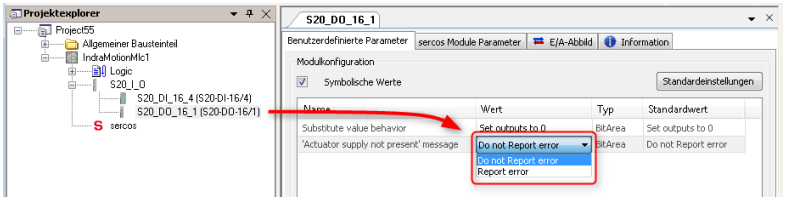


Abb. 12-6: S20 Feldbus Diagnose, Steuerung online, eingeloggt, ein Modul deaktiviert

Das Ziehen der Versorgungsspannung an einem S20 Modul führt nicht zwingend zu einer Diagnose.

Bei einigen Modulen ist die Reaktion auf fehlende Versorgungsspannung einem eigenen Modulparameter zugeordnet.

Hier am Beispiel des Moduls "S20_DO_16_1":



Der zugehörige Parameter (sofern vorhanden) steht per Default auf "Do not Report error".

Dies führt dazu, dass keine Meldung generiert wird.

Wenn eine Diagnosemeldung ausgegeben werden soll, muss der Parameter auf "Report error" eingestellt werden.

Registerkarte 'E/A-Abbild'

Buszyklus-Optionen

- **Buszyklus-Task:** gibt an, mit welcher SPS-Task das E/A-Abbild durchgeführt wird,
  - **Default:** Zykluseinstellungen des übergeordneten Busses verwenden
  - Anwender Task

Siehe Kap. 4.5 "Buszyklus-Task: Kopieren der E/A-Daten zwischen SPS und Feldbus-Stack" auf Seite 29

Registerkarte 'Status'

Die Registerkarte zeigt den Status des gesamten S20-Busses an.

Offline: n/a

Online: "Läuft", "Läuft nicht (n/a)"

Angezeigt wird ferner die "Letzte Diagnosemeldung", die über "Bestätigen" quittiert werden kann.

Diagnose im Fehlerfall

Im Fehlerfall wird eine detaillierte Diagnose übergeben. Hier ist an der realen Steuerung das dritte Modul falsch bestückt.

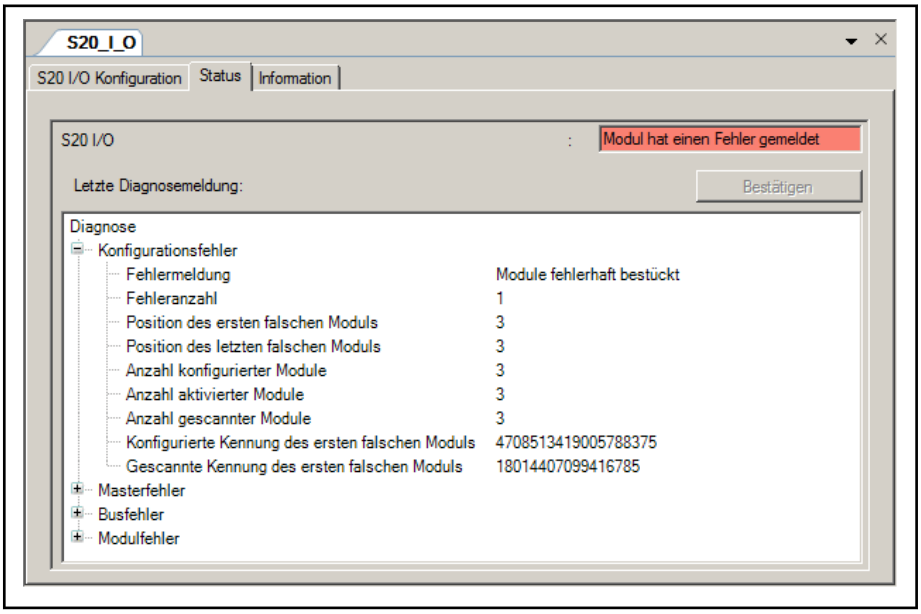


Abb. 12-7: Detaillierte Fehlermeldung im S20-IO-Bus

1. **Konfigurationsfehler**: Bei einem solchen Fehler stimmt die Konfiguration der Module im Projekt nicht mit den Modulen überein, die physikalisch am Bus vorhanden sind.

   Mögliche Meldungen sind:

   - Konfiguration fehlerfrei
   - Zu viele Module bestückt

     D.h. es sind mehr Module am Bus vorhanden als im Projekt konfiguriert.
   - Zu wenige Module bestückt

     D.h. es sind weniger Module am Bus vorhanden als im Projekt konfiguriert. Die fehlenden Module werden im Projektbaum mit einem gelben Warndreieck versehen.
   - Module fehlerhaft bestückt

     D.h. es sind andere Module am Bus vorhanden als im Projekt konfiguriert. Die falsch konfigurierten Module werden im Projektbaum mit einem gelben Warndreieck versehen.
   - Sonstiger Konfigurationsfehler

   Zusätzlich werden noch weitere Informationen geliefert:

   - Anzahl der Fehler

- Position des ersten und letzten falsch konfigurierten Moduls
- Anzahl der im Projekt konfigurierten Module
- Anzahl der im Projekt aktivierten Module
- Anzahl der am Bus vorhandenen Module
- Für das erste falsch konfigurierte Modul:
  - Kennung des Moduls im Projekt
  - Kennung des Moduls am Bus

2. **Masterfehler**: Hierbei handelt es sich um einen internen Fehler des S20-Masters.

   Mögliche Meldungen sind:

   - Master fehlerfrei
   - Allgemeiner Masterfehler

     Als Zusatzinformation wird hier eine Fehlernummer geliefert.

3. **BusStatus**

   Mögliche Meldungen sind:

   - Bus fehlerfrei
   - Allgemeiner Busfehler: Bei einem solchen Fehler läuft der Bus nicht, z.B. weil ein Modul ausgefallen ist.

4. **BusInfo**

   Mögliche Meldungen sind:

   - Meine Meldung vorhanden
   - Zusätzliches Modul erkannt
   - Zusätzliches Modul entfernt

5. **BusSyncInfo**

   Mögliche Meldungen sind:

   - Synchronisation OK
   - Synchronisation verloren
   - Synchronisation undefiniert

6. **Modulfehler**: Mindestens ein Modul meldet einen Fehler (z.B. Kurzschluss). Der Bus läuft weiter. Die letzte erkannte Diagnose wird angezeigt. Fehlerhafte Module werden außerdem im Projektbaum mit einem gelben Warndreieck versehen.

   - **Fehlermeldung:**
     - Modul fehlerfrei
     - Modulspezifischer Fehler
   - **Modulposition:** Slot, Position des S20-Moduls
   - **Fehlercode:** 4-stelliger Diagnose Code, zu finden in der Dokumentation des S20-Moduls, oder in Sercos Busdiagnose, da die Fehlercodes identisch sind. Siehe Kap. 9.3.3 "Sercos III-Master, Sercos-Busdiagnose" auf Seite 248
   - **Fehlerstelle:** zeigt den Kanal an, beim Wert 255 ist das komplette S20-Modul betroffen

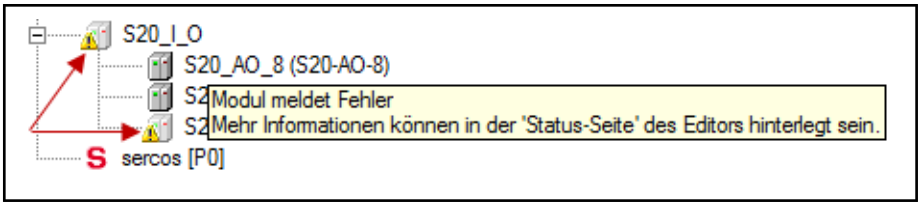


Abb. 12-8: S20-IO-Bus, Fehlermeldung im Projektnavigator

Registerkarte 'Information'

Die Registerkarte zeigt einige allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

Name, Hersteller, Kategorien, Version, Bestellnummer, Beschreibung, ggf. Bild.

12.2.3 S20-Module einfügen

Alle für die jeweilige Steuerung zur Verfügung stehenden S20-Module befinden sich in der Bibliothek "Peripherie" im Ordner "S20".

Ziehen Sie aus der Bibliothek die benötigten S20-Module in das betreffende "S20-IO"-Objekt. Neue S20-Module lassen sich im Projektexplorer auch zwischen bereits vorhandene S20-Module einfügen.

Alternativ können Sie S20-IO-Module im Kontextmenü über **Hinzufügen ▸ Modul ▸ <S20-Modul>** des "S20-IO"-Objekts hinzufügen. Hierbei wird das neue Modul als letztes Modul unter "S20-IO" eingefügt.

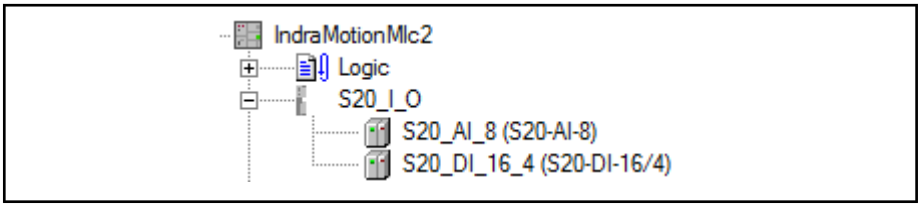


Abb. 12-9: S20-Objekt mit zwei Modulen (Beispiel)

Wenn ein benötigtes S20-Modul nicht standardmäßig in der IndraWorks Gerätebibliothek vorhanden ist, kann es mittels Import einer SDDML-Datei in die IndraWorks Gerätedatenbank hinzugefügt werden.

Wählen Sie hierzu den Menübefehl:

Extras ▸ Gerätedatenbank ▸ Geräte hinzufügen.

Siehe auch Kap. 4.1 "Gerätedatenbank" auf Seite 15.

12.2.4 S20-Module aktivieren / deaktivieren

Die an einer Steuerung konfigurierten S20-Module können im Projektexplorer aktiviert bzw. deaktiviert werden.

Dies erfolgt durch Anwählen / Abwählen der Schaltfläche, die am Icon des S20-Moduls erscheint.

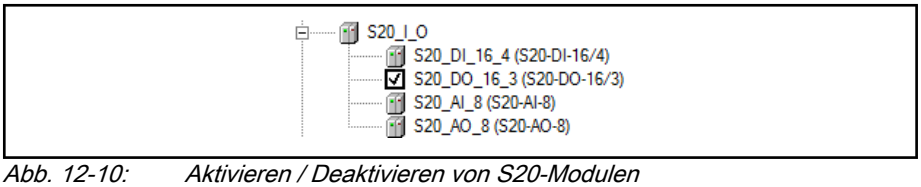


Abb. 12-10: Aktivieren / Deaktivieren von S20-Modulen

Ist der Haken gesetzt, ist das Modul aktiviert. Es wird in der Diagnose des Busses berücksichtigt.

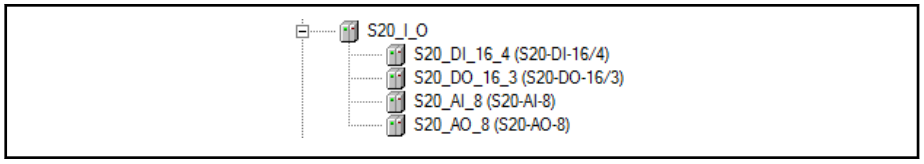


Abb. 12-11: S20-Bus mit aktivierten Modulen

Ist der Haken nicht gesetzt, ist das Modul deaktiviert. Es wird in der Diagnose des Busses nicht berücksichtigt.

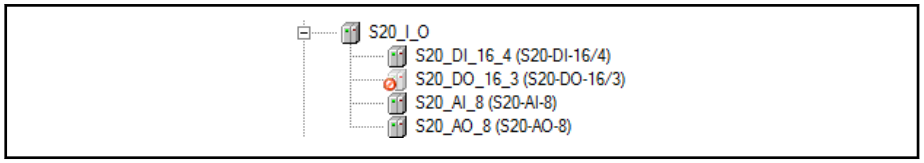


Abb. 12-12: S20-Bus mit einem deaktivierten Modul

Wird ein neues S20-Modul angelegt, ist es per Voreinstellung aktiviert.

Beim Laden der Konfiguration werden alle aktiven S20-Module berücksichtigt. Deaktivierte S20-Module werden ignoriert.

Ein aktiviertes jedoch nicht vorhandenes Modul erzeugt in der Diagnose eine Fehlermeldung.

Ein deaktiviertes nicht vorhandenes Modul erzeugt in der Diagnose keine Fehlermeldung.

12.2.5 S20-Module deaktivieren mittels 'UserDefs.cfg'

Ab IndraWorks 14V22 können S20 IO-Module am Lokalbus des Steuerungssystems XMx, die in IndraWorks konfiguriert sind, durch Definition in einer auf der Steuerungs-FlashCard abliegenden Datei deaktiviert werden.

Dadurch werden Diagnose-Meldungen für Module verhindert, die zwar konfiguriert, aber nicht vorhanden sind.

Aktivierung der Funktion

Die Funktion wird durch das Vorhandensein einer gültigen Datei "UserDefs.cfg" auf der Steuerungs-FlashCard in folgenden Partitionen aktiviert:

Bei XM- und VPx-Steuerungen: **\USER**

Optional können durch Einträge in der "cfg-Datei" eines Systems ein anderer Pfad und Dateiname definiert werden.

Beispiel:

```
[IOSpecialDefs]
IODeactFilePath=MyFilePath
IODeactFileName=MyFileName
```

Die Datei "UserDefs.cfg" kann Einträge für weitere Busse enthalten:

Inline-Module, Sercos, Profinet IO, Profibus DP.

Zeitpunkt der Modul-Deaktivierung

Die Auswertung der Datei „UserDefs.cfg“ und das Deaktivieren der dort eingetragenen Module erfolgt zum Zeitpunkt und im Kontext des internen Events ´PrepareUpdateConfiguration´.

Das Event tritt bei folgenden Aktionen ein:

- Anlauf nach Power-ON
- Reset (warm)
- Reset (kalt)

- Programm laden nach Reset (Ursprung)

Struktur der Datei "UserDefs.cfg"

Die Datei ist strukturiert in einfach, oder mehrfach vorhandene Sektionen und den darin enthaltenen Einträgen.

- **Sektionen**: Die Bezeichnung einer Sektion ist in ´**[**´ und ´**]**´ geklammert. Eine Sektion ist begrenzt durch den Beginn der nächsten Sektion oder durch das Ende der Datei.

  Die Bezeichnung besteht aus dem Symbol **S20IO.Master.** und dem konstanten Index **0** (kompatibel zum Deaktivieren eines Profibus-Slaves).

  Beispiel:

  ```
  [S20IO.Master.0]
  ```

- **Sonderfunktion**: Die Definition des Zeichens ´**X**´ für den Index anstelle einer Ziffer führt dazu, dass diese Sektion in der Auswertung ignoriert wird.

  Es ist möglich, für den Master-Index **0** mehrere Sektionen zu definieren. Die darin enthaltenen Einträge werden alle dem S20-Master zugeordnet.

  In Kombination mit der oben beschriebenen "X-Funktion" lassen sich damit unterschiedliche Gruppen von Modulen in einfacher Weise behandeln.

- **Modul-Eintrag**: Zum Deaktivieren wahlfreier Module unterhalb eines Slaves werden

  das Schlüsselwort `ModuleDeactPosi=` und die 1- bis 2-stelligen Angabe der Modul-Position in dezimaler Form angegeben.

  Beispiel:

  ```
  ModuleDeactPosi= 3
  ```

  Der zulässige Wertebereich ist **1…64**.

  Der Wert ´**0**´ ist ebenfalls zulässig, aber unwirksam.

- **Kommentare**: Als Kommentarzeichen ist das Semikolon (´**;**´) definiert. Ab der Position des Semikolons bis zum Ende der Zeile wird der Text nicht ausgewertet.

Größe der Datei "UserDefs.cfg"

Entsprechend der gegebenen Größe der Datei wird zu deren Auswertung Speicher allokiert. Die zulässige Größe ist daher praktisch nicht begrenzt.

Beispiel der Datei "UserDefs.cfg"

```
;---------- Master.0 ----------
[S20IO.Master.0]
ModuleDeactPosi = 3
ModuleDeactPosi = 7
ModuleDeactPosi = 15
ModuleDeactPosi = 16
;------------------------------
```

12.2.6 S20-Module konfigurieren

Allgemeines

Klicken Sie doppelt im Projektexplorer auf das S20-Modul, das Sie konfigurieren möchten.

Der S20-Modul-Editor enthält drei Registerkarten, die Sie durch Anklicken öffnen können.

Register:

- Registerkarte 'Benutzerdefinierte Parameteraufe 315
- Registerkarte 'E/A-Abbild' auf Seite 315

- Registerkarte 'Information' auf Seite 316

Registerkarte 'Benutzerdefinierte Parameter'

Die Registerkarte ermöglicht die Voreinstellung von Modul-Parametern und wird nur für Module angezeigt, bei denen diese Voreinstellung erforderlich oder möglich ist.

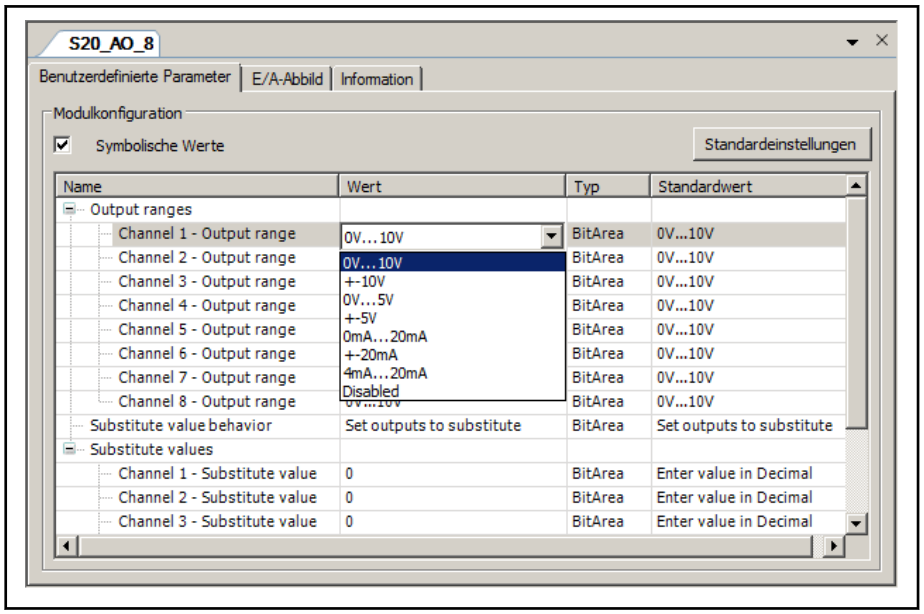


Abb. 12-13: S20-Modul: Benutzerdefinierte Parameter

Modulkonfiguration

Im Beispiel handelt es sich um ein Modul mit acht Analogausgängen.

Für Analogausgang 1 wird im Bild der Wertebereich ausgewählt, in dem die Ausgangsinformation zu erwarten ist.

Für jeden "Wert" wird dazu ein Auswahlfenster, welches auf einem Aufzählungsdatentyp basiert, angeboten.

Symbolische Werte: entscheidet, ob der Wert als Klartext ("0V...10V") oder als Ganzzahl ("0") angezeigt wird.

Standardeinstellungen: setzt alle Parameter zurück auf deren Initialwert.

Registerkarte 'E/A-Abbild'

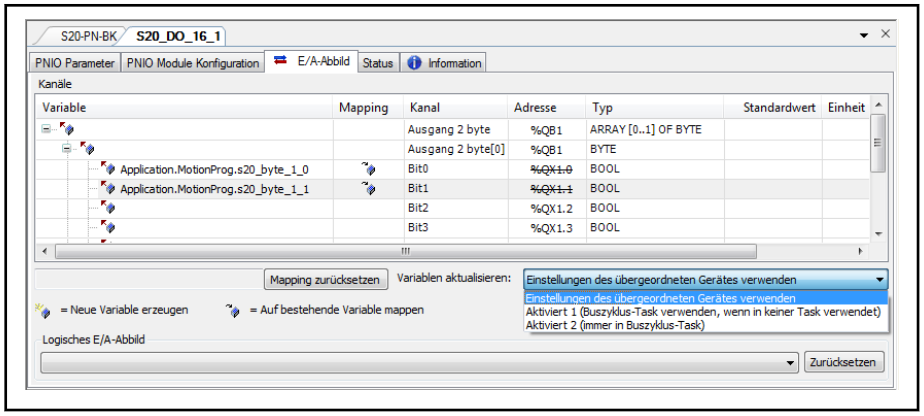


Abb. 12-14: S20-Modul: Registerkarte 'E/A-Abbild'

Die Registerkarte dient der Zuordnung der S20-Modul-Ein- und Ausgänge zu Variablen, die als lokale oder globale Variablen in den einzelnen POUs verwendet werden können.

Im Online-Modus wird der aktuelle Wert der Variablen angezeigt.

Mapping zurücksetzen: Über diese Schaltfläche können die Mapping-Einstellungen auf die durch die Gerätebeschreibungsdatei definierten Standardwerte zurückgesetzt werden.

Variablen aktualisieren: Festlegung für das Modul , ob die E/A-Variablen in der Buszyklus-Task aktualisiert werden.

Der Default-Wert ist in der Gerätebeschreibung definiert.

- **Einstellungen des übergeordneten Geräts verwenden:** Aktualisierung gemäß den Einstellungen des übergeordneten Geräts.
- **Aktiviert 1 (Buszyklus-Task verwenden, wenn in keiner Task verwendet):** Die E/A-Variablen werden in der Buszyklus-Task aktualisiert, falls sie in keiner anderen Task verwendet werden
- **Aktiviert 2 (immer im Buszyklus-Task):** Alle Variablen werden in jedem Zyklus der Buszyklus-Task aktualisiert, egal ob sie verwendet werden und ob sie auf einen Ein- oder Ausgangskanal abgebildet sind.

Die Zuordnung ist unter Mapping der Feldbus-Ein- und Ausgänge auf Seite 19 beschrieben.

Registerkarte 'Information'

Die Registerkarte zeigt allgemeine Informationen an, die aus der Gerätebeschreibungsdatei stammen:

- Name
- Hersteller
- Kategorien
- Version
- Bestellnummer
- Beschreibung
- Gerätebild (wenn vorhanden)

12.2.7 S20-Module, Bus scannen

Voraussetzung Der Befehl "Bus scannen..." steht erst dann zur Verfügung, wenn zuvor die SPS Applikation geladen wurde.

Funktion Der Dialog "Bus scannen" zeigt im linken Fenster die Geräte an, die bereits konfiguriert sind, und im rechten Fenster die Geräte, die nach dem Scannen am Bus gefunden wurden.

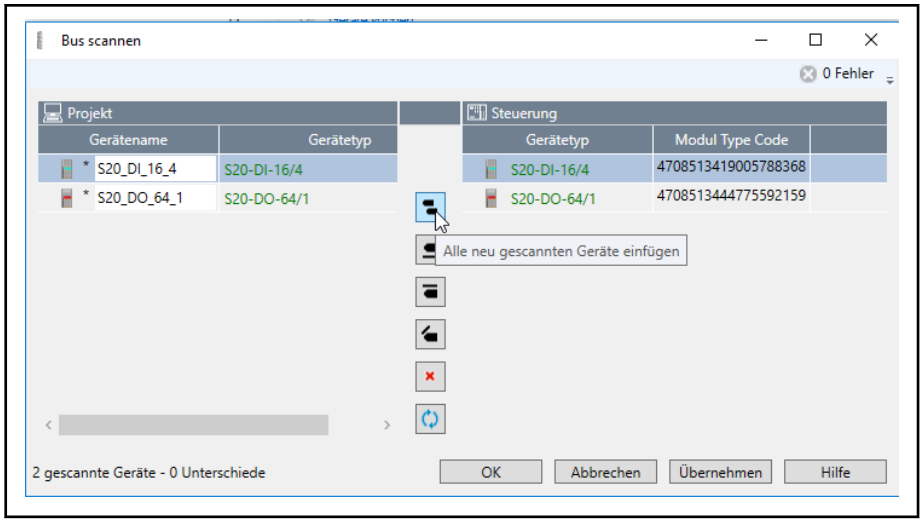


Abb. 12-15: S20IO - Dialog "Bus scannen"

Über die Befehlsleiste in der Dialogmitte können die gefundenen Geräte (rechts) in die Projektkonfiguration (links) übernommen werden. Dabei können entweder alle neuen Geräte eingefügt werden, oder einzelne Geräte markiert und bearbeitet werden (siehe Tooltip-Unterstützung der Buttons).

Aufruf Im Projektexplorer über das Kontextmenü des S20IO Masters

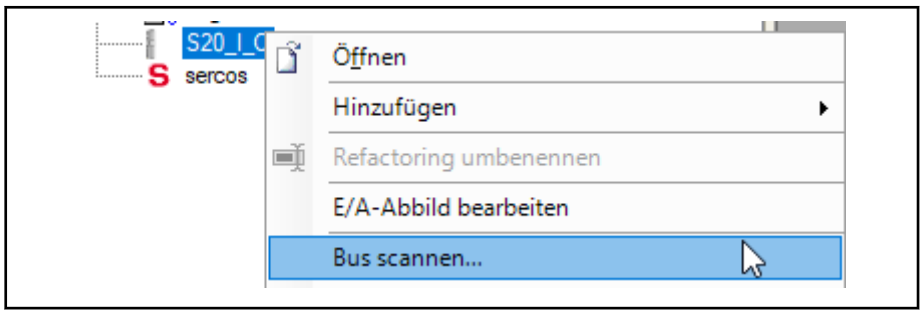


Abb. 12-16: S20IO - "Bus scannen" Aufruf im Kontextmenü des Projektexplorers.

13 Gebäudeautomatisierung

13.1 Allgemein

Einführung und Überblick

Bosch Rexroth bietet verschiedene Dienste und Schnittstellen für Anwendungen der Gebäudeautomation an.

Das folgende Kapitel gibt einen Überblick über die unterstützten Systeme.

Systemübersicht

- **BACnet** (ab IndraWorks 15VRS)

  BACnet ist ein Netzwerkprotokoll zur herstellerübergreifenden Anbindung an eine Leitebene bzw. zur Kommunikation zwischen BACnet Geräten und BACNet Controllern.

  Eine weitere Funktion ist die Querkommunikation zwischen unterschiedlichen BACnet Systemen zum Datenaustausch.

- **KNX**

  KNX ist ein Standard zur Gebäudeautomation in der Feldebene.

  KNX sammelt und verwaltet die Daten bzw. Schaltzustände, von Tastern, Schaltern, Präsenzsensoren, Helligkeitssensoren, Temperatur- oder Luftgüte-Sensoren.

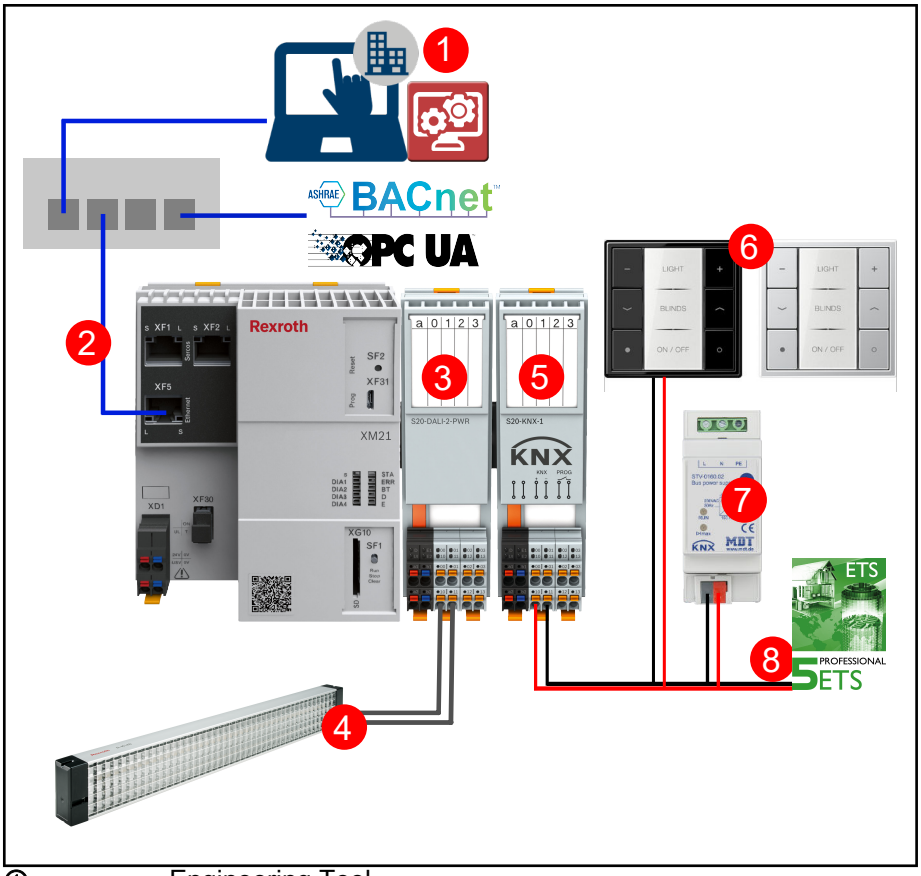
  Über die KNX Applikation auf einer IndraControl XM-Steuerung können viele Aktivitäten gesteuert werden, beispielsweise:

  – Ansteuerung von Aktoren
  – Steuerung von Leuchten
  – Kommandierung von Rolläden
  – Erfassung von Zustandsmeldern (z. B. Zustand-LED am Taster)

  Mit der XM-Steuerung können die Information zu anderen BACnet-Controllern, oder an andere Systeme weitergeben werden z.B. an DALI-Leuchtmittel.

- **DALI**

  "Digital Addressable Lighting Interface" zur Steuerung von Vorschaltgeräten, oder Dimmern.



① Engineering Tool
② Ethernet-Anschluss konfigurierbar (Engineeringport, BACnet, OPC-UA usw.)
③ S20 DALI-PWR-2 Modul (DALI Netzteil integriert)
④ DALI Geräte
⑤ S20 KNX-1 Modul (enthält die ETS / KNX Konfiguration)
⑥ KNX Geräte, z.B. Taster, Präsenzsensoren, Rollladen, Fensterkontakte
⑦ KNX Netzteil, versorgt die KNX Geräte mit Bus-Spannung (erforderlich)
⑧ KNX Programmiertool, z.B. ETS 5 (zur Konfiguration des S20 KNX-1 Moduls)

Abb. 13-1: Übersicht Gebäudeautomation Komponenten

Rexroth Komponenten zur Anbindung an Systeme der Gebäudeautomatisierung

System	Komponenten	Funktion
Dali	S20-DALI-2-PWR	2 Kanal DALI-Master Siehe Dokumentation im Rexroth Medienverzeichnis DOK-CONTRL-S20*DALI*2*-DA
	RIH_DALI	SPS Bibliothek mit Funktionsbausteinen zur Kommunikation mit DALI Geräten mit Hilfe dem S20-DALI-2-PWR.

KNX	S20-KNX-1	1 Kanal KNX Master Siehe Dokumentation im Rexroth Medienverzeichnis DOK-CONTRL-S20*KNX*1**-DA
	RIH_IndraControl_S20_KNX	SPS Bibliothek mit Funktionsbausteinen zur Kommunikation mit KNX Geräten mit Hilfe dem S20 KNX-1 Modul
BACnet	Package für das Engineering Tool, Konfigurator unterstützte Projektierung	Siehe Dokumentation im Rexroth Medienverzeichnis: DOK-IWORKS-FB******V15-AP01-DE-P R911393282
OPC-UA	Package für Steuerung	Siehe OPC-UA Dokumentation im Rexroth Medienverzeichnis: DOK-IWORKS-OPC*UA*****-APRS-DE-P R911379308

Tab. 13-1: Übersicht – Verfügbare Komponenten für die Gebäudeautomation

13.2 BACnet

13.2.1 Features der BACnet Implementation

Die aktuelle BACnet Implementation beinhaltet folgende Funktionalität:

Funktionsumfang BACnet/IP Server und Client

Unterstützte BACnet-Objekte:

Name - Objekt Typ	Objekt Type ID	Objekt Typ Abkürzung	Kommentar
OBJ_DEVICE	8	DEV	Information und Eigenschaft über BACnet Gerät, existiert genau einmal je BACnet Gerät (DEV)
OBJ_ANALOG_INPUT	0	AI	Analoger Eingang (AI), Messen
OBJ_ANALOG_OUTPUT	1	AO	Analoger Ausgang (AO), Stellen
OBJ_ANALOG_VALUE	2	AV	Analoger Wert (AV), Virtueller Analogwert, z.B. Rechenergebnis
OBJ_AVERAGING	18	AVG	Mittelwert (AVG)
OBJ_BINARY_INPUT	3	BI	Binärer Eingang (BI), Melden
OBJ_BINARY_OUTPUT	4	BO	Binärer Ausgang (BO), Schalten
OBJ_BINARY_VALUE	5	BV	Binärzustand (BV), Virtueller Binärwert aus logischer Verknüpfung
OBJ_CALENDAR	6	CAL	Kalender (CAL)
OBJ_COMMAND	7	CMD	Befehl, Ausführen von vordefinierten Aktionen (CMD)
OBJ_EVENT_ENROLLMENT	9	EE	Ereignisregistrierung, Ereigniskategorie, Reaktion auf Ereignisse (EE)
OBJ_FILE	10	FIL	Datei, für externe Programme oder Datensicherung
OBJ_GROUP	11	GRP	Gruppe, gruppieren beliebiger Objekte für eine Grafik oder Protokoll von einem Gerät
OBJ_GLOBAL_GROUP	26	GGRP	Globale Gruppe in einem BACnet Netzwerk

Name - Objekt Typ	Objekt Type ID	Objekt Typ Abkürzung	Kommentar
OBJ_LOOP	12	LP	Regler PID
OBJ_MULTISTATE_INPUT	13	MI	Mehrstufige Eingabe
OBJ_MULTISTATE_OUTPUT	14	MO	Mehrstufige Ausgabe
OBJ_MULTISTATE_VALUE	19	MV	Mehrstufiger Zustand
OBJ_NOTIFICATION_CLASS	15	NC	Meldungsklasse, Verteilerliste für Ereignismeldungen
OBJ_PROGRAM	16	PR	Programm
OBJ_SCHEDULE	17	SCHED	Zeitprogramm, Zeitplan
OBJ_TRENDLOG	20	TLOG	Einfache Trendaufzeichnung
OBJ_TRENDLOG_MULTIPLE	27	TLOGM	Mehrfache Trendaufzeichnung
OBJ_LIFE_SAFETY_POINT	21	LSP	Gefahrenmeldeobjekt
OBJ_LIFE_SAFETY_ZONE	22	LSZ	Sicherheitsbereich, Zusammenfassung von Gefahrenmeldeobjekten
OBJ_ACCUMULATOR	23	ACC	Zählereingang, akkumuliert
OBJ_PULSE_CONVERTER	24	PC	Zähler
OBJ_EVENTLOG	25	ELOG	Ereignis-Log
OBJ_LOAD_CONTROL	28	LC	Laststeuerung
OBJ_STRUCTURED_VIEW	29	SV	Struktursicht
OBJ_ACCESS_DOOR	30	AD	Tür-Zutrittskontrolle
OBJ_LIGHTING_OUTPUT	31	LO	Licht Ausgang, Lichtsteuerung
OBJ_ACCESS_CREDENTIAL	32	AC	Zugangsberechtigung, Authentifizierungs-Daten (AC)
OBJ_ACCESS_POINT	33	AP	Zugangspunkt
OBJ_ACCESS_RIGHTS	34	AR	Zugangsrechte, Bedingung, Zeitraum
OBJ_ACCESS_USER	35	AU	Zugang-Benutzer und dessen Rechte
OBJ_ACCESS_ZONE	36	AZ	Zugangs-Zone
OBJ_CREDENTIAL_DATA_INPUT	37	CD	Berechtigung Daten Eingang
OBJ_NETWORK_SECURITY	38		Netzwerk Sicherheit
OBJ_BITSTRING_VALUE	39	BSV	Bit String
OBJ_CHARACTERSTRING_VALUE	40	CSV	Charakter String / Text
OBJ_DATE_PATTERN_VALUE	41	DPV	Zusammenfassung von Wiederkehrenden Datumsangaben
OBJ_DATE_VALUE	42	DV	Datum
OBJ_DATETIME_PATTERN_VALUE	43	DTPV	Zusammenfassung von Wiederkehrenden Datumsangaben inklusive Uhrzeit
OBJ_DATETIME_VALUE	44	DTV	Datum inklusive Uhrzeit
OBJ_INTEGER_VALUE	45	IV	Integer Wert
OBJ_LARGE_ANALOG_VALUE	46	LAV	Float Wert
OBJ_OCTETSTRING_VALUE	47	OSV	Oktett String
OBJ_POSITIVE_INTEGER_VALUE	48	PIV	Integer Wert Positiv

Name - Objekt Typ	Objekt Type ID	Objekt Typ Abkürzung	Kommentar
OBJ_TIME_PATTERN_VALUE	49	TPV	Zusammenfassung von Wiederkehrende Uhrzeit
OBJ_TIME_VALUE	50	TV	Uhrzeit
_OBJ_MAX_SIZE	16#7FFFFFFF		

Tab. 13-2:

Beispiel in einer SPS-Applikation für die Adressierung in Verbindung mit einem Client Funktionsbaustein für den Input **objType**: `BACnet.CmpBACnet.IEC_BACNET_OBJECT_TYPE.OBJ_ANALOG_INPUT`

Bei einem BACnet Server bildet sich das Objekt als Element unter den BACnet Server Knoten ab, inklusive seiner Eigenschaften. Siehe Kap. "BACnet Server und Objekte anlegen und BACnet Server starten" auf Seite 353.

☞ Benutzerdefinierte Objekte oder benutzerdefinierte Eigenschaften können nicht erstellt werden.

Funktionsumfang BACnet/IP Client

Unterstützte BACnet-Objekte:

Funktion / Funktionsbaustein	Kommentar
BACnetClientAcknowledgeAlarm	Alarm / Bedienerquittierung
BACnetClientCreateObject	Objekt erzeugen/anlegen
BACnetClientDeleteObject	Objekt löschen
BACnetClientDeviceCommControl	Aktivieren/Deaktivieren der Kommunikation eines Gerätes
BACnetClientGetAlarmSummary	Liste aktueller Alarme
BACnetClientGetEnrollmentSummary	Liste der Objekte die für Alarme konfiguriert sind
BACnetClientEventInfo	Liste aller aktuellen Alarme und Events
BACnetClientReadStreamFile	Datei lesen
BACnetClientWriteStreamFile	Datei schreiben
BACnetClientReintializeDevice	Neustart eines Gerätes
BACnetClientTimeSynchronization	Uhrzeit Synchronisation
BACnetClientUTCTimeSynchronization	UTC Uhrzeit Synchronisation
BACnetClientAddListElement	Objekt einer Liste hinzufügen
BACnetClientRemoveListElement	Objekt aus Liste entfernen
BACnetClientReadProperty	Objekt Wert oder Eigenschaft Lesen
BACnetClientReadPropertyMultiple	mehrere Objekteigenschaften lesen
BACnetClientReadRange	Bereich eines Objektes lesen z.B. bei Trendaufzeichnung
BACnetClientWriteProperty	Objekt Wert oder Eigenschaft schreiben
BACnetClientWritePropertyMultiple	mehrere Objekteigenschaften schreiben

Funktion / Funktionsbaustein	Kommentar
BACnetClientSubscribeCOV	Anmelden auf Änderung des PresentValue / StatusFlags eines Objektes
BACnetClientSubscribeCOVProperty	Anmelden auf Änderung auf Eigenschaften eines Objektes

Tab. 13-3:



Für Details zu den unterstützen Client-Funktionen und Code Beispiele finden Sie in der BACnet Bibliothek unter Programmierbeispiele

BACnet Standard

Die BACnet/IP Integration implementiert den ANSI/ASHRAE Standard 135-2012 Protokoll Revision 14 und orientiert sich an der Norm AMEV AS-A und AS-B.

Der Funktionsumfang entspricht dem Geräteprofil B-ASC (in Vorbereitung: B-BC).

Zielsysteme

Die BACnet/IP Implementation kann mit folgenden Systemen umgesetzt werden:

- MLC (ab 15V02)

Steuerungs-Hardware

Folgende Steuerungs-Hardware wird unterstützt über die Engineering-Port Schnittstelle:

- XM2
- XM4

Technische Daten

Eigenschaft	Wert
BACnet Adressierung	• IP-Adresse des Engineering-Ports der Steuerung. • UDP-Port: 47808 *) (Default). • Netzwerknummer: 1 (Default).
Max. Anzahl von BACnet Objekten	Die Anzahl der Objekte wird durch Speicher und Laufzeit limitiert. Ein BACnet Objekt benötigt ungefähr 12 KB RAM. Der zur Verfügung stehende Speicher wird auch durch das SPS-Programm belegt (inkl. der sonstigen Feldbus-Konfiguration).

Eigenschaft	Wert
Aktualisierungsrate des BACnet Stacks	Der BACnet Stack wird von einer internen, freilaufenden Systemtask aufgerufen (d.h. die Buszyklus-Task hat hier keine Bedeutung). Die SPS-Schnittstellen zum Lesen / Schreiben der BACnet Objekte aus der SPS können unabhängig davon aufgerufen werden.
SPS Online-Change	**>> Nicht möglich! <<** In der SPS-Schnittstelle zum BACnet Stack werden mögliche Online-Change Effekte (z.B. Verschiebung von Pointer) nicht unterstützt! Daher sollte ein SPS Online-Change nicht ausgeführt werden, da sonst inkonsistente Daten entstehen können.

*) Der UDP-Port 47808 (= 0xBAC0) muss von jedem BACnet/IP Gerät unterstützt werden.

⚠ VORSICHT **SPS Online-Change nicht möglich!**

SPS Programm Download bei Code Änderungen durchführen.

13.2.2 BACnet Grundlagen

Allgemeines

BACnet ist ein standardisiertes Datenkommunikationsprotokoll für Gebäudeautomatisierungs- und Gebäudesteuerungsnetzwerke.

BACnet wurde durch die ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers) entwickelt und im Jahr 1995 in der ANSI/ASHRAE-Norm 135 standardisiert.

Seit 2003 gilt die BACnet ISO-Norm 16484-5.

BACnet verwendet einen objektorientierten Ansatz, d. h. jedes Element wird als Objekt betrachtet. Die Objekte werden durch Objekteigenschaften ("Properties") beschrieben. Es können herstellerspezifische Objekte, Eigenschaften oder Dienste definiert werden.

BACnet basiert auf dem Client/Server Modell. Der Server stellt die Daten bereit. Der Client ist der Nutzer, der auf die Daten zugreift. Die Kommunikation erfolgt abhängig vom Dienst entweder gezielt (Punkt-zu-Punkt) oder an alle Teilnehmer (Broadcast).

Weiterführende Informationen

BACnet PICS ANNEX A – PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT(PICS)	R911401885 Reference book

Begriffe und Abkürzungen

ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Engineers (Interessenvertretung der Ingenieure der Heizungs-, Lüftungs-, Klimatechnik der USA)
B/IP	Abkürzung für BACnet/IP
BACnet	Building Automation and Control networks (BACnet® ist Schutzmarke der ASHRAE)
BBMD	BACnet/IP Broadcast Management Device Bei zwei oder mehreren IP-Subnetzen muss in jedem Subnetz ein BBMD-Gerät vorhanden sein, das die BACnet Broadcasts in die anderen Subnetze weiterleitet. Die Konfiguration im BBMD erfolgt anhand einer Broadcast Distribution Table (BDT). Man spricht von einem asymmetrischen BBMD, wenn das BBMD in einem Subnetz nur das BBMD einer Leitstation kennt, aber das BBMD in der Leitstation alle BBMD in allen Subnetzen kennt.
BIBB	BACnet Interoperability Building Block BIBBs sind eine Ansammlung von einer oder mehrerer BACnet Funktionalitäten und Diensten. Zwei Geräte funktionieren miteinander, wenn das eine die A-Seite des BIBB (Client) und das andere Gerät die B-Seite des BIBB (Server) unterstützt.
BTL	BACnet Testing Laboratories Prüflabor für BACnet Konformitätstests
Client / Server	• Der **Server** ist der Bereitsteller von Daten. Jedes BACnet Gerät ist ein Server, da mindestens das "Device" Objekt unterstützt werden muss. Ein Gerät, das nur das Device Objekt unterstützt und sonst keine weiteren Objekte, wird üblicherweise als Client bezeichnet. • Der **Client** ist der Nutzer, der auf die Daten zugreift. Software-Tools (Clients) werden typischerweise auch als BACnet Browser bezeichnet. Auch Client Tools haben ein Device Objekt und eine entsprechende "Device-ID" im BACnet Netzwerk.
COV	Change Of Value
COV Notification	Automatische Benachrichtigung bei Wertänderung
Device-ID	Die Device-ID ist die Instanznummer des "Device" Objekts. Alle BACnet Geräte im gesamten BACnet Netzwerk brauchen eine eindeutige Device-ID. Die Konfiguration erfolgt herstellerspezifisch z.B. durch DIP-Schalter am Gerät, WebServer, proprietäre Tools, etc. Eine Remote Vergabe über einen Client ist eine optionale Funktion und eher selten genutzt.

EDE-Datei	Engineering Data Exchange Tatsächliche Konfiguration eines Gerätes (projektierte Objekte) in Form einer spezifizierten Excel-Tabelle (EDE-Format), z. B. für den Austausch von Projektkonfigurationen (Ex-/Import).
FD	Foreign Device Ein Foreign Device ist ein BACnet Gerät (typischerweise ein Client), das nicht zu einem BACnet IP-Netzwerk gehört, sich aber bei einem BBMD Gerät im BACnet IP-Netzwerk registrieren kann, um Teilnehmer dieses BACnet IP-Netzwerkes zu werden (z.B. für Remote-Zugriffe).
Geräte-Profil	Ein Gerät muss eine bestimmte Menge an BIBBs unterstützen um ein BACnet Geräte-Profil zu erfüllen. Beispiel für ein Profil: B-BC (BACnet Building Controller)
GLT	Gebäudeleittechnik
PICS	Protocol Implementation Conformance Statement Das Protocol Implementation Conformance Statement ist die Selbstdeklaration des Herstellers, welche die implementierten BACnet Funktionen eines Gerätes beschreibt (z.B. die unterstützen BIBBs, Objekttypen, Eigenschaften, ...)
API	**Application Programming Interface**: Schnittstelle zur Programmierung von Anwendungen
PDU	**Protocol Data Unit**: Kompletter Datensatz, der als Einheit zwischen Peer-Entitäten eines Netzwerks bereitgestellt wird und der Steuerinformationen, Adressinformationen oder Daten enthalten kann.
APDU	**Application Protocol Data Unit**: Kombinierter Kommando-/Datenblock des Kommunikationsprotokolls

Tab. 13-4: BACnet Begriffe und Abkürzungen

Netzwerke und Topologien

Der BACnet Standard basiert auf den ISO/OSI Schichtenmodell für Kommunikationsnetzwerke. Es können BACnet Daten über mehrere Medien mit unterschiedlichen Zugriffsverfahren ausgetauscht werden.

Für Schicht 1 und 2 sind folgende Alternativen definiert:

- BACnet/IP (UDP/IP)
- BACnet MS/TP (Master-Slave/Token-Passing, RS-485)
- BACnet PTP (Point-To-Point, RS-232)
- BACnet LonTalk
- BACnet ARCNET
- BACnet Ethernet

Folgende Abbildung zeigt ein Netzwerk Beispiel:

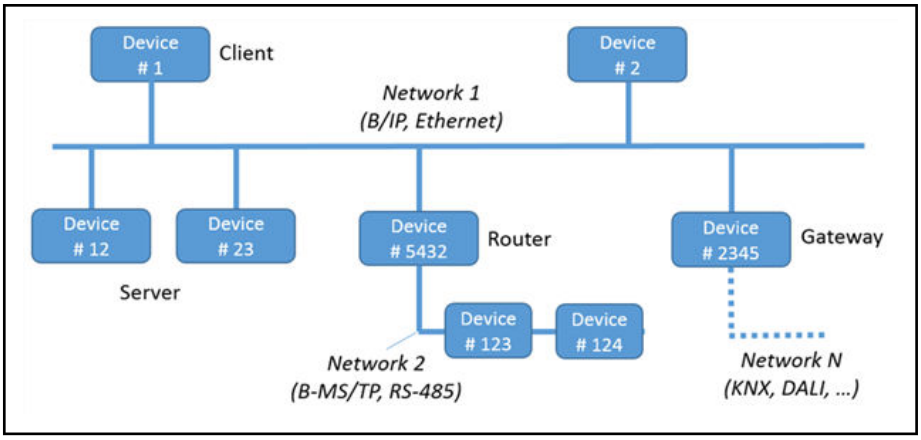


Abb. 13-2: BACnet Netzwerk Beispiel

Allgemeine Hinweise zu BACnet Netzwerken:

- Ein BACnet Router verbindet BACnet Subnetzwerke, die aus unterschiedlichen Medien bestehen.
- Ein BACnet Client hat Zugriff auch auf die BACnet Geräte in den BACnet Subnetzwerken (da Router, z.B. B/IP ⇔ B-MS/TP, auch Broadcasts weiterleiten).
- Bei Zusammenschaltung mehrerer BACnet Teilnetzwerke mit unterschiedlichen Medien ist pro Teilnetzwerk eine eindeutige BACnet Netzwerknummer erforderlich (bis zu 65.535 Teilnetzwerke möglich).
- Ein BACnet Gateway verbindet ein BACnet Netzwerk mit anderen Netzwerktechniken bzw. Geräten mit anderen Protokollen. Typischerweise ist ein BACnet Gateway ein BACnet Server, dessen Objekte die Daten der Geräte der zweiten Netzwerkschnittstelle abbilden.
- Jedes BACnet Gerät benötigt eine eindeutige Device-ID (= Instanznummer des Device-Objekts des Gerätes). Die Konfiguration erfolgt herstellerspezifisch z.B. durch DIP-Schalter am Gerät, WebServer, proprietäre Tools, etc.

Hinweise zu BACnet/IP Netzwerken:

- BACnet/IP Netzwerke bestehen aus einem oder mehreren IP-Subnetzen, die die gleiche BACnet Netzwerknummer besitzen.
- Die Adressierung erfolgt über eine "MAC" Adresse, die nicht der Ethernet MAC Adresse entspricht, sondern eine Kombination aus IP-Adresse des Gerätes und dem verwendeten Port ist.

  Beispiel: B/IP-Adresse = 192.168.1.1:47808.

  Der Port 47808 (= 0xBAC0) ist die Defaulteinstellung und muss von allen B/IP Geräten unterstützt werden.
- Siehe auch Hinweise zu "BACnet/IP Broadcast Management Device" (BBMD) und "Foreign Device" (FD) in Begriffe und Abkürzungen, Seite 326.

Objekte und Eigenschaften

Die BACnet Spezifikation definiert mehr als 50 Standardobjekte. Es können auch herstellerspezifische Objekte und Eigenschaften definiert werden, was aber nicht von allen Geräten unterstützt wird.

Objekt Identifizierung

Jedes BACnet Objekt hat drei Identifizierungseigenschaften:

- ObjectIdentifier

  Der "ObjectIdentifier" (32 Bits) ist eine Kombination aus dem "Object-Type" (10 Bits) und der Objekt-Instanznummer (22 Bits):

  – Es kann also max. 1024 (= 2^{10}) Objekttypen geben.

  – Es kann pro Objekttyp max. 4.194.304 (= 2^{22} - 1) Instanzen geben.

  – Der "ObjectIdentifier" wird z.B. in der Form dargestellt: "AI 1"

  (d.h. "AI" ist die Abkürzung für den Objekttyp AnalogInput und "1" ist die Objekt-Instanznummer).

  – Der "ObjectIdentifier" muss in einem Gerät eindeutig sein.

- ObjectType

  (als separate Eigenschaft, obwohl der "ObjectType" auch im "ObjectIdentifier" enthalten ist)

- ObjectName

  Der "ObjectName" muss im Gerät eindeutig sein und wird benutzt um den "ObjectIdentifier" zu ermitteln (Who-Has, I-Have).

Device Objekt

Das Device Objekt ist ein spezielles BACnet Objekt. Ein BACnet Gerät darf nur ein Device Objekt haben.

Die Instanznummer des Device Objekts wird auch als Device-ID bezeichnet und muss im gesamten BACnet Netzwerk eindeutig sein.

Eigenschaften

BACnet Objekte haben verschiedene Eigenschaften ("Properties").

Der Typ eines BACnet Objekts bestimmt, welche Eigenschaften es enthält.

Eine Eigenschaft verhält sich immer gleich, unabhängig davon in welchem Objekt sie verwendet wird.

Eigenschaft		AI	AO	AV	BI	BO	BV	MI	MO	MV
Identification	ObjectIdentifier	R	R	R	R	R	R	R	R	R
	ObjectName	R	R	R	R	R	R	R	R	R
	ObjectType	R	R	R	R	R	R	R	R	R
	Description	O	O	O	O	O	O	O	O	O
	DeviceType	O	O	-	O	O	-	O	O	-
Status	PresentValue	R	W	R	R	W	R	R	W	R
	StatusFlags	R	R	R	R	R	R	R	R	R
	EventState	R	R	R	R	R	R	R	R	R
	Reliability	O	O	O	O	O	O	O	O	O
	OutOfService	R	R	R	R	R	R	R	R	R
Analog Properties	UpdateInterval	O	-	-	-	-	-	-	-	-
	Units	R	R	R	-	-	-	-	-	-
	MinPresValue	O	O	-	-	-	-	-	-	-
	MaxPresValue	O	O	-	-	-	-	-	-	-
	Resolution	O	O	-	-	-	-	-	-	-
	COVIncrement	O	O	O	-	-	-	-	-	-

Eigenschaft		AI	AO	AV	BI	BO	BV	MI	MO	MV
Binary Properties	Polarity	-	-	-	R	R	-	-	-	-
	InactiveText	-	-	-	O	O	O	-	-	-
	ActiveText	-	-	-	O	O	O	-	-	-
	ChangeOfStateTime	-	-	-	O	O	O	-	-	-
	ChangeOfStateCount	-	-	-	O	O	O	-	-	-
	TimeOfStateCountReset	-	-	-	O	O	O	-	-	-
	ElapsedActiveTime	-	-	-	O	O	O	-	-	-
	TimeOfActiveTimeReset	-	-	-	O	O	O	-	-	-
	MinimumOffTime	-	-	-	-	O	O	-	-	-
	MinimumOnTime	-	-	-	-	O	O	-	-	-
Multistate Properties	NumberOfStates	-	-	-	-	-	-	R	R	R
	StateText	-	-	-	-	-	-	O	O	O
Commandable	PriorityArray	-	R	O	-	R	O	-	R	O
	RelinquishDefault	-	R	O	-	R	O	-	R	O

Eigenschaft		AI	AO	AV	BI	BO	BV	MI	MO	MV
Alarm Properties	TimeDelay	O	O	O	O	O	O	O	O	O
	Notification-Class	O	O	O	O	O	O	O	O	O
	HighLimit	O	O	O	-	-	-	-	-	-
	LowLimit	O	O	O	-	-	-	-	-	-
	Deadband	O	O	O	-	-	-	-	-	-
	LimitEnable	O	O	O	-	-	-	-	-	-
	AlarmValue	-	-	-	O	-	O	-	-	-
	FeedbackValue	-	-	-	-	O	-	-	O	-
	AlarmValues	-	-	-	-	-	-	O	-	O
	FaultValues	-	-	-	-	-	-	O	-	O
	EventEnable	O	O	O	O	O	O	O	O	O
	AckedTransitions	O	O	O	O	O	O	O	O	O
	NotifyType	O	O	O	O	O	O	O	O	O
	EventTimeStamps	O	O	O	O	O	O	O	O	O
	EventMessageTexts	O	O	O	O	O	O	O	O	O

Tab. 13-5: Übersicht Eigenschaften von wichtigen Objekten

R = Required (muss vorhanden sein, muss lesbar sein).

W = Writeable (muss vorhanden sein, muss lesbar sein, muss schreibbar sein).

O = Optional (kann vorhanden sein).

Eigenschaft	Datentyp	Beschreibung
AckedTransitions	`STRING`	Bestätigte Zustandsänderungen, '1' = Event bisher nicht aufgetreten • `TO_OFFNORMAL` • `TO_FAULT` • `TO_NORMAL`
EventEnable	`STRING`	Ereignis-Meldungen aktiv, '1' = aktiviert • `TO_OFFNORMAL` • `TO_FAULT` • `TO_NORMAL`
EventMessage-Texts	`ARRAY [0..2] OF WSTRING(255)`	Ereignis-Meldungstexte • `TO_OFFNORMAL` • `TO_FAULT` • `TO_NORMAL`

Eigenschaft	Datentyp	Beschreibung
LimitEnable	STRING	Grenzwerte aktiv, '1' = aktiv • LowLimit • HighLimit
StatusFlags	STRING	Zustandsangaben, '1' = aktiv • IN_ALARM • FAULT • OVERRIDDEN • OUT_OF_SERVICE
TimeDelay	IEC_BACNET_UNSIGNED	Meldungsverzögerung (in Sekunden)

Tab. 13-6: Beschreibung von ausgewählten Eigenschaften (mit dem Datentyp in der Schnittstelle zur SPS)

EventState Die EventState Eigenschaft wird von vielen Objekten unterstützt und beschreibt den allgemeinen Zustand des Objekts, z.B. ob ein Alarm ("OffNormal") vorliegt oder ob die Zuverlässigkeit des Objektwertes einen Fehler ("Fault") anzeigt.

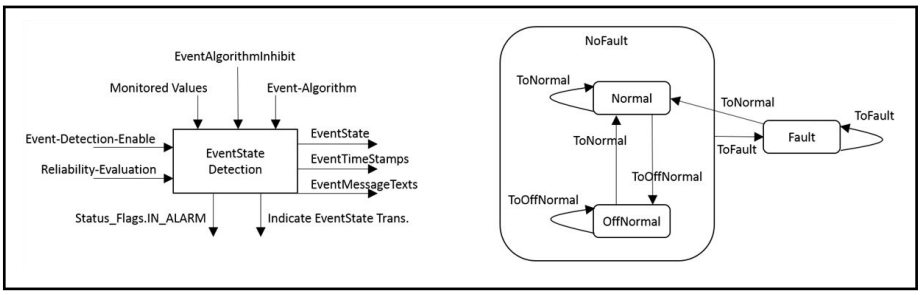


Abb. 13-3: EventState Abhängigkeiten (links) und Zustandsmaschine (rechts)

Dienste

Übersicht

Im BACnet Standard sind mehr als 38 Dienste definiert, die den Datenaustausch zwischen BACnet Geräten beschreiben.

Die Dienste werden in 5 Kategorien unterschieden:

Objektzugriff	• Eigenschaften lesen / schreiben • Objekte anlegen / löschen • Listenverarbeitung
Dateitransfer	• Lesen / Schreiben von Dateien
Alarm und Event Funktionen	• Abonnieren, automatische Benachrichtigung bei Wertänderungen (COV) • Alarm- und Ereignisnachrichten • Sicherheitssysteme (Life-Safety)

Remote Device Management	• Uhrensynchronisation • Geräte neu starten • Dynamic Device Binding (DDB): Geräte per Broadcast ermitteln (Who-Is, I-Am) • Dynamic Object Binding (DOB): Objekte dynamisch ermitteln (Who-Has, I-Have)
Virtual Terminal	• Öffnen / Schließen / Datenübertragung an eine virtuelle Schnittstelle des Gerätes

Tab. 13-7: BACnet Kommunikationsdienste (Beispiele)

Änderungsdienst (COV Notification)

Das "Change Of Value" (COV) Verfahren ist eine ereignisgesteuerte Benachrichtigung, d. h. der Server benachrichtigt den Client über Änderungen von Werten.

Hier wird unterschieden in:

- COV: Wertänderung des "Present_Value" (aktueller Wert des Objekts)
- COVproperty: Wertänderung von einer anderen Eigenschaft

Ablauf:

- Clients können die COV-Benachrichtigungen abonnieren (SubscribeCOV / UnsubscribeCOV).
- Die Benachrichtigungen können eine "Time-To-Live" haben. Wenn die "Time-To-Live" auf unendlich eingestellt ist, muss die Subscription aktiv zurück genommen werden (Resubscribe).
- Der Server benachrichtigt die Clients über Wertänderungen (COVnotification).

BACnet COV Notification Beispiel

Das Beispiel zeigt ein einfaches Programm zur Auswertung einer COV Notifikation. Als BACnet Server dient eine IndraControl XM Steuerung mit einem BACnet-Objekt "Analog value" und eine weitere IndraControl XM Steuerung, die als BACnet Client zum abonnieren und abholen der COV Notifikation dient. Die einzelnen Programm-Schritte im unten gezeigten Code-Beispiel können bei Bedarf erweitert und verändert werden.

Die Vorgehensweise zur Initialisierung des BACnet Clients ist identisch zur der Initialisierung des Basis BACnet Servers.

Das Programm `COV` des BACnet Clients enthält `CASE` Strukturen, welche für die Initialisierung und Bearbeitung der COV Notification zuständig sind. Im `COV` Programm wird Funktionsbaustein `FB_SendCOVnotification` auserhalb der CASE Anweisung zyklisch bearbeitet.

Mögliche Fehlerbehandlungen sind im Code als Kommentar enthalten, müssen bei Verwendung individuell implementiert werden.

Im CASE Schritt "run" erfolgt die eigentliche Abarbeitung in folgender Reihenfolge:

1. Die COV Notifikation wird abonniert
2. Anschließend wird der Callback ausgeführt.

   Im unten gezeigten Beispiel für eine bestimmte Zeit (60 Sekunden).
3. Nachdem die COV Notification einen Wert gesendet hat oder nachdem die vorgegebene Zeit abgelaufen ist, muss das Abonnement wieder abgemeldet werden und der Callback muss gestoppt werden.

Beim Abonnieren wird angegeben, für welchen Zeitraum das Abonnement gelten soll. Mit der Vorgabe des Werts "0" am Funktionsbaustein-Eingang `lifetime`, ist es möglich, den Zeitraum für das Abonnement auf unendlich Laufzeit zu konfigurieren. Ob eine unendliche Laufzeit sinnvoll ist, muss applikationsabhängig entscheiden werden. Ein Aspekt ist beispielsweise eine zyklische Überprüfung, ob BACnet Objekt noch existiert. In unten gezeigten Beispiel gilt das Abonnement für 3600 Sekunden. Spätestens nach dieser Zeit muss das Abonnement erneuert werden.

Im CASE Schritt "`finish`" kann ausgewählt werden, ob der Ablauf wieder erneut ausgeführt werden soll, oder ob er beendet werden soll. Die Wiederholung des Ablaufs wird über die Variable "`bAgain`" gesteuert. Wichtig ist, dass die Programmteile außerhalb der CASE Schleifen immer durchlaufen werden. Außerhalb der CASE Schleifen werden die eingehenden Nachrichten empfangen und quittiert. Über die Variable "`FB_SendCOVnotification.iState`" kann die Abarbeitung überwacht werden.

Voraussetzung:

Es existiert ein BACnet Server mit einem BACnet Objekt `BACnet_Analog_Value`*, das einen Wert liefert, der für den BACnet Client für das abonnieren der COV Notifikation verwendet wird.*

1. Erstellen Sie ein neues Projekt, das einen BACnet Server enthält, der als BACnet Client verwendet wird.

   Siehe Kap. 13.2.4 "BACnet/IP Client konfigurieren" auf Seite 364.

2. erstellen Sie die Enumeration `TestState`, wie folgt:

Programm:

```
TYPE TestState :
(
    once := 0,    // is passed through only once
    init,            // initialization
    start,           // start settings
    run,             // Test run
    finish,          // Conclusion settings
    end,             // End
    error,           // by error
    stop             // stopped
);
END_TYPE
```

3. Erstellen Sie für die COV Notifikation den Funktionsbaustein `FB_CB_CONF_COV_NOTIFICATION`:

Programm:

```
FUNCTION_BLOCK FB_CB_CONF_COV_NOTIFICATION IMPLEMENTS BACnet.IBACnetEventConsumer
VAR_INPUT
  ResetOutputs        : BOOL := FALSE;
END_VAR
VAR_OUTPUT
  EvtCnt              : UINT := 0; // call count
  LastEvtValid        : BOOL := FALSE;
  LastCovNotifyInfo   : BACnet.CmpBACnet.IEC_BACNET_COV_NOTIF_INFO;
  LastPropertyValue   : BACnet.CmpBACnet.IEC_BACNET_PROPERTY_VALUE;
  LastRealValue       : REAL := 0;
END_VAR
VAR
  ResetOutputs_old    : BOOL := FALSE;
  prEvtConfCovNotify  : POINTER TO BACnet.CmpBACnet.EVT_BACNET_CONFCOVNOTIFICATION;
END_VAR
```

Programm:

```
(* Reset Event-Count*)
 IF(TRUE = ResetOutputs  AND  FALSE = ResetOutputs_old) THEN
   EvtCnt := 0;
   LastEvtValid := FALSE;
 ELSE
   ResetOutputs_old := ResetOutputs;
 END_IF
```

4. Fügen Sie die Callback Methode `BACnetEventCallback` zum Funktionsbaustein `FB_CB_CONF_COV_NOTIFICATION` hinzu:

Programm:

```
(* This method gets called, whenever a event occurs for which the implementing
  FB is registered via |BACnetServer.RegisterHook| or
 |BACnetServer.RegisterCallback|. Since several FBs implementing IBACnetEventConsumer
 can be registered for one event or callback the
 following rules are used for the returned BOOL:
 If the method returns FALSE, all other registered IBACnetEventConsumer objects for
 the specified Hook/Callback are further called.
 If the method return TRUE, the other registeres IBACnetEventConsumer objects are not
 called anymore. The calling order is as
 the objects were registered. All calls have to treat on their own, whether the
 EventIds are correct (see CmpBACnet.EVENTIDS) and/or
 BACnet-Object-IDs or other additional data fits the needs of the registered FB
 (i.e. the specific BACnet-Object-FB).*)

METHOD BACnetEventCallback : BOOL
VAR_INPUT
    // If only one event is possibly called, the following Ids can be ignored. If there are more
 // at least the EventId should be checked, to get the right type of pParameter.

 // ID of the occured Event. Contains the class and the event (see CmpBACnet.EVENTIDS)
    EventId    : DWORD;

    // Id of the parameter structure (see pParameter and CmpBACnet.EVENTIDS)
    usParamId    : WORD;

    // Version of the parameter structure (see pParameter and CmpBACnet.EVENTIDS)
    usVersion    : WORD;

    // Pointer to the event specific parameter, that is specified by Id
 // (see CmpBACnet.EVENTIDS and CmpBACnet, i.e. EVT_BACNET_ACKALARM)
    pParameter    : POINTER TO BYTE;
END_VAR
```

Programm:

```
// METHOD BACnetEventCallback
// NOTE: no breakpoint possible, because this event callback is called from the
// BACstack task, which is not an IEC task!

  (* Check Event *)
  IF( EventId   = BACnet.CmpBACnet.EVT_ConfCOVNotification AND
      usParamId = BACnet.CmpBACnet.EVTPARAMID_CmpBACnet_ConfCOVNotification
    ) THEN

    LastEvtValid := TRUE;
    EvtCnt := EvtCnt + 1;

    (* set event parameter *)
    prEvtConfCovNotify   := pParameter;
    LastCovNotifyInfo    := prEvtConfCovNotify^.pServiceInfo^;
    LastPropertyValue    := LastCovNotifyInfo.listOfValues^;

// Note: the buffer in 'LastPropertyContents' is stack internal,
// that means you have to copy the value here in the callback!

    (* analog object COV notificatin *)
    IF( LastCovNotifyInfo.monitoredObjectID.typ = BACnet.CmpBACnet.OBJ_ANALOG_INPUT OR
        LastCovNotifyInfo.monitoredObjectID.typ = BACnet.CmpBACnet.OBJ_ANALOG_OUTPUT OR
        LastCovNotifyInfo.monitoredObjectID.typ = BACnet.CmpBACnet.OBJ_ANALOG_VALUE
      ) THEN
```

```
      LastRealValue := BACnet.GetRealFromContents(LastPropertyValue.value);

      (* Event found Meldung *)

    ELSE
      LastRealValue := 0.0;
    END_IF

    (* event handled *)
    BACnetEventCallback := TRUE; // TRUE = no calls of other registered callbacks of this event

  ELSE // unexpected EventId
    (* not my event *)
    BACnetEventCallback := FALSE; // FALSE = call the other registered callbacks of this event
  END_IF

(* Note:  prEvtConfCovNotify^.pCBStatus^ := BACnet.CmpBACnet.IEC_BACNET_CB_STATUS.CB_STATUS_DEFAULT
    => default: the stack handles the default actions (and not the hook / callback)

   Note:  prEvtConfCovNotify^.pCBStatus^ := BACnet.CmpBACnet.IEC_BACNET_CB_STATUS.CB_STATUS_OK
    => the hook or application generates the reply, see the ((Response-)-'CbCompletion' functions,
       which are mentioned in the corresponding event structures,
       e.g. see documentation of EVT_BACNET_CONFCOVNOTIFICATION.
*)
```

5. Erstellen Sie das COV Programm COV:

Programm:

```
PROGRAM COV
VAR

    (* Test_00 - Example COV Notification *)
    Test_00 : TestState := once;


  (* COV Notification *)
  StartCOV : BOOL := FALSE;
  FB_SendCOVnotification : BACnet.BACnetClientSubscribeCOV;

  COVTargetDeviceNumber : DWORD := 0; // DeviceId of the device (BACnet-Server) at the BACnet network
  COVObjectType : DINT := 0;  // AI = 0, AO = 1, AV = 2, BI = 3, BO = 4, BV = 5
  COVObjectInstance : DWORD := 0;
  COVSubscribe : BOOL := TRUE;  // Subscribe or Un-Subscribe
  CovValue : REAL := 0.0; // for analog values, with GetRealFromContent()
  COVState : UINT := 0; // at the busy state

  (* COV Callback *)
  StartCallback : BOOL := FALSE;
  FB_CB_Conf_COV_Notification : FB_CB_CONF_COV_NOTIFICATION; // Instance of the Calback-FBs
  BACnetStatus : BACnet.CmpBACnet.IEC_BACNET_STATUS;
  EventCount : UINT := 0;

  (* Timeout Timer *)
  Timeout : TON;
  tTimeout: TIME := T#1M;
    waitTon: Ton;
    eventTon:    Ton;
    Run_case: INT;
    bAgain: BOOL := TRUE;            //if bAgain = false then reading of the value is stopp
    wfinish: WORD;
END_VAR
```

Programm:

```
IF TRUE = UserVarGlobal.BACnetReady THEN

    CASE Test_00 OF

        once: // run through only once at the beginning
        //--------------------------------------
            (* Registering the FB with your own BACnet server *)
            FB_SendCOVnotification.RegisterToServer(BACnet_Server);

            (* Registering the Callbacks with your own BACnet-Server *)
      BACnetStatus := BACnet_Server.RegisterHook(
```

```
            BACnet.CmpBACnet.IEC_BACNET_CB_TYPE.CB_CONF_COV_NOTIFICATION,
            FB_CB_Conf_COV_Notification );
    IF BACnetStatus = BACnet.CmpBACnet.IEC_BACNET_STATUS.BACNET_STATUS_OK THEN
        (* success message *)
        (* ... *)
        (* set Event-Count *)
        EventCount := FB_CB_Conf_COV_Notification.EvtCnt;
    ELSE
        (* error message *)
        (* ... *)
    END_IF


    Test_00 := init;

init: // init
//---------------------------------------
    FB_SendCOVnotification(xExecute:=FALSE);
    COVState := 0; // if busy

    COVTargetDeviceNumber := 21; // TargetDeviceNumber of the BACnet-Server of the BACnet Partner
    COVObjectInstance := 1; // Instance
    COVObjectType := 2; // 0 = AI, 1 = AO, 2 = AV, 3 = BI, 4 = BO, 5 = BV

    Run_case := 0;
    Test_00 := start;

start: // start COV
//---------------------------------------

    Test_00 := run;


run: // run
//---------------------------------------

    StartCallback := TRUE;
    waitTon(IN:= NOT (WaitTon.q), PT:= T#1S, Q=> , ET=> );

    //Test COV
    CASE Run_case OF
        0:

            IF waitTon.Q = TRUE THEN
                Run_case := Run_case + 1;
            END_IF

        1:                                  //send COV subcribe
            StartCOV := TRUE;  // FB start
            COVSubscribe := TRUE; // Subscribe
            Run_case := Run_case + 1;

        2:                                  // wait until FB done
       // see below section


        3:                                  // wait of events
            (* Start evaluation *)
            StartCallback := TRUE;
            eventTon(IN:= NOT (eventTon.Q), PT:= T#60s, Q=> , ET=> );
            IF eventTon.Q = TRUE THEN
                Run_case := Run_case + 1;
            END_IF

        4:                                  // send COV unsubcribe
            StartCOV := TRUE;                // FB start
            COVSubscribe := FALSE;           // unsubscribe
            Run_case := Run_case + 1;

        5:                                  // wait until FB done
       // see below section


        6:                                  // Test end
            (* Stopp evaluation *)
            StartCallback := FALSE;
            IF waitTon.Q = TRUE THEN
                Test_00 := finish;
                Run_case := 0;
            END_IF

    END_CASE
```

```
        finish: // finish
        //---------------------------------------
        wfinish := wfinish + 1;

        (* Stopp evaluation *)
        StartCallback := FALSE;


        IF bAgain THEN
            Test_00 := init; // Neustart
        ELSE
            Test_00 := end;
        END_IF

        end: // exit
        //---------------------------------------
            FB_SendCOVnotification(xExecute:=FALSE);    //Exit FB cleanly

            (* Un-registering the FB at the own BACnet server *)
            FB_SendCOVnotification.UnregisterFromServer();

            (* remove the callback *)
            BACnetStatus := BACnet_Server.UnregisterHook(
                     BACnet.CmpBACnet.IEC_BACNET_CB_TYPE.CB_CONF_COV_NOTIFICATION,
                     FB_CB_Conf_COV_Notification );

            IF BACnetStatus = BACnet.CmpBACnet.IEC_BACNET_STATUS.BACNET_STATUS_OK THEN
                (* success message *)
            ELSE
                (* error message *)
            END_IF

            Test_00 := stop;


        error: // error
        //---------------------------------------

            Test_00 := finish;
            bAgain := FALSE;

        stop: // exit
        //---------------------------------------

    END_CASE
    (* ################ ENDE USER ################ *)

    (* ********************************************************************** *)
      IF StartCOV THEN

        (* call Function Block *)
        FB_SendCOVnotification(
          xExecute  := TRUE,  // BOOL : Rising edge: starts the FB execution
          xAbort    := FALSE,  // BOOL : TRUE: Abort of the FB-Operation
          dwTargetDeviceNumber := COVTargetDeviceNumber,
    // DWORD : DeviceId of the addressed device in the BACnet network

    //pTargetDeviceAddress := 0,
    // [Optional:] POINTER TO IEC_BACNET_ADDRESS : Device address of the
    // addressed device in the BACnet network.

    // If an address is set here, the dwTargetDeviceNumber is ignored.
          xSubscribe := COVSubscribe,  // BOOL : TRUE: registers or renews the COV subscription.
                                       //        FALSE: delete the COV Subscription.
          xIssueConfirmedNotifications := TRUE,  //  (TRUE) "confirmed" *
                                                                          //   (FALSE) "unconfirmed" COV message
          lifetime := 3600, // IEC_BACNET_UNSIGNED: Lifetime of COV notifications in seconds
                      //                     (0 = infinite)

 // Object type to be observed
          //monitoredObjType := BACnet.CmpBACnet.IEC_BACNET_OBJECT_TYPE.OBJ_ANALOG_INPUT,
          //monitoredObjType := BACnet.CmpBACnet.IEC_BACNET_OBJECT_TYPE.OBJ_ANALOG_OUTPUT,
          //monitoredObjType := BACnet.CmpBACnet.IEC_BACNET_OBJECT_TYPE.OBJ_BINARY_INPUT,
          //monitoredObjType := BACnet.CmpBACnet.IEC_BACNET_OBJECT_TYPE.OBJ_BINARY_OUTPUT,

          monitoredObjType := COVObjectType, // or ... 0 = AI, 1 = AO, 2 = AV, 3 = BI, 4 = BO, 5 = BV
          monitoredObjInst := COVObjectInstance
   // IEC_BACNET_DWORD :  Object instance of the object to be observed

(* at SendCOV Property *)
          //monitoredPropID := 85,  // IEC_BACNET_PROPERTY_ID : PROP_PRESENT_VALUE = 85
          //monitoredPropIndex := 0,  // IEC_BACNET_SIGNED : Property-Index, which is to be observed
          //covIncrement := 0.5  // IEC_BACNET_REAL : COV increment for the COV notifications
```

```
    );

    (* Evaluation done *)
    IF (TRUE = FB_SendCOVnotification.xDone) THEN
      (* End execution of the FB *)
      FB_SendCOVnotification(xExecute := FALSE);
      (* message "success" *)

      (* finish *)
      // ? COVState := 0;  // reset busy State ?

      StartCOV := FALSE;  // finish  ?
      Run_case := Run_case + 1;
    END_IF

    (* Evaluation busy *)
    IF (TRUE = FB_SendCOVnotification.xBusy) THEN
      CASE FB_SendCOVnotification.iState OF
        0: // no evaluation, Reset-state
          COVState := COVState + 1;
        1: // Searching for the addressed device
          COVState := COVState + 10;
        2: // Sending the BACnet request
          COVState := COVState + 100;
        3: // Waiting for response to the request
          COVState := COVState + 1000;
        4: // Processing the response and setting the outputs
          COVState := COVState + 10000;
      END_CASE
    END_IF

    (* evaluation error *)
    IF (TRUE = FB_SendCOVnotification.xError) THEN
      (* End execution of the FB *)
      FB_SendCOVnotification(xExecute:= FALSE);
      (* error message *)
      (* Error evaluation *)
      // ? COVState := 0;  // busy State reset ?
      StartCOV := FALSE;  // finish execution ?
      Test_00 := error;
    END_IF

  END_IF // StartCOV


(* ********************************************************************************************** *)
  (* for the Callback *)
  (* real evaluation ... *)
  IF StartCallback THEN
    // either the value changes ... CovValue = old value
    // or the EventCount increases ... EventCount = old value

    (* get value from object *)
    CovValue := FB_CB_Conf_COV_Notification.LastRealValue; // PresentValue Analog Objects
    //CovValue := FB_CB_Conf_COV_Notification.LastBinaryValue; // PresentValue Binary Objects
    //CovValue := FB_CB_Conf_COV_Notification.LastPropertyValue.value; // Object-Property


    (* CovValue change *)
    IF FB_CB_Conf_COV_Notification.LastRealValue <> CovValue THEN

      (* save CovValue  *)
      CovValue := FB_CB_Conf_COV_Notification.LastRealValue;

      (* stopp timer *)
      Timeout(IN:=FALSE);

      (* next step *)
    END_IF

    (* Event Count increase *)
    IF FB_CB_Conf_COV_Notification.EvtCnt > EventCount THEN

      (* save EventCount *)
      EventCount := FB_CB_Conf_COV_Notification.EvtCnt;

      (* stopp Timer *)
      Timeout(IN:=FALSE);

      (* next step *)
    END_IF

    (* Verification Timeout *)
```

```
        Timeout(IN:=TRUE, PT:=tTimeout);
        IF (TRUE = Timeout.Q) THEN
          Timeout(IN:=FALSE);
          (* Timeout message *)
        END_IF

        (* Reset Callback *)
        //FB_CB_Conf_COV_Notification(ResetOutputs := TRUE);  // resets the values
        //FB_CB_Conf_COV_Notification(ResetOutputs := FALSE);  // resets the flag (can be done better)

      END_IF // StartCallback

END_IF // BACnetReady
```

6. Fügen Sie das Programm `COV` in den gewünschten SPS Programmteil ein. Zusätzlich kann das Ergebnis der COV Notifikation bearbeitet werden.
   - `COV();`
   - `Temp := cov.CovValue;`

Auftragspriorisierung (Outputs)

Bei Objekten mit Ausgangswerten (z.B. AO, BO, MO) besteht das Problem, dass verschiedene Clients unterschiedliche Werte vorgeben könnten, z.B. ein manueller Bedienereingriff schaltet einen Ausgang "Aus" und eine übergeordnete Steuerung schaltet den gleichen Ausgang "Ein".

Um einen Konflikt zu vermeiden besitzen kommandierbare BACnet Objekte die Priorisierungseigenschaft "PriorityArray".

Durch die Priorisierungseigenschaft wird ein Schreibbefehl nicht direkt auf das "PresentValue" ausgeführt, sondern der Wert wird intern in ein Feld der Prioritätsliste eingetragen.

Eigenschaften der Prioritätsliste

- Die Prioritätsliste hat 16 Felder, das Feld 1 hat die höchste Priorität.
- Ein Feld muss mit einem Client belegt werden, damit es wirksam wird. Der Wert "`Null`" löscht den Schreibzugriff eines Client, d.h. ein Feld muss explizit auf "`Null`" gesetzt werden um diese Priorität zu entfernen.
- Das "PresentValue" erhält den Wert von dem Client mit der höchsten Priorität.
- Ist die Prioritätsliste leer, d. h. alle Felder sind nicht belegt bzw. haben den Wert "`Null`", wird die Eigenschaft "RelinquishDefault" als "PresentValue" übernommen.
- Diese Priorisierung ist immer aktiv und kann nicht umgangen bzw. deaktiviert werden.

Folgende Prioritäten sind in BACnet für besondere Zwecke reserviert:

Prio	Anwendung
1	Sicherheitssteuerung manuell (Manual-Life Safety)
2	Sicherheitssteuerung Automatik (Automatic-Life Safety)
3	frei
4	frei
5	Übergeordnete Anlagensteuerung (Critical Equipment Control)
6	Ein-/Aus-Zeitbegrenzung (Minimum On/Off)
7	frei

Prio	Anwendung
8	Manueller Eingriff / Hand (Manual Operator)
9	frei
10	frei
11	frei
12	frei
13	frei
14	frei
15	frei
16	frei

Tab. 13-8: Reservierte Prioritäten in BACnet

Alarmbehandlung

Allgemein

BACnet unterscheidet zwei Methoden zur Alarmbehandlung:

- **Intrinsic Reporting**
  - Ein BACnet Objekt (z.B. AI, BI, ...) löst einen Alarm aus.
  - Dieses auslösenden Objekt bestimmt auch die Auslösung.
  - Versenden des Alarms.
- **Algorithmic Reporting**
  - Ein BACnet Objekt (z. B. AI, BI, ...) löst einen Alarm aus.
  - Ein anderes BACnet Objekt (EventEnrollment) bestimmt die Auslösung.
  - Versenden des Alarms.

Intrinsic Reporting

Die Bedingungen für die Alarmerzeugung stammen aus den Eigenschaften des auslösenden Objekts.

Das auslösende Objekt, in nachfolgender Abbildung z. B. ein AnalogInput (AI), hat eine Referenz zum Meldekanal (NotificationClass (NC)), der die Alarmnachricht verteilt.

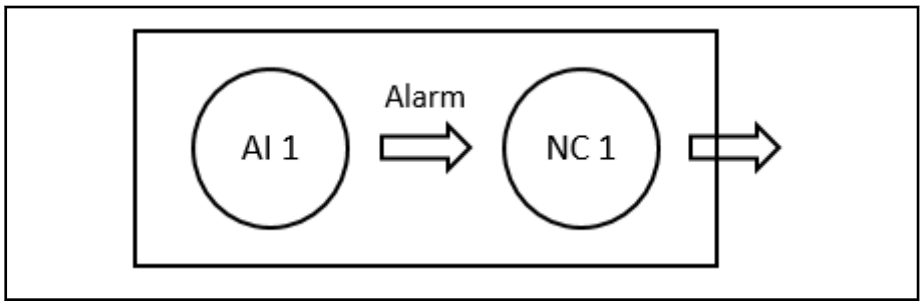


Abb. 13-4: Beispiel für Intrinsic Reporting

Folgende Algorithmen werden häufig für Intrinsic Reporting verwendet:

Algorithmus	Objekt
OUT_OF_RANGE	AnalogInput, AnalogOutput, AnalogValue, PulseConverter
CHANGE_OF_STATE	BinaryInput, BinaryValue, MultistateInput, MultistateValue
COMMAND_FAILURE	BinaryOutput, MultistateOutput
BUFFER_READY	EventLog, TrendLog, TrendLogMultiple
FLOATING_LIMIT	Loop

Tab. 13-9: Exemplarische Algorithmen für Intrinsic Reporting

Algorithmus OUT_OF_RANGE

Der Algorithmus OUT_OF_RANGE ist typischerweise eine Grenzwertüberwachung eines Analogwertes (AI, AO, AV).

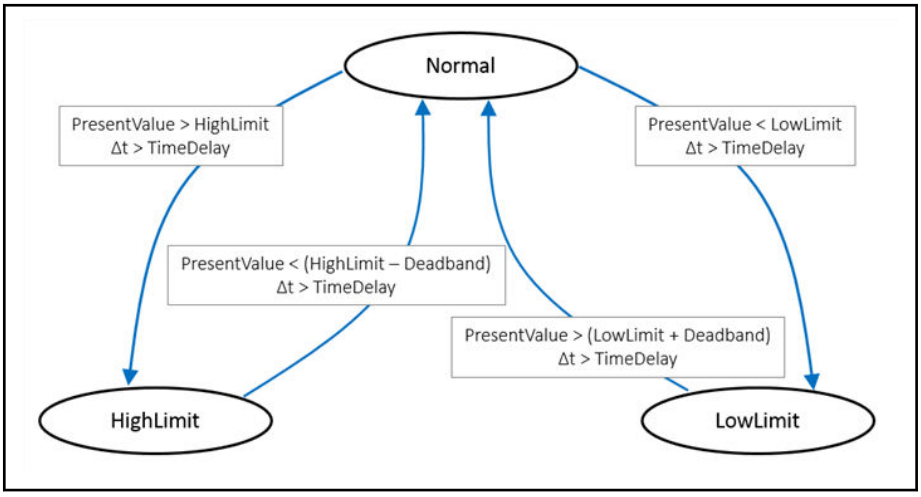


Abb. 13-5: Zustandsmaschine OUT_OF_RANGE

Algorithmus CHANGE_OF_STATE

Der Algorithmus CHANGE_OF_STATE ist typischerweise eine Zustandsüberwachung eines Binär- oder Multistate-Eingangs oder -wertes (BI, BV, MI, MV).

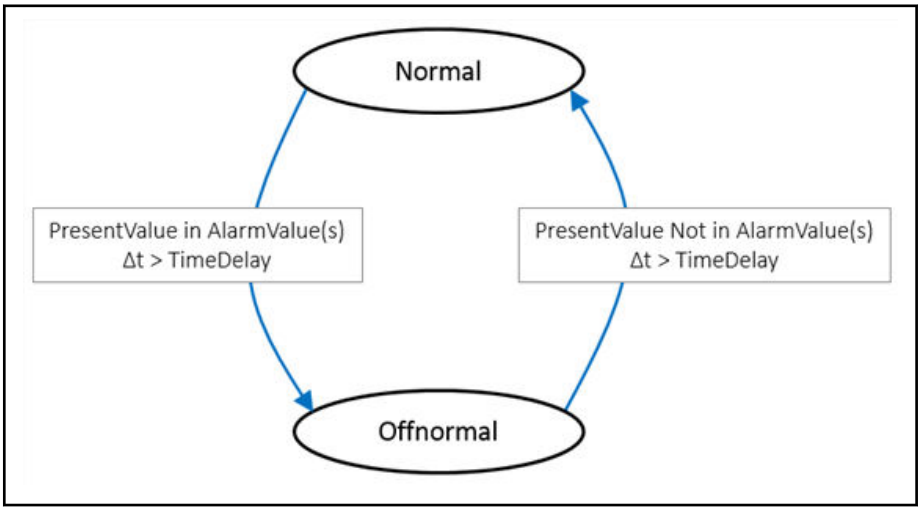


Abb. 13-6: Zustandsmaschine CHANGE_OF_STATE

Algorithmus COMMAND_FAILURE

Der Algorithmus COMMAND_FAILURE ist typischerweise eine Befehlsausführkontrolle eines Binär- oder Multistate-Ausgangs (BO, MO).

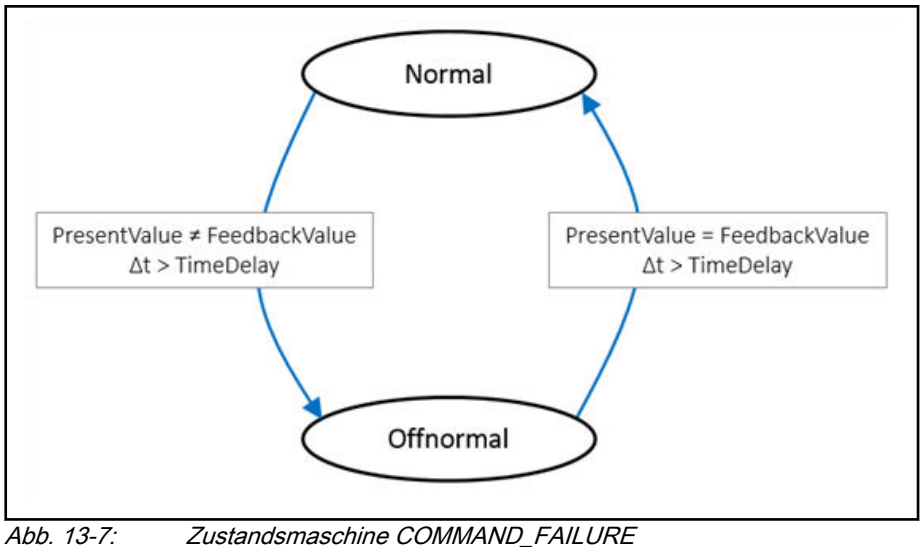


Abb. 13-7: Zustandsmaschine COMMAND_FAILURE

Algorithmic Reporting

Die Bedingungen für die Alarmerzeugung werden nicht im auslösenden Objekt konfiguriert, sondern in einem weiteren Objekt (EventEnrollment (EE)).

Das EventEnrollment Objekt hat dann die Referenz zum Meldekanal (NotificationClass (NC))

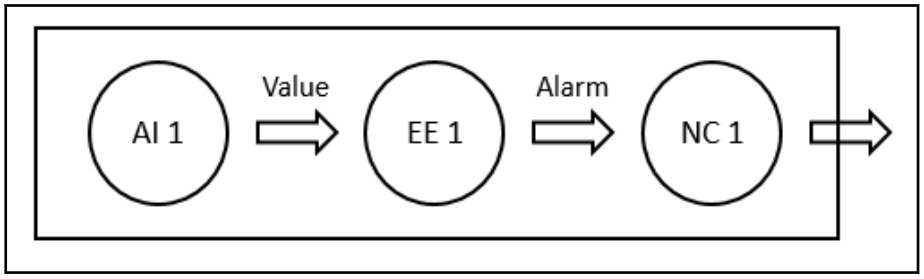


Abb. 13-8: Beispiel für Algorithmic Reporting

Mit Algorithmic Reporting können komplexere Alarmfunktionen umgesetzt werden, da der Algorithmus frei wählbar ist (da er nicht im auslösenden Objekt integriert ist, sondern durch ein zusätzliches Objekt realisiert wird).

Alarmverteilung

Als Meldekanal der Alarmerzeugungsverfahren wird das BACnet Objekt NotificationClass (NC) verwendet.

- Im NC Objekt werden die relevanten Übertragungseigenschaften eingestellt.
- Das NC Objekt hat eine Liste von Empfängern (Recipients), die über Alarme informiert werden.
- Es muss mindestens ein NC Objekt im Gerät vorhanden sein (wenn Alarme verwendet werden sollen).
- Das NC Objekt ist unabhängig von der Alarmerzeugung (sondern dient nur als Meldekanal).
- Mehrere NC Objekte sind z. B. dann sinnvoll, wenn man zwischen Warnungen und kritischen Alarmen unterscheiden möchte.

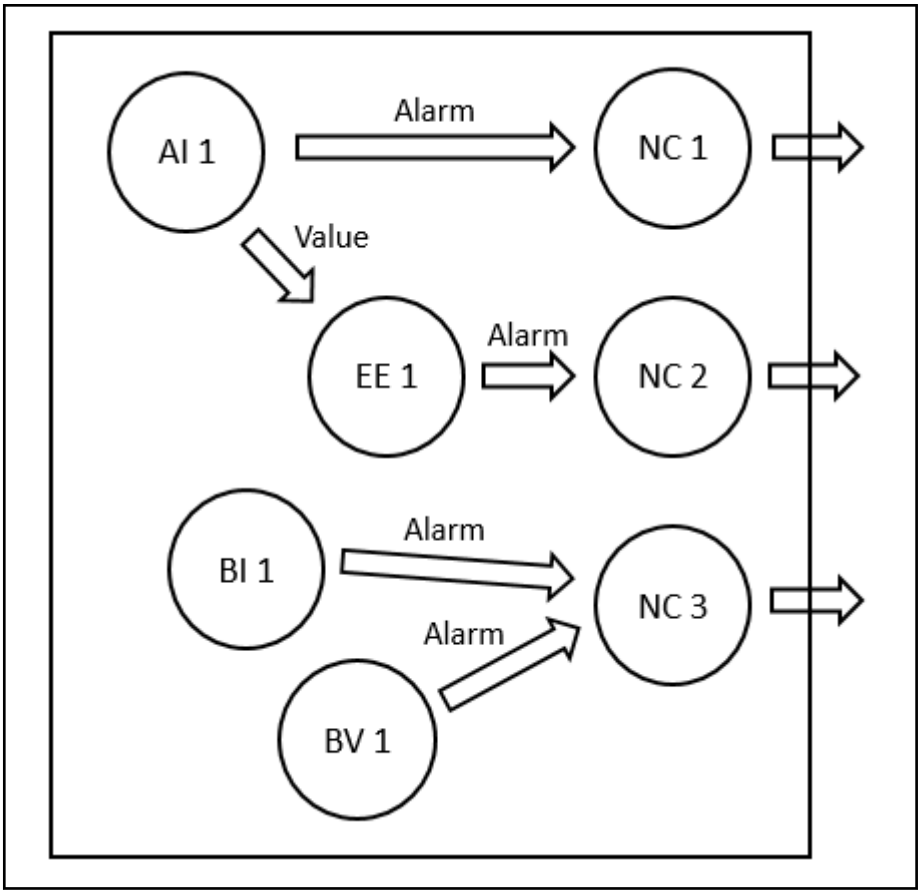


Abb. 13-9: Beispiel für Alarmverteilung

Datenaufzeichnung (TrendLog)

- Das TrendLog (TL) Objekt sammelt Datenpaare (Wert + Zeitstempel).
- Daten werden regelmäßig oder ereignisgesteuert z. B. durch Wertänderung aufgezeichnet.
- Das referenzierte Objekt kann sich im eigenen Geräte ("internal trending") oder in einem anderen Gerät ("external trending") befinden.
- Die Aufzeichnung wird über die Eigenschaften des TL Objekts gesteuert.
- Es wird die Auslastung des LogBuffers überwacht (Intrinsic Reporting). Hierbei wird der Algorithmus BUFFER_READY eingesetzt .

  Es erfolgt eine Benachrichtigung zum Abholen der Werte, wenn `EventState == NORMAL` und `RecordsSinceNotification >= NotificationThreshold` ist.

  Die aufgezeichneten Daten können mit dem Dienst ReadRange ausgelesen werden.

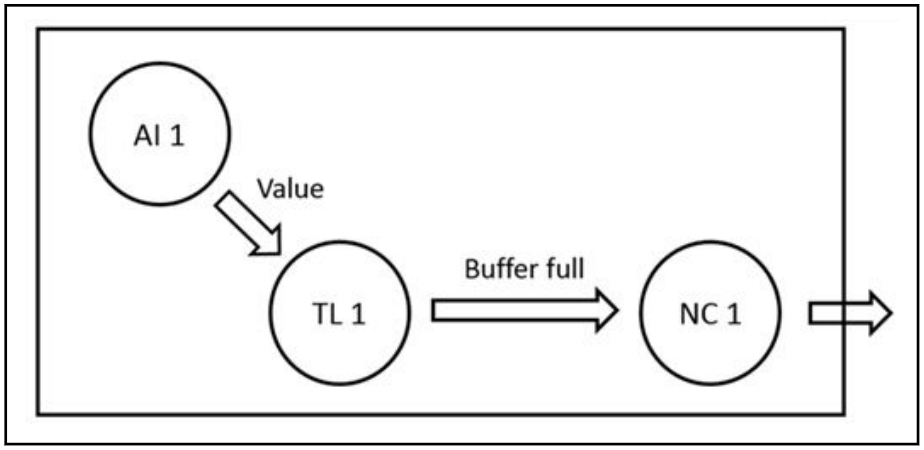


Abb. 13-10: Beispiel für TrendLog

Geräte-Profile

In der BACnet Spezifikation werden standardisierte Profile für BACnet Geräte beschrieben. Jedes Profil beschreibt einen Funktionsumfang, den ein BACnet Gerät unterstützen muss, damit das Gerät das Geräte-Profil erfüllt. Die wichtigsten Geräte-Profile sind:

B-AWS	BACnet-Advanced Workstation	Management- und Bedienstation mit größerem Leistungsumfang als B-OWS
B-OWS	BACnet-Operator Workstation	Management- und Bedienstation
B-OD	BACnet-Operator Display	Bediengerät (vor Ort)
B-BC	BACnet-Building Controller	Frei programmierbares Automationsgerät
B-AAC	BACnet-Advanced Application Controller	Automationsgerät mit Programmauswahl
B-ASC	BACnet-Application Specific Controller	Automationsgerät mit festem Programm
B-SA	BACnet-Smart Actuator	Schalt- oder Stellgerät
B-SS	BACnet-Smart Sensor	Sensor

Tab. 13-10: BACnet Geräte-Profil

BACnet Funktionalitäten und Dienste werden in Form von BIBB (BACnet Interoperability Building Block) beschrieben. Man unterscheidet folgende Kategorien:

- Data Sharing (DS)
- Alarm / Event Management (AE)
- Scheduling (SCH)
- Trending (TR)
- Device and Network Management (DN)

Die Funktionalitäten werden mit "A" (Client) oder "B" (Server) klassifiziert.

Beispiele für BIBB:

- DS-RP-A (/-B): Data Sharing - ReadProperty - Client (/-Server)

- DS-WP-A (/-B): Data Sharing - WriteProperty - Client (/-Server)

BACnet Fehler-Codes

Fehler Code	Name
16#7FFFFFFF	_ERR_CODE_MAX_SIZE
0	ERR_CODE_OTHER
1	ERR_CODE_AUTHENTICATION_FAILED
2	ERR_CODE_CONFIG_IN_PROGRESS
3	ERR_CODE_DEVICE_BUSY
4	ERR_CODE_DYN_CREATION_NOT_SUPP
5	ERR_CODE_FILE_ACCESS_DENIED
6	ERR_CODE_WRONG_SECURITY_LEVELS
7	ERR_CODE_INCONSISTENT_PARAMS
8	ERR_CODE_INVALID_PARAMETERS
9	ERR_CODE_INVALID_DATA_TYPE
10	ERR_CODE_INVALID_FILE_ACCESS
11	ERR_CODE_INVALID_FILE_START_POS
12	ERR_CODE_INVALID_OPERATOR_NAME
13	ERR_CODE_INVALID_PARAMETER_TYPE
14	ERR_CODE_INVALID_TIME_STAMP
15	ERR_CODE_KEY_GENERATION_ERROR
16	ERR_CODE_MISSING_PARAMETER
17	ERR_CODE_NO_OBJS_OF_SPEC_TYPE
18	ERR_CODE_NO_SPACE_FOR_OBJECT
19	ERR_CODE_NO_SPACE_FOR_ELEMENT
20	ERR_CODE_NO_SPACE_TO_WRITE_PROP
21	ERR_CODE_NO_VT_SESSIONS_AVAIL
22	ERR_CODE_PROP_IS_NOT_A_LIST
23	ERR_CODE_OBJ_DELETION_DENIED
24	ERR_CODE_OBJ_ID_ALREADY_EXISTS
25	ERR_CODE_OPERATIONAL_PROBLEM
26	ERR_CODE_PASSWORD_FAILURE
27	ERR_CODE_READ_ACCESS_DENIED
28	ERR_CODE_SECURITY_NOT_SUPPORTED
29	ERR_CODE_SERVICE_REQUEST_DENIED
30	ERR_CODE_TIMEOUT
31	ERR_CODE_UNKNOWN_OBJECT
32	ERR_CODE_UNKNOWN_PROPERTY
34	ERR_CODE_UNKNOWN_VT_CLASS

Fehler Code	Name
35	ERR_CODE_UNKNOWN_VT_SESSION
36	ERR_CODE_UNSUPPORTED_OBJ_TYPE
37	ERR_CODE_VALUE_OUT_OF_RANGE
38	ERR_CODE_VT_SESSION_CLOSED
39	ERR_CODE_VT_SESSION_TERM_FAILED
40	ERR_CODE_WRITE_ACCESS_DENIED
41	ERR_CODE_CHAR_SET_NOT_SUPPORTED
42	ERR_CODE_INVALID_ARRAY_INDEX
43	ERR_CODE_COV_SUBS_FAILED
44	ERR_CODE_NOT_COV_PROPERTY
45	ERR_CODE_OPTIONAL_FUNC_NOT_SUPP
46	ERR_CODE_INVALID_CONFIG_DATA
47	ERR_CODE_DATATYPE_NOT_SUPPORTED
48	ERR_CODE_DUPLICATE_NAME
49	ERR_CODE_DUPLICATE_OBJECT_ID
50	ERR_CODE_PROP_IS_NOT_AN_ARRAY
51	ERR_CODE_ABORT_BUFFER_OVERFLOW
52	ERR_CODE_ABORT_INVALID_APDU
53	ERR_CODE_ABORT_PREEMPTED
54	ERR_CODE_ABORT_SEGMENT_NOT_SUPP
55	ERR_CODE_ABORT_PROPRIETARY
56	ERR_CODE_ABORT_OTHER
57	ERR_CODE_INVALID_TAG
58	ERR_CODE_NETWORK_DOWN
59	ERR_CODE_REJECT_BUFFER_OVERFLOW
60	ERR_CODE_REJECT_INCONSISTENT_PARAMS
61	ERR_CODE_REJECT_INVALID_DATA_TYPE
62	ERR_CODE_REJECT_INVALID_TAG
63	ERR_CODE_REJECT_MISSING_REQUIRED_PARAM
64	ERR_CODE_REJECT_PARAMETER_OUT_OF_RANGE
65	ERR_CODE_REJECT_TOO_MANY_ARGUMENTS
66	ERR_CODE_REJECT_UNDEFINED_ENUMERATION
67	ERR_CODE_REJECT_UNRECOGNIZED_SERVICE
68	ERR_CODE_REJECT_PROPRIETARY
69	ERR_CODE_REJECT_OTHER
70	ERR_CODE_UNKNOWN_SERVICE
71	ERR_CODE_UNKNOWN_ROUTER
72	ERR_CODE_VALUE_NOT_INITIALIZED

Fehler Code	Name
73	ERR_CODE_INVALID_EVENT_STATE
74	ERR_CODE_NO_ALARM_CONFIGURED
75	ERR_CODE_LOG_BUFFER_FULL
76	ERR_CODE_LOGGED_VALUE_PURGED
77	ERR_CODE_NO_PROPERTY_SPECIFIED
78	ERR_CODE_NOT_CONFIGURED_FOR_TRIGGERED_LOGGING
79	ERR_CODE_UNKNOWN_SUBSCRIPTION
80	ERR_CODE_PARAMETER_OUT_OF_RANGE
81	ERR_CODE_LIST_ELEMENT_NOT_FOUND
82	ERR_CODE_BUSY
83	ERR_CODE_COMMUNICATION_DISABLED
84	ERR_CODE_SUCCESS
85	ERR_CODE_ACCESS_DENIED
86	ERR_CODE_BAD_DESTINATION_ADDRESS
87	ERR_CODE_BAD_DESTINATION_DEVICE_ID
88	ERR_CODE_BAD_SIGNATURE
89	ERR_CODE_BAD_SOURCE_ADDRESS
90	ERR_CODE_BAD_TIMESTAMP
91	ERR_CODE_CANNOT_USE_KEY
92	ERR_CODE_CANNOT_VERIFY_MESSAGE_ID
93	ERR_CODE_CORRECT_KEY_REVISION
94	ERR_CODE_DESTINATION_DEVICE_ID_REQUIRED
95	ERR_CODE_DUPLICATE_MESSAGE
96	ERR_CODE_ENCRYPTION_NOT_CONFIGURED
97	ERR_CODE_ENCRYPTION_REQUIRED
98	ERR_CODE_INCORRECT_KEY
99	ERR_CODE_INVALID_KEY_DATA
100	ERR_CODE_KEY_UPDATE_IN_PROGRESS
101	ERR_CODE_MALFORMED_MESSAGE
102	ERR_CODE_NOT_KEY_SERVER
103	ERR_CODE_SECURITY_NOT_CONFIGURED
104	ERR_CODE_SOURCE_SECURITY_REQUIRED
105	ERR_CODE_TOO_MANY_KEYS
106	ERR_CODE_UNKNOWN_AUTHENTICATION_TYPE
107	ERR_CODE_UNKNOWN_KEY
108	ERR_CODE_UNKNOWN_KEY_REVISION
109	ERR_CODE_UNKNOWN_SOURCE_MESSAGE
110	ERR_CODE_NOT_ROUTER_TO_DNET

Fehler Code	Name
111	ERR_CODE_ROUTER_BUSY
112	ERR_CODE_UNKNOWN_NETWORK_MESSAGE
113	ERR_CODE_MESSAGE_TOO_LONG
114	ERR_CODE_SECURITY_ERROR
115	ERR_CODE_ADDRESSING_ERROR
116	ERR_CODE_WRITE_BDT_FAILED
117	ERR_CODE_READ_BDT_FAILED
118	ERR_CODE_REGISTER_FOREIGN_DEVICE_FAILED
119	ERR_CODE_READ_FDT_FAILED
120	ERR_CODE_DELETE_FDT_ENTRY_FAILED
121	ERR_CODE_DISTRIBUTE_BROADCAST_FAILED
122	ERR_CODE_UNKNOWN_FILE_SIZE
123	ERR_CODE_ABORT_APDU_TOO_LONG
124	ERR_CODE_ABORT_APPLICATION_EXCEEDED_REPLY_TIME
125	ERR_CODE_ABORT_OUT_OF_RESOURCES
126	ERR_CODE_ABORT_TSM_TIMEOUT
127	ERR_CODE_ABORT_WINDOW_SIZE_OUT_OF_RANGE
128	ERR_CODE_FILE_FULL
129	ERR_CODE_INCONSISTENT_CONFIGURATION
130	ERR_CODE_INCONSISTENT_OBJECT_TYPE
131	ERR_CODE_INTERNAL_ERROR
132	ERR_CODE_NOT_CONFIGURED
133	ERR_CODE_OUT_OF_MEMORY
134	ERR_CODE_VALUE_TOO_LONG
135	ERR_CODE_ABORT_INSUFFICIENT_SECURITY
136	ERR_CODE_ABORT_SECURITY_ERROR
65535	ERR_CODE_MAX

Tab. 13-11: TYPE IEC_BACNET_ERROR_CODE

BACnet Status-Codes

Name	Initial	Kommentar
_BACNET_STATUS_MAX_SIZE	16#7FFFFFFF	
BACNET_STATUS_OK	16#0	die Anfrage (Request) ist erfolgreich.
BACNET_STATUS_UNKNOWN_ERROR	16#1	Unbekannter Fehler.
BACNET_STATUS_SERVICE_NOT_FOUND	16#2	Die Applikation kann sich nicht mit dem "Datalink-Layer" verbinden.
BACNET_STATUS_NOT_INITIALIZED	16#3	Das Interface API ist nicht korrekt initialisiert und kann keine Anfragen bearbeiten.

Name	Initial	Kommentar
BACNET_STATUS_NOT_SUPPORTED	16#4	die Anfrage (Request) wird nicht unterstützt.
BACNET_STATUS_REJECT	16#5	Die Anfrage (Request) wurde abgelehnt.
BACNET_STATUS_DCC_DISABLED	16#6	Die Anfrage (Request) kann nicht verarbeitet werden, weil das device communication control nicht aktiv ist.
BACNET_STATUS_CHARSET_NOT_SUPPORTED	16#7	String (Charakter) wird nicht unterstützt.
BACNET_STATUS_FUNCTION_NOT_SUPPORTED	16#8	Ein optionales Objekt oder eine Funktionseigenschaft (property function) wird nicht unterstützt.
BACNET_STATUS_INVALID_PARAM	16#10	Die Anfrage (Request) wurde aufgrund ungültiger Parameterwerte abgelehnt.
BACNET_STATUS_BUSY	16#11	Die Anfrage kann nicht bearbeitet werden, weil der Datalink oder das Gerät mit einer anderen Task beschäftigt ist.
BACNET_STATUS_ALREADY_EXISTS	16#12	Das Objekt kann nicht generiert werden, weil bereits ein Objekt, oder eine Instanz mit dem selben Namen existiert.
BACNET_STATUS_BAD_CONFIG	16#13	die Konfiguration ist ungültig für BACnet stack Datalink-Layer .
BACNET_STATUS_OBJECT_NOT_FOUND	16#14	Das angefragte Objekt wurde nicht gefunden.
BACNET_STATUS_PROPERTY_NOT_FOUND	16#15	Die angefragte Objekteigenschaft wurde nicht gefunden.
BACNET_STATUS_MISSING_REQUIRED_PARAM	16#16	Notwendige Parameter wurden nicht gefunden, oder fehlen.
BACNET_STATUS_SECURITY_ENTRY_NOT_FOUND	16#17	Der angeforderte Sicherheitszugang konnte nicht gefunden werden.
BACNET_STATUS_OUT_OF_MEMORY	16#20	Speicherüberlauf.
BACNET_STATUS_OUT_OF_RESOURCES	16#21	Die System Ressourcen sind verbraucht (sockets/timers...).
BACNET_STATUS_XMT_OVERFLOW	16#22	Der Applikations- / Anwender-Sendeseite fehlen Daten in der Queue.
BACNET_STATUS_RCV_OVERFLOW	16#23	Der Applikations- / Anwender-Empfangsseite fehlen Daten in der Queue.
BACNET_STATUS_TIMEOUT	16#30	Es wurde keine Antwort (Response) auf eine Anfrage (Request) erhalten. Bei der Verwendung der Funktionen `BACnetReadProperty`, `BACnetReadPropertyMultiple`, oder `BACNET_READ_PROP_COMPLETE_CB` wurde keine Antwort (Response) innerhalb des konfigurierten APDU Timeout erhalten.
BACNET_STATUS_REQUEST_TOO_LONG	16#31	Die Anfrage (Request) ist zu groß bzw. zu lang für den Datalink.
BACNET_STATUS_ILLEGAL_RESPONSE	16#32	Die Applikation versucht eine ungültige Antwort (Response) auf eine empfangene Anfrage (Request) zu senden.
BACNET_STATUS_RESPONSE_TOO_LONG	16#33	Die Antwort (Response) ist zu groß bzw. zu lang für den Datalink.
BACNET_STATUS_INVALID_DATA_TYPE	16#34	Ungültiger Datentyp.
BACNET_STATUS_VAL_OUT_OF_RANGE	16#35	Wert außerhalb des Wertebereichs.
BACNET_STATUS_VAL_OUT_OF_SPACE	16#36	Wert zu groß, oder zu lang.

Name	Initial	Kommentar
BACNET_STATUS_SEG_NOT_SUPPORTED	16#37	Die Anwendung versucht eine segmentierte Anforderung zu senden, aber die Segmentierung wird nicht unterstützt oder ist deaktiviert.
BACNET_STATUS_TRANSACTION_ABORTED	16#38	Die BACnet Transaktion wurde abgebrochen.
BACNET_STATUS_BACNET_ERROR	16#40	Eine Anfrage (Request) ist mit Fehler "BACnetError" in der Antwort (Response) beantwortet. Bei der Verwendung der Funktionen `BACnetReadProperty`, `BACnetReadPropertyMultiple`, oder `BACNET_READ_PROP_COMPLETE_CB` enthält die Struktur **BACNET_ERROR** zusätzliche Fehlerinformationen.
BACNET_STATUS_BACNET_REJECT	16#41	Eine Anfrage (Request) ist mit der Antwort (Response) "BACnetReject" abgelehnt worden. Bei der Verwendung der Funktionen `BACnetReadProperty`, `BACnetReadPropertyMultiple`, oder `BACNET_READ_PROP_COMPLETE_CB` enthält die Struktur **BACNET_ERROR** zusätzliche Fehlerinformationen.
BACNET_STATUS_BACNET_ABORT	16#42	Eine Anfrage (Request) ist mit der Antwort (Response) "BACnetAbort" abgebrochen worden. Bei der Verwendung der Funktionen `BACnetReadProperty`, `BACnetReadPropertyMultiple`, oder `BACNET_READ_PROP_COMPLETE_CB` enthält die Struktur **BACNET_ERROR** zusätzliche Fehlerinformationen.
BACNET_STATUS_INVALID_PACKET	16#43	Ungültiges Paket empfangen (obsolete code).
BACNET_STATUS_SEG_ERROR	16#44	Ungültige Paketsegmente empfangen (obsolete code).
BACNET_STATUS_INCONSISTENT_TAGS	16#45	Ungültiges Paket mit Inkonsistenten Tags empfangen (obsolete code).
BACNET_STATUS_CANNOT_BIND_TO_PORT	16#49	Die Applikation versucht sich auf einen Datalink-Port zu verbinden, der nicht existiert.
BACNET_STATUS_DRV_NOT_FOUND	16#50	Die Applikation versucht sich auf einen Datalink zu verbinden, der nicht existiert. (obsolete code)
BACNET_STATUS_DRV_NOT_SUPPORTED	16#51	Ein unbekannter, oder nicht unterstützter Datalink ist konfiguriert (obsolete code).
BACNET_STATUS_DRV_CANT_INIT	16#52	Der Datalink wurde nicht korrekt initialisiert (obsolete code).
BACNET_STATUS_DRV_CANT_SEND	16#53	Die Applikation versucht eine Anfrage (Request) zu senden, aber die Datalink-Queue des dazugehörigen Datalink -Gegenstelle ist ausgelastet (voll).
BACNET_STATUS_WRITEACCESS_DENIED	16#60	Die Eigenschaft ist nicht schreibbar / änderbar (obsolete code).
BACNET_STATUS_NO_ROUTE_FOUND	16#61	Das Telegramm (Frame) kann nicht über den Router gesendet werden (obsolete code).
BACNET_STATUS_SEC_DIS_NON_TRUST	16#70	Security ist deaktiviert, unsicheres Routing.
BACNET_STATUS_SEC_INVAL_KEY_DATA	16#71	Ungültiger Datenschlüssel (key data) angegeben.
BACNET_STATUS_SEC_INVAL_DEST_ID	16#72	Ungültige Zielgeräte-ID.
BACNET_STATUS_SEC_INVAL_SRC_ID	16#73	Ungültige Quellgeräte-ID.

Name	Initial	Kommentar
BACNET_STATUS_SEC_BAD_VALIDATION	16#74	Die Nachricht konnte nicht validiert werden.
BACNET_STATUS_SEC_DEC_NO_DECRYPT	16#75	Die Nachricht kann nicht entschlüsselt und verifiziert werden (möglicherweise falscher Schlüssel zum Decodieren verwendet).
BACNET_STATUS_RAW_VALUE	16#F0	Es konnten nicht alle RAW-Werte dekodiert werden (only for internal use).
BACNET_STATUS_PRIMITIVE_VALUE	16#F1	Es konnten nicht alle Primitive RAW-Werte dekodiert werden (only for internal use).
BACNET_STATUS_STANDARD_VALUE	16#F2	Es konnten nicht alle Standard RAW-Werte dekodiert werden (only for internal use).
BACNET_STATUS_INVALID_VALUE	16#F3	Es wurden ungültige RAW-Werte dekodiert (only for internal use).

Tab. 13-12: TYPE IEC_BACNET_STATUS

13.2.3 BACnet Server konfigurieren

BACnet Server Inbetriebnahme

Damit der BACnet Server korrekt startet sind folgende Maßnahmen zu treffen

1. Es ist eine separate SPS-Task (langsam 100ms, ohne Watchdog) zum Starten erforderlich, da manche Initialisierungen mehr Zeit benötigen.
2. Es ist ein "BACnetStartPRG"-Programm erforderlich, in dem die unten genannten Schritte der Reihe nach ausgeführt werden. In den Schritten können weitere Initialisierungen vorgenommen werden, bevor der Status "BACnetReady" gesetzt wird.

   Erforderliche Schritte:

   1. Zeitzone setzen
   2. BACnet Server Kommunikationseinstellungen mit IP-Adresse einstellen
   3. Plugins und Objekte registrieren, die vor der Ausführung der Funktion `StartBACnetStack()` registriert werden müssen
   4. BACnet Server mit der Funktion `StartBACnetStack()` starten und ein BACnet Ready Flag setzen
   5. Plugins und Objekte registrieren, die nach der Ausführung der Funktion `StartBACnetStack()` registriert werden müssen
3. Es ist ein "BACnetStartPRG"-Programm erforderlich, das ein regelmäßiges Triggern der Uhrzeit und eine Aktualisierung der BACnet Objekte ausführt, siehe "Anlegen eines BACnet-Projekts" auf Seite 353 Schritt 4.

Es wird empfohlen, alle verwendeten Plugins vor der Ausführung der Funktion `StartBACnetStack()` zu initialisieren.

Plugins, bei denen die Initialisierung absolut notwendig ist, enthalten in der Beschreibung einen entsprechenden Hinweis.

Folgende Plugins sind zwingend vor der Ausführung der Funktion `StartBACnetStack()` zu initialisieren:

- TrendLog
- ObjectPersistence
- TimeSync und UtcTimeSync
- deviceDateTime

BACnet Server und Objekte anlegen und BACnet Server starten

Anlegen eines BACnet-Projekts

Einer Steuerung kann maximal ein BACnet Server hinzugefügt werden. Dieser Server repräsentiert das Gerät als Teilnehmer im BACnet Netzwerk.

Der beschriebene Ablauf beschreibt das Prinzip, einen BACnet Server anzulegen und zu starten.

Zur Konfiguration eines BACnet-Projekts gehen Sie wie folgt vor:

1. Legen Sie eine Steuerung im IndraWorks Projektexplorer an (z. B. XM2x).
2. Fügen Sie der Steuerung einen "BACnet Server" hinzu.

   Öffnen Sie hierzu das Kontextmenü der Steuerung und wählen Sie den Kontextbefehl **Hinzufügen ▸ BACnet Server**.
3. Fügen Sie dem "BACnet Server" das Objekt "BACnet IP Datalink" hinzu. Dies ist notwendig für das konfigurieren der BACnet Kommunikationseinstellungen innerhalb der SPS. Mit dem Objekt "BACnet IP Datalink" wird unter anderem die Netzwerk Nummer (NetworkNumber, Default = 1), IP-Adresse und der BACnet Port konfiguriert. Im System wird automatisch die Datei bacstac.ini generiert.

   Die Zuweisung der IP-Adresse erfolgt im nächsten Schritt mittels SPS-Programm.
4. Erstellen sie ein SPS Programm "`BACnetSartPRG`" zum Initialisieren des BACnetServers.
5. Erstellen Sie eine neue SPS-Task (z.B. BACnetTask) und fügen Sie das Programm "`BACnetStartPRG`" in der Task-Aufrufliste hinzu.

   Die Task "BACnetTask" initialisiert den BACnet-Server und kann in einem langsamen Intervall laufen (Empfohlen 100ms Taskzykluszeit).

   → Es darf keinen Watchdog verwendet werden!
6. Initialisieren der Zeitzoneninformation für die Synchronisation von Uhrzeit und Datum des BACnet Servers. Erfolgt vor dem Start des BACnet Server. Dazu ist die Zeitzoneninformation mit Hilfe der SysTimeRTC Bibliothek zu setzen, damit BACnet Lokalzeit und UTC korrekt anzeigt. Setzen Sie passend zu ihren Land die Werte der Struktur "**TimezoneInformation**" und berücksichtigen Sie die Zeitpunkte der Zeitumstellung in Ihren Land. Es werden die beiden Globalen Variablen "`g_ulDaylightDate : UDINT := TO_UDINT(DT#2020-03-29-2:0:0);`" und"`g_ulStandardDate : UDINT := TO_UDINT(DT#2020-10-25-3:0:0);`" verwendet, diese sind zu de-

klarieren. Mit Hilfe der globalen Variablen kann man den Zeitpunkt der Zeitumstellung jährlich anpassen.

Im Beispiel Programm werden dazu die "TimeSync" Plugins registriert.

7. Das Starten und Stoppen des BACnet-Servers erfolgt mit dem SPS Programm "BACnetStartPRG", wie im folgenden Beispiel gezeigt.

   Folgende Funktionen kommen im Programm zum Einsatz:

   - Zum Starten des BACnet-Servers: `StartBACnetStack`
   - Zum Stoppen des BACnet-Servers: `StopBACnetStack`
   - Zur Zuweisung der IP-Adresse: `BACnet_IP_datalink.IPaddress := [AAA.BBB.CCC.DDD];`

Programm:

```
PROGRAM BACnetStartPRG
VAR
(* BACnet Init Case *)
    BACnetInit : INT := 0; // init case counter
    InitError : UINT := 0; // error counter

(* to set Timezone ... *)
    TZI : TimezoneInformation;  //form SysTimeRTC.library

(* get IP from PLC *)
    FB_GetIP : ML_GetIpAddress; // from ML_BASE.library
    (* IP als String *)
    IPString : STRING;
    (* IP als DWORD *)
    dwIPAdr : DWORD;
    (* IP als array of Byte *)
    IPAdr : ARRAY [0..3] OF BYTE := [192,168,1,1]; // default

// TimeSync --> from  BACnetDefaultImpl.library
    timeSync : BACnetDefaultimpl.TimeSync_SvcAppHook;
    utcTimeSync : BACnetDefaultimpl.UTCTimeSync_SvcAppHook;
    deviceDateTime : BACnetDefaultImpl.DeviceDateTime;

// trigger time sync of BACnet
    TimeTON : TON;
    TimeWait: TIME := T#1S;

END_VAR
```

Programm:

```
IF UserVarGlobal.BACnetReady = FALSE THEN
// ...
// mandatory steps to initialize BACnet
// ...
    CASE BACnetInit OF
    0: (* set TimeZoneInformation of your local location in this example Germany *)
        // iBias - unit mintues - distance to UTC
        TZI.iBias := -60;
        // iDaylightBias - unit minutes - distance daylight saving time UTC if exist in your country
        TZI.iDaylightBias := -60;
        TZI.szDaylightName := 'CEST'; // Timzone shortcut
        TZI.szStandardName := 'CET'; // Timzone shortcut
        TZI.ulDaylightDate := UserVarGlobal.g_ulDaylightDate; // last Sunday in Mar start daylight
        TZI.ulStandardDate := UserVarGlobal.g_ulStandardDate; // last Sunday in Oct end daylight
        (*  *)
        SysTimeRtcSetTimezone(TZI); // set Timzone in runtime
        (* ... *)
        BACnetInit := BACnetInit + 1;

    1: (* get/set IP-Adress and start BACnet-Server *)
        // get IPAdress of control ...
        FB_GetIP(Execute := TRUE);
        (* run FB *)
        IF FB_GetIP.Done THEN
```

```
            FB_GetIP(Execute := FALSE);
            (* get IP-String *)
            IPString := FB_GetIP.IpAddress; // as string
            (* convert string to array of byte *)
            dwIPAdr := IL_IPV4Decode(IPString);
            (* convert DWORD to array of byte *)
            IPAdr[3] := DWORD_TO_BYTE( SHR( dwIPAdr,  0));
            IPAdr[2] := DWORD_TO_BYTE( SHR( dwIPAdr,  8));
            IPAdr[1] := DWORD_TO_BYTE( SHR( dwIPAdr, 16));
            IPAdr[0] := DWORD_TO_BYTE( SHR( dwIPAdr, 24));
            (* set datalink *)
            BACnet_IP_datalink.IPaddress := IPAdr;
            (* ... *)
            BACnetInit := BACnetInit + 1;
        END_IF // done
        (* *)
        IF FB_GetIP.Error THEN
            FB_GetIP(Execute := FALSE);
            (* ... *)
            InitError := InitError + 1;
            (* set default Adress ... *)
            BACnet_IP_datalink.IPaddress := IPAdr;
            (* ... *)
            BACnetInit := BACnetInit + 1;
        END_IF // error

    2: (* prepare BACnet services for time synchronization see BACnetDefaultImpl *)
        timeSync.Logging := FALSE;
        timeSync.Init();
        BACnet_Server.RegisterPlugin(timeSync); // register Service Request Time
        (* ... *)
        utcTimeSync.Logging := FALSE;
        utcTimeSync.Init();
        BACnet_Server.RegisterPlugin(utcTimeSync); // register Service Request UTCTime
        (* ... *)
        BACnet_Server.RegisterPlugin(deviceDateTime); // register Service DateTime
        (* ... *)
        BACnetInit := BACnetInit + 1;

    3:    // ...
        // Register other plugins, schedule, notification and/or calendar,
  // if component must be registered before start of BACnet server
        // ...
        BACnetInit := BACnetInit + 1;

    4: (* Start BACnet Server *)
        (* ... *)
        BACnet_Server.StartBACnetStack();
        (* ... *)
        BACnetInit := BACnetInit + 1;

    5:    // ...
        // Register other plugins, schedule, notification and/or calendar,
  // if component must be registered after start of BACnet server
        // ...
        BACnetInit := BACnetInit + 1;

    6: (* finish *)
        IF InitError = 0 THEN
            UserVarGlobal.BACnetReady := TRUE;
        ELSE
            ;
        END_IF
        (* *)

        BACnetInit := 0;

    END_CASE // BACnetInit

END_IF // UserVarGlobal.BACnetReady

IF UserVarGlobal.BACnetReady THEN

    (* BACnet Server - set Time and Date *)
    (* must be called cyclic that the BACnet time is correct *)
    (* currently its once a second, can be adjusted for application need *)
    TimeTON(IN:=TRUE, PT:=TimeWait);
    IF (TRUE = TimeTON.Q) THEN
        TimeTON(IN:=FALSE);
        // need to call deviceDateTime, because of datetime calculation
        deviceDateTime();
```

```
    END_IF
    (* ... *)
    (* ... *)
    (* ... *)

END_IF // BACnetReady
```

8. Deklarieren Sie die globale Variable "`BACnetReady`" in einer globalen Variablenliste.

   Die Variable wird vom SPS Programm "`BACnetStartPRG`" verwendet und zeigt an, dass der BACnet Server gestartet ist.

9. Zur Zeitsynchronisation des BACnet Server ist zyklisch die Funktion "**deviceDateTime()**" aufzurufen, in dem Beispiel erfolgt das jede Sekunde im SPS Programm "BACnetStartPRG". Das ist notwendig, das z.B. Schedule Objekte zum erwarteten Zeitpunkt schalten.

10. Fügen Sie über das Kontextmenü des "BACnet Server" die gewünschten BACnet Objekte hinzu.

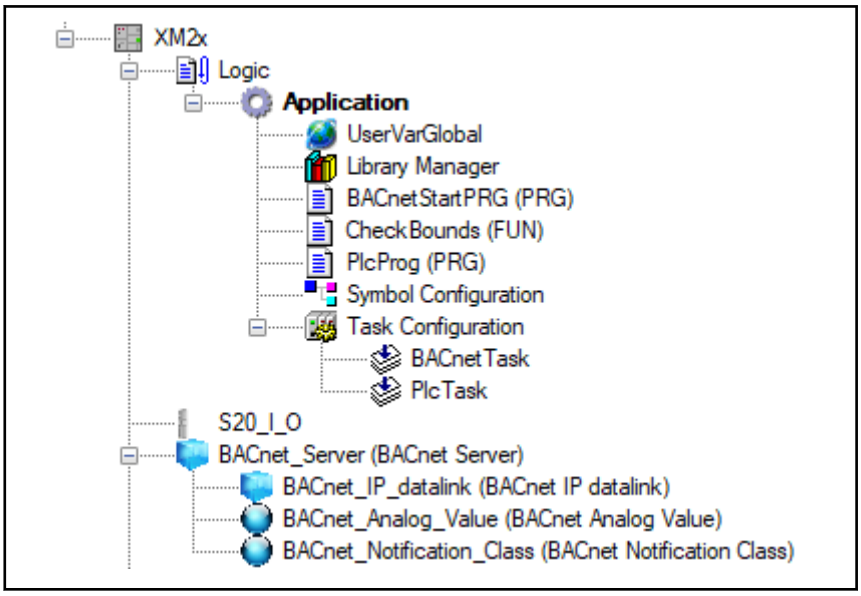
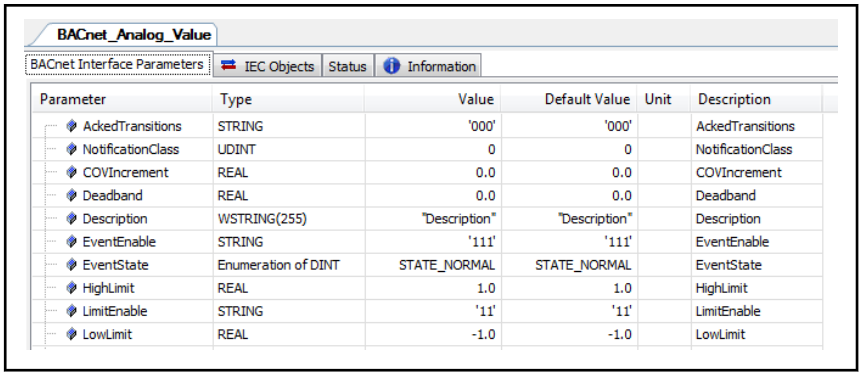


Abb. 13-11: BACnet Server Konfigurations Beispiel

11. Zur Einstellung der Geräteeigenschaften öffnen Sie den generischen Geräteeditor des Objekts durch Doppelkick im Projektexplorer.

    In der Registerkarte "BACnet Interface Parameter" können in der Spalte "Wert" die Eigenschaften des Objekts editiert werden.



Parameter	Type	Value	Default Value	Unit	Description
AckedTransitions	STRING	'000'	'000'		AckedTransitions
NotificationClass	UDINT	0	0		NotificationClass
COVIncrement	REAL	0.0	0.0		COVIncrement
Deadband	REAL	0.0	0.0		Deadband
Description	WSTRING(255)	"Description"	"Description"		Description
EventEnable	STRING	'111'	'111'		EventEnable
EventState	Enumeration of DINT	STATE_NORMAL	STATE_NORMAL		EventState
HighLimit	REAL	1.0	1.0		HighLimit
LimitEnable	STRING	'11'	'11'		LimitEnable
LowLimit	REAL	-1.0	-1.0		LowLimit

Abb. 13-12: BACnet Geräteeditor – Konfiguration der Objekteigenschaften

Es können nicht alle **Eigenschaften** über den Geräteeditor konfiguriert werden.

Manche Einstellungen (z. B. Properties mit Array-Datentypen / BACnet-Listen) können nur im SPS-Programm über die Schnittstellen der automatisch instanziierten Funktionsbausteine der Objekte gesetzt werden.

Siehe Kap. "SPS-Schnittstelle zu den Objekteigenschaften" auf Seite 361.

Eine **Online-Anzeige** der aktuellen Werte der Objekteigenschaften ist im eingeloggtem Zustand in der Registerkarte "BACnet Interface E/A-Abbild" innerhalb der FB-Instanzvariable möglich.

Die Registerkarte "BACnet Interface Parameter" zeigt auch im eingeloggten Zustand nur die konfigurierten Werte an und nicht die aktuellen Werte der Objekteigenschaften.

BACnet Server Einstellungen

InstanceNumber (in Device Properties)	Der BACnet Server stellt das Device Objekt dar und repräsentiert das Gerät im BACnet Netzwerk. Jedes BACnet Gerät im Netzwerk benötigt eine eindeutige Device-ID (= Instanznummer des Device Objekts). D.h. die Instanznummer des Server Gerätes ist entsprechend der Netzwerk-Konfiguration zu setzen (Defaultwert ist "1").
AutoStart	• "AutoStart" = FALSE: (Default, Empfohlen) Der BACnet-Server wird nicht automatisch gestartet. Der BACnet-Server kann mit den BACnet-Server FB-Methoden `StartBACnetStack` und `StopBACnetStack` gestartet bzw. gestoppt werden. ☞ Beachten Sie, dass alle BACnet-Objekte mit ihren Eigenschaften beim erstmaligen Aufruf initialisiert werden. Die Initialisierung kann einige Zeit in Anspruch nehmen, nutzen Sie daher die Funktion `StartBACnetStack` in einer separaten Task ohne Watchdog. • "AutoStart" = TRUE: Der BACnet-Server wird automatisch gestartet, wenn die SPS-Applikation geladen wird, die den BACnet-Server enthält. ☞ Es wird eine "BACstac.ini" Datei in der User Partition der Steuerung benötigt. Die Datei enthält die IP-Adresse, Netzwerk Nummer usw. des BACnet-Servers

Tab. 13-13: BACnet Server Einstellungen

Übersicht Registerkarten

BACnet Interface Parameter	Konfiguration der Objekteigenschaften
BACnet Interface E/A-Abbild	Mapping der FB-Instanzvariable, dient auch als Online-Anzeige
Status	Zeigt an, ob der Server bzw. das Objekt "läuft"
Information	Allgemeine Gerätebeschreibung des Objekts

Tab. 13-14: BACnet – Übersicht Registerkarten

Beispiel Konfiguration

Folgendes Beispiel zeigt eine exemplarische Konfiguration von zwei BACnet Objekten, um zur Laufzeit einen analogen Wert in der SPS zu schreiben und mittels der BACnet "NotificationClass" einen Alarm auszugeben, wenn der aktuelle BACnet-Wert einen bestimmten Bereich verlässt.

Führen Sie zur Konfiguration des Beispiels bitte folgende Schritte aus:

1. Fügen Sie der Steuerung über das Kontextmenü einen BACnet Server hinzu

2. Fügen Sie dem BACnet Server über dessen Kontextmenü die Objekte BACnet "AnalogValue" und BACnet "NotificationClass" hinzu
3. Öffnen Sie den Geräteeditor des Objekts "BACnet_Analog_Value" durch Doppelklick im Projektexplorer.

   Öffnen Sie im Geräteeditor die Registerkarte "BACnet Interface Parameter".

   Stellen Sie in der Tabellenspalte "Wert" folgende Eigenschaften ein:

   - AckedTransitions: `'111'`
   - NotificationClass: `1`
   - HighLimit: `10`
   - LimitEnable: `'11'`
   - LowLimit: `0`

   Damit sind für den Analogwert ("PresentValue") die Grenzwerte `[0, 10]` aktiviert und die "NotificationClass" (Instanz 1) wird entsprechende Alarme generieren, wenn der Wertebereich verlassen wird.

4. Deklarieren Sie im SPS-Programm eine Variable `lrValue` vom Typ `LREAL` und fügen Sie folgenden Programm-Code im Implementierungsteil ein:

Programm:

```
BACnet_Analog_Value_1.PresentValue := BACnet_Analog_Value_1.PresentValue + 0.0001;
lrVal := BACnet_Analog_Value_1.PresentValue;
```

5. Laden Sie die Applikation in die Steuerung und starten Sie diese.

   Ergebnis:
   Der "`PresentValue`" des Objekts "`BACnet_Analog_Value`" wird hochgezählt.

6. Starten sie jetzt einen externen BACnet Client, beispielsweise "Innea BACnet Explorer" oder "MBS BACeye"

   Ergebnis:
   Durch den Start des externen Client kann nun folgendes Verhalten beobachtet werden:
   Wenn der "`PresentValue`" den oberen Grenzwert `10` überschreitet, wird vom Server Gerät ein entsprechender Alarm der "`NotificationClass`" generiert

Dieses Beispiel zeigt, wie im SPS-Programm zur Laufzeit auf die Eigenschaften eines BACnet Objekts zugegriffen werden kann.

Wenn kein explizites "Property" eines BACnet Objekt-Funktionsbausteins zur Verfügung steht, können entsprechende Methoden des Funktionsbausteins, oder Hilfsfunktionen der "BACnet" Bibliothek verwendet werden.

Siehe Kap. "SPS-Schnittstelle zu den Objekteigenschaften" auf Seite 361.

BACnet Plugin - Persistence

☞ Das Plugin "Persistence" muss vor BACnet Start Kommando `BACnet_Server.StartBACnetStack();` ausgeführt werden.

Das Plugin kann nicht verwendet werden, wenn der BACnet-Server AutoStart konfiguriert ist

Das Plugin "Persistence" speichert ausgewählte Eigenschaften von Objekten in einer binären Datendatei pro Objekt und führt eine Indexdatei für die Speicherung.

Die Namenskonvention für die Objektpersistenzdatei lautet: `pdata_{deviceId}_{objectType}_{objectInstanceNumber}.dat`.

Die Namenskonvention muss zwingend eingehalten werden, um sicherzustellen, dass Objektinformationen für ein BACnet-Objekt, die mit dem Tupel {deviceId},{objectType},{objectInstanceNumber} übereinstimmen, identifiziert und wiederhergestellt werden können.

Das Plugin "Persistence" bietet eine optionale Funktion zum Speichern und Wiederherstellen von Objekten, die vom Client Service `CreateObject-Requests` erstellt wurden (diese Option ist standardmäßig aktiviert).

Das Plugin "Persistence" muss vor dem Start des BACnet-Servers registriert werden, um die Wiederherstellung von Objekten aus der Persistenz während der Stack-Initialisierung über das Kommando `StartBACnetStack()` sicherzustellen.

Programm:

```
(* Persistenz *)
            objectPersistence.Logging := TRUE;
            objectPersistence.HardShutdown := TRUE;
            objectPersistence.PersistentPath := '/USER/BACnet/bacnetfiles/persistence';
            objectPersistence.RestoreClientCreatedObjects := FALSE;

            // BACnet_PersistentServer.RegisterPlugin(objectPersistence);

            IF BS_Otto.RegisterPlugin(objectPersistence) <> CmpErrors.Errors.ERR_OK THEN
                // Log an Error that it was not successful
                InitError := InitError + 1;
            END_IF

            //BACnet_PersistentServer.RegisterPlugin(logPropertyAttributes);
            IF BS_Otto.RegisterPlugin(logPropertyAttributes) <> CmpErrors.Errors.ERR_OK THEN
                // Log an Error that it was not successful
                InitError := InitError + 1;
            END_IF

            (* ! ! ! Important ! ! ! *)
            // ObjectPersistence.DeleteAllData(); -> Delete only the latest files not the temp files
            (*
                The function ObjectPersistence.DeleteAllData();
                must be called, after the PLC application is changed
          e.g. Download all, new BACnet objects added
                Otherwise a corrupt BACnet configuration is the result ! ! !
            *)
            (* ... *)
(* use DumpDataBase to save property attributes, which are marked to be persistent,
   cyclic executed in PLC program e.g. once a day
   need to be called after ObjectPersistence.DeleteAllData(); , too
*)

   // ObjectPersistence.DumpDataBase(); *)

            (* ... *)
            (* ... *)
```

⚠ VORSICHT **Wenn an BACnet Objekten grundlegende Änderungen vorgenommen werden, können die Persistenz-Daten korrupt werden.**

Löschen Sie die alten Persistenz Daten im jeweils verwendeten Pfad, z.B. mit einem FTP-Client.

In diesem Code-Beispiel lautet der Pfad: `/USER/BACnet/bacnetfiles/persistence`

BACnet Plugin - Schedule

☞ Das Plugin "Schedule" muss nach dem BACnet Start Kommando `BACnet_Server.StartBACnetStack();` ausgeführt werden.

Plugin Funktion:

Das Plugin ermöglicht die Erstellung und Verwaltung von Zeitplänen zur Ausführung wiederkehrender Aktivitäten, sowie Definition einmaliger Ausnahmen zu einer bestimmten Uhrzeit.

BACnet Plugin - Calendar

☞ Das Plugin "Calendar" muss nach dem BACnet Start Kommando `BACnet_Server.StartBACnetStack();` ausgeführt werden.

Plugin Funktion:

Das Plugin ermöglicht die Erstellung und Verwaltung von Abwesenheits-, Feiertags- und Ferienlisten, zum verändern oder pausieren von Scheduler Ereignissen.

Die konfigurierbaren Zeiträume sind ohne Zeitbezug, d.h. nur für ganze Tage möglich.

BACnet Plugin - Notification Class

☞ Das Plugin "Notification Class" muss nach dem BACnet Start Kommando `BACnet_Server.StartBACnetStack();` ausgeführt werden.

Plugin Funktion:

Über das Plugin können Verteilerlisten für Ereignismeldungen erstellt und verwaltet werden.

Einzelne BACnet Objekte werden zu einer Verteilerliste zusammengefasst / verlinkt.

BACnet Plugin - Trend Log

☞ Das Plugin "Trend Log" muss nach dem BACnet Start Kommando `BACnet_Server.StartBACnetStack();` ausgeführt werden.

Plugin Funktion:

Das Plugin liefert eine Trend-Aufzeichnung von BACnet Objektwerten über einen definierten Zeitraum mit der Möglichkeit zur visuellen Anzeige oder zur Übertragung in eine Datenbank.

SPS-Schnittstelle zu den Objekteigenschaften

Für den BACnet Server und die BACnet Objekte wird automatisch eine entsprechende Funktionsbaustein-Instanz angelegt (aus der "BACnet" Bibliothek). Diese IEC-Instanz hat den gleichen Namen wie die Objekte im Projektexplorer und stellt die Schnittstelle der BACnet Objekte zur SPS dar. Das Mapping der Funktionsbaustein-Instanzvariablen wird auch auf der Registerkarte E/A-Abbild des jeweiligen Geräteeditors angezeigt.

Für die am häufigsten verwendeten Objekteigenschaften stellen die Objekt-Funktionsbausteine entsprechende Funktionsbaustein-Propertys zur Verfügung, die einen direkten Zugriff zum Lesen und Schreiben der Eigenschaften

ermöglichen. Daneben gibt es noch spezifische Objekt-Methoden oder allgemeine Zugriffsfunktionen.

Die spezifischen Objekt-Funktionsbausteine leiten von dem Basis-Funktionsbaustein "BACnetStaticObjectBase" und damit von "BACnetObjectBase" ab. Die Basis-Funktionsbausteine stellen Methoden zur Verfügung, die dann in allen Objekt-Funktionsbausteinen zur Verfügung stehen.

SetPropertyInstance

Die Möglichkeit eine beliebige BACnet Eigenschaft zu verändern bietet zum Beispiel die Methode SetPropertyInstance:

- Die Daten des zu setzenden BACnet-Property werden in Form der Struktur IEC_BACNET_PROPERTY_CONTENTS übergeben.
- Diese Struktur beinhaltet neben einer Enumeration, die den Datentyp beschreibt, auch einen Puffer (Buffer), der einen POINTER TO BYTE auf die tatsächlichen Daten besitzt.
- Damit ist es möglich, beliebige BACnet-Property-Datenstrukturen innerhalb der IEC_BACNET_PROPERTY_CONTENTS-Struktur zu übergeben. Der angegebene Datentyp definiert dabei, auf welche Strukturen oder Daten der Puffer zeigt. Der SPS-Datentyp IEC_BACNET_X entspricht hierbei den BACnet-Datentyp DATA_TYPE_X.

Für einige häufig verwendete Datentypen bieten die Basis-Funktionsbausteine auch Hilfsfunktionen, z. B. SetBACnetRealProperty, SetBACnetBooleanProperty, u.a. .

Folgendes Beispiel zeigt die Verwendung von IEC_BACNET_PROPERTY_CONTENTS, wie ein BACnet-Property vom Typ REAL auf den Wert 0.0 gesetzt wird.

Programm:

```
METHOD SetBACnetRealProperty
VAR_INPUT
 propID    : CmpBACnet.IEC_BACNET_PROPERTY_ID;
 rVal      : REAL;
END_VAR
VAR
 BACreal   : CmpBACnet.IEC_BACNET_REAL              := 0.0;
 conts     : CmpBACnet.IEC_BACNET_PROPERTY_CONTENTS := ( tag := CmbBACnet.IEC_BACNET_DATA_TYPE.DATA_TYPE_REAL,
 nElements := 1,
 buffer    := ( nBufferSize := SIZEOF(LREAL), pBuffer := ADR(BACreal) ),
 rawBuffer := ( nBufferSize := 0, pBuffer :=0) );
END_VAR

BACreal  := rVal;
SetPropertyInstance(propID, -1, conts);
```

GetPropertyInstance

Um eine Objekteigenschaft auszulesen, steht die Methode GetPropertyInstance zur Verfügung:

- Mit der Methode kann innerhalb der Struktur IEC_BACNET_PROPERTY_CONTENTS ein Puffer übergeben werden, der bereits mit dem korrekten Datentyp vorinitialisiert ist und der dann befüllt wird.
- Des Weiteren kann mit der Methode ein Puffer der Größe "0" übergeben werden, der dann automatisch initialisiert wird.

☞ Nach der Benutzung muss der Puffer wieder freigegeben werden!
Verwenden Sie hierzu die Funktion FreeStackAllocatedMemory.

Unabhängig von diesem allgemeinen Fall stehen für die häufigsten Datentypen auch bereits entsprechende Hilfsfunktionen (GetBACnet-x-Property) bereit.

Datentypen

Die Objekt-Funktionsbausteine der "BACnet" Bibliothek verwenden zumeist BACnet-Datentypen in den Funktionsbaustein-Schnittstellen (Methoden-Argumente, Methoden-Rückgabewerte, FB-Properties).

Wenn an den Schnittstellen andere Datentypen verwendet werden, z. B. IEC-Grunddatentypen, dann führen die entsprechenden Methoden und Propertys eine implizite Konvertierung von/nach den entsprechenden intern verwendeten BACnet-Datentypen durch.

`IEC BOOL` ⇔ `CmpBACnet.IEC_BACNET_BOOLEAN`	Die Verwendung von `BOOL` (`TRUE`/`FALSE`) anstelle von `CmpBACnet.IEC_BACNET_BOOLEAN` macht den IEC-Code besser lesbar.
`IEC WSTRING` ⇔ `CmpBACnet.IEC_BACNET_STRING`	Die Verwendung von `WSTRING` ist nötig, da eine Konvertierung von/nach den intern verwendeten BACnet-Datentypen `CmpBACnet.IEC_BACNET_STRING_TYPE.BACNET_STRING_UTF_8` oder `CmpBACnet.IEC_BACNET_STRING_TYPE.BACNET_STRING_ISO_8859_1` erfolgt.
`IEC DATE` / `TIME` / `BacnetDateTime` ⇔ `CmpBACnet.IEC_BACNET_DATE` / `CmpBACnet.IEC_BACNET_TIME`	Die Verwendung von IEC `DATE`/`TIME` ist für den IEC-Programmierer gebräuchlicher und wird daher am Interface bevorzugt.

Tab. 13-15: Beispiele für Datentyp-Konvertierung

Export und Import einer BACnet-Konfiguration im EDE-Format

Voraussichtlich ab Version 15V10 verfügbar.

Die aktuelle BACnet-Konfiguration, die im Projektexplorer des Projekts vorliegt, kann in eine *.csv-Datei in EDE-Format (Engineering Data Exchange) exportiert werden.

- Der Befehl **BACnet ▸ Export in EDE-Datei** öffnet den Dialog EDE-Export zur Angabe des Projektnamens, der Dateiversion und des Autors der letzten Änderung. Danach öffnet sich der Dialog zur Ablage der Datei im Dateisystem.
- Mit dem Befehl **BACnet ▸ Import aus EDE-Datei** kann eine BACnet-Konfiguration aus einer EDE-Datei in ein Projekt geladen werden. Der Befehl öffnet den Standarddialog zum Auswählen der entsprechenden *.csv-Datei im Dateisystem. Danach erscheint der Dialog EDE-Import mit Informationen zur gewählten EDE-Datei.

  Wenn die gewählte EDE-Datei die BACnet-Konfigurationen mehrerer Geräte enthält, wählen Sie die gewünschte Gerät aus (EDE-Datei aus BACnet-Gerät). Wenn Ihr Projekt mehrere Steuerungen enthält, wählen Sie, unter welchem Gerät die BACnet-Konfiguration importiert werden soll.

13.2.4 BACnet/IP Client konfigurieren

Unter einem BACnet Server können neben BACnet Objekten auch Client Funktionsbausteine als "Geräte" hinzugefügt werden.

Die BACnet Client Funktionalität setzt einen konfigurierten BACnet Server voraus, siehe Kap. 13.2.3 "BACnet Server konfigurieren" auf Seite 352

Wie auch beim BACnet Server und bei den BACnet Objekten wird auch bei den BACnet Client Funktionsbausteinen automatisch eine IEC-Instanz mit dem gleichen Namen angelegt.

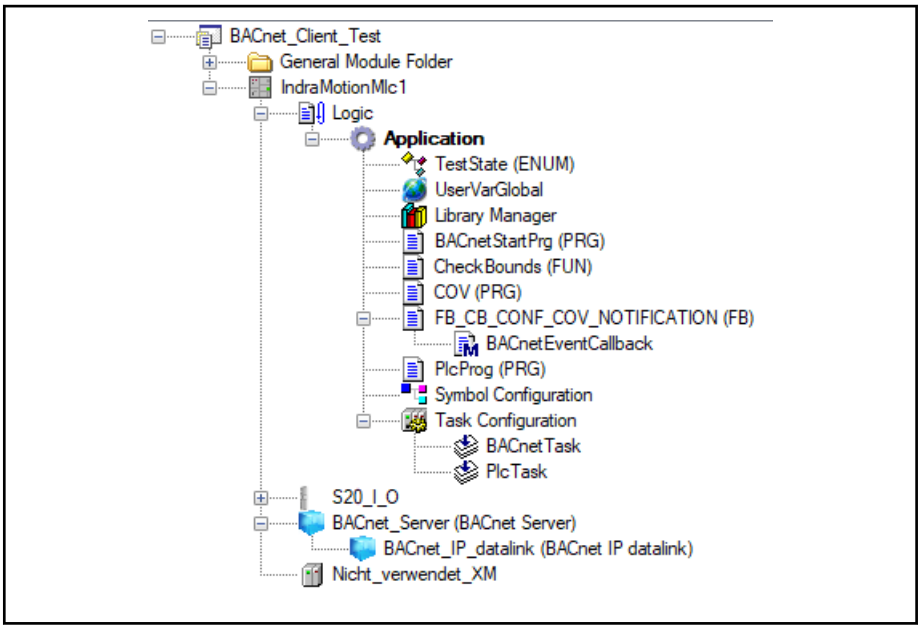


Abb. 13-13: BACnet Client Konfiguration

Da die Client Funktionalität keine statische Konfiguration benötigt, ist das Anlegen der Client "Geräte" im Projektbaum nicht zwingend erforderlich. Die Funktionsbausteine können auch manuell im SPS-Programm instanziiert werden.

Bei einer manuellen Instanzierung der Client-Funktionsbausteine muss dann allerdings noch die Verknüpfung mit dem Server hergestellt werden. Die Verknüpfung muss im SPS-Programm mit der Client FB-Methode `RegisterToServer(BACnet_Server)` erfolgen.

Bei der Konfiguration der Client FBs im Projektexplorer erfolgt dies automatisch.

☞ Auch wenn eine Steuerung nur als BACnet Client eingesetzt werden soll, muss der BACnet Server konfiguriert werden, siehe Kap. 13.2.3 "BACnet Server konfigurieren" auf Seite 352.

Das Server-Gerät im Projektbaum instanziert das "Device-Objekt", das jedes BACnet Gerät benötigt.

Die BACnet Client-Funktionsbausteine folgen dem "Common Behaviour Model", verfügen also über Ein- und Ausgänge zur Ablaufsteuerung:

- Eingänge `xExecute` und `xAbort`
- Ausgänge `xBusy`, `xDone`, `xError` und `xAborted`

- Ein- und Ausgänge, die die Client Anfrage-Parameter oder -Ergebnisse repräsentieren.

Beispiel SPS-Programm

Folgender Code sendet mit Hilfe des Funktionsbausteins "`BACnetClient-ReadProperty`" aus der BACnet.library einen "`Client-Read-Property-Request`" an ein bestimmtes BACnet Gerät im BACnet Netzwerk.

Zum Auslesen der Ausgangsvariablen `readProp.result` vom Typ `IEC_BACNET_PROPERTY_CONTENTS` wird die Hilfsfunktion "`GetRealFrom-Contents()`" verwendet.

Programm:

```
PROGRAM PlcProg
VAR
    initDone          : BOOL := FALSE;
    readProp          : BACnet.BACnetClientReadProperty;
    readPropCnt       : INT := 0;
    readPropErrorCnt  : INT := 0;
    readPropCancelCnt : INT := 0;
    readPropVal       : REAL;
END_VAR

IF NOT initDone THEN
    readProp.RegisterToServer(BACnet_Server);
    readProp(
        dwTargetDeviceNumber := 43,
        objType:= BACnet.CmpBACnet.IEC_BACNET_OBJECT_TYPE.OBJ_ANALOG_VALUE,
        objInst:= 1,
        propID:= BACnet.CmpBACnet.IEC_BACNET_PROPERTY_ID.PROP_PRESENT_VALUE,
        nIndex:= -1);
END_IF

IF readProp.xExecute AND readProp.xBusy AND readProp.iState = 3 THEN
    IF readPropCnt MOD 2 = 0 THEN // cancel every second request
        readProp.xAbort := TRUE;
        readPropCancelCnt := readPropCancelCnt + 1;
    END_IF
END_IF

IF NOT readProp.xExecute AND NOT readProp.xAborted AND NOT readProp.xDone THEN
    readProp.xExecute := TRUE;
    readPropCnt := readPropCnt + 1;
END_IF

IF readProp.xExecute THEN
    IF readProp.xDone THEN
        readProp.xExecute := FALSE;
    END_IF
    IF readProp.xError THEN
        readProp.xExecute := FALSE;
        readPropErrorCnt := readPropErrorCnt + 1;
    END_IF
    IF readProp.xAborted THEN
        readProp.xExecute := FALSE;
        readProp.xAbort := FALSE;
    END_IF
END_IF

readProp();

IF readProp.xDone OR readProp.xError THEN
    IF readProp.xError THEN
        ; // ToDo handle error
    ELSE
        readPropVal := BACnet.GetRealFromContents(readProp.result);
    END_IF
END_IF
```

13.2.5 Hinweise zu implementierten BACnet Objekten und Diensten

Allgemein

In der hier beschriebenen Implementierung eines BACnet API/Stacks ist ein Großteil des Objektverhaltens, das im BACnet-Standard spezifiziert ist, bereits vorimplementiert.

Dennoch gibt es bestimmte Objektaktionen, die in der Applikation implementiert werden müssen, damit die Aktionen dem BACnet-Standard entsprechen (z.B. um die BTL-Zertifizierung zu erhalten).

Die nachfolgende Beschreibung zeigt eine Auflistung aller Objektaktionen und des Objektverhaltens, die im aktuellen BACnet API/Stack implementiert sind bzw. nicht implementiert sind.

Ergänzend finden Sie Beschreibungen und ein Beispiel dafür, wie fehlendes Objektverhalten implementiert werden kann.

Objekt - Property - Aktionen

Die nachfolgend beschriebenen, allgemeinen Objekt-Property Aktionen sind für alle Objekte implementiert, welche die aufgeführten Propertys besitzen.

Property StatusFlags

Das `StatusFlags`-Property hat einen `Bit-String`-Wert von 4 Bits:

- `IN_ALARM`
- `FAULT`
- `OVERRIDDEN`
- `OUT_OF_SERVICE`

Die Werte von drei dieser Bits beziehen sich auf die Werte anderer Propertys im selben Objekt.

Die Beziehung ist im BACnet-Standard definiert und wird als Aktion im beschriebenen API implementiert:

Logisches Verhalten:

- Das Bit `IN_ALARM` ist logisch `FALSE`, wenn das `EventState`-Property den Wert von `NORMAL` hat.

  Ansonsten ist `IN_ALARM` logisch `TRUE`.
- Das Bit `FAULT` ist logisch `TRUE`, wenn das `Reliability`-Property vorhanden ist und nicht den Wert von `NO_FAULT_DETECTED` hat.

  Ansonsten ist das Bit `FAULT` logisch `FALSE`.
- Das Bit `OVERRIDDEN` zeigt bei einem Wert von `TRUE` an, dass das `PresentValue`-Property und das `Reliability`-Property nicht mehr den physikalischen Eingang abbilden, sondern durch einen anderen Mechanismus überschrieben wurden.

  Ansonsten ist das Bit `OVERRIDDEN` logisch `FALSE`.
- Das Bit `OUT_OF_SERVICE` ist logisch `TRUE` wenn das `OutOfService`-Property vorhanden ist und nicht den Wert von `TRUE` hat.

  Ansonsten ist das Bit `OUT_OF_SERVICE` logisch `FALSE`.

Property OutOfService

Das `OutOfService`-Property (`BOOL`) zeigt den Betriebszustand des physikalischen Punkts an, der das BACnet Objekt repräsentiert:

- `TRUE` = In Betrieb
- `FALSE` = Nicht in Betrieb

Wenn der physikalische Punkt nicht in Betrieb ist, müssen verschiedene andere Propertys beschreibbar sein.

Unter anderem müssen die Propertys `PresentValue` und `Reliability` beschreibbar sein, beispielsweise um verschiedene Ereignisse und Zustände eines Objekts zu simulieren, die während des Normalbetriebs nicht auftreten.

Wenn das `OutOfService`-Property auf `TRUE` gesetzt ist, sind die Ereignis- und Zustands-Propertys schreibbar, auch wenn die Anwenderapplikation keine Zugriffsrechte für die Propertys definiert hat.

Propertys in Zusammenhang mit intrinsischem Reporting

Objekte, die im Standard intrinsische Reporting-Prozeduren definieren, und die auch vom vorliegenden API unterstützt werden, bekommen ihre Propertys bezüglich intrinsischen Reportings automatisch von internen Aktionen im API aktualisiert.

Beispiele:

- Das `EventState`-Property wird im entsprechenden Enumerationswert für den intrinsischen Status gesetzt, in dem sich das Objekt befindet.
- Das `EventTimeStamps`-Property wird aktualisiert, wenn die Ereignisbestätigung gesendet wird.
- Bits im `AckedTransitions`-Property werden ebenfalls entsprechend verändert.

Propertys in Zusammenhang mit Wertänderung Reporting (COV)

Objekte, für die im Standard COV-Reporting-Prozeduren definiert sind, und die ebenso von dem API unterstützt werden, bekommen ihre Propertys bezüglich des COV-Reportings automatisch durch interne Aktionen im API aktualisiert. Auch das Senden von Benachrichtigungen über Werteänderungen an registrierte Teilnehmer für Objekte oder Objekt-Propertys wird vom API durchgeführt.

Propertys in Zusammenhang mit Befehlspriorisierung (Outputs)

Für BACnet-Objekte werden Befehle auf Basis einer fixen Anzahl von Propertys priorisiert, die befehlserzeugenden Einheiten zugewiesen sind. Ein priorisierter Befehl, der an eine kommandierbare Property eines Objekts gerichtet ist, wird über eine `WriteProperty`-Serviceanfrage oder eine `WritePropertyMultiple`-Serviceanfrage ausgeführt. Die Basisanfrage schließt einen bedingten `Priority`-Parameter ein, der von 1 bis 16 reicht. Jede kommandierbare Property eines Objekts hat eine zugehörige Priorisierungstabelle, die durch das `Priority_Array`-Property repräsentiert wird. Das `Priority_Array` besteht aus einem Array von befohlenen Werten, angeordnet nach absteigender Priorität. Der erste Wert im Array entspricht Priorität 1 (höchste) der zweite Wert entspricht Priorität 2, usw., bis zum sechzehnten Wert, der Priorität 16 entspricht (niedrigste). Wenn ein solcher, priorisierter Befehl vom API empfangen wird, werden alle in Absatz 19 im BACnet-Standard beschriebenen Aktionen stattfinden, indem interne Aktionen den priorisierten Befehl korrekt behandeln.

Wenn die Applikation Callbacks für das `Present_Value`-Property und das `Priority_Array`-Property registriert hat, wird sie über den aktuellen Befehlsstatus im Objekt informiert.

Objekt Accumulator

Der `Accumulator`-Objekttyp definiert ein standardisiertes Objekt, dessen Propertys die extern sichtbaren Charakteristiken eines Geräts repräsentieren.

Das Objekt zeigt Messungen an, die durch Impulszählung durchgeführt werden.

Die Impulseingabe am Objekt kann auf zwei Wegen erfolgen:

1. Durch Hinzufügen der `Prop_Pulse_Value_Source`-Property mit einer Referenz auf eine `BINARY` oder `BOOLEAN` Wertequelle
2. Durch Aufruf der Funktion `BACnet.BACnetServer. UpdateAccumulatorDataSourceValue`, die für jeden eingefügten BACnet-Server im Projektexplorer verfügbar ist.

In jedem Fall wird das Toggeln von Werten vom Objekt gezählt und alle im BACnet-Standard definierten Property-Aktionen werden vom API durchgeführt. Die Prozedur des intrinsischen Reportings wird ebenfalls durch interne Aktionen in dem API für dieses Objekt durchgeführt..

Objekt Analog-Input

Der Objekttyp `Analog-Input` definiert ein standardisiertes Objekt, dessen Propertys die extern sichtbaren Charakteristiken eines physikalischen analo-

gen Eingangs repräsentieren. Die Propertys `Min_Pres_Val` und `Max_Pres_Val`, zeigen die die niedrigste und höchste Zahl in EngineeringEngineering-Einheiten an, welche zuverlässig für das `Present_Value`-Property dieses Objekts erreicht werden kann.

`Min_Pres_Val` und `Max_Pres_Val` erzeugen `value out of range`-Fehler, wenn `Present_Value` außerhalb dieses Bereichs liegt.

Auch die Werte der Propertys `High_Limit`, `Low_Limit` und `Deadband` werden überwacht, ob sie außerhalb dieses Bereichs geraten.

Die optional vorhandene `Resolution`-Property, welche die kleinste erkennbare Änderung in `Present_Value` in Engineering-Einheiten angibt, beeinflusst die Einstellung der `Present_Value`-Property direkt und passt ihren Wert automatisch an.

Die `Change-Of-Value`-Reporting-Prozedur und die Prozedur für das intrinsische Reporting sind durch interne API-Aktionen für diesen Objekttyp voll implementiert.

Objekt Analog-Output

Der Objekttyp `Analog-Output` definiert ein standardisiertes Objekt, dessen Propertys die extern sichtbaren Charakteristiken eines physikalischen analogen Ausgangs repräsentieren.

Alle Aktionen, die für das `Analog-Input`-Objekt implementiert sind, sind auch für den Objekttyp `Analog-Output` implementiert.

Objekt Analog-Value

Der Objekttyp `Analog-Value` definiert ein standardisiertes Objekt, dessen Propertys die extern sichtbaren Charakteristiken eines analogen Werts repräsentieren.

Das Objekt ist ein Kontrollsystemparameter, der im Speicher des BACnet-Geräts liegt und kein physikalisches Gerät hat.

Alle Aktionen, die für das `Analog-Input`-Objekt implementiert sind, sind auch für den `Analog-Value`-Objekttyp implementiert.

Objekt Averaging

Der Objekttyp `Averaging` definiert ein standardisiertes Objekt, dessen Propertys die extern sichtbaren Charakteristiken eines Werts repräsentiert, der periodisch über eine bestimmte Zeitspanne gesammelt wird. Das `Averaging`-Objekt zeichnet den Minimum,- Maximum- und Durchschnittswert über die Zeitspanne auf und macht die Aufzeichnungswerte als Propertys des `Averaging`-Objekts sichtbar.

Der aufgezeichnete Wert kann der Wert eines Propertys folgendes Typs sein:

- `BOOLEAN`
- `INTEGER`
- `UNSIGNED`
- `ENUMERATED`
- `REAL`

Der Wert des Objekts, das über `Object_Property_Reference` referenziert ist, wird ermittelt und alle abhängigen Property-Werte werden durch interne Aktionen im API aktualisiert (z. B. die Werte `Minimum_Value`, `Maximum_Value`, oder `Average_Value`).

Objekte ohne Implementierung von Standard-Aktionen

In der aktuellen Version des API gibt es keine interne Implementierung von Aktionen zu einer Property, die mit folgenden Objekttypen zusammenhängt:

- Objekttyp `Notification-Class`
- Objekttyp `Program`
- Objekttyp `Structured-View`

Dienste mit Implementierung von Standard-Aktionen

Im Folgenden finden Sie eine Auflistung von Service-Indication Service Hooks, die bereits Standardaktionen implementiert haben und normalerweise nicht in der Applikation implementiert werden müssen:

AcknowledgeAlarm Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, versucht das API das angegebene Objekt zu finden.

Folgende Aktionen werden durchgeführt, wenn das Objekt vorhanden ist und wenn der `Time Stamp`-Parameter mit dem Zeitpunkt übereinstimmt, zu dem das Ereignis zuletzt bestätigt wurde:

- Das Bit in der `Acked_Transitions`-Property des Objekts, das mit dem Wert des `Event State Acknowledged`-Parameters übereinstimmt, wird auf `1` gesetzt.
- Ein Basisergebnis `result(+)` wird angelegt
- Eine Ereignismeldung mit einem `Notify Type`-Parameter "gleich `ACK_NOTIFICATION`" wird ausgegeben

Wenn bei der Suche kein Objekt gefunden wird, wird das Basisergebnis `result(-)` ausgegeben.

Eine Bestätigungsmeldung verwendet denselben Service-Typ (bestätigt oder unbestätigt).

Die Bestätigungsmeldung ist an dieselben Empfänger gerichtet, an welche die ursprüngliche bestätigte oder unbestätigte Ereignismeldung gesendet worden war.

Der `Time Stamp`, der in der Bestätigungsmeldung übertragen wird, wird nicht vom `Time Stamp` der ursprünglichen Ereignismeldung abgeleitet, sondern enthält die Zeit, zu der die Bestätigungsmeldung generiert wird.

GetAlarmSummary Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, sucht das API alle ereignisauslösenden Objekte, die folgende Propertys aufweisen:

- Eine `Event_State`-Property ungleich `NORMAL`
- Und eine `Notify_Type`-Property mit Wert `ALARM`

Wenn bei der Suche Objekte gefunden wurden, wird eine positive Antwort erstellt, welche die `Alarm Summaries` für die gefundenen Objekte enthält.

Wenn keine Objekte gefunden wurden, die den Kriterien entsprechen, wird eine Liste der Länge `0` zurückgegeben.

GetEnrollmentSummary Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, sucht das API alle ereignisauslösenden Objekte anhand von Suchkriterien, die in der Basisanfrage angegeben wurden.

Suchkriterien sind die logischen Verknüpfungen aller explizit gesetzten Filter, die in der Basisanfrage ausgelassen wurden.

Wenn bei der Suche Objekte gefunden wurden, wird eine positive Antwort erstellt, welche die `Enrollment Summaries` für die gefundenen Objekte enthält.

Wenn keine Objekte gefunden wurden, die diesen Kriterien entsprechen, wird eine Liste der Länge `0` zurückgegeben.

SubscribeCOV Service Hook

Wenn die Parameter `Lifetime` oder `Issue Confirmed Notifications` nicht existieren, wird die Anfrage als Stornierung interpretiert.

Jeder COV-Inhalt, der bereits für die gleiche BACnet-Adresse existiert und in der PDU enthalten ist, welche die `SubscribeCOV`-Anfrage befördert und denselben `Subscriber Process Identifier` und `Monitored Object`

`Identifier` enthält, wird deaktiviert und ein `SimpleAck` wird zurückgegeben.

Wenn für eine erstellte Stornierung kein passender COV-Inhalt gefunden wird, wird die Stornierung trotzdem durchgeführt, so als ob ein Kontext existieren würde.

In diesem Fall liefert die Stornierung ein `SimpleAck` als Ergebnis zurück.

Wenn es keine `Lifetime`- Parameter, aber einen `Issue Confirmed Notifications`-Parameter gibt, dann wird ein Wert Null (unbegrenzte Lebensdauer) für `lifetime` angenommen.

Wenn der `Issue Confirmed Notifications`-Parameter existiert, aber das zu überwachende Objekt das COV-Reporting nicht unterstützt, wird ein Fehler ausgegeben.

Wenn das zu überwachende Objekt COV-Reporting unterstützt wird folgende Bedingung geprüft:

- Existiert ein COV-Kontext für die gleiche BACnet-Adresse,
  - die in der PDU enthalten ist,
    - welche die `SubscribeCOV`-Anfrage enthält **und** welche den gleichen `Subscriber Process Identifier` **und** `Monitored Object Identifier` hat.

  Wenn ein existierender COV-Kontext gefunden wird, dann wird die Anfrage so behandelt als ob die Subskription neu erstellt wird.

Wenn bei der Anfrage kein COV-Kontext gefunden wird, dann wird ein neuer COV-Kontext erstellt, der die BACnet-Adresse der PDU beinhaltet, welche die `SubscribeCOV`-Anfrage mit folgenden `Identifiern` enthält:

- `Subscriber Process Identifier` **und** `Monitored Object Identifier`

Wenn kein Kontext erstellt werden kann, es wird ein Fehler zurückgegeben.

Wenn ein neuer Kontext erstellt wird, oder eine neue Subskription erhalten wird, dann wird der COV-Kontext initialisiert und erhält eine Lebensdauer gemäß `Lifetime`-Parameter (sofern vorhanden).

Wenn der `Lifetime`-Parameter nicht existiert erhält die Lebensdauer den Wert `0`.

Die Subskription wird automatisch abgebrochen wenn keine erneute Subskription erhalten wurde.

Eine Lebensdauer von `0` zeigt an, dass die Subskription unbegrenzt ist und es wird nicht automatisch abgebrochen.

In jedem Fall wird `SimpleAck` zurückgegeben.

Nach erfolgreichem Abschluss einer Subskriptions- oder Resubskriptionsanfrage wird im `Issue Confirmed Notifications`-Parameter der Status `ConfirmedCOVNotification` oder `UnconfirmedCOVNotification` ausgegeben.

AddListElementService Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, versucht das API, das Objekt zu verändern, das im `Object Identifier`-Parameter identifiziert wird.

Wenn das identifizierte Objekt existiert und die `Property` besitzt, die im `Property Identifier`-Parameter angegeben ist, wird versucht, alle im

List of Elements-Parameter aufgelisteten Elemente zur angegebenen Property hinzuzufügen.

Wenn das Hinzufügen erfolgreich war, wird ein Basis-SimpleAck erzeugt.

Wenn eines, oder mehrere der Elemente bereits in der Liste stehen, werden sie ignoriert und nicht mehr zur Liste hinzugefügt.

Das Ignorieren eines bereits vorhandenen Elements verursacht kein Fehlschlagen des Services.

In den folgenden Fällen scheitert der Service und erzeugt eine Basis-Fehlermeldung:

- Wenn das angegebene Objekt nicht existiert
- Wenn die angegebene Property nicht existiert
- Wenn die angegebene Property keine Liste ist

Wenn ein- oder mehrere Elemente nicht zur Liste hinzugefügt werden können und auch noch nicht in der Liste enthalten waren, dann wird eine Basis-Fehlermeldung erzeugt und es werden keine Elemente zur Liste hinzugefügt.

Die Wirkung des Services ist, alle angegebenen Elemente der Liste hinzuzufügen, die noch nicht enthalten sind, oder andernfalls überhaupt keine Elemente zur Liste hinzuzufügen.

RemoveListElement Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, versucht das API das Objekt zu verändern, das im Object Identifier-Parameter identifiziert wird.

Wenn das identifizierte Objekt existiert und die Property besitzt, die im Property Identifier-Parameter angegeben ist, wird versucht, alle im List of Elements-Parameter aufgelisteten Elemente aus der angegebenen Property zu entfernen.

Wenn ein- oder mehrere Elemente nicht existieren, oder aufgrund mangelnder Berechtigung oder anderer Probleme nicht entfernt werden können, wird keines der Elemente entfernt und es wird eine Basis-Fehlermeldung erzeugt.

ReadProperty Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, versucht das API, auf die angegebene Property des angegebenen Objekts zuzugreifen.

Das API behandelt den Object Identifier als ob er dem lokalen Device-Objekt entspricht, wenn folgende Bedingungen erfüllt sind:

- Der object-type im Object Identifier-Parameter enthält den Wert Device Object **und** die Instanz im Object Identifier-Parameter enthält den Wert 4194303

Dies ermöglicht eine Device-Instanz eines Geräts, das keine I-Am-Meldungen erzeugt, um bestimmt werden zu können.

Meldungen / Fehlerreaktion:

Wenn der Zugriff gelingt, wird eine Basis-ComplexAck erzeugt, die den gelesenen Wert zurückgibt.

Wenn der Zugriff fehlschlägt, wird eine Basis-Fehlermeldung erzeugt, welche die Ursache für das Fehlschlagen anzeigt.

ReadPropertyMultiple Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, versucht das API, auf die angegebene Property des angegebenen Objekts zuzugreifen.

Es wird die Liste List of Read Access Results erstellt, deren Reihenfolge in der Anfrage spezifiziert ist.

Wenn der List of Property References-Anteil des List of Read Access Specifications-Parameters den Property-Identifier ALL, RE-

QUIRED, oder OPTIONAL enthält, dann wird die List of Read Access Results so zusammengebaut, als ob jedes zurückgelieferte Property explizit referenziert worden wäre.

Das API stellt sicher, dass alle Lesevorgänge automatisch in der kürzestmöglichen Zeit erfolgen und dabei nur höherprioren Abarbeitungen unterworfen sind.

Die Anfrage wird so lange abgearbeitet, bis ein Versuch unternommen wird, auf alle angegebenen Propertys zuzugriefen.

Das API behandelt den Object Identifier als ob er dem lokalen Device-Objekt entspricht, wenn folgende Bedingungen erfüllt sind:

- Der object-type im Object Identifier-Teil des Read Access Specification-Parameters enthält den Wert Device Object **und** die Instanz des Object Identifier-Parameters enthält den Wert 4194303.

Dies ermöglicht eine Device-Instanz eines Geräts, das keine I-Am-Meldungen erzeugt, um bestimmt werden zu können.

Meldungen / Fehlerreaktion:

Wenn keines der angegebenen Objekte gefunden wird, oder auf keine der angegebenen Parameter der angegebenen Objekte zugegriffen werden kann, dann wird ein Basis-Fehler oder ein Basis-ComplexAck erzeugt, das Fehler-Codes für alle Propertys zurückgibt.

Wenn mindestens auf eines der angegebenen Propertys für die angegebenen Objekte zugegriffen werden kann, wird ein Basis-ComplexAck erzeugt, das die gelesenen Werte zurückgibt.

Für alle Propertys, auf die nicht zugegriffen werden konnte wird ein Fehlercode ausgegeben.

☞ **Hinweis für BACnet-Server-Applikationen, welche die interne Datenbank nicht nutzen können und deshalb Unterstützung für den ReadPropertyMultiple-Service implementieren müssen:**

Implementieren Sie den ReadProperty-Service Hook, da ReadPropertyMultiple-Anfragen intern in einzelne ReadProperty-Anfragen aufgeteilt werden und die Fertigstellungen gesammelt werden um die Antwort zu erstellen.

WriteProperty Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, versucht das API, das angegebene Property im angegebenen Objekt zu verändern, wobei es den im Property Value-Parameter bereitgestellten Wert verwendet.

Meldungen / Fehlerreaktion:

Wenn der Änderungsversuch erfolgreich ist, wird ein Basis-SimpleAck erzeugt.

Wenn der Änderungsversuch fehlschlägt, wird eine Basis-Fehlermeldung zur Fehlerursache erzeugt.

WritePropertyMultiple Service Hook

Für jedes Write Access Specification das in der List of Write Access Specifications enthalten ist, wird der Wert jeder angegebenen Property mit dem Property-Wert ersetzt, der in Write Access Specification angegeben ist.

Im Anschluss wird ein Basis-SimpleAck erzeugt, das die erfolgreiche Abarbeitung der kompletten Service-Anfrage anzeigt.

Die vorgesehenen Propertys werden in der Reihenfolge abgearbeitet, die in der List of Write Access Specifications vorgegeben ist.

Wenn während der Abarbeitung ein Property bearbeitet wird, das nicht geändert werden kann, dann gibt der PI eine Basis-Fehlermeldung mit der Fehlerursache aus.

Das Ergebnis dieses Services ist:

- Entweder alle der angegebenen Propertys werden geändert, **oder** es werden nur die Propertys geändert, bis zu dem Property, das im `First Failed Write Attempt`-Parameter angegeben ist (nicht einschließlich des angegebenen Propertys)

Meldungen / Fehlerreaktion:

Wenn keine Schreibaktion erfolgreich ausgeführt wurde und die Service-Anfrage komplett zurückgewiesen wurde, wird `BACnet-Reject-PDU` angezeigt.

Wenn eine der Schreiboperationen aus `List of Write Access Specifications` nicht erfolgreich ausgeführt werden konnte, wird eine Fehlermeldung mit der Fehlerursache zurückgegeben.

ReadRange Service Hook

Das API wird verifiziert zunächst die Gültigkeit der Parameter:

- `Object Identifier`
- `Property Identifier`
- `Property Array Index`

Wenn der `Range`-Parameter nicht existiert, liest das API alle verfügbaren Elemente in der Liste, oder im Array aus und gibt die ausgelesenen Elemente zurück.

Wenn der `Range`-Parameter existiert und er die `By Position`-Parameter angibt, liest das API alle angegebenen Elemente aus und gibt die ausgelesenen Elemente zurück.

Die angegebenen Elemente schließen das Element an der Indexposition ein, die durch `Reference Index` definiert ist, plus die bis `Count` - 1 folgenden Elemente, wenn `Count` positiv ist, oder die bis -1 - `Count` vorangehenden Elemente, wenn `Count` negativ ist.

Das erste Element einer Liste wird mit Index 1 verknüpft.

Wenn es den `Range`-Parameter gibt und er den `By Time`-Parameter angibt, oder die `By Sequence Number`-Parameter, dann weist das API die Anfrage zurück, indem es eine Fehlermeldung mit entsprechender Fehlerklasse und Fehler-Code zurückgibt.

Die zurückgelieferte Antwort überträgt mit Hilfe des `Item Count`-Parameter die Anzahl der gelesenen und zurückgegebenen Elemente.

Die aktuellen Elemente werden im `Item Data`-Parameter zurückgegeben.

Wenn die zurückgegebene Antwort den ersten Positionsindex einschließt, dann ist im `Result Flags`-Parameter das `FIRST_ITEM`-Flag auf `TRUE` gesetzt, andernfalls auf `FALSE`.

Wenn die zurückgelieferte Antwort den letzen Positionsindex einschließt, dann ist in `Result Flags` das `LAST_ITEM`-Flag auf `TRUE` gesetzt, andernfalls auf `FALSE`.

Wenn in der Liste keine Elemente sind, die den `Range`-Parameterkriterien entsprechen, dann wird ein `ComplexAck` mit einem `Item Count` von `0` und keinem `First Sequence Number`-Parameter zurückgegeben.

Die Standard-Implementierung dieses Services unterstützt NICHT den Zugriff auf das `log-buffer`-Property der Trend- und Event-logging Objekte.

Meldungen / Fehlerreaktion:

Falls erforderlich gibt das API gibt eine Fehlermeldung mit entsprechender Fehlerklasse und Fehler-Code zurück.

Potentielle Fehlerursachen:

- Das Objekt oder die Property sind unbekannt
- Die referenzierten Daten sind keine Liste
- Die referenzierten Daten sind ein Array
- Es ist kein Zugriff möglich

☞ Wenn die Applikation Logging-Objekte unterstützen muss, dann müssen die Speicherung und der Zugriff auf die Daten in der Applikation implementiert werden.

SubscribeCOVProperty Service Hook

Die Anfrage als Abbruch interpretiert, wenn weder der `Lifetime`-Parameter noch der `Issue Confirmed Notifications`-Parameter existiert.

Jeder COV-Kontext, der bereits für die gleiche BACnet-Adresse existiert, die in der PDU enthalten ist, die die `SubscribeCOVProperty`-Anfrage befördert und denselben `Subscriber Process Identifier`, `Monitored Object Identifier` und `Monitored Property Identifier` hat, wird deaktiviert und ein `SimpleAck` wird zurückgegeben.

Abbrüche, für die kein passender COV-Kontext gefunden wird, werden erfolgreich abgehandelt, so als ob ein Kontext vorhanden ist und es wird ein `SimpleAck` zurückgegeben.

Wenn ein COV-Kontext gefunden wird, wird er aus dem `Active_COV_Subscriptions`-Property im Device-Objekt entfernt.

Wenn es den `Issue Confirmed Notifications`-Parameter gibt, jedoch die zu überwachende Property das COV-Reporting nicht unterstützt, dann wird eine Fehlermeldung zurückgegeben.

Wenn das zu überwachende Property COV-Reporting unterstützt, dann wird geprüft, ob ein COV-Kontext für die gleiche BACnet-Adresse existiert, die in der PDU enthalten ist, die die `SubscribeCOVProperty`-Anfrage befördert und den gleichen `Subscriber Process Identifier`, `Monitored Object Identifier` und `Monitored Property Identifier` hat.

Wenn ein bestehender COV-Kontext gefunden wird, dann wird die Anfrage als Resubskription interpretiert und erfolgreich behandelt, als wäre es eine neu erzeugte Subskription.

Wenn zur Anfrage kein passender COV-Kontext gefunden wird, dann wird ein neuer COV-Kontext erzeugt.

Der neue COV-Kontext enthält die BACnet-Adresse aus der PDU, welche die `SubscribeCOVProperty`-Anfrage befördert, und den gleichen `Subscriber Process Identifier`, `Monitored Object Identifier` und `Monitored Property Identifier`.

Der neue Kontext wird in das `Active_COV_Subscriptions`-Property des Device-Objekts eingeschlossen.

Wenn kein Kontext erzeugt werden kann, wird eine Fehlermeldung zurückgegeben.

Wenn ein neuer Kontext erzeugt wurde, oder eine Resubskription erhalten wurde, dann wird der COV-Kontext initialisiert und er erhält eine Lebensdauer wie sie im `Lifetime`-Parameter definiert ist.

Die Subskription wird automatisch abgebrochen, wenn keine Re-Subskription erhalten wurde.

Ein SimpleAck wird zurückgegeben und eine ConfirmedCOVNotification oder UnconfirmedCOVNotification wird nach dem erfolgreichen Abschluss einer Subskriptions- oder Resubskriptionsanfrage erzeugt, wie im Issue Confirmed Notifications-Parameter angegeben.

GetEventInformation Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, sucht das API nach allen ereignisauslösenden Objekten, welche die unten aufgeführten Bedingungen erfüllen.

Die Suche beginnt mit dem Objekt nach dem Objekt, welches mit dem Last Received Object Identifier-Parameter definiert ist (in aufsteigender interner Reihenfolge der Objektinstanzen).

Bedingungen:

- Das Objekt enthält ein Event_State-Property, dessen Wert ungleich NORMAL ist, **oder** das Objekt enthält ein Acked_Transitions-Property, in dem mindestens eines der folgenden Bits auf FALSE steht:
  - TO-OFFNORMAL
  - TO-FAULT
  - TO-NORMAL

Eine positive Antwort wird erstellt, welche die "Event Summaries" für Objekte enthält, die bei dieser Suche gefunden wurden.

Wenn keine Objekte gefunden werden, welche den Kriterien entsprechen, wird eine Liste der Länge Null zurückgeliefert.

Es werden so viele Objekte zurückgeliefert, wie sie innerhalb der APDU zurückgeliefert werden können.

Wenn mehr Objekte existieren, welche den Kriterien entsprechen, aber nicht in der APDU zurückgeliefert werden können, wird der More Events-Parameter auf TRUE gesetzt, andernfalls wird er auf FALSE gesetzt.

I-Am Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, fügt das API die erhaltene Information dem Device-Objekt Address_Binding-Property hinzu.

Dieses wird verwendet, um die Geräteinstanznummern in BACnet MAC-Adressen aufzulösen, die für die Kommunikation mit dem Gerät benötigt werden.

I-Have Service Hook

Nachdem die Gültigkeit der Anfrage verifiziert wurde, fügt das API die erhaltene Information einer internen, sortierten Liste hinzu.

Diese wird verwendet, um die Objektnamen im Object_Identifier aufzulösen, die für den Zugriff auf die Objekte benötigt werden.

Who-Has Service Hook

Das API übermittelt die unbestätigte Who-Has-Anfrage, normalerweise unter Verwendung einer Broadcast-Adresse.

Wenn die Parameter Device Instance Range Low Limit und Device Instance Range High Limit vorliegen, dann ist dasjenige API qualifiziert zu antworten, dessen Object_Identifier Instance Number in folgenden Bereich fällt:

Device Instance Range Low Limit ≥ Object_Identifier Instance Number ≤ Device Instance Range High Limit.

Das API

Wenn der Object Name-Parameter vorliegt, dann erfolgt eine I-Have Service-Anfrage von dem API, das ein Objekt mit einem Object_Name-Property-Wert hat, welcher dem Object Name-Parameter entspricht.

Wenn der Object Identifier-Parameter vorliegt, dann erfolgt eine I-Have Service-Anfrage von dem API, das ein Objekt mit einen Object_Identifier-Property-Wert hat, welcher dem Object Identifier-Parameter entspricht.

Who-Is Service Hook

Das API übermittelt die unbestätigte Who-Is-Anfrage normalerweise unter Verwendung einer Broadcast-Adresse.

Wenn die Parameter Device Instance Range Low Limit und Device Instance Range High Limit fehlen, liefert das API ihre Device-Object_Identifier in individuellen Antworten unter Verwendung des I-Am-Service Hook zurück.

Wenn die Parameter Device Instance Range Low Limit und Device Instance Range High Limit vorliegen, dann liefert dasjenige API ihre Device- Object_Identifier über den I-Am-Service Hook zurück, dessen Device Object_Identifier Instance Number in den folgenden Bereich fällt:

Device Instance Range Low Limit ≥ Device Object_Identifier Instance Number ≤ Device Instance Range High Limit

AtomicReadFile Service Hook

Das API verifiziert zunächst die Gültigkeit des Object Identifier.

Falls das Objekt unbekannt ist, gibt das API eine Fehlermeldung result(-) mit entsprechender Fehlerklasse und Fehler-Code zurück.

Der AtomicReadFile-Service Hook unterstützt nur Dateien, auf die über STREAM zugegriffen wird.

Danach wird die Gültigkeit des lokalen Dateizugriffs auf dem lokalen Laufwerk geprüft.

Wenn bei der Gültigkeitsprüfung festgestellt wird, dass die Datei-Startposition ungültig ist, oder nicht auf die Datei zugegriffen werden kann, wird eine Antwort result(-) mit der entsprechenden Fehlerklasse und Fehler-Code zurückgeliefert.

Wenn die Gültigkeitsprüfung ein positive Ergebnis liefert, wird die angegebene Request Record Count Oktett-Anzahl aus der lokalen Datei gelesen.

Wenn die tatsächliche Anzahl der gelesenen Oktetts kleiner ist als die angegebene Anzahl, zeigt der Returned Record Count-Wert der zurückgelieferten result(+)-Antwort die Anzahl der gelesenen Oktetts an.

Wenn das Ende der Datei erreicht wurde, wird der End Of File-Parameter der zurückgelieferten result(+)-Antwort auf TRUE gesetzt.

Der Service Hook wird unter Verwendung der SysFile-Schnittstelle implementiert und unterstützt nur STREAM-Zugriff.

Somit wird die Applikation speziell diesen Service Hook für den Dateizugriff abändern wollen.

AtomicWriteFile Service Hook

Das API verifiziert zunächst die Gültigkeit des Object Identifier.

Falls das Objekt unbekannt ist, gibt das API eine Fehlermeldung result(-) mit entsprechender Fehlerklasse und Fehler-Code zurück.

Der Service Hook unterstützt nur Dateien, auf die über STREAM zugegriffen wird.

Falls die Startposition in der Datei ungültig ist, oder auf die Datei nicht zugegriffen werden kann, wird die Gültigkeit des lokalen Dateizugriffs auf dem lokalen Laufwerk geprüft und eine Antwort result(-) mit der entsprechenden Fehlerklasse und Fehler-Code wird zurückgeliefert.

Wenn der `File Start Position`-Parameter der Service-Anfrage die lokale Dateigröße überschreitet, wird die Datei für das Schreiben der eingehenden Oktetts erweitert.

Wenn der `File Start Position`-Parameter den Wert `-1` aufweist, werden die eingehenden Oktetts am Ende der Datei angehängt.

Wenn das Schreiben der lokalen Datei fehlgeschlagen ist, wird eine `result(-)` Antwort zurückgegeben.

Wenn das Schreiben der lokalen Datei erfolgreich war, wird die Antwort `result(+)` zurückgegeben, welche die `File Start Position` enthält.

Der Service Hook wird unter Verwendung der `SysFile`-Schnittstelle implementiert und unterstützt nur `STREAM`-Zugriff.

Somit wird die Applikation versuchen speziell diesen Service Hook für den Dateizugriff abzuändern.

CreateObject Service Hook

Das API verifiziert zunächst die Gültigkeit der Anfrage indem es prüft, ob der `Object Specifier` der Anfrage einen erzeugbaren Objekttyp angibt (entsprechend der internen Regel: `BACnet.IsBACnetObjectAMEVCreatable`).

Wenn kein erzeugbarer Objekttyp angegeben werden kann, wird die Fehlermeldung `result(-)` von dem API zurückgegeben.

Wenn es im `Object Specifier`-Parameter so angegeben ist, wird die Gültigkeit der Objektinstanznummer geprüft und eine Fehlermeldung `result(-)` zurückgegeben, wenn die Instanznummer nicht mehr existiert.

Wenn keine `List of Initial Values` existiert, werden alle Properties des erzeugten Objekts unter Verwendung der Standardwerte angelegt.

Wenn das Erzeugen eines Objekts fehlschlägt, wird eine Fehlermeldung `result(-)` von dem API zurückgegeben und der `First Failed Element Number`-Wert der Antwort auf `0` gesetzt.

Wenn das Erzeugen erfolgreich war, wird eine `result(+)`-Antwort von dem API zurückgegeben, die den `Object Identifier` der erzeugten Objekte enthält.

DeleteObject Service Hook

Das API verifiziert zunächst die Gültigkeit des `Object Identifier`.

Wenn das Objekt unbekannt ist gibt das API eine Fehlermeldung `result(-)` mit entsprechender Fehlerklasse und Fehler-Code zurück.

Wenn das angegebene Objekt gelöscht werden kann, wird von dem API eine `result(+)` Antwort zurückgeliefert.

Wenn das Löschen fehlschlägt, wird von dem API eine `result(-)` Antwort zurückgegeben, beispielsweise wenn das Objekt nicht nach der internen Regel `BACnet.IsBACnetObjectAMEVCreatable` dynamisch gelöscht werden kann.

Dienste mit internen Implementierungen

Die folgenden Dienste haben keine Standard-Aktionen implementiert, aber das Advanced Client API verwendet dieses Set von Funktionen, um die Daten von entfernten Geräten zu erhalten:

- `ConfirmedCOVNotification` Service Hook
- `ConfirmedEventNotification` Service Hook
- `UnconfirmedCOVNotification` Service Hook
- `UnconfirmedEventNotification` Service Hook

Diese Service Indication Hooks können verwendet werden, um die auf eine `SubscribeCOVProperty`-Anfrage zurückgelieferte Information zu empfangen.

Beispielsweise wird nach einer Subskription eines bestimmten Propertys des Objekts eines Geräts der `COVNotification` Service Hook aufgerufen, um über eine Änderung des Propertys zu informieren.

Dienste ohne Implementierung von Standard-Aktionen

Die folgenden Dienste haben keine Standard-Aktion implementiert. Die Applikation muss eigene Routinen zum Anfrage-Handling implementieren, falls ein Service unterstützt werden soll:

- `DeviceCommunicationControl` Service Hook
- `ReinitializeDevice` Service Hook
- `TimeSynchronization` Service Hook
- `UtcTimeSynchronization` Service Hook

Die hier gelisteten Service Hooks müssen gegebenenfalls von der Anwenderapplikation implementiert werden (abhängig von der Anwenderapplikation und ihren Spezifikationen bezüglich des BACnet-Standards).

Beispiele zur Implementierung von Service Indication Hooks

Implementierungsbeispiel "CreateObject" Service Hook

Das folgende Beispiel zeigt, wie ein `CreateObject` Service Hook implementiert werden kann.

Zuerst muss der Service Hook im Programm-Code eingetragen werden.

Dies erfolgt durch Deklaration des Funktionsbausteins `CallbackFB`, der die Schnittstelle `IBACnetEventConsumer` mit der Funktion `RegisterHook` des BACnetServer-FBs implementiert.

Eintragen des Funktionsbausteins, der für das lokale BACnetServer-Gerät im Projektexplorer eingefügt ist:

```
BACnetServer.RegisterHook(CmpBACnet.IEC_BACNET_CB_TYPE.CB_CREATE_OBJECT, CallbackFB);
```

Sobald eine CreateObject-Service-Anfrage vom BACnetServer aufgefangen wird, wird die Funktion des Funktionsbausteins "BACnetEventCallback" aufgerufen (definiert durch die Schnittstelle `IBACnetEventConsumer`).

Das folgende Beispiel zeigt die Deklaration der `BACnetEventCallback`-Funktion des Funktionsbausteins `CallbackFB`.

Deklarationsteil:

Programm:

```
METHOD BACnetEventCallback : BOOL
VAR_INPUT
   (*If only one event is possibly called, the following Ids can be ignored.
   If there are more at least the EventId should be checked, to get the right type of pParameter.
      ID of the occured Event. Contains the class and the event.*)
   EventId    : DWORD;
   (*Id of the parameter structure (see pParameter)*)
   usParamId  : WORD;
   (*Version of the parameter structure (see pParameter)*)
   usVersion  : WORD;
   (*Pointer to the event specific parameter, that is specified by Id (see GVL EVENTID in CmpBACnet)*)
   pParameter : POINTER TO BYTE;
END_VAR
VAR
   pCreateObj : POINTER TO CmpBACnet.EVT_BACNET_CREATEOBJECT;
   destInstNumber : CmpBACnet.IEC_BACNET_DWORD;

   objSpec    : CmpBACnet.IEC_BACNET_OBJECT_SPECIFIER;
```

```
   objID      : CmpBACnet.IEC_BACNET_OBJECT_ID;

   i          : UDINT;
   bacObj     : BACnetObjectBase;
   runObjID   : CmpBACnet.IEC_BACNET_OBJECT_ID;
   objCount   : UDINT := 0;

   strBuffer  : STRING;
   errCode    : CmpBACnet.IEC_BACNET_ERROR_CODE := 0;
   bacResult  : CmpBACnet.IEC_BACNET_STATUS;
END_VAR
```

Beachten Sie die `pCreateObj`-Variable, welche die Basisinformation der `CreateObject`-Anfrage trägt.

Das folgende Beispiel zeigt die Implementierung der `BACnetEventCallback`-Funktion des Funktionsbausteins `CallbackFB`.

Implementierungsteil:

Programm:

```
BACnetEventCallback := FALSE;

//Check if the issued service event is the one we are interested in
//meaning the create object event
IF EventId = CmpBACnet.EVT_CreateObject AND
   usParamId = CmpBACnet.EVTPARAMID_CmpBACnet_CreateObject AND
   usVersion = CmpBACnet.EVTVERSION_CmpBACnet_CreateObject THEN

   //Get the service information using the proper
   //event structure CmpBACnet.EVT_BACNET_CREATEOBJECT
   pCreateObj := pParameter;

   //The pCBStatus property of the event structure allows to define
   //the result of this service hook. Choosing the default action
   //will advice the API to use its default implementations if available
   //(see also documentation of CB_STATUS_DEFAULT)
   pCreateObj^.pCBStatus^ := CmpBACnet.IEC_BACNET_CB_STATUS.CB_STATUS_DEFAULT;

   //Get the device instance number for the service request and check if it is
   //our device that the request is for (only important for the case of multiple
   //devices in one PLC)
   bacResult := CmpBACnet.BACnetDeviceAddressToInstNumber(pCreateObj^.destinationAddress,
                        ADR(destInstNumber));

   IF bacResult = CmpBACnet.IEC_BACNET_STATUS.BACNET_STATUS_OK
                        AND destInstNumber = m_Device.DeviceInstNumber THEN

      //Get the object specifier defining what to create
      objSpec := pCreateObj^.pServiceInfo^.objectSpecifier;
      //Check if the object type specified is creatable
      IF IsBACnetObjectAMEVCreatable(objSpec.objectSpec.typ) THEN

         //If no object ID is given find a free one
         IF objSpec.tag = CmpBACnet.IEC_BACNET_CREATE_OBJECT_TYPE.CREATE_OBJECT_TYPE THEN
            objID.typ := objSpec.objectSpec.typ;
            objID.instNumber := 1;
            objCount := GetObjectCount();
            i := 1;
            WHILE i <= objCount DO
               GetObject(i, bacObj);
               runObjID := bacObj.ObjectID;
               IF runObjID.typ = objID.typ AND
                                 runObjID.instNumber = objID.instNumber THEN
                  objID.instNumber := objID.instNumber + 1;
                  i := 1;
               END_IF
               i := i + 1;
            END_WHILE
            BACnetEventCallback := TRUE;

         //If the object ID is given check if it is a valid/free object ID
         ELSIF objSpec.tag = CmpBACnet.IEC_BACNET_CREATE_OBJECT_TYPE.CREATE_OBJECT_ID THEN
            objID := objSpec.objectSpec.objectID;
            objCount := GetObjectCount();
            i := 1;
            BACnetEventCallback := TRUE;
            WHILE i <= objCount DO
               GetObject(i, bacObj);
               runObjID := bacObj.ObjectID;
```

```
            IF runObjID.typ = objID.typ AND
                                runObjID.instNumber = objID.instNumber THEN
               errCode := ERR_CODE_OBJ_ID_ALREADY_EXISTS;
               BACnetEventCallback := FALSE;
               EXIT;
            END_IF
            i := i + 1;
         END_WHILE

      ELSE
         //No object specification given which can be interpretated, so
         //issue a standard error
         BACnetEventCallback := FALSE;
      END_IF

   ELSE
      //If the object type is not supported to be created issue the
      //appropriate error
      errCode := ERR_CODE_DYN_CREATION_NOT_SUPP;
      BACnetEventCallback := FALSE;
   END_IF

   //If everything is all right till now (BACnetEventCallback is TRUE) try to
   //create the object
   IF BACnetEventCallback THEN
      IF pCreateObj^.pServiceInfo^.fValuesPresent > 0 THEN
         //If property values are given by the service request use them
         bacResult := CmpBACnet.BACnetConstructObject(destInstNumber,
                           ADR(objID), pCreateObj^.pServiceInfo^.pValues,
                           pCreateObj^.pServiceInfo^.nValuesCount,
                           m_SupportUnsolicitedCOV, 1, m_HideToOutside);
      ELSE
         //If no property values are given by the service request just
         //create the object using the default implementation
         bacResult := ConstructDefaultObject(objID.typ, objID.instNumber,
                           STRING_TO_WSTRING(CONCAT('CreatedObject', DWORD_TO_STRING(objID.instNumber))),
                           bacObj, strBuffer);
      END_IF
   END_IF

   //Check if everything went well
   IF BACnetEventCallback AND bacResult = BACNET_STATUS_OK THEN
      //If everything succeeded issue an result(+) response containing the
      //object ID
      CmpBACnet.BACnetCreateObjectResponseCbCompletion(pCreateObj^.phTransaction,
                     ADR(objID));
      //We treated the service request, so no default API actions are
      //necessary anymore
      pCreateObj^.pCBStatus^ := CmpBACnet.IEC_BACNET_CB_STATUS.CB_STATUS_OK;
   ELSE
      //If something went wrong issue an error
      BACnetEventCallback := FALSE;
      //If no explicit error code was given till now, issue an internal error
      IF errCode = 0 THEN
         errCode := ERR_CODE_INTERNAL_ERROR;
      END_IF
      //Issue the result(-) response with the appropriate error code
      CmpBACnet.BACnetCreateObjectErrorCbCompletion(pCreateObj^.phTransaction,
      ERR_CLASS_OBJECT, errCode, 0);
      //We treated the service request, so no default API actions are
      //necessary anymore
      pCreateObj^.pCBStatus^ := CmpBACnet.IEC_BACNET_CB_STATUS.CB_STATUS_OK;
   END_IF
 END_IF
END_IF
```

Die Hauptaufgabe dieses Code-Beispiels ist es, die Information der Service-Anfrage zu erhalten, um darin auf die Information in `pParameter` zugreifen zu können.

Der Zugriff auf `pParameter` erfolgt, indem ein Casting von `pCreateObj (EVT_BACNET_CREATEOBJECT)` auf `pParameter` durchgeführt wird.

Die Bibliothek `CmpBACnet` enthält die Hook-Anfragestruktur `pCreateObj`, in der die Variable `pCBStatus` enthalten sein kann und die der Rückgabewert an das API ist.

Der Rückgabewert `pCBStatus` entscheidet, ob das API die Service-Anfrage weiter bearbeitet (`Default`) oder nicht (`OK`).

Eine weitere Funktion des Code-Beispiels ist die Prüfung der Propertys in der Service-Anfrage und die Objekterzeugung, wobei auftretende Fehler in der `errCode`-Variable gespeichert werden.

Als Ergebnis werden `result(+)` oder `result(-)` Antworten erzeugt, unter Verwendung der Funktionen:

- `CmpBACnet.BACnetCreateObjectResponseCbCompletion`, **oder** `CmpBACnet.BACnetCreateObjectErrorCbCompletion`

Das Code-Beispiel zeigt die typische Implementierung eines Service Hook, in dem die Event-Mechanismen des BACnet Stack/API genutzt werden.

Um immer dann einen Aufruf zu erhalten, wenn eine Service-Anfrage empfangen wurde, kann ein Event/Callback eingetragen werden.

Die Implementierung des Service Hooks muss dann den Spezifikationen des BACnet-Standards folgen, um einen konformen Service bereitzustellen.

Implementierung von Logging-Objekten (EL / TL / TLM)

Für die folgenden Objekttypen wird vom BACnet Stack/API keine vollständige Implementierung ihrer Funktionalitäten bereitgestellt, da das Speichern von geloggten Daten hardware-spezifisch ist und von der jeweiligen Applikation implementiert wird:

- `Event-Log` (EL)
- `Trend-Log` (TL)
- `Trend-Log-Multiple` (TLM)

Das mögliche Verhalten der Logging-Objekte ist in dem API vorimplementiert, so dass nur das Aufnehmen und Lesen der Trenddaten in der Anwenderapplikation implementiert werden muss.

Implementierung von Datenaufzeichnung für Logging-Objekte

Jedes Logging-Objekt verwendet ein eigenes `Log_Buffer`-Property zur Speicherung der aufgenommenen Daten.

Der Zugriff auf das `Log_Buffer`-Property erfolgt über den `ReadRange` Service Hook, eine interne Darstellung in der Applikation ist daher nicht erforderlich.

Das BACnet Stack/API behandelt den `Log_Buffer` so, als wäre das Log nur eine Aufzeichnung lang und schreibt die eingehenden Daten in diese Aufzeichnung wann immer nötig (z.B. wenn das geloggte Objekt neue Daten produziert).

Für die Implementierung einer Aufzeichnung der geloggten Daten ist es ausreichend, wenn die Schreib-Events auf die `Log_Buffer`-Property erfasst werden und die eingehenden Einzelaufzeichnungen gespeichert werden.

Zur Erfassung kann das `Write-Callback` des BACnet Stack/API verwendet werden.

Der `Callback` wird immer dann aufgerufen, wenn das Property eines Objekts geschrieben wird und der `Callback` des Objekts mit folgender Funktion aktiviert wurde: `BACnet.BACnetObjectBase.SetPropertyCallbackAttachment`

Das Eintragen eines `Write-Callback` erfolgt ähnlich zum Eintragen eines Service Hooks, mit dem einzigen Unterschied, dass eine andere Eintragefunktion verwendet wird.

Mit Hilfe der Funktion `RegisterCallback` des BACnetServer-FBs können bis zu zwei `CallbackFB` Funktionsbausteine eingetragen werden, welche die Schnittstelle `IBACnetEventConsumer` implementieren:

```
BACnetServer.RegisterCallback(CmpBACnet.IEC_BACNET_CALL-
BACK_TYPE.BACNET_CALLBACK_WRITE, CallbackFB, 0);
```

Um die Aufzeichnungen zu sichern, die mit den `EVT_BACNET_WRITEPRO-PERTYCALLBACK` Callback-Daten bei jedem Aufruf geliefert werden, kann die Funktion `BACnetEventCallback` im `CallbackFB` Funktionsbaustein implementiert werden.

Stellen Sie bei der Implementierung sicher, dass nach der korrekten Objekt-ID und der korrekten Property-ID für die Callback-Daten gefiltert wird und dass ausschließlich Aufzeichnungsdaten in das `Log_Buffer` eines Logging-Objekts geschrieben werden.

Implementierung des Datenlesens für Logging-Objekte

Die `Log_Buffer`-Property eines Logging-Objekts kann nur unter Verwendung des `ReadRange` Service Hook ausgelesen werden.

Daher ist die Implementierung des `ReadRange` Service Hook ausreichend, um die Funktion des Datenlesens für die Logging-Objekte zu implementieren.

Der `ReadRange` Service Hook kann durch Eintragen eines `CallbackFB` zum entsprechenden Register Hook implementiert werden, wie in folgendem Beispiel dargestellt: Kap. "Implementierungsbeispiel CreateObject Service Hook" auf Seite 378

Innerhalb der `BACnetEventCallback`-Funktion des `CallbackFB` kann nach der korrekten Objekt-ID und Property-ID gefiltert werden und die korrekten Service-Daten ausgelesen werden (EVT_BACNET_READRANGE).

Innerhalb des ReadRange-Hook müssen exakt die Daten zurückgegeben werden (abhängig von der angegebenen Objekt-ID), die zuvor im `Write-Callback` aufgezeichnet wurden (siehe oben).

Dabei ist der `ReadRange` Service Hook für das Aussortieren von Daten zuständig, abhängig von den angeforderten Aufzeichnungsnummern und Aufzeichnungszeitstempeln.

Zusätzliche Hinweise

Die oben beschriebene Prozedur gilt für alle verfügbaren Logging-Objekte:

- `Event-Log` Objekt
- `Trend-Log` Objekt
- `Trend-Log-Multiple` Objekt

☞ Voraussetzung für das Speichern der Daten ist die Implementierung des `Write-Callback` Service Hook.

Voraussetzung für das Wiederauslesen der Daten ist die Implementierung des `ReadRange` Service Hook.

Ansonsten sind keine Objekt-Aktionen bezüglich Propertys erforderlich.

---

14 Service und Support

Für Ihre schnelle und optimale Unterstützung verfügen wir über ein dichtes weltweites Servicenetz. Unsere Experten stehen Ihnen mit Rat und Tat zur Seite. Sie erreichen uns täglich **rund um die Uhr – auch an Wochenenden und Feiertagen**.

Service Deutschland

Unser technologieorientiertes Competence Center in Lohr deckt alle Belange rund um den Service für elektrische Antriebe und Steuerungen ab.

Sie erreichen unsere **Service-Hotline** und unseren **Service-Helpdesk** unter:

Telefon:	**+49 9352 40 5060**
Fax:	**+49 9352 18 4941**
E-Mail:	service.svc@boschrexroth.de
Internet:	http://www.boschrexroth.com

Auf unseren Internetseiten finden Sie ergänzende Hinweise zu Service, Reparatur (z. B. Anlieferadressen) und Training.

Service weltweit

Außerhalb Deutschlands nehmen Sie bitte zuerst Kontakt mit Ihrem Ansprechpartner auf. Die Hotline-Rufnummern entnehmen Sie bitte den Vertriebsadressen im Internet.

Vorbereitung der Informationen

Wir können Ihnen schnell und effizient helfen, wenn Sie folgende Informationen bereithalten:

- Eine detaillierte Beschreibung der Störung und der Umstände
- Angaben auf dem Typenschild der betreffenden Produkte, insbesondere Typenschlüssel und Seriennummern
- Ihre Kontaktdaten (Telefon-, Faxnummer und E-Mail-Adresse)

Index

0 … 9

0000**** - Sercos I/O status code (für S20 Lokal)..... 248

A

Adresse Sercos-Slave ändern..... 261, 264
Aktivieren / deaktivieren
- Profibus DP-Slaves..... 60
- Profinet IO-Devices..... 98
- Profinet IO-Module..... 110
- S20-Module..... 312
- Sercos-Module..... 267
- Sercos-Slaves..... 261

ANSI Z535.6-2006..... 2
AXIS_REF
- Anwendung Feldbusantrieb..... 43

AxisData[],
- Anwendung Feldbusantrieb..... 45

B

BACnet
- BACnet Server Einstellungen..... 358
- BACnet Server konfigurieren..... 352
- BACnet/IP Client konfigurieren..... 364
- Beispiel Konfiguration..... 358
- Features der BACnet Implementation..... 321
- Funktionsumfang BACnet/IP Client..... 323
- Funktionsumfang BACnet/IP Server / Client..... 321
- Grundlagen..... 325
- Hinweise Implementierungshinweise..... 365
- Übersicht Registerkarten..... 358

BACnet - Dienste mit Implementierung von Standard-Aktionen
- AcknowledgeAlarm Service Hook..... 369
- AddListElementService Hook..... 370
- AtomicReadFile Service Hook..... 376
- AtomicWriteFile Service Hook..... 376
- CreateObject Service Hook..... 377
- DeleteObject Service Hook..... 377
- GetAlarmSummary Service Hook..... 369
- GetEnrollmentSummary Service Hook..... 369
- GetEventInformation Service Hook..... 375
- I-Am Service Hook..... 375
- I-Have Service Hook..... 375
- ReadProperty Service Hook..... 371
- ReadPropertyMultiple Service Hook..... 371
- ReadRange Service Hook..... 373
- RemoveListElement Service Hook..... 371
- SubscribeCOV Service Hook..... 369
- SubscribeCOVProperty Service Hook..... 374
- Who-Has Service Hook..... 375
- Who-Is Service Hook..... 376
- WriteProperty Service Hook..... 372
- WritePropertyMultiple Service Hook..... 372

BACnet Fehler Codes..... 346
BACnet Grundlagen
- Allgemeines..... 325
- Begriffe und Abkürzungen..... 326
- Dienste..... 332
- Dienste - Alarmbehandlung..... 341
- Dienste - Änderungsdienst (COV)..... 333
- Dienste - Auftragspriorisierung (Outputs)..... 340
- Dienste - Datenaufzeichnung (TrendLog)..... 344
- Dienste - Übersicht..... 332
- Fehler Codes..... 346
- Geräte-Profile..... 345
- Netzwerke und Topologien..... 327
- Objekte und Eigenschaften..... 328
- Status Codes..... 349

BACnet Implementierungshinweise
- Allgemein..... 365
- Beispiele zur Implementierung von Service Indication Hooks..... 378
- Dienste mit Implementierung von Standard-Aktionen..... 369
- Dienste mit internen Implementierungen..... 377
- Dienste ohne Implementierung von Standard-Aktionen..... 378
- Implementierung des Datenlesens für Logging-Objekte..... 382
- Implementierung von Logging-Objekten (EL / TL / TLM)..... 381
- Implementierungsbeispiel CreateObject Service Hook..... 378
- Objekt - Property - Aktionen..... 366
- Objekt Accumulator..... 367
- Objekt Analog-Output..... 368
- Objekt Analog-Value..... 368
- Objekt AnalogInput..... 367
- Objekt Averaging..... 368
- Objekte ohne Implementierung von Standard-Aktionen..... 368
- Property OutOfService..... 366
- Property StatusFlags..... 366
- Propertys in Zusammenhang mit intrinsischem Reporting..... 367
- Propertys in Zusammenhang mit Wertänderung Reporting (COV)..... 367

BACnet PICS
- IMPLEMENTATION CONFORMANCE STATEMENT..... 325

BACnet Server
- BACnet Server Inbetriebnahme..... 352
- BACnet Server und Objekte anlegen..... 353
- Calendar..... 361
- COV Notification Beispiel..... 333
- EDE Import / Export..... 363
- Notification Class..... 361
- Persistence..... 359

Schedule..... 361
SPS-Schnittstelle zu den Objekteigenschaften..... 361
Trend Log..... 361
BACnet Status Codes..... 349
Basisadresse modifizieren
S20-Module..... 315
Sercos III-Module..... 274
Befehl
Bus scannen (EtherCAT Master)..... 239
Bus Scannen (S20IO)..... 316
Busdiagnose Ringbruch..... 256
Buskoppler..... 8
Buszyklus-Task
Beschreibung der Buszyklus-Task..... 29
Bus-synchrone SPS-Verarbeitung..... 31
Einstellung "Variablen aktualisieren"..... 32
Konfiguration der Buszyklus-Task..... 30

C
C10A**** - Sercos I/O status code..... 248
C10C**** - Sercos I/O status code..... 248
C10E**** - Sercos I/O status code..... 248
C10F**** - Sercos I/O status code..... 248

D
Diagnose
Sercos III..... 266
Dokumentation
Änderungsverlauf..... 1

E
E/A-Abbild
Sercos III-Module..... 274
E/A-Adresszuweisung der Feldbusantriebe..... 45
E/A-Mapping..... 19
Engineering-Port..... 11
EtherCAT
EtherCAT FAQs..... 240
EtherCAT Master konfigurieren..... 209
EtherCAT Master, Bus Scannen..... 239
EtherCAT Master, Ereignis für extern getriggerte SPS-Task..... 239
EtherCAT Master, Slaves konfigurieren..... 217
Features des EtherCAT Master..... 175
Konfiguration Übersicht..... 205
EtherCAT FAQs
Buskoppler schaltet nicht in OP (Hinweise zu AL Status Codes)..... 243
Übersicht..... 240
Wie erstellt man eine Anlaufparametrierung für die S20-Module?..... 244
Wie konfiguriert man SoE mit Free Run?.... 242
Wie verbindet man IndraWorks Ds über EtherCAT?..... 240
EtherCAT Grundlagen..... 176
Adressierungs- und Identifikationsverfahren..... 179
Azyklische Kommunikation (Mailbox)..... 182
Begriffe und Abkürzungen..... 176
Diagnose..... 186
Diagnose AL Status Codes..... 187
Diagnose Allgemein..... 186
Diagnose CoE SDO Abortcodes..... 190
Diagnose FoE Fehlercodes..... 193
Diagnose SoE Fehlercodes..... 191
EtherCAT Slave Controller..... 203
EtherCAT Slave Controller / Aufbau..... 203
EtherCAT Slave Controller / Kommandos... 204
Hot Connect..... 186
Synchronisation..... 193
Synchronisation / Allgemein..... 193
Synchronisation / Free Run..... 194
Synchronisation mit DC-Sync..... 197
Synchronisation mit SM-Event..... 195
SyncUnits..... 185
Telegramm-Aufbau..... 184
Topologie..... 177
Topologie / Kabelredundanz..... 178
WorkingCounter..... 185
Zustandsmaschine..... 180
EtherCAT Konfiguration Übersicht
Download der Konfiguration..... 207
Funktionspaket EtherCAT Master..... 207
Übersicht Konfigurations-Dialoge..... 208
Verkabelung..... 205
EtherCAT Master konfigurieren
EtherCAT Master - Registerkarte Busdiagnose..... 214
EtherCAT Master, Gerät setzen..... 209
Registerkarte 'Allgemein'..... 209
Registerkarte 'CoE'..... 216
Registerkarte 'Diagnose'..... 217
Registerkarte 'Distributed Clocks'..... 211
Registerkarte 'EoE'..... 216
Registerkarte 'Information'..... 217
Registerkarte 'Sync-Unit-Zuordnung'..... 210
EtherCAT Master, Slaves konfigurieren
Dialog - Diagnose..... 220
Dialog - E/A-Abbild / Mapping zurücksetzen..... 237
Dialog - E/A-Abbild / Variablen aktualisieren..... 237
Dialog - EoE-Einstellungen / IP-Einstellungen..... 232
Dialog - EoE-Einstellungen / Virtueller Ethernet Port..... 232
Dialog - Identifikation..... 223
Dialog - Prozessdaten / Ausgänge auswählen..... 228
Dialog - Prozessdaten / Eingänge auswählen..... 228
Dialog - Prozessdaten Expertenmodus / Bearbeite PDO-Liste..... 226

Dialog - Prozessdaten Expertenmodus / Download ... 227
Dialog - Prozessdaten Expertenmodus / PDO-Info vom Gerät laden ... 227
Dialog - Prozessdaten Expertenmodus / PDO-Inhalt ... 227
Dialog - Prozessdaten Expertenmodus / PDO-Liste ... 226
Dialog - Prozessdaten Expertenmodus / PDO-Zuordnung (SM x) ... 226
Dialog - Prozessdaten Expertenmodus / Sync-Manager ... 226
Dialog - Startparameter / IDN ... 229
Dialog - Startparameter / SDO ... 229
Dialog - Startup-Überprüfung und Time-outs ... 221
Dialog - Watchdog ... 222
Konfiguration der Topologie ... 218
Module hinzufügen ... 217
Registerkarte 'Allgemein' ... 219
Registerkarte 'CoE' ... 234
Registerkarte 'Diagnose' ... 235
Registerkarte 'E/A-Abbild' ... 236
Registerkarte 'EoE-Einstellungen' ... 232
Registerkarte 'ESC Register' ... 233
Registerkarte 'Information ... 238
Registerkarte 'Online' ... 233
Registerkarte 'Prozessdaten Experten-modus' ... 225
Registerkarte 'Prozessdaten' ... 227
Registerkarte 'Startparameter' ... 228
Registerkarte 'SyncManager' ... 224
Slaves am E-BUS hinzufügen ... 218
Slaves und Module hinzufügen ... 217
SoE Parametereditor ... 238
EtherNet/IP
Allgemeines ... 149
Begriffe und Abkürzungen ... 150
FAQs ... 170
Generic Device ... 158
Scanner hinzufügen ... 151
Scanner-Adapter-Kopplung ... 165
EtherNet/IP - FAQs
Häufig auftretende Probleme bei der EtherNet/IP Konfiguration ... 170
Hinweise zur EtherNet/IP Verbindungs-länge (O => T / T => O Größe) ... 172
EtherNet/IP Adapter
Registerkarte 'Adapter-Einstellungen' ... 161
Registerkarte 'E/A-Abbild' ... 163
Registerkarte 'EtherNetIP Parameter' ... 164
Registerkarte 'Feldbusmapping' ... 164
Registerkarte 'Information' ... 165, 169
Registerkarte 'IP-Kanal-Einstellungen' ... 163
Registerkarte 'Status' ... 165
Steuerung als Adapter konfigurieren ... 160
EtherNet/IP Adapter (Engineering Port)
Adapter-Module ... 169
Adapter-Module - Registerkarte 'E/A-Ab-bild' ... 169
Registerkarte 'Information' ... 168
Registerkarte 'Status' ... 168
Übersicht ... 166
EtherNet/IP Adapter (Engineering)
Features ... 149
EtherNet/IP Adapter-Module
Basisadresse modifizieren ... 169
Mapping ... 169
EtherNet/IP Scanner
Adapter - Assemblies ... 158
Adapter - Verbindungen ... 156
Adapter - Zielsystem einstellen ... 154
Adapter hinzufügen ... 154
Features ... 145
Registerkarte 'Allgemein' ... 152
Registerkarte 'E/A-Abbild' ... 153
Registerkarte 'EthernetIP Scanner Para-meter' ... 152
Registerkarte 'Information' ... 154
Registerkarte 'IP-Kanal-Einstellungen' ... 152
Registerkarte 'Status' ... 154
Scanner Einstellungen ... 152
Scanner konfigurieren ... 152
EtherNet/IP Scanner konfigurieren
Übersicht ... 150
EtherNet/IP Scanner, Adapter
Registerkarte 'Allgemein' ... 154
Registerkarte 'Assemblies' ... 158
Registerkarte 'Status' ... 158
Registerkarte 'Verbindungen' ... 156

F

FAQs
EtherNet/IP ... 170
Features des EtherNet/IP Adapter
Features ... 145
Feldbus-Diagnose
Sercos III ... 248
Feldbusantrieb
Antrieb im Urzustand ... 38
Antrieb referenzieren ... 40
AXIS_REF ... 43
AxisData[] ... 45
azyklischen Parameterkommunikation ... 45
Betriebsarteneinstellungen ... 40
Bibliotheken einbinden ... 40
Busteilnehmeradresse ... 40
E/A-Adressierung ... 45
im Projekt anlegen ... 37
Kommunikationseinstellungen IndraDrive ... 40
kommunikationsnotwendige Einstellungen ... 39
Länge des Parameterkanals ... 40
Prozessdaten und Profiltype ... 40
Signalstatuswort ... 40
Signalsteuerwort ... 40

SPS-Programm, notwendige Deklaration..... 43
zyklischer Datenaustausch..... 45
Feldbuskonfiguration
UserDefs.cfg..... 28
Feldbusmapping
EtherNet/IP Adapter..... 24
Profibus DP Slave..... 24
Profinet IO-Device..... 24
Feldbusse und Feldbuskommunikation..... 11
Freeze-Modus
Profibus DP Master..... 51
Profibus DP Slave..... 70, 72

G

Gebäudeautomatisierung
Allgemein..... 319
Einführung und Überblick..... 319
Systemübersicht..... 319
Gefahrenhinweise..... 2
Geräte suchen
Profinet IO..... 113
Gerätedatenbank..... 15
Geräte entfernen..... 19
Geräte hinzufügen..... 16
Gerätedatenbank Dialog..... 15
Gerätedatenbank verwalten..... 15
Übersicht..... 15
Geräteeditor
Profibus DP Master - Registerkarte 'E/A-Abbild'..... 57
Profibus DP Master - Registerkarte 'Feldbus-Diagnose'..... 54
Profibus DP Master - Registerkarte 'Information'..... 58
Profibus DP Master - Registerkarte 'Profibus DP Master IEC-Objekte'..... 57
Profibus DP Master - Registerkarte 'Profibus DP Master Parameter'..... 53
Profibus DP Master - Registerkarte 'Status'..... 58
Profibus DP Slave - Registerkarte 'Allgemein'..... 64
Profibus DP Slave - Registerkarte 'Information'..... 66
Profibus DP Slave - Registerkarte 'Profibus DP Parameter'..... 66
Profibus DP Slave - Registerkarte 'Status'..... 66
Profinet IO Controller - Registerkarte 'Allgemein'..... 88
Profinet IO Controller - Registerkarte 'E/A-Abbild'..... 96
Profinet IO Controller - Registerkarte 'Einstellungen'..... 92
Profinet IO Controller - Registerkarte 'Feldbus Diagnose'..... 93
Profinet IO Controller - Registerkarte 'Information'..... 97
Profinet IO Controller - Registerkarte 'IP-Kanal-Einstellungen'..... 93
Profinet IO Controller - Registerkarte 'Profinet IO Controller Parameter'..... 93
Profinet IO Controller - Registerkarte 'Status'..... 97
Profinet IO Controller - Registerkarte 'Topologie'..... 94
Profinet IO Controller - Registerkarte 'Überblick'..... 91
Profinet IO Device..... 102
Profinet IO Device - Registerkarte 'Allgemein'..... 103, 128
Profinet IO Device - Registerkarte 'E/A-Abbild'..... 129
Profinet IO Device - Registerkarte 'Feldbusmapping'..... 129
Profinet IO Device - Registerkarte 'Information'..... 109, 130
Profinet IO Device - Registerkarte 'IOxS'..... 107
Profinet IO Device - Registerkarte 'Optionen'..... 105
Profinet IO Device - Registerkarte 'PNIO Parameter'..... 108, 129
Profinet IO Device - Registerkarte 'Status'..... 109, 129
Profinet IO Device-Koppelmodule - Registerkarte 'Allgemein'..... 131
Profinet IO Device-Koppelmodule - Registerkarte 'Information'..... 132
Profinet IO Device-Koppelmodule - Registerkarte 'PNIO Module E/A-Abbild'..... 131
Profinet IO Device-Koppelmodule - Registerkarte 'PNIO Module Parameter'..... 131
Profinet IO Device-Koppelmodule - Registerkarte 'Status'..... 132
Profinet IO Module - Registerkarte 'Allgemein'..... 111
Profinet IO Module - Registerkarte 'E/A-Abbild'..... 112
Profinet IO Module - Registerkarte 'Information'..... 113
Profinet IO Module - Registerkarte 'PNIO Parameter'..... 112
Profinet IO Module - Registerkarte 'Status'..... 113
Profinet IO, Controller-Device-Kopplung - Registerkarte 'Allgemein'..... 134
Profinet IO, Controller-Device-Kopplung - Registerkarte 'Information'..... 137
Profinet IO, Controller-Device-Kopplung - Registerkarte 'IOxS'..... 136
Profinet IO, Controller-Device-Kopplung - Registerkarte 'PNIO Parameters'..... 137
Profinet IO, Controller-Device-Kopplung - Registerkarte 'Status'..... 137
Gruppeneigenschaften
Profibus DP Master..... 51

Profibus DP Slave..... 70, 71, 72
GVL_FieldBusDrives..... 44

H
Helpdesk..... 383
Hotline..... 383

I
I/O-Mapping
Adressblöcke..... 19
Globale Variablen..... 22
Lokale Variablen..... 21
IndraDrive
Busteilnehmeradresse..... 40
Einstellungen Prozessdaten und Profiltype... 40
Länge des Parameterkanals..... 40
Signalstatuswort..... 40
Signalsteuerwort 40
IndraWorks DS
Einstellungen Feldbusantriebe..... 39
Indraworks Engineering Desktop
Einstellungen Feldbusantriebe..... 39
IO-Komponenten der XM-Steuerung..... 11
IO-Link..... 287
Einführung und Übersicht..... 287
IO-Link-Devices
Analoge Werte..... 287
Diagnose..... 289
Erstinbetriebnahme..... 288
Gerätetausch..... 290
IO Device Description..... 288
Parametrierung aus der SPS..... 289
Prozessdaten..... 288
Serieninbetriebnahme..... 288
IP-Kanal
Allgemeines..... 33
Anwendungsbeispiele..... 34
Konfiguration..... 33
Konfiguration / Randbedingungen..... 34
Steuerungssystem / Unterstützung..... 33
IP-Kanal Anwendungsbeispiele
IP-Routing über den Engineering Port der Steuerung..... 34
OPC-UA Kommunikation..... 36
UDP oder TCP Kommunikation mit der SPS..... 36

K
Klemme..... 8
Kommunikation und Feldbusse
Allgemeine Informationen..... 7
BACnet..... 321
Begriffe und Abkürzungen..... 7
Buszyklus-Task..... 29
EtherCAT..... 175
EtherNet/IP..... 145
EtherNet/IP Adapter (Engineering Port)..... 166
EtherNet/IP Grundlagen..... 149
EtherNet/IP Scanner..... 150
EtherNet/IP Scanner konfigurieren..... 150
Features der Feldbusse..... 13
Features des EtherNet/IP Adapter..... 145
Features des EtherNet/IP Scanners..... 145
Feldbus übergreifende Features..... 15
Feldbus-Master..... 13
Feldbus-Schnittstellen und Eigenschaften..... 11
Feldbus-Slaves..... 14
IP-Kanal..... 33
Lokales S20 IO..... 303
Profibus DP..... 47
Profibus DP, Grundlagen..... 48
Profinet IO..... 79
Sercos III IO..... 245
Konfiguration
Profinet IO Controller..... 88
Konfigurationsabgleich
Profinet IO..... 118

L
Länge des Parameterkanals, IndraDrive..... 40
Lokales S20 IO
Features..... 303

M
Mapping..... 19
EtherNet/IP Adapter Module..... 169
Sercos III-Module..... 274
Modbus..... 277, 280
'Information'..... 286
Modbus Master..... 280
Modbus Slave Device..... 284
Modbus-Konfigurator..... 278
Registerkarte 'E/A-Abbild'..... 285
Registerkarte 'Modbus - Parameter'..... 285
Registerkarte 'Modbus Master - Allgemein'..... 280
Registerkarte 'Modbus Serial Device - Allgemein'..... 284
Registerkarte 'Modbus Serieller Port - Allgemein'..... 280
Registerkarte 'Modbus Slave - Allgemein'... 281
Registerkarte 'Modbus Slave - Kanal'..... 282
Registerkarte 'Modbus Slave Init'..... 283
Registerkarte 'ModbusTCP Master - Allgemein'..... 280
Registerkarte 'ModbusTCP Slave - Allgemein'..... 281
Registerkarte 'ModbusTCP Slave Device - Allgemein'..... 284
Registerkarte 'Status'..... 286
Modbus Buszyklus-Task..... 277
Modbus RTU..... 277
Modbus Slave Device..... 284
Modbus-Konfigurator..... 278

Modul..... 8

N
Name space..... 42
Namensraum..... 42

P
Profibus DP
Begriffe und Abkürzungen..... 48
Grundlagen..... 48
Konfiguration Übersicht..... 48
Master..... 50
Master konfigurieren..... 50
Module zum Slave hinzufügen..... 67
Profibus DP-Geräteeditor..... 48
Slave (Steuerung als Slave)..... 69
Profibus DP..... 47
Profibus DP Master
Features..... 47
Module zum Slave hinzufügen..... 67
Registerkarte 'E/A-Abbild'..... 57
Registerkarte 'Feldbus-Diagnose'..... 54
Registerkarte 'Information'..... 58
Registerkarte 'Profibus DP Master IEC-Objekte'..... 57
Registerkarte 'Profibus DP Master Parameter'..... 53
Registerkarte 'Status'..... 58
Slaves hinzufügen..... 58
Slaves konfigurieren..... 58, 63
Profibus DP Master-Objekt
DP-Parameter..... 51
Profibus DP Master, Module zum Slave hinzufügen
Allgemeines..... 67
Registerkarte 'Allgemein'..... 67
Registerkarte 'E/A-Abbild'..... 68
Registerkarte 'Information'..... 69
Registerkarte 'Status'..... 69
Profibus DP Slave
Allgemeines..... 69
Features..... 47
konfigurieren..... 69
Koppelmodule hinzufügen..... 74
Koppelmodule hinzufügen - Allgemeines..... 74
Master-Slave-Kopplung..... 76
Registerkarte 'Allgemein'..... 64
Registerkarte 'Information'..... 66
Registerkarte 'Profibus DP Parameter'..... 66
Registerkarte 'Status'..... 66
Steuerung als Slave..... 47
Profibus DP Slave-Objekt
Profibus DP-Parameter..... 72
Profibus DP Slave, konfigurieren
Registerkarte 'Allgemein'..... 70
Registerkarte 'E/A-Abbild'..... 71
Registerkarte 'Feldbusmapping'..... 71
Registerkarte 'Information'..... 74
Registerkarte 'Profibus DP IEC-Objekte'..... 71
Registerkarte 'Profibus DP Konfiguration'..... 73
Registerkarte 'Profibus DP-Parameter'..... 72
Registerkarte 'Status'..... 73
Profibus DP Slave, Module hinzufügen
Registerkarte 'Allgemein'..... 75
Registerkarte 'DP-Module E/A-Abbild'..... 75
Registerkarte 'DP-Module Parameter'..... 75
Registerkarte 'Information'..... 76
Registerkarte 'Status'..... 76
Profibus DP-Slave deaktivieren
UserDefs.cfg..... 61
Profibus DP-Slaves
aktivieren / deaktivieren..... 60
Profinet IO
Ablauf des Gerätetausches..... 120
Anbindung an VPx-Steuerungen..... 85
Anbindung an XM2-Steuerungen..... 84
Anbindung an XM4-Steuerungen..... 85
Austausch eines IO-Devices..... 119
Begriffe und Abkürzungen..... 81
Benennung von IO-Devices - meistgenutzte Verfahren..... 82
Controller konfigurieren..... 88
Controller-Device-Kopplung..... 133
FAQs..... 141
Geräte suchen..... 113
Gerätebeschreibungsdatei..... 84
Geräteeditor..... 88
Gerätenamen..... 82
Gerätetausch..... 119
Grundlagen..... 81
GSDML-Datei..... 84
Identifizieren..... 84
IO-Link-Gateways..... 290
Konfiguration Übersicht..... 84
Konfigurationsabgleich..... 118
Media Redundancy Protokoll (MRP)..... 83
Namensbasierte Geräteadressierung..... 82
Profinet IO-Geräteeditor..... 84
Shared Device..... 83
Topologie..... 86
Topologie und Gerätetausch..... 119
Topologie und Gerätetausch - Allgemeines..... 119
Unterstützung der Topologie- und Gerätetauschfunktionen durch die SPS..... 121
Profinet IO Controller..... 88
Devices hinzufügen..... 97
Devices hinzufügen - Allgemeines..... 97
Devices zum Anschluss von I/O-Modulen... 102
Features..... 79
Geräte suchen..... 113
Module zum Device hinzufügen..... 109
Module zum Device hinzufügen - Allgemeines..... 109
Registerkarte 'Allgemein'..... 88
Registerkarte 'E/A-Abbild'..... 96

Registerkarte 'Einstellungen'..... 92
Registerkarte 'Feldbus-Diagnose'..... 93
Registerkarte 'Information'..... 97
Registerkarte 'IP-Kanal-Einstellungen'..... 93
Registerkarte 'Profinet IO Controller Parameter'..... 93
Registerkarte 'Status'..... 97
Registerkarte 'Topologie'..... 94
Registerkarte 'Überblick'..... 91
Profinet IO Controller-Device-Kopplung
Allgemeines..... 133
Registerkarte 'Allgemein'..... 134
Registerkarte 'Information'..... 137
Registerkarte 'IOxS'..... 136
Registerkarte 'PNIO Parameters'..... 137
Registerkarte 'Status'..... 137
Profinet IO Device
Allgemeines..... 130
Features..... 80
Geräteeditor..... 102, 126
konfigurieren..... 126
Koppelmodule hinzufügen..... 130
Registerkarte 'Allgemein'..... 103, 128
Registerkarte 'E/A-Abbild'..... 129
Registerkarte 'Feldbusmapping'..... 129
Registerkarte 'Information'..... 109, 130
Registerkarte 'IOxS'..... 107
Registerkarte 'IP-Kanal-Einstellungen'..... 128
Registerkarte 'Optionen'..... 105
Registerkarte 'PNIO Parameter'..... 108, 129
Registerkarte 'Status'..... 109, 129
Steuerung als Slave..... 126
Profinet IO Device (Steuerung als Slave)..... 126
Profinet IO Device-Koppelmodule
Registerkarte 'Allgemein'..... 131
Registerkarte 'Information'..... 132
Registerkarte 'PNIO Module E/A-Abbild'..... 131
Registerkarte 'PNIO Module Parameter'..... 131
Registerkarte 'Status'..... 132
Profinet IO IO-Link-Gateways
Beispielhafte Anwendung des Rexroth Profinet IO IO-Link-Gateways S67E-PN-IOL8-DI4-M12-6P..... 296
Beispielhafte Anwendung eines Profinet IO IO-Link-Gateways der Firma Balluff..... 292
Einführung und Übersicht..... 290
Funktionsbausteine für Profinet IO IO-Link-Gateways..... 291
Profinet IO Module
Registerkarte 'Allgemein'..... 111
Registerkarte 'E/A-Abbild'..... 112
Registerkarte 'Information'..... 113
Registerkarte 'PNIO Parameter'..... 112
Registerkarte 'Status'..... 113
Profinet IO-Device
Austausch..... 119
Profinet IO-Devices
aktivieren / deaktivieren..... 98
Profinet IO-Devices deaktivieren
UserDefs.cfg..... 99
Profinet IO-Module
aktivieren / deaktivieren..... 110
Profinet IO-Module deaktivieren
UserDefs.cfg..... 99
Profinet IO, FAQs
IP-Routing zwischen Eng.-Port der Steuerung und Profinet Netzwerk funktioniert nicht?..... 142
PNIO Controller: PN-Device umbenennen funktioniert nicht? (im laufenden Betrieb)..... 142
Profinet IO Controller: wird eine Ringtopologie / Redundanz unterstützt?..... 143
Profinet IO Device (Steuerung als Slave): XMx oder VPx als PN-Device wird sporadisch nicht erkannt..... 143
SPS-Logger Meldung: PN_Topology <station-name> IP x.x.x.x multiple No-of-Peers n..... 142
Übersicht..... 141
Wo findet man die Profinet Gerätebeschreibungsdateien (GSDML) für die Steuerungssysteme?..... 141

R

Redundanz
Sercos III Master..... 255
Registerkarte
EtherNet/IP Adapter - 'Adapter-Einstellungen'..... 161
EtherNet/IP Adapter - 'Information'..... 168
EtherNet/IP Adapter - 'Status'..... 168
EtherNet/IP Adapter - E/A-Abbild..... 163
EtherNet/IP Adapter - EtherNetIP Parameter..... 164
EtherNet/IP Adapter - Feldbusmapping..... 164
EtherNet/IP Adapter - Information..... 165
EtherNet/IP Adapter - IP-Kanal-Einstellungen..... 163
EtherNet/IP Adapter - Status..... 165
EtherNet/IP Scanner - E/A-Abbild..... 153
EtherNet/IP Scanner - EthernetIP Scanner Parameter..... 152
EtherNet/IP Scanner - Information..... 154
EtherNet/IP Scanner - IP-Kanal-Einstellungen..... 152
EtherNet/IP Scanner - Status..... 154
EtherNet/IP Scanner, Adapter - Allgemein..... 154
EtherNet/IP Scanner, Adapter - Assemblies..... 158
EtherNet/IP Scanner, Adapter - Status..... 158
EtherNet/IP Scanner, Adapter - Verbindungen..... 156
IP-Kanal-Einstellungen..... 33

Profibus DP Master - 'E/A-Abbild'..... 57
Profibus DP Master - 'Feldbus-Diagnose'..... 54
Profibus DP Master - 'Information'..... 58
Profibus DP Master - 'Profibus DP Master IEC-Objekte'..... 57
Profibus DP Master - 'Profibus DP Master Parameter'..... 53
Profibus DP Master - 'Status'..... 58
Profibus DP Slave - 'Allgemein'..... 64
Profibus DP Slave - 'Information'..... 66
Profibus DP Slave - 'Profibus DP Parameter'..... 66
Profibus DP Slave - 'Status'..... 66
Profibus DP Slave, konfigurieren - Allgemein..... 70
Profibus DP Slave, konfigurieren - E/A-Abbild..... 71
Profibus DP Slave, konfigurieren - Feldbusmapping..... 71
Profibus DP Slave, konfigurieren - Information..... 74
Profibus DP Slave, konfigurieren - Profibus DP IEC-Objekte..... 71
Profibus DP Slave, konfigurieren - Profibus DP Konfiguration..... 73
Profibus DP Slave, konfigurieren - Profibus DP-Parameter..... 72
Profibus DP Slave, konfigurieren - Status..... 73
Profinet IO Controller - 'Allgemein'..... 88
Profinet IO Controller - 'Feldbus Diagnose'.... 93
Profinet IO Controller - 'Topologie'..... 94
Profinet IO Controller - 'Überblick'..... 91
Profinet IO Controller - E/A-Abbild..... 96
Profinet IO Controller - Information..... 97
Profinet IO Controller - IP-Kanal-Einstellungen..... 93
Profinet IO Controller - Profinet IO Controller Parameter..... 93
Profinet IO Controller - Registerkarte 'Einstellungen'..... 92
Profinet IO Controller - Status..... 97
Profinet IO Device - Allgemein..... 103, 128
Profinet IO Device - E/A-Abbild..... 129
Profinet IO Device - Feldbusmapping..... 129
Profinet IO Device - Information..... 109, 130
Profinet IO Device - IOxS..... 107
Profinet IO Device - Optionen..... 105
Profinet IO Device - PNIO Parameter..... 108, 129
Profinet IO Device - Status..... 109, 129
Profinet IO Device-Koppelmodule - Allgemein..... 131
Profinet IO Device-Koppelmodule - Information..... 132
Profinet IO Device-Koppelmodule - PNIO Module E/A-Abbild..... 131
Profinet IO Device-Koppelmodule - PNIO Module Parameter..... 131
Profinet IO Device-Koppelmodule - Status..... 132
Profinet IO Module - Allgemein..... 111
Profinet IO Module - E/A-Abbild..... 112
Profinet IO Module - Information..... 113
Profinet IO Module - PNIO Parameter..... 112
Profinet IO Module - Status..... 113
Profinet IO, Controller-Device-Kopplung - Allgemein..... 134
Profinet IO, Controller-Device-Kopplung - IOxS..... 136
Profinet IO, Controller-Device-Kopplung - PNIO Parameters..... 137
Profinet IO, Controller-Device-Kopplung - Registerkarte 'Information'..... 137
Profinet IO, Controller-Device-Kopplung - Status..... 137
S20-IO-Objekt - E/A-Abbild..... 309
S20-IO-Objekt - Einstellungen..... 304
S20-IO-Objekt - Information..... 312
S20-IO-Objekt - Registerkarte 'Feldbus Diagnose'..... 307
S20-IO-Objekt - S20 Module Parameter..... 307
S20-IO-Objekt - Status..... 310
S20-Module - Benutzerdefinierte Parameter..... 315
S20-Module - E/A-Abbild..... 315
S20-Module - Information..... 316
Sercos III Modul - Oversampling..... 271
Sercos III-Modul - Allgemeine Ein- und Ausgänge..... 269
Sercos III-Modul - Benutzerdefinierte Parameter..... 271
Sercos III-Modul - E/A-Abbild..... 274
Sercos III-Modul - Funktionsgruppen..... 269
Sercos III-Modul - Information..... 275
Sercos III-Modul - Sercos Modul..... 268
Sercos III-Modul - Sercos Module Konfiguration..... 273
Sercos III-Slave - Allgemein..... 265
Sercos III-Slave - Information..... 266
Sercos III-Slave - sercos Konfiguration..... 266
Sercos III-Slave - Status..... 266
Registerkarte 'E/A-Abbild'..... 285
Registerkarte 'Information'..... 286
Registerkarte 'Modbus - Parameter'..... 285
Registerkarte 'Modbus Master - Allgemein'..... 280
Registerkarte 'Modbus Serial Device - Allgemein'..... 284
Registerkarte 'Modbus Serieller Port - Allgemein'..... 280
Registerkarte 'Modbus Slave - Allgemein'..... 281
Registerkarte 'Modbus Slave - Kanal'..... 282
Registerkarte 'Modbus Slave Init'..... 283
Registerkarte 'ModbusTCP Master - Allgemein'..... 280
Registerkarte 'ModbusTCP Slave - Allgemein'..... 281

Registerkarte 'ModbusTCP Slave Device - Allgemein'... 284
Registerkarte 'Status'... 286
Ring wiederherstellen, Sercos... 258
Ringbruch, Busdiagnose ... 256
RMB_PLCopenFieldBus.library
Anwendung... 41

S

S20 Buszyklustask
Ereignisgesteuerte SPS-Task als Buszyklustask für den S20-Lokalbus konfigurieren... 306
S20 Module... 12
S20-IO-Objekt... 304
Registerkarte 'E/A-Abbild'... 309
Registerkarte 'Feldbus Diagnose'... 307
Registerkarte 'Information'... 312
Registerkarte 'S20 Module Parameter'... 307
Registerkarte 'Status'... 310
S20-Module Parameter... 304
S20-Modul
IO konfigurieren... 304
S20-Module
aktivieren / deaktivieren... 312
Registerkarte 'Benutzerdefinierte Parameter'... 315
Registerkarte 'E/A-Abbild'... 315
Registerkarte 'Information'... 316
S20 Master, Bus Scannen... 316
S20-Module einfügen... 312
S20-Module deaktivieren
UserDefs.cfg... 313
S20-Module konfigurieren... 314
Allgemeines... 314
S20-Objekt und S20-Module
Übersicht... 304
Sercos
IO-Link-Gateways... 300
Registerkarte 'Einstellungen'... 257
Sercos III
Inbetriebnahme von Sercos-IO-Geräten/Modulen... 264
konfigurieren... 246
Module zum Slave hinzufügen... 267
Slave hinzufügen... 259
Slave-Adresse ändern... 259
Sercos III IO-Master
Begriffe... 245
Sercos III konfigurieren
Übersicht... 246
Sercos III Master
Übersicht Busadressen... 247
Verkabelung und Redundanz... 255
Sercos III Modul
Registerkarte 'Oversampling'... 271
Sercos III Module zum Slave hinzufügen
Allgemeines... 267
Sercos III-IO-Master
Features... 245
Sercos III-Master
Feldbus-Diagnose... 248
Sercos III-Modul
Registerkarte 'Allgemeine Ein- und Ausgänge'... 269
Registerkarte 'Benutzerdefinierte Parameter'... 271
Registerkarte 'E/A-Abbild'... 274
Registerkarte 'Funktionsgruppen'... 269
Registerkarte 'Information'... 275
Registerkarte 'Sercos Modul'... 268
Registerkarte 'Sercos Module Konfiguration'... 273
Sercos III-Slave
Registerkarte 'Allgemein'... 265
Registerkarte 'Information'... 266
Registerkarte 'sercos Konfiguration'... 266
Registerkarte 'Status'... 266
Sercos IO-Link-Gateways
Einführung und Übersicht... 300
Funktionsbausteine für Sercos IO-Link-Gateways... 300
Sercos-Adresse ändern... 261, 264
Sercos-IO-Statuscode... 250
Sercos-Module
aktivieren / deaktivieren... 267
Sercos-Module deaktivieren
UserDefs.cfg... 262
Sercos-Slaves
aktivieren / deaktivieren... 261
Sercos-Slaves deaktivieren
UserDefs.cfg... 262
Sercos, Ring wiederherstellen ... 258
Service-Hotline... 383
Sicherheitshinweise... 2
Signalgrafik... 2
Signalwörter... 2
SPS steuert Antriebe über Feldbusverbindung... 36
Steuerung als Adapter konfigurieren
Übersicht... 160
Support... 383
Symbole... 4
Symbolkonfiguration
Feldbusmapping... 24
Sync-Modus
Profibus DP Master... 51
Profibus DP Slave... 70, 72

U

Übertragungsarten
Feldbus-Master... 8
Feldbus-Slaves... 8
UserDefs.cfg

Feldbuskonfiguration..... 28
Profibus DP-Slave deaktivieren..... 61
Profinet IO-Devices / Profinet IO-Module deaktivieren..... 99
S20-Module deaktivieren..... 313
Sercos-Slaves / Sercos-Module deaktivieren..... 262

V
Verkabelung und Redundanz
Sercos III Master..... 255

W
Warnhinweise..... 2

Z
Zu dieser Dokumentation..... 1
Gültigkeit der Dokumentation..... 1

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. +49 932 18 0
Fax +49 9352 18 8400
www.boschrexroth.com/electrics



R911393282

DOK-IWORKS-FB******V15-AP05-DE-P